StorageTek Crypto Key Management Solution

Version 2.0

Management Practices

Whitepaper
July 2008

We welcome your feedback. Please contact Sun StorageTek Test Engineering at:

martha.sammartano@sun.com

or

warner.hersey@sun.com

or

Sun StorageTek Open Systems Test Engineering
Sun Microsystems, Inc.
500 Eldorado Blvd.
Broomfield, CO 80021
USA

Change History

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

Revision	Date	Description
Version 1.0	03/14/2008	Initial Draft
Version 1.1	03/20/2008	First revision incorporating review suggestions (pre-release)
Version 1.2	03/28/2008	Release version
Version 2.0	07/03/2008	Added examples specific to mainframe applications and support for T10000B, T9840D and HP LTO4 encryption capable drives

Audience

This documentation is intended for Sun employees, field personnel, partners and customers who are interested in learning more about the StorageTek encryption solution. We assume that readers are familiar with the information contained in the product documentation listed in the Related Publications section.

Contents

Related Publications

- Key Management System (KMS) 2.0 Systems Assurance Guide, PN 316194801
- Key Management System (KMS) 2.0 Installation and Service Manual, PN 316194901
- Key Management System (KMS) 2.0 Administration Guide, PN 316195101
- Key Management System (KMS) 2.0 Open Systems Implementation Practices White Paper

Glossary

AES—Advanced Encryption Standard

Agent—a storage device used to encrypt and decrypt data

ANSI—American National Standards Institute

API—application programming interfaces

BOT—beginning of tape

Credentials—see Role

Data unit—abstract entity that represents a physical storage object (tape volume)

ELOM—Embedded Lights-Out Manager; a dedicated system of hardware and supporting software that allows management of a Sun server independent of the operating system

FIPS—Federal Information Processing Standard

Key—data encryption key used to encode and decode data

Key group—group of keys associated with a key policy; used to enforce access to key material by encryption agents

Key ID—public identifier used to reference an encryption key

Key policy—policy that defines the lifecycle of keys in key groups associated with it

Key split credentials—a set of userid/passphrase pairs that must be provided to the system to perform certain security-critical operations.

Keystore—secure memory location used to store encryption keys

KMA—Key Management Appliance; contains the key management database, key manager and key store

KMS—Key Management System; a clustered group of KMAs

MVC- Multiple Virtual Cartridge

NIST—National Institute of Standards and Technology

Quickstart—configuration menu executed automatically when a KMA is powered on that collects the configuration data required to initialize the KMA

Quorum—key split credentials

Role—set of permissions that is granted to a KMS user to allow the performance of certain operations: Auditor, Backup Operator, Compliance Officer, Operator or Security Officer

RSA—algorithm for public key encryption

SOAP—Simple Object Access Protocol

TLS—Transport Layer Security

VOLSER—tape volume serial number

VOP—Virtual Operator Panel interface to StorageTek tape drives

VSM—-Virtual Storage Manager

VTCS—Virtual Tape Control System

VTSS—Virtual Tape Storage Subsystem

XML—eXtensible Mark-up Language

Introduction

KMS 2.0 is a comprehensive key management platform designed to address the rapidly growing enterprise commitment to storage-based data encryption. Developed to comply with open security standards, KMS 2.0 provides the capacity, scalability and interoperability to centrally manage encryption keys over widely distributed and heterogeneous storage infrastructures.

KMS 2.0 is specifically designed to meet the unique challenges of storage key management including:

- **Long-term key retention**: To ensure that archived data is always available, KMS 2.0 securely retains encryption keys for the full data lifecycle, which can exceed a decade in length.
- **Interoperability**: Developed to open standards, KMS 2.0 provides the level of interoperability needed to support a diverse range of storage devices attached to mainframe or open systems under a single storage key management service.
- **High Availability**: With active N-node clustering, dynamic load-balancing and automated failover, KMS 2.0 provides high availability whether the appliances are together in the same room or distributed around the world.
- **High Capacity**: KMS 2.0 was built to manage large numbers of storage devices and even more storage keys. A single clustered KMS 2.0 appliance pair can provide key management services for thousands of storage devices and millions of storage keys.

KMS 2.0 encryption supports StorageTek T10000, T10000B and T9840D and HP LTO4 encryption capable tape drives.

The basics of installation and operation of the components of the KMS 2.0 encryption solution are presented in the product documents referenced in the Related Publications section. This paper assumes the reader is familiar with the content of these documents and purposely does not include that material except as needed for emphasis or to provide context for new information.

This paper will address the following topics:

- KMS 2.0 overview
- Basic operations
- Security features
- Key management practices
- Partner key transfer
- Backup and recovery practices
- Disaster recovery practices
- Migration path from KMS 1.x to KMS 2.0

Chapter 1: KMS 2.0 Overview

Architecture

A KMS 2.0 Key Management System (KMS) consists of two main components:

- **Key Management Appliance (KMA)**—Security-hardened box that delivers policy-based key management, authentication, access control and key provisioning services. As a trust authority for storage networks, the KMA ensures that all storage devices (agents) are registered and authenticated and that all encryption key creation, provisioning and deletion are in accordance with prescribed policies. Multiple KMAs are connected via an IP network to form a KMS cluster.
- **KMS Manager**—Graphical user interface that is executed on a workstation and communicates with the KMS cluster over an IP network to configure and manage the system.

The KMS cluster provides redundancy and increased bandwidth. Two networks are provided:

1. Service network for communication between the encryption agents and the KMAs;
2. Management network for inter-KMA traffic and communication with remote management stations.

This isolates the storage devices from heavy corporate network traffic and improves response time for key requests.

KMS 2.0 is designed to interoperate with a variety of agents. Supported agents include StorageTek T10000, T10000B and T9840D and HP LTO4 encryption-enabled tape drives. Content in the main body of this paper describes the behavior of the KMS 2.0 encryption solution with StorageTek tape drives. Behavior specific to the HP LTO4 tape drive is included in the appendix.

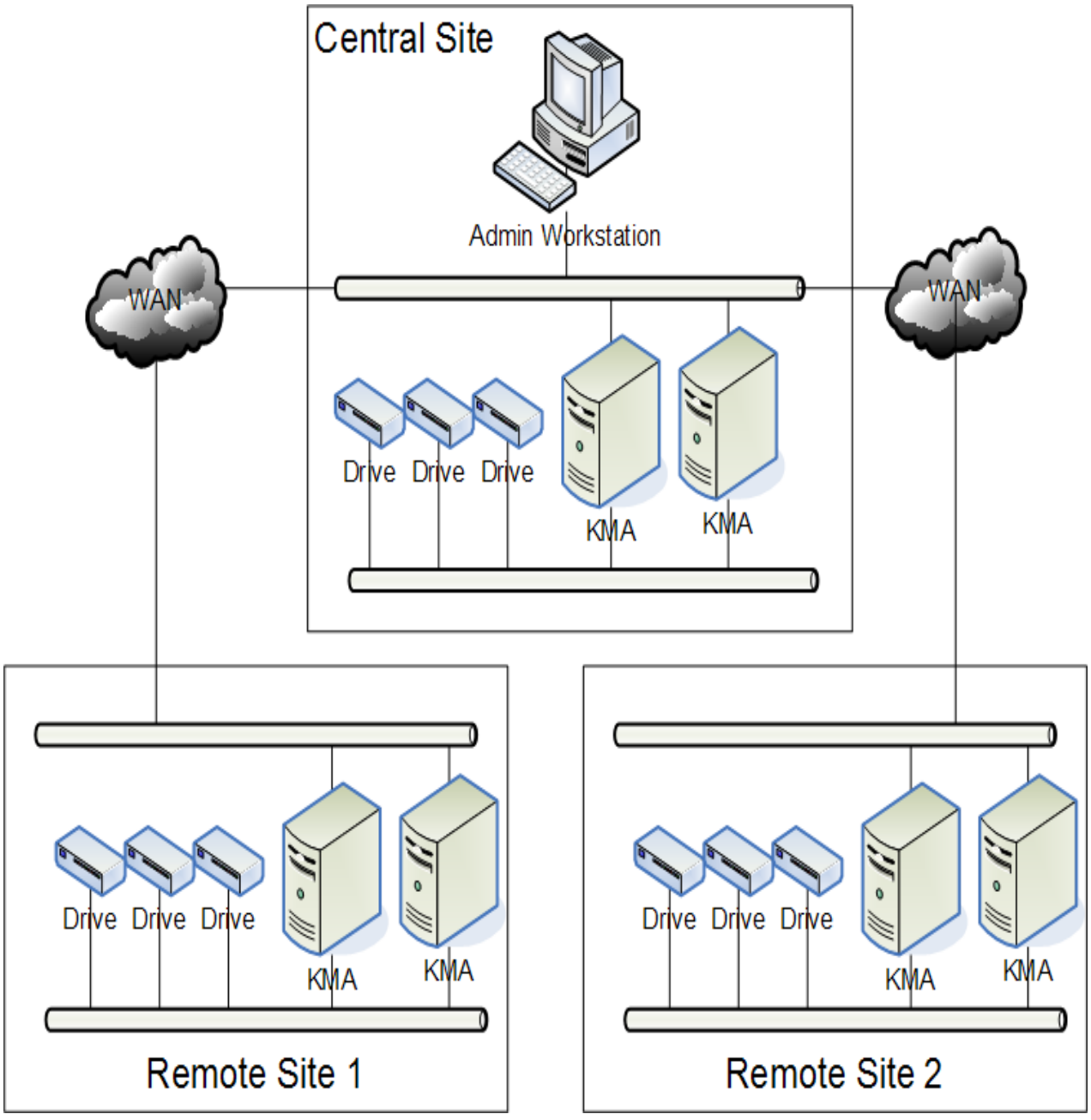
KMS 2.0 supports active N-node clustering with fully automated failover. Drive encryption agents know about all KMAs in the system, and any KMA in the cluster can service any agent. By default, agents are serviced by a local KMA, if available.

In the current implementation, drive networks at different sites (i.e., on different subnets) are isolated. Agents are connected only to KMAs local to their site. Therefore, each site should have a pair of KMAs to provide continuous availability in the event that one KMA fails.

Any KMA can be used for administration functions, and changes made at any KMA are replicated to all other KMAs in the cluster. Keys generated at a site are replicated to all other KMAs in the cluster to allow for easy key sharing among sites and disaster recovery. Likewise,

administrative changes at a site are propagated to all other KMAs in the cluster. All administration functions are executed through the KMS Manager, and one management station can administer all KMAs in the cluster.

The diagram below shows a typical multi-site configuration:



NOTE: Each KMA will have a second connection to the management network (not pictured above) for the ELOM interface used to perform initial configuration tasks.

Features

The features and associated benefits provided by KMS 2.0 are presented in the table below:

Features	*Benefits*
Security	
FIPS certified cryptography	Utilizes FIPS 140-2 certified cryptography to provide AES 256-bit encryption keys for stored data.
Role-based access control	Supports NIST SP800-60 operational roles to segregate operational functions.
Quorum	Requires a minimum number of quorum members to activate a KMA and to restore KMS 2.0 database backups; quorum parameters are fully configurable.
Hardened OS	Provides additional security capabilities to prevent direct attacks on the KMS 2.0 appliance.
TLS communication security	Utilizes TLS to protect all KMS 2.0 appliance and device communication.
High Availability	
Active clustering and failover	Provides high availability through active N-node clustering with fully automated failover.
Load-balancing	Provides active load-balancing for optimization.
Near-synchronous replication	Provides near-synchronous, secure replication of transaction data among appliance nodes.
Roles and Role Segregation	
Granular role segregation	Provides operational segregation through five distinct role definitions—Security Officer, Compliance Officer, Operator, Backup Operator and Auditor. Each role is access controlled and functionally restricted, although users can have multiple roles if desired.
Centralized role assignment	Provides centralized administration that allows operational roles to be centrally assigned and administered. Automated data replication automatically distributes role data to all remote sites for disaster recovery purposes.
Centralized policy management	Provides central management of data encryption policies through the Compliance Officer role. Policies are automatically replicated for disaster recovery purposes.

Operations	
Robust API library	Provides a robust library of APIs to enable operational automation and third-party product integration; allows the rapid integration of third-party storage services, such as storage or backup application managers, to utilize KMS2.0's policy and key management functions.
Audit logging	Maintains audit logs for all operational and key material events and transactions.
Open standards	Supports open standards including standard certificate format X.509v3 certificates, SOAP and TLS.
Management	
Secure management client	Provides a rich, cross-platform compatible client for local and remote management.
Ease-of-use	Provides an easy-to-use and fully configurable management client.
Certification	
FIPS 140-2	Utilizes FIPS 140-2 certified cryptography for all key and cryptographic functions.
NIST algorithm matching	Conforms to NIST guidelines for algorithm matching, including symmetric and asymmetric key pairing and hash and digital signatures.

Chapter 2: Basic Operations

The KMS 2.0 encryption system manages the following objects:

- Key policies and groups
- Agents
- Keys
- Data units

Example 1

We use the following somewhat over-simplified example to illustrate the basic working of the system as it operates with StorageTek encryption drives:

1. A backup or archive application makes a request to write data to a new tape volume.
2. The volume is mounted in a drive whose encryption agent has been enrolled in the KMS and given access to a key group, which is associated with a key policy that defines the characteristics of the keys in that group.
3. The agent requests the creation of a data unit corresponding to the mounted volume and an encryption key. The KMA creates a key in the agent's default key group, provides the key to the agent, associates the new key and data unit, stores both in its database and replicates these changes across the network to all KMAs in the cluster.
4. The agent accepts unencrypted data across the data path and encrypts that data with the new key. The volume is dismounted, and the encryption key is flushed from the drive's memory.
5. The tape volume is mounted again for another write operation.
6. The agent reads the data unit ID written on the media on the first mount and requests keys associated with that data unit from the KMA. Depending on the state of the key used on the previous write, which is determined by its key policy and the time elapsed since its activation, the agent either reuses that key to encrypt the new data or asks the KMA to create a new key to use on the current write operation.
7. The new data is encrypted, the volume is dismounted and the key(s) are flushed from the drive's memory.
8. Sometime later, a request is made to read data written on the tape volume.
9. The volume is again mounted in an encryption drive. The agent requests keys associated with that data unit from the KMA. The agent selects the appropriate key, decrypts the data and returns it in unencrypted form across the data path to the application. The volume is dismounted and the key(s) are flushed from the drive's memory.

The following sections provide more details about the objects managed by the KMS and how they interact.

Key Policies and Groups

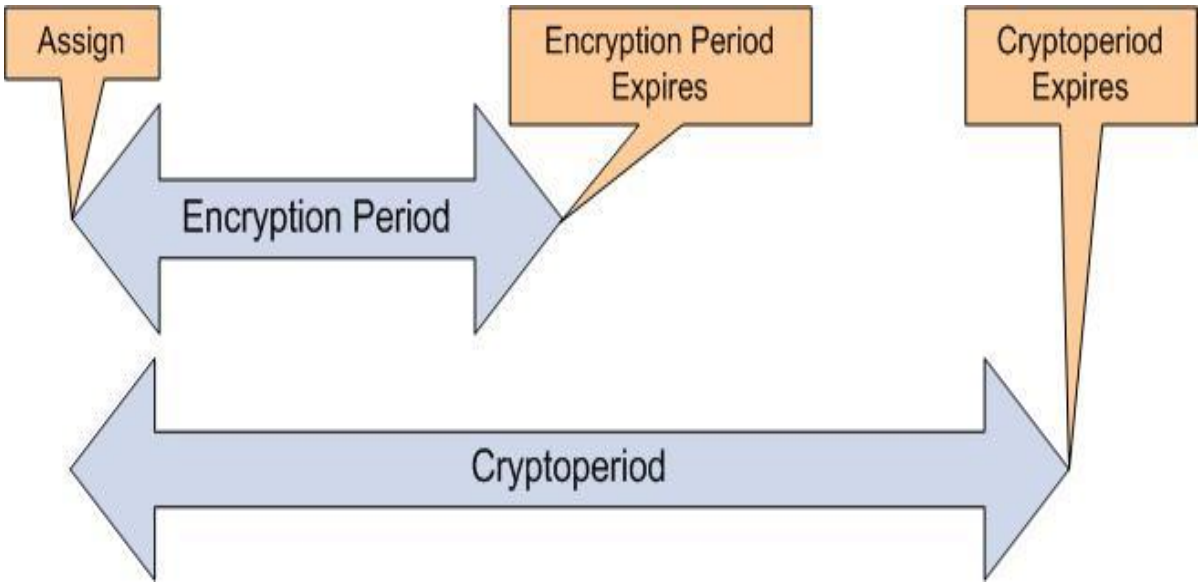
In a KMS 2.0 system, keys are the primary managed object. Keys used to encrypt data are aggregated into key groups, which in turn are associated with a key policy that defines the lifecycle of all keys in the group.

A key policy specifies two important parameters.

Encryption period: The length of time a key may be used to encrypt data.

Cryptoperiod: The length of time that a key is expected to be needed to decrypt data.

The encryption period and the cryptoperiod each start when the key is first used to encrypt data. The cryptoperiod must be at least as long as the encryption period and is typically much longer. The relationship between these two periods is shown in the following diagram:



A key may be used to encrypt data until its encryption period expires. Then that key may be used only to decrypt data written with it. A key whose cryptoperiod has expired can still be used to decrypt data if necessary but is considered to be deactivated and can be destroyed.

For a given tape volume, the goal in most open-systems environments will be to write all data on that volume using one encryption key and to have that key's cryptoperiod expire when all of the data on the volume has reached the projected end of its useful life. The following simple example demonstrates how a key policy can be defined to simplify key management and to allow a key's lifecycle to closely mirror the lifecycle of the data it protects.

Example 2

A backup application in an open-systems environment backs up 50GB of non-compressible data each day from each of 20 servers, uses a common drive pool with a different tape pool for each server's backups and fills each tape in the pool completely before starting a new tape. Each data set must be retained for a period of one year. Data from one server fills a 500GB tape volume every 10 days. We create a key policy with encryption period of 10 days and cryptoperiod of 54 weeks. We create a key group that uses this key policy and assign this key group to be the default key group for every drive in the pool. With this configuration, the same key is used to encrypt all backup files written to one volume (regardless of what drive is used), and that key is deactivated a few days after the retention period for the last backup file written to the tape expires.

Example 3

A backup application in a MVS mainframe environment executes a daily backup. The application is assigned to a particular pool of tapes using drives that are not shared by other applications. Data retention is set for a 1 year period managed by the KMS software, and the data is scattered upon an unknown number of tape volumes within that pool each day. We create a key policy with encryption period of 1 year and cryptoperiod of 2 years. We create a key group that uses this key policy and assign this key group to be the default key group for every drive in the pool for the backup application. All data written to a single tape volume by a drive in that pool will be encrypted with the same key for a period of 1 year. Call this key Key 1. After the 1 year encryption period for Key 1 expires, when the tape volume is mounted next a request for a write operation results in the creation by the KMS of a new key, say Key 2. Data written to this tape volume during the next year is encrypted with the new key Key 2. The data written with Key 1 continues to be accessible although Key 1 is deactivated 2 years after its activation date.

A key policy also indicates whether key groups associated with it allow exporting of keys from the group or importing of keys into the group. Use of these attributes is discussed more fully in Chapters 5 and 9.

Agents

An agent is a peripheral storage device that performs encryption and decryption operations. For this version of the KMS, only tape drive encryption agents are supported. A tape drive is shipped in non-encrypting mode. Configuring the drive for encryption at a customer site is typically done in two steps: enablement and enrollment. Enablement is done by a CSE, who supplies a license key to enable encryption.

Enrollment requires the following steps. Steps 1 and 2 are preparatory steps that are done through KMS Manager by a user with Operator privileges. Step 3, the actual agent enrollment, is done using the VOP interface to the drive.

1. Create an agent for the drive in a KMA, specifying an agent ID and passphrase.

2. Assign one or more key groups to the agent and designate one group as its default key group. (When the agent requests the creation of a new write key, the key will be assigned to the agent's default key group.)
3. Use the VOP interface to set the drive offline. Navigate to Configure → Drive Data. The Encrypt tab will be selected. Enter the following information on the display screen:
    a. Use tokens: Select "No".
    b. Permanently encrypting: Select "Yes" to place the agent permanently in encryption mode. Select "No" to allow encryption mode to be disabled in the future.
    c. Agent ID: Enter the agent ID specified in Step 1.
    d. Pass Phrase: Enter the passphrase specified in Step 1. (This passphrase will be displayed in plain text. Since a new passphrase is required to re-enroll the drive, securing this passphrase is unnecessary.)
    e. KMS IP address: Enter the IP address of a KMA port on the drive network. (The agent will use this IP address to contact the cluster and obtain IP addresses of all KMAs in the cluster.)

NOTE: If "No" is selected in Step 3b, the VOP interface may be used to reset the drive and disable encryption at some later time. However, the drive will remain in encryption mode until this manual mode switch is executed.

When a tape volume is mounted for the first time in a drive whose agent is enrolled in a KMS 2.0 cluster, the agent requests the KMA to create a new data unit and a new key in its default key group that it will use to encrypt data written to the volume. Typically, the agent passes to the KMA the barcode ID (VOLSER) of the tape volume. The KMA populates the External Tag field of the data unit with this 6-digit identifier, which appears on the barcode label on the exterior of the tape cartridge and is communicated to the drive by the library controller (in an automated library setting) or is read from an ANSI label on the tape. (A VOLSER is not available for data units associated with tape volumes without ANSI labels that are used in stand-alone drives.) However, neither the agent nor the KMS uses this tag for locating or processing data units. The External Tag field is provided as a convenient way for the user to easily associate abstract data units displayed by KMS Manager to physical tape volumes.

NOTE: Keys that are transferred to a drive are stored in its memory only as long as the data unit with which they are associated is mounted. When the tape volume is dismounted, all keys associated with it are flushed from its memory.

Key state changes that occur while a key is resident in the drive's memory are unnoticed by the agent. The KMS does not initiate communication with the agent, even when objects it manages change state. When a tape volume is mounted in the drive, the agent requests the KMA to send the keys associated with this data unit. The agent checks the state of each key as it is received and only then. Therefore, a state change for a key in use by the

drive goes unnoticed by the agent until it receives that key from the KMA on a subsequent tape volume mount.

An agent should have access to all key groups associated with data units that it may be asked to process. When it requests the creation of a new key, that key will always be assigned to its default key group, but it can use a key in any other key group to which it has access to encrypt or decrypt data as allowed by the key's current state. For example, if a data unit with an active write key is mounted into a drive for a write operation, the agent will use this key if it has access to the key group to which the key belongs. If not, a write error may be returned to the application requesting the write. (The exception is a write from BOT when the drive does not have access to the active write key for that data unit. In this case, the KMA supplies a new key in the drive's default key group, and the write proceeds.)

Keys

Each key used for encrypting data has a lifecycle that is determined by its associated key policy. It moves through a sequence of states that determine at each stage the operations for which the key may be used.

Key State Transitions

At startup, the KMS system generates a pool of pre-operational keys in the **Generated** state. Before a key can be used, it must be protected against loss by automatic replication across a multi-node cluster or, in a single-node system, by manual creation of a system backup. When one of these actions completes, the key may be used to encrypt data and is moved to the **Ready** state. (Because of the lack of automatic key protection in a single-node system, only multi-node systems will be sold to customers.)

When a key is first used to encrypt data, it transitions to what is known as the **Protect-and-process** state. Both its encryption period and it cryptoperiod begin at this time. In this state, it may be used both to encrypt and to decrypt data. When its encryption period expires, it transitions to the **Process-only** state, at which time it may be used only to decrypt data. Eventually, the key's cryptoperiod expires, and it moves to the **Deactivated** state. The expiration of the key's cryptoperiod is intended to coincide with the end of the usefulness of the data that it protects, although the transition is purely logical. The key may still be used to decrypt data if needed.

In normal operations, a key will transition from the Generated state to the Deactivated state as dictated by its key policy and remain in that state indefinitely, allowing it to be used to decrypt data as long as the data it protects exists. However, allowance is made for events that would necessitate operator intervention into a key's normal lifecycle.

At any point in a key's life cycle, if the security of the key is known or suspected to have been breached, it may be manually declared to be compromised by a Compliance Officer. A key in

the **Compromised** state will no longer be used to encrypt data but may be used to decrypt data as required. Loss of an encrypted tape volume does not require declaring the keys associated with it to be compromised, since the keys used to encrypt the data are secure. However, if a protect-and-process key is inadvertently exported and shared with a key partner, marking the key as compromised might be appropriate to ensure that no more data is encrypted with it.

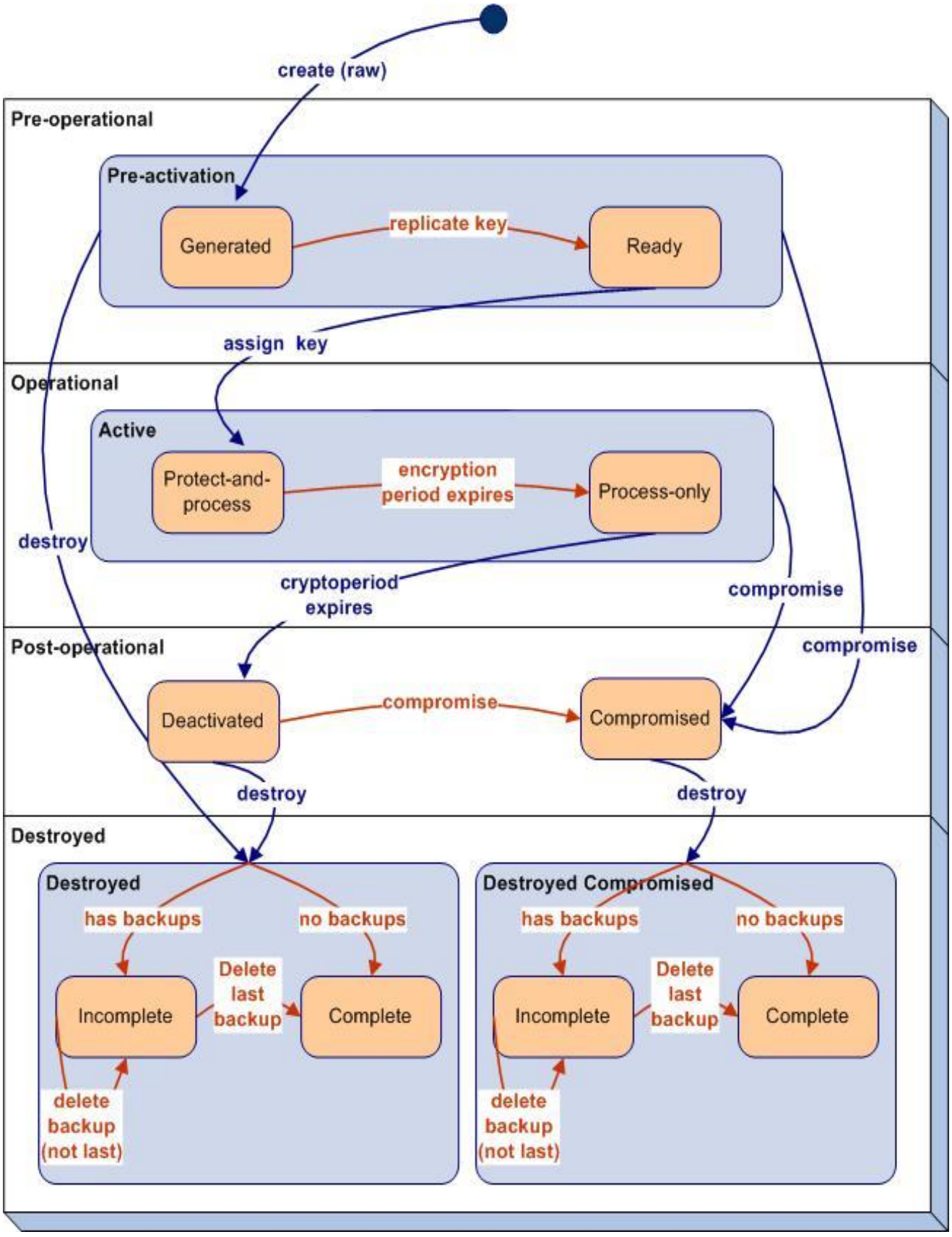
Customers may wish to deny access to some data altogether to enforce retention policies. To allow this, a key that is either deactivated or compromised may be manually placed in the **Destroyed** state. Once a key is marked as destroyed, it will no longer be sent to an agent. If no backup containing the destroyed key exists, the key is marked as **Completely Destroyed**. Otherwise, it is marked as **Incompletely Destroyed**.

Since management of KMS backups is outside the control of the KMS, the key state transition from Incompletely Destroyed to Completely Destroyed requires operator intervention. Once a backup file has been manually deleted, the Backup Operator may mark it as destroyed through KMS Manager. When all backups containing a destroyed key have been marked as destroyed, the key's state will transition to Completely Destroyed.

> NOTE: The system makes this key state transition automatically once all relevant backups have been marked as destroyed. However, it cannot verify the destruction of the backups containing the destroyed key.

For completeness, the system also allows a destroyed key in either sub-state to be compromised, moving it to the **Destroyed Compromised** state with the same sub-state. The Destroyed Compromised state is logically equivalent to the Destroyed state.

The diagram on the next page shows the states and transitions between states in the lifecycle of an encryption key:


create (raw)
Pre-operational
Pre-activation
Generated
replicate key
Ready
assign key
Operational
Active
Protect-and-process
encryption period expires
Process-only
destroy
cryptoperiod expires
compromise
compromise
Post-operational
Deactivated
compromise
Compromised
destroy
destroy
Destroyed
Destroyed
has backups
no backups
Incomplete
Delete last backup
Complete
delete backup (not last)
Destroyed Compromised
has backups
no backups
Incomplete
Delete last backup
Complete
delete backup (not last)

Key Destruction

A destroyed key is removed from the KMS system, leaving only metadata attesting to its previous existence. It can no longer be used to decrypt data, and any data encrypted with it is effectively destroyed. Therefore, destruction of keys should be undertaken only after very careful consideration.

WARNING: Key destruction can result in unintended loss of access to useful data. A KMS 2.0 encryption agent cannot position past data for which it has no key. Therefore, data located past destroyed data on a tape is inaccessible even if the key used to encrypt it still exists.

In addition, imported KMS 1.x data units may share a key across multiple data units. Destroying a key associated with one data unit may have the unforeseen side effect of destroying data on other data units.

The operator may select one or more data units and destroy post-activation keys associated with these data units. Options are available to destroy only deactivated keys, only compromised keys or both. Because of the potential side effects described above, this operation should be done very cautiously. Expiring the data through the backup or archive application has the same effect without the risk of unintended consequences.

If the decision is made to destroy keys, the safest policy is to wait until all keys associated with a data unit are deactivated, and then destroy all keys associated with the data unit. Such a policy reduces the risk of unintended data loss. Having only one key associated with a data unit makes it much easier to adhere to this policy.

WARNING: Any user with Operator credentials can destroy deactivated or compromised keys.

Some protections are in place. No active key can be destroyed. An active key must be marked as compromised before it can be destroyed. Marking a key as compromised requires Compliance Officer credentials, which cannot be used to destroy a key. Thus, destroying an active key requires both Compliance Officer and Operator credentials.

A single user with Operator credentials can destroy a deactivated key. However, no user can deactivate a key. That transition happens only when the key's cryptoperiod expires, which should coincide with the last stage of the lifecycle of the data it protects. Therefore, if the cryptoperiod of a key is at least as long as the expected useful life of the data it protects, no single (rogue) user can destroy data encrypted with it until that data has outlived its usefulness.

Data Units

A data unit is any unit of media that holds data that is encrypted. In KMS 2.0, a data unit is a tape volume. In the current KMS system, once a key is created, it is associated with a data unit and remains logically associated with that data unit, even if the data it protects has been overwritten or its key material has been destroyed. The state of the data unit is a function of the state of the key(s) associated with it.

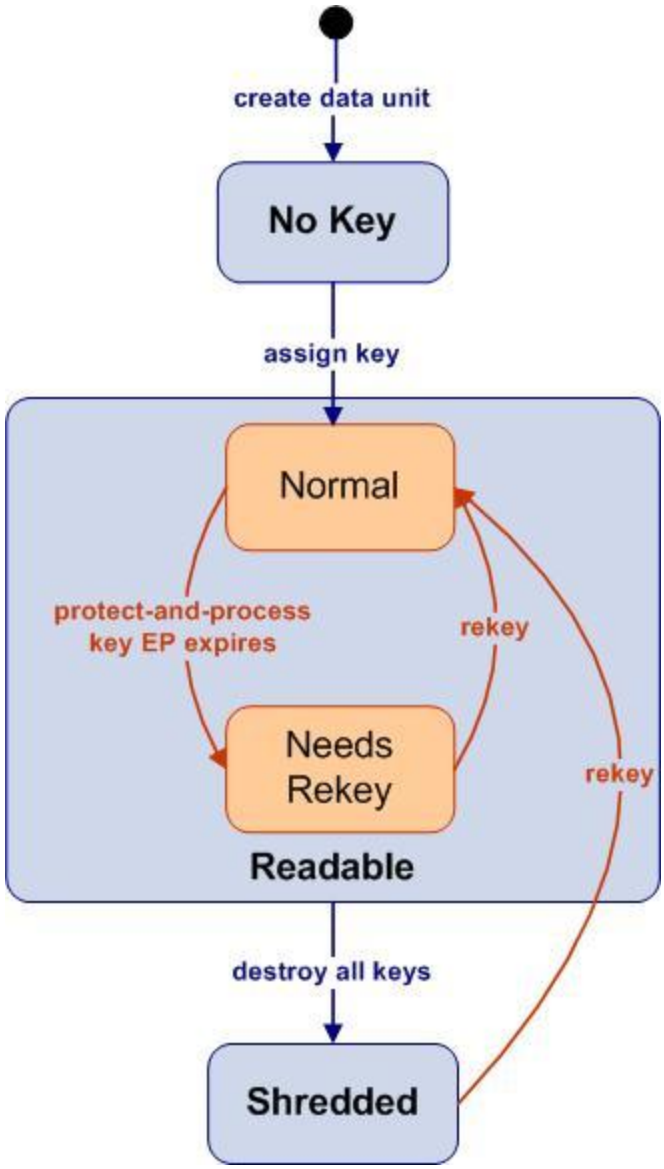
When the data unit is initially created, it is in **No Key** state (after the agent's request to create a data unit has completed and before its request to create a key has completed). As soon as a key is created for the data unit, the data unit moves to **Normal** state. In this state, encrypted data can be written to the data unit. The data unit can be read, and new data can be written to the data unit. New data can either overwrite existing data or can be written to a location that does not contain data. The details of this behavior are unknown to the KMS and are determined by the agent.

When the encryption period for the data unit's protect-and-process key expires, the data unit moves to the **Needs Rekey** state.

> NOTE: This transition may occur while data is being written to the data unit using the key whose encryption period has expired. The agent will continue to write data to the data unit using the expired key until the application managing the operation requests a dismount of the tape volume.

When a tape volume is mounted, the agent requests all keys associated with the data unit. If the data unit is in Needs Rekey state, none of the keys transferred to the drive is in protect-and-process state. The agent then requests the KMA to create a new key for the data unit, and the data unit returns to Normal state. When the data unit is full or no longer in use, it moves back to the Needs Rekey state upon expiration of its protect-and-process key and typically remains in that state. However, if all of the keys associated with the data unit are destroyed, it moves to the **Shredded** state. In that state, no data can be read from it, but new data may be written from BOT using a new key, returning it to the Normal state.

Data unit states and allowed transitions for a data unit with a single key are shown in the following diagram:



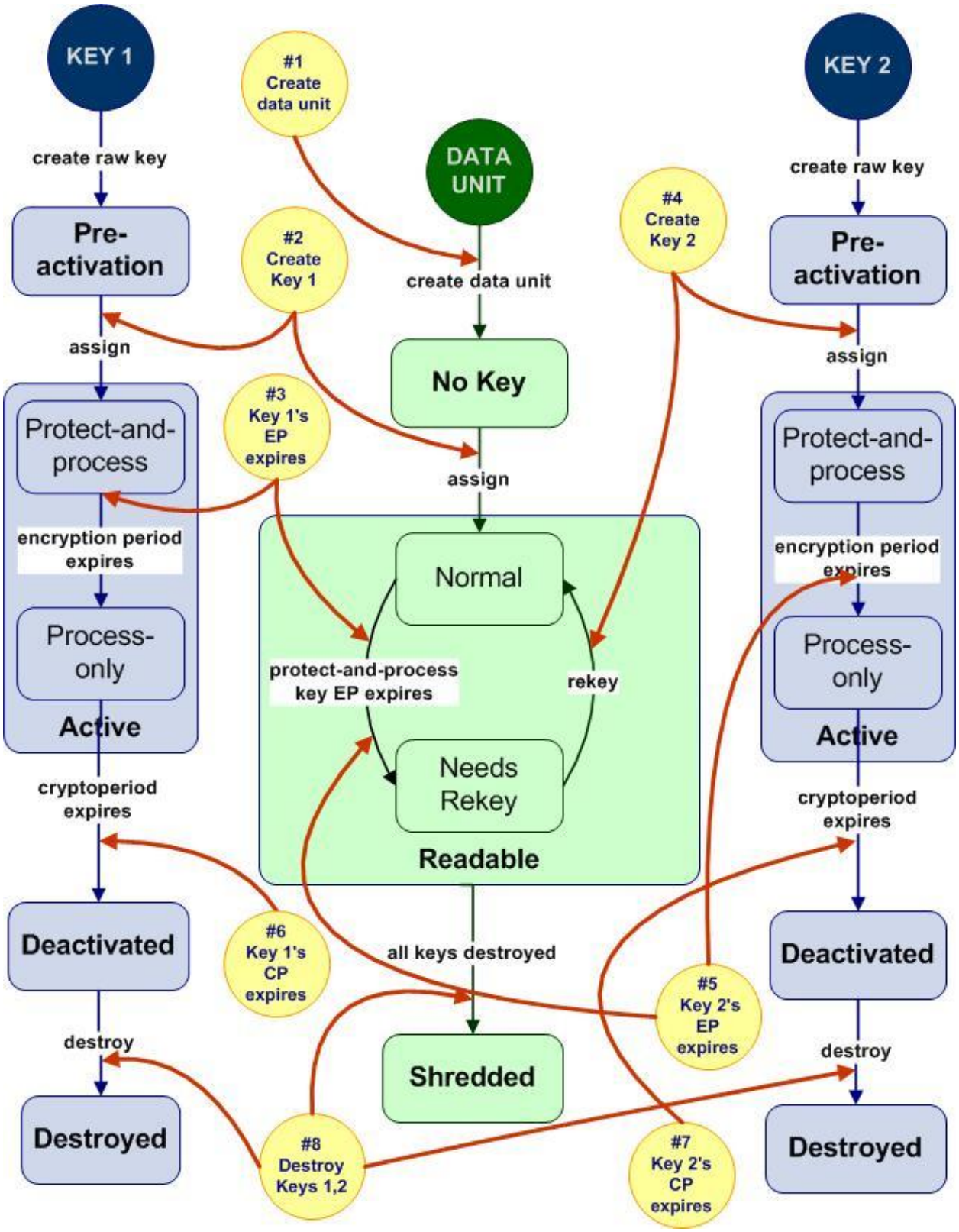
Consider the following sequence of events:

1. New media is mounted into a KMS 2.0 encrypting drive as part of a backup operation. The agent requests the KMA to create a new data unit.
2. The agent requests the KMA to create a key (Key 1) and encrypts and writes the backup data. The data unit moves to **Normal** state.
3. Sometime later, Key 1's encryption period expires, causing the data unit to change to **Needs Rekey** state.
4. The same media is again mounted for another backup operation. The agent requests the KMA to create a key (Key 2) and encrypts and writes the backup data. The data unit once again moves to **Normal** state.

5. Sometime later, Key 2's encryption period expires, causing the data unit to return to **Needs Rekey** state.
6. Eventually, Key 1's cryptoperiod expires, leaving it deactivated.
7. Likewise, Key 2's cryptoperiod expires, leaving it deactivated.
8. The operator destroys Key 1 and Key 2, causing the data unit to move to the **Shredded** state.

The diagram on the next page provides a visual representation of the states of the data unit and two keys involved in this sequence. The diagram consists of the following parts:

- State transitions for the data unit are shown in the center of the diagram.
- State transitions for the two keys are pictured on either side. (The key state diagrams are simplified for this example and do not contain all possible states.)
- Yellow circles correspond to events in the sequence.
- Orange arrows indicate the key and data unit state transitions triggered by the yellow events.


KEY 1
create raw key
Pre-activation
assign
Protect-and-process
encryption period expires
Process-only
Active
cryptoperiod expires
Deactivated
destroy
Destroyed
#1 Create data unit
DATA UNIT
create data unit
#2 Create Key 1
No Key
assign
#3 Key 1's EP expires
Normal
protect-and-process key EP expires
rekey
Needs Rekey
Readable
all keys destroyed
Shredded
#6 Key 1's CP expires
#8 Destroy Keys 1,2
KEY 2
create raw key
Pre-activation
assign
Protect-and-process
encryption period expires
Process-only
Active
cryptoperiod expires
Deactivated
destroy
Destroyed
#4 Create Key 2
#5 Key 2's EP expires
#7 Key 2's CP expires

Recycling/Scratching Data Units

A tape volume may be recycled or set for scratch when all of the data on it is no longer needed. When this happens, new data is written from BOT, overwriting existing data on the media. Data that is overwritten is lost, and data not yet overwritten can be recovered only using special recovery tools. Therefore, the keys previously associated with the data unit should not be needed again. To reduce the number of keys that must be transmitted to the agent when the tape volume is mounted again, when the agent receives a request to write the media from BOT it disassociates all keys previously associated with the data unit that are not in protect-and-process state. (The agent will reuse an existing protect-and-process key to write new data until its encryption period expires.)

The term “disassociates” must not be taken literally. Actually, a dissociated key continues to appear in the key list for the data unit, but two changes occur as a result of this action:

1. The key’s In Use By Data Unit attribute changes from True to False.
2. The key is no longer transmitted to the agent when it requests keys for this data unit.

If a key is imported from a KMS 1.x system, it may be associated with multiple data units. The disassociation operation applies only to the data unit currently mounted.

Chapter 3: Security Features

An in-depth discussion of the various levels of security implemented in this solution is outside the scope of this paper. The sections below address only a few of these features.

Secure Communication

The communication protocol between a drive agent and a KMA, between a new KMA added to the cluster and an existing KMA and between a user and a KMA is all the same. In each case, the system uses the passphrase for the entity initiating the communication to perform a challenge/response protocol. If successful, the entity is provided with a certificate and its corresponding private key. This certificate and private key are used to establish a TLS 1.0 (secure sockets) channel. Establishing this secure sockets channel is done using 2048-bit RSA. Establishing this session results in the endpoints agreeing on an AES 256-bit key. All subsequent communications are encrypted with this AES-256 key. Mutual authentication is performed; each end of any connection authenticates the other party.

For a drive, the authentication process of an encryption agent is performed during the VOP enrollment session. For a KMA, authentication is part of the Quickstart process described in the Key Management System Version 2.0 Installation and Service Manual referenced in the Related Publications section. For either of these, the certificate and private key are retained and used to re-establish the session after the drive or KMA reboots. For KMS users, the process is repeated every time the user logs in, since we assume users may log in from different workstations. All the later communications—an agent requesting a key, one KMA sending updates to another or the KMS Manager making requests to a KMA—are done using the already-established secure sockets session.

Key Replication

When the first KMA of the cluster is initialized, a large pool of raw keys is generated. When another KMA is added to the cluster, the raw keys are replicated to the new KMA and are then ready to be used to encrypt data. When a data unit without a protect-and-process key is mounted in a drive, the drive agent contacts a KMA in the cluster and requests a new key. A raw key is drawn from this pool, assigned to the agent's default key group and to the data unit associated with the tape volume mounted in the drive. The database updates from this transaction are then replicated across the network to all KMAs in the cluster. **At no time is any clear-text key material transmitted across the network.**

Role-based Software Access

Access to the Key Management System is limited to users who have been trusted with permission to perform certain roles. The following user roles are defined in the system:

- Security Officer—manages security settings, users, sites and transfer partners
- Compliance Officer—manages key policies and key groups and assigns key groups to agents and transfer partners
- Operator—manages agents, data units and keys
- Backup Operator—performs KMS database backup and restore operations
- Auditor—views information about the KMS cluster

A single user account may be given permission to perform multiple roles, and multiple user accounts may be given permission to perform the same role. Some operations, such as sharing keys between trusted partners, may be multi-step tasks that involve more than one user role. For maximum security, each role should be performed by a different user. For maximum convenience, one “super-user” may be assigned all roles. A compromise that divides the roles among two or three users may provide an acceptable level of security and increased flexibility.

Quorum Protection

Some operations are deemed critical enough to require an additional level of security. These operations include adding a KMA to a cluster, unlocking a KMA, restoring a KMS from backup and configuring key transfer partners. To implement this security, the system uses a set of key split credentials in addition to the role-based access described above.

Key split credentials consist of a set of userid/passphrase pairs, together with the minimum number of these pairs that must be provided to the system to allow completion of certain operations. The key split credentials are also referred to as “the quorum” and the minimum number as “the quorum threshold”; operations that require the key split credentials to be provided are referred to as “quorum operations”.

The KMS system allows a maximum of 10 userid/passphrase pairs. The quorum threshold can be set anywhere from one to the number of userid/passphrase pairs defined. Setting it to one allows the completion of a quorum operation with only one quorum member present. Setting it to the total number of pairs defined requires that all quorum members be present for these operations. The most common case would be to set this to a value greater than one, but less than the total, e.g., three of five. This choice ensures that completion of these operations requires more than a single (possibly rogue) quorum member but allows for the situation where a member is unavailable.

NOTE: The userid/passphrase pairs defined for the quorum are unrelated to the user roles defined above. Quorum members will normally not have access to the KMS system. Their sole function will be to validate critical security operations performed by a KMS user with the appropriate role assignment.

Chapter 4: Key Management

Key Policies and Groups

A drive agent requests the KMA to create keys in its default key group as needed for data encryption. The key policy associated with this key group defines the lifecycle of these keys as well as whether keys can be exported from or imported into the group. As mentioned earlier, for open-systems environments we recommend that the key policy be defined so that the encryption period is sufficiently long to allow all data on one data unit to be encrypted with the same key. This reduces the total number of keys in the system and allows all of the data on one data unit to be managed as a whole, rather than on an individual file basis.

A good rule of thumb is to set the encryption period to at least three months and longer if small amounts of data will be written at intervals until the tape is full. For example, a full year of bi-weekly payroll data might be stored on a single tape volume. To ensure that all data on this data unit is written with the same key would require a key policy with a much longer encryption period.

Most open-systems backup applications and some MVS mainframe applications will fill one data unit before writing to another, making the time required to fill a data unit typically fairly short. However, choosing a longer than needed encryption period is not a problem. Even if the encryption period is longer than the time required to fill a data unit, the key will not be reused, effectively terminating its encryption period when the tape is full. However, choosing a shorter than recommended encryption period may result in an agent rekeying a data unit several times before it's full. This will result in large numbers of keys being created, degrading KMS performance.

The VSM application of VTCS/VTSS and other MVS mainframe applications will not necessarily fill a tape data unit before writing to another data unit. Because of this, it makes sense to maintain longer encryption periods for the data units so the number of keys created for each data unit is kept to a minimum. The data written to MVC's by VSM in the form of a Virtual Tape Volume is compressed by VTSS then encrypted by the encryption agent.

Multiple key groups may use the same key policy. However, for many customers, a single key group per KMS cluster should suffice. This guarantees that any data unit known to the KMS can be read by any drive enrolled in the cluster. If data is to be shared with another KMS within the enterprise or with another enterprise partner, a new key group with the Export From attribute set could be created. Keys to be shared would be moved to this new key group, which would be assigned to the key transfer partner. (Chapter 5 discusses the partner key transfer operation in much more detail.)

KMS Clusters

KMS 2.0 allows users the flexibility to structure their key management systems to provide secure data protection while ensuring access to authorized users and ease of management. Clustering KMAs allows for replication of database entries and workload balancing. An enterprise may have encrypting drives in different locations within the same site and/or at multiple, geographically separated sites. Adherence to the following guidelines will ensure efficient and reliable performance:

- To reduce network latency, the KMAs servicing a pool of drives should be connected to the same network subnet as the drives.
- To spread the workload and allow for KMA outages (software upgrade, hardware failures, etc.), each pool of drives should have access to at least two KMAs.
- To prevent catastrophic loss of key data, KMAs located at two (or more) geographically separated sites should be clustered. This ensures business continuance in the event of a disaster that disables an entire site.

We assume that critical data generated at any site is replicated and vaulted off-site. If a site is lost, this backup data may be transferred to another operational site. Data units and keys associated with these tape volumes will be known to the KMAs at the sister site, and encrypted data required to continue business operations will be available. The damaged portion of the cluster can be restored easily at the same or a different location once site operations resume. Chapter 6 describes how this restoration is accomplished.

Chapter 5: Partner Key Transfer

KMAs at sites within an enterprise that routinely share data may be clustered so that all keys in use at any site are automatically available to all sites. In that situation, only tape media must be transported between sites to allow the desired sharing. However, the need may arise to share encrypted data between KMS clusters within the same enterprise or between enterprise partners. KMS 2.0 provides a secure mechanism by which such key sharing may be accomplished.

The key sharing mechanism is based on each partner having a public/private key pair generated by the KMS at its site. A transfer key file containing a list of data units and keys to be shared is generated at the sender's site. This file is encrypted using the receiver's public key and signed with the sender's private key to ensure that only the receiver may access the shared keys and to allow the receiver to verify that the expected sender is the source.

The process is described below:

- A Security Officer at each site obtains the public key generated by the KMS, or creates a new public/private key pair, using the KMS Manager. (Only the public key information is displayed.) The partners exchange public keys. For each key, a fingerprint is generated at the sending site and computed at the receiving site. This fingerprint should be compared across sites to verify that no corruption has occurred during transfer of the key. For security reasons, this fingerprint is not sent along with the public key but verbally verified later.
- At each site, a Security Officer creates a key transfer partner through the KMS Manager interface, supplying the public key received from the partner site. Creating a key transfer partner is a quorum operation, so the threshold number of quorum members must be present to validate this operation. When the partner's key is input, a fingerprint for that key is computed by the KMS. If no corruption has occurred in transit, this fingerprint will match the one that was generated at the partner's site at the time the public key was created.
- Each site assigns to its key transfer partner one or more key groups from which keys may be exported or to which keys may be imported. (Key transfer can be two-way, so a site may be both an import site and an export site.) These key groups must be associated with key policies that allow the desired export or import operation. (This attribute of a key policy is modifiable if it was not set when the key policy was created.)
- An operator at the sending site creates one or more key transfer files containing data units and keys to be shared. A key may be exported only if it satisfies the following conditions:

    1. It belongs to a key group associated with the transfer partner.
    2. It is active (in Protect-and-process or Process-only state), deactivated or compromised, with its In Use By Data Unit attribute set to True.

    Each file is created by selecting data units that are to be shared. Only data units that have associated keys that meet the stated conditions may be included, and only keys

associated with those data units that meet the stated conditions will be included in the transfer file.

NOTE: To simplify the sharing process, write data to be shared to new or recycled tape volumes, change the key group for the keys associated with each of these data units to the export key group and export keys for all of these data units. This process will ensure that all data intended to be shared will be accessible at the receiving site and that only data intended to be shared is sent to the transfer partner.

- The sender's KMS encrypts the transfer key file using the receiving site's public key and signs the file using the sender's private key.
- The sender transfers the key file and the associated data tapes to the receiver.
- The receiver enters the data tapes into the library and imports the tape volumes into the backup application.
- The receiver loads the transfer key file onto the management station and imports the keys (and associated data units) from the file, specifying a key group assigned to the key partner (sender) as the destination key group. (Importing of keys does not require quorum validation.)
- The receiver's KMA decrypts the transfer file and verifies the sender as its source. The KMA creates entries in its database for the shared keys and data units and associates each new key with the appropriate data unit.
- The keys and data units imported are replicated to all KMAs in the receiver's cluster. (This process may take some time if a large number of items are shared. Reading tapes that require the imported keys may fail until sufficient time has elapsed for replication to complete.)

NOTE: Data unit and key IDs contain the ID of the creating KMS. Therefore, data units and keys imported into the receiver's KMS will have IDs distinct from those created by the receiver's KMS.

Chapter 6: KMA Recovery

KMS 2.0 allows for two types of recovery: 1) recovery of individual KMAs, and 2) recovery of the entire KMS cluster.

Recovery of a single KMA can be accomplished with no impact to the rest of the cluster as long as one KMA for each drive pool remains operational. Furthermore, recovery of N-1 nodes of an N-node cluster can be accomplished without loss of any critical data.

The following sections address scenarios that require recovery of a single KMA. Recovery of an entire cluster is addressed in Chapter 8.

Software Upgrade

Software upgrades may be done without interrupting KMA service to active encryption agents. Download of the new software may be done concurrently on all KMAs in the cluster. Activation of the new software requires a reboot of the KMA server. Therefore, activations must be staggered so that at least one KMA connected to each drive pool is active at all times. As each KMA comes online, database updates completed while it was offline are replicated to it so that all KMAs in the cluster are re-synchronized.

Network Disconnect

When a KMA is disconnected from the management network, the remaining KMAs in the cluster continue to attempt to contact it and will report communication errors in the audit event log. (The server reboot required to activate new software, as described above, produces this same behavior.) Drives in its drive pool communicate without interruption with the remaining operational KMAs attached to their service network. When the KMA is reconnected to the network, its database is updated with all updates made during its absence.

Hardware Failure

The KMA server is a single field-replaceable unit. The entire unit must be replaced if any component of a KMA server fails—NIC failure, database corruption, hard disk crash, etc. First, the KMA should be deleted from the cluster so that the remaining KMAs will no longer attempt to communicate with it. If the KMA console is still accessible, the option to reset the KMA may be executed. The reset operation will return the unit to its factory defaults. This operation offers the option to scrub the server's hard disk as an extra security precaution. Disposition of the failed server is handled by the customer.

NOTE: The reset operation may also be used prior to moving a KMA from one cluster to another. When a KMA has been reset, it will appear exactly like a new KMA fresh from the factory except that the ELOM network configuration persists. These network parameters can be reset during the Quickstart process. The optional disk scrub may be used to remove all traces of its former cluster.

A replacement KMA server is configured and added to the cluster as described in the Key Management System Version 2.0 Installation and Service Manual referenced in the Related Publications section. Once the new server is known to the cluster, the KMS database is replicated to it, and it becomes an active member of the cluster.

Chapter 7: KMS Backup

Backing up the KMS involves two types of backups: 1) core security backup, and 2) database backup. The following sections describe the purpose of each and when each type of backup should be taken.

Core Security Backup

The core security backup backs up the system master key and quorum information and is required if a system is to be restored from a database backup. This backup should be taken immediately after a KMS cluster has been configured and again whenever a change is made to the quorum.

As described in Chapter 3, the passphrases of the quorum members are used to protect the system master key. The master key is split into N pieces, where N is the size of the quorum, in such a way that the entire key can be reconstructed from M pieces, where M is the quorum threshold. The passphrase of each quorum member is used to generate a key that is used to encrypt one of the N pieces of the master key. When the userids and passphrases of any M of the N quorum members are provided, the system can reconstruct the master key. This master key is required to complete critical security operations.

The core security backup file contains the system master key, split and wrapped as described above. The file is an XML file in human-readable form. The userids of the quorum members are in plain text, which can be useful to verify the identities of the quorum members. Key and passphrase information in the file is securely encrypted. In particular, the master key is well-protected. Reconstruction of the master key would require access to the algorithm used to split the master key, the userids and passphrases of the required subset of quorum members, the algorithm used to generate keys from these passwords and the algorithm used to encrypt the key splits with these keys. Even so, **each copy of the core security backup file should be kept in a very secure place.**

The quorum information should be modified in the event a quorum member is added or removed or a member's passphrase is compromised. A new core security backup should be taken after the quorum information is modified, and all copies of the previous core security backup file should be completely destroyed.

The core security backup operation is done using KMS Manager. The file name for the core security backup file generated by this operation is stored in the database. However, the core security backup file itself is stored on the KMS management station or some remote location accessible from it, not on any KMA server, and managed entirely outside the KMS system.

To provide the required level of protection, the core security backup file might be placed on a thumb drive and given to a highly trusted employee. That employee would keep the file in his possession at all times. When a quorum change is necessary, the thumb drive containing the first file would be crushed to prevent any possibility of future access. A new core security backup would be taken, placed on a new thumb drive and protected with the same level of security.

Database Backup

In most business-critical systems, the security of the database is protected by regular backups with journaled updates that can be applied to ensure that the most recent changes are included if a restore from backup is required. Backing up these systems is critical because only a single instance of the database may exist.

The security of the KMS 2.0 database is provided by its replication across multiple clustered KMAs located at geographically separated sites. **Restoration of the KMS database from backup will be necessary only if every KMA at every site is destroyed at the same time.** KMS Manager provides a database backup facility to create a point-in-time copy of the KMS database for use in such a circumstance.

Backing up the KMS database is done by a user with Backup Operator credentials who is connected to a single KMA in the cluster. The backup operation produces a copy of the instance of the KMS database residing on the KMA to which the user is connected. The operation is non-disruptive as long as every agent has access to at least two KMAs. Other KMAs in the cluster provide uninterrupted service while the backup operation is in progress. The instance of the database on the KMA creating the backup is re-synchronized after the operation is complete.

Backing up the KMS database produces two files: 1) the backup file, and 2) the backup key file. The backup file is a copy of the KMS database encrypted using the key stored in the backup key file. The backup key in the key file is encrypted using the system master key, which is contained in the core security backup file. Therefore, the core security backup file, the backup key file and the presence of the required subset of quorum members are all required to restore a KMS system from a backup file. The passphrases of the quorum members are needed to decrypt the system master key, which is needed to decrypt the backup key, which in turn is needed to decrypt the backup file.

The file names for the backup files produced are stored in the KMS database. However, the backup files themselves are stored on the KMS management station or some remote location accessible to it, not on any KMA server. These files are totally outside the control of the KMS system and should be managed carefully. Backup files should be backed up and stored off-site, away from the cluster and separately from the core security backup file needed to unlock them. The system currently provides no utility to set up automated backup schedules, but such a utility will be available soon.

The KMS database is not journaled, so updates done by another KMA and not yet replicated to the KMA executing the backup or made after a backup is taken will not be included if that backup is used as the source for a cluster restore. However, the restored system may be able to recover key and data unit information not included in the backup.

Example 4

The creation of a key and a new data unit occurs on one KMA at the same time that a database backup is initiated on another KMA in the cluster. The backup operation captures the state of the database on the KMA executing the backup and does not include the new key or the new data unit. Sometime later a disaster causes the loss of the entire KMS cluster, and the cluster is restored using this backup. A request is made to the new cluster to restore the data written at the time the backup was in progress. The drive reads the data unit ID from the media and requests all keys associated with that data unit. The KMS has no record of that data unit, so the drive requests the specific key associated with the requested data. The KMS has no record of this key being assigned. However, it does have a list of unassigned keys generated in the original cluster before the backup was taken. If it finds the ID of the required key in this list, it assumes that this is a recovery operation, creates a new data unit, associates the key with the new data unit and provides the key to the drive.

Some scenarios exist in which an assigned key may not be recoverable from a backup copy of the database. Nevertheless, the existence of multiple active copies of the database at geographically separated locations provides an extremely high level of security. In addition, no key material is ever assigned until it has been replicated across the cluster. Therefore, the system ensures access to critical data with a very high level of confidence.

WARNING: No restore from backup can be done without a core security file, but any core security backup file can be used to restore a cluster from backup as long as the requisite threshold of quorum userid/passphrase pairs stored in that file is provided. Therefore, copies of outdated core security backup files should be destroyed, and copies of the current core security backup file should be stored securely in different locations from each other and from the database backup files.

Chapter 8: Disaster Recovery

A KMS 2.0 clustered environment that spans multiple, geographically separated sites dramatically reduces the risk of a disaster destroying the entire cluster. In the unlikely event that an entire cluster must be recreated, most key data can be recovered by recreating the KMS 2.0 environment from a recent database backup.

Many companies employ the services of a third-party disaster recovery (DR) site to allow them to restart their business operations as quickly as possible. Periodic unannounced DR tests demonstrate the company's degree of preparedness to recover from a cataclysmic event. A number of possible scenarios exist, three of which are discussed here.

Scenario 1: The company maintains a KMA that is part of its KMS cluster at the DR test site. The test site provides a pool of drives with network access to the KMA.

Scenario 2: All of the company's KMAs are destroyed. The DR test site provides a pool of drives with network access to a standalone KMA that is used at different times by different client companies.

Scenario 3: The company maintains a KMA that is part of its KMS cluster at DR Site 1, but the DR test is run at Site 2, which has a standalone KMA attached to a pool of drives as described in Scenario 2.

Scenario 1

This scenario is optimal; the company's KMS database is intact even though all of their sites have been destroyed. The primary focus is restoration of the data center and critical business systems. Move vaulted tapes to the DR test site. Connect a laptop running KMS Manager to the KMA network, and enroll the drives provided by the DR site in the company's KMA. A user with Operator credentials (but no quorum) is required. Connect a host running the required backup application to the drives at the DR site and use it to restore the company's data center.

Since the KMS database is intact, restoration of the company's KMS cluster is simple. The KMA at the DR site serves as the first KMA in the cluster. Remove the agents for the drives at the DR site. Replace the destroyed KMAs and add them, one at a time, to the cluster as described in the Hardware Failure section of Chapter 6. The database is replicated across the cluster. Create and enroll agents as replacement drives are brought into service.

Scenario 2

This scenario requires restoration of the company's KMS from a backup file. The process for restoring the cluster is described below.

The DR test site provides:

- A KMA, either fresh from the factory or reset to factory defaults by the previous client;

NOTE: Resetting the KMA is a console operation that requires the Security Officer login. Unless a new KMA is provided each time, the DR site must ensure that each client resets the KMA using his Security Officer login before he leaves the facility. This protects the client's data and allows the KMA to be reused by the next client. The scrub option on the reset should always be used to wipe all trace of the client's data from the server hard disk.

- A pool of drives attached to the KMA network;
- A management station on the KMA network running KMS Manager.

The company supplies:

- A trusted operator;
- Removable media containing a core security backup file, the most recent database backup file and the corresponding database backup key file;
- The required threshold number of quorum members whose userid/passphrase pairs are contained in the core security backup file;
- Backup tapes needed to restore the company's data center;
- A host running the backup application needed to restore data from the backup tapes.

The operator follows the steps prescribed in the Getting Started chapter of the Key Management System (KMS) Version 2.0 Administration Guide referenced in the Related Publications section, using the option to Restore Cluster from Backup. DR site personnel must provide relevant network information required by the Quickstart process.

Once the Quickstart procedure is completed, the operator configures the cluster as follows:

1. Load the removable media containing the core security backup file, the backup key file and the backup file into the management station. For security, do not load the backup files onto the management station.
2. Start KMS Manager and connect to the KMA using the Security Officer login created during the Quickstart process.
3. Navigate to Backup List → Restore, and supply the complete path name for each of the three backup files.
4. Allow each quorum member present to input his userid and passphrase when prompted by the software.
5. Click Start to initiate the restore.

NOTE: The time required to complete the restore operation may vary from an hour to several hours, depending on the size of the original cluster. The restore process is being reworked. A future release will contain improvements that reduce the restore time to a few minutes.

6. Once the restore is complete, remove the media containing the backup files from the management station.
7. Create a new user, e.g. KmsAdm, and assign to that user all defined roles. This creates a single user who can execute all of the operations required to complete the configuration of the KMA.
8. Disconnect from the KMA and reconnect using the new “superuser” login.
9. Create agents for the drives provided by the DR site and assign key groups to these agents as needed.
10. Provide the agent IDs and passphrases and the IP address of a KMA port on the drive network to a DR site operator whose job it is to enroll the drives in the KMA.

Once the drives are enrolled, the backup application may be configured to use these drives and the restoration of the company’s data center can be accomplished.

When the restoration is complete, log in to the KMA console and select the option to reset the KMA to factory defaults, choosing the scrub option to remove all traces of the company’s KMS database from the DR site KMA.

Scenario 3

This scenario is a hybrid of the previous two scenarios. A copy of the company’s KMS database exists at DR Site 1, but the DR test is being run at Site 2, which provides the same environment as the DR test site in Scenario 2. Several options exist for handling this scenario, depending in part on what network connections exist between the two DR sites.

Option 1: Connect the drives at Site 2 directly to the management network of the KMA at Site 1.

Option 2: Add the KMA at Site 2 to the company’s (single-node) cluster at Site 1.

Option 3: Back up the KMA at Site 1 and restore it to the KMA at Site 2.

Option 4: Transfer keys from the KMA at Site 1 to the KMA at Site 2.

Option 1

This option is simplest and quickest, if the drive network at Site 2 can be connected to a WAN that connects to the company’s KMA at Site 1. Create agents for the drives at Site 2 in the KMA at Site 1 and assign key groups as needed. (Creating the agents requires Operator credentials; assigning the key groups requires Compliance Officer credentials.) Provide the agent IDs and passphrases and the IP address of the management port of the KMA at Site 1 to a DR site operator at Site 2 whose job it is to enroll the drives in the KMA. Once the enrollment is complete, the drives at Site 2 can be used to process the company’s encrypted data. Configure

a host with the required backup application to use the drives at Site 2 and proceed with the restoration of the company's data center. This option allows the restoration of the company's data center from Site 2, and the required keys never go into the DR site KMA.

Option 2

This option requires the following resources at each site.

Site 1

- A WAN connection between the KMAs at Site 1 and Site 2
- Security Officer
- A threshold number of quorum members

Site 2

- A KMA, either fresh from the factory or reset to factory defaults
- A pool of drives attached to the KMA service network

At Site 1, the Security Officer executes the following steps:

1. Connect a laptop to the KMA network and start KMS Manager.
2. Create a new KMA, specifying a KMA name and passphrase.
3. Use the ELOM interface on the KMA at Site 2 to execute the Quickstart procedure, selecting the option to Add a KMA to a Cluster. Supply the name and passphrase of the new KMA created in Step 2.
4. Allow each quorum member present to enter her userid and passphrase to complete the Quickstart procedure.
5. Connect to the new KMA through KMS Manager and check the progress of the replication of the KMS database to the new KMA.
6. When the replication is complete, execute the Lock/Unlock KMA function.
7. Click on Unlock and allow each quorum member present to enter his userid and passphrase to complete the unlock operation.
8. Create a new user, e.g. KmsAdm, and assign to that user all defined roles. This creates a single user who can execute all of the operations required to complete the configuration of the KMA.
9. Disconnect from the KMA and reconnect using the new "superuser" login.
10. Create agents for the drives at Site 2 and assign key groups to these agents as needed.
11. Provide the agent IDs and passphrases to a DR site operator at Site 2 whose job it is to enroll the drives in the KMA.

Once the drives are enrolled, the drives at Site 2 can be used to process the company's encrypted data. Configure a host running the required backup application to use these drives, and proceed with the restoration of the data center.

When the restoration is complete, delete the KMA at Site 2 from the cluster and use the ELOM interface to the KMA at Site 2 to reset the KMA to factory defaults, choosing the scrub option to remove all traces of the company's KMS database from the DR site KMA. The "superuser" created in Step 8 may be deleted as well.

Option 3

This option requires creating a backup of the company's KMS database at Site 1 and using that backup to create a copy of the company's cluster on the KMA at Site 2. Backup Operator and Security Officer credentials are required at Site 1 to create the database backup and a core security backup required to unlock the backup key file at Site 2. Site 2 requirements are the same as those for the DR test site in Scenario 2, including the required number of quorum members to complete the restore operation.

Once the backup files are created at Site 1, the process defined in Scenario 2 is used to complete the restoration of the company's data center. The restore from backup operation is slow, but effective if no quicker option is available.

Option 4

This option entails setting up a key transfer partner at each site. It is attractive because importing keys is very quick, much quicker than restoring keys from a backup file. The process to follow is described below.

Site 2 must have a fully functioning standalone KMA with a pool of drives enrolled. This configuration may be done ahead of time or as part of the DR test. Setting up the KMA at Site 2 is done by a trusted operator of the company performing the DR test. Site 1 operations require Security Officer credentials.

The steps to be taken at each site are listed below.

Site 2

1. Use the ELOM interface on the KMA at Site 2 to complete the Quickstart procedure, using the Install First KMA in Cluster option.
2. Use KMS Manager to connect to the KMA using the newly created Security Officer login.
3. Create a key policy with Import Allowed and a key group that uses this key policy.
4. Obtain the public key information from the KMA at Site 1.
5. Create a key transfer partner using this public key information and verify that the key fingerprint matches that at Site 1.

6. Create agents for the drives at Site 2 and assign the import key group to them.
7. Provide the agent IDs and passphrases to a DR site operator at Site 2 whose job it is to enroll the drives in the KMA.

Once the drives are enrolled, a host running the required backup application may be configured to use these drives.

Site 1

1. Obtain the public key information from the KMA at Site 2.
2. Create a key transfer partner using this public key information and verify that the key fingerprint matches that at Site 2.
3. If necessary, modify all key policies to allow Export From.
4. Create a single “superuser” as described in Step 7 of Scenario 2.
5. Disconnect from the KMA and reconnect using the “superuser” login.
6. Assign all key groups to the transfer partner.
7. Export keys for all data units to be used at Site 2.
8. Transfer the export file containing the data units and keys to be shared to Site 2.
9. Delete the key transfer partner and “superuser” and undo the key policy changes, if desired.

Site 2

1. Load the export file received from Site 1 onto the management station.
2. Import the data units and keys from the export file into the KMA database.
3. Verify that a data unit exists in the KMA database for each backup tape volume required to restore the company’s data center.
4. Remove the export file from the management station.

Restoration of the company’s data center can proceed from this point at Site 2. Once the restoration is complete, log in to the console of the KMA at Site 2 and select the option to reset the KMA to factory defaults, using the scrub option to remove all traces of the company’s private information.

Chapter 9: Migration from KMS 1.x to KMS 2.0

The migration path from KMS 1.x to KMS 2.0 is simple and straightforward. The following sections outline the steps required to accomplish the task.

Preparation on KMS 1.x System

1. If the KMS 1.x system is running version 1.0 or 1.1 software, upgrade it to version 1.2. Only keys contained in an export key file created using version 1.2 can be imported into KMS 2.0. (Refer to KMS 1.2 product documentation for the process for upgrading to version 1.2.)
2. Log in to the 1.2 KMS as Administrator.
3. Navigate to Keys → Media Key Export, select the keys to be exported, enter a name for the export file and click Apply. An export file with the specified name will be created in the /export/home/kms/mnt_keys directory on the KMS 1.2 server. This file will have the following format:

   <Key ID>, <Key Value>[, <Description>] where

   Key ID is a 64-character (hexadecimal) value that uniquely identifies each key;

   Key Value is a 64-character (hexadecimal) value that is the cipher value of the key **(not encrypted)**;

   Description is an optional word or sentence used to describe each key.

   The Key ID and Description fields will be the same as those displayed in the KMS Keys → Media Key Export display.

WARNING: The key values in this file are not encrypted. This file should be stored in a highly secure location and completely destroyed once the key import into the KMS 2.0 database is complete.

Preparation on KMS 2.0 System

1. Install and configure a KMS 2.0 cluster.

   Using KMS Manager, create a key policy with the desired cryptoperiod for the KMS 1.2 keys, an encryption period less than or equal to the cryptoperiod and the Allow Import attribute enabled. An existing key policy with an appropriate cryptoperiod that enables import may also be used.

NOTE: A KMS 1.2 key is placed in the Process-only state upon import, and its encryption period end date is set to current date and time, making the encryption period set in the key policy irrelevant. Its Created Date and Activation Date are both set to current date and time when the import is executed, so the cryptoperiod should be chosen accordingly.

2. Create a key group and associate it with the key policy created in Step 1. Again an existing key group with the appropriate attributes may be used.
3. Create an agent for each KMS 1.x drive that is to be enrolled in the KMS 2.0 environment, recording the agent ID and passphrase input for each agent. Each agent in the agent list should have Enrolled = False.
4. Assign an appropriate default key group to each new agent.
5. Assign the destination key group for KMS 1.x keys to all agents.
6. Transfer the KMS 1.2 key export file to the KMS 2.0 management station.

Key Import

1. Log in to KMS Manager using the Compliance Officer login.
2. Select the function to Import 1.0 Keys, input the destination key group and the path name for the KMS 1.0 key export file and click Start.

   The KMS system executes the following steps:

   - The entire file is read and each line checked to ensure that the Key ID and Key Value are the appropriate length and format. The first 4 characters of the Key ID are stripped off. (KMS 2.0 Key IDs have length 30 bytes rather than the 32 byte length in KMS 1.2.) In addition, the Key ID is checked against the KMS 2.0 KMS database to ensure it is unique. If the Key ID is not unique, the Key Value is checked against the KMS 2.0 KMS Keystore for that Key ID.
   - If a key exists in the KMS 2.0 KMS database with the same Key ID and Key Value, that Key ID is noted and processing continues. When the key import is completed, the number of duplicate keys is returned.
   - If a key exists in the KMS 2.0 KMS database with the same Key ID but a different Key Value, then the operation is aborted immediately and an error is returned on the assumption that the KMS 1.2 KMS export file may be corrupt.
   - If no error occurs in the validation phase, each key is processed. Its Key Value is encrypted and added to the Keystore, and its metadata (Key ID, state, creation date, etc.) is added to the database. An error in the processing of any key causes the processing of keys to halt.

- If no error occurs in the processing phase, all keys are processed, and the total number of keys imported is reported by KMS Manager.
- If an error occurred in processing a key, the Key Value that was added to the Keystore is removed, the transaction to insert the metadata into the database is rolled back and an error message is displayed in KMS Manager.
- Imported key information is replicated across the network to all KMAs in the cluster.

3. Ensure that the drives that are to be enrolled in the KMS 2.0 environment have access to the service network of the KMS 2.0 KMS.
4. Rewrite all 1.x tokens on the KMS 2.0 service network to remove information for the drives being transferred.
5. Use the VOP interface to reset each drive that is to be enrolled in the KMS 2.0 environment. This will clear all KMS 1.x keys from its memory.
6. Place each drive offline and load the latest 2.0 drive firmware.
7. After each drive reboots, place it offline again, and select Configure → Drive Data → Encrypt. Enter the following information:

   - Agent ID and passphrase specified when the agent was created in the.
   - "Use token" = "No".
   - IP address of a KMA on the drive's network.

   (No license key is required for a drive previously enabled for KMS 1.x encryption.) The VOP log reports that the enrollment was successful, and the drive reboots again.
8. Verify that KMS Manager now reports Enrolled = True for each agent.

NOTE: A drive previously enabled for KMS 1.x encryption is permanently encrypting. Encryption may not be disabled on the drive.

Retrieving 1.x Encrypted Data

1. Import the tape volumes containing data written with the imported keys into whatever application environment will be used to retrieve data from them.
2. Through the application, issue a request to retrieve a file from one of the imported volumes.
3. When the tape volume is mounted in a 2.0 drive, the agent determines that the media is encrypted but has no data unit ID.

4. The agent requests the KMA to create a data unit (with the VOLSER provided by the library as the external tag).
5. The KMA returns the data unit, and the agent requests the Key ID for the specified file.
6. The KMA finds the requested Key ID, verifies that the agent has access to its key group, associates it with the new data unit and returns its Key Value to the agent.
7. The agent decrypts the specified file and returns the data to the requesting application.

Management of Imported Data

KMS 1.x keys will never be reused since their lifecycles start in the Process-only state. Otherwise, they are managed exactly as KMS 2.0 keys. Data may be appended to tape volumes encrypted with KMS 1.x keys. The drive agent will rekey the data unit just as it would a KMS 2.0 data unit without a protect-and-process key.

> NOTE: A tape volume imported from KMS 1.x to KMS 2.0 can no longer be processed in a KMS 1.x environment.

Data units imported from a KMS 1.x system differ from those encrypted with a KMS 2.0 system in one important way. Multiple such data units will likely contain data encrypted with the same key. Therefore, operations on one data unit may have side effects. For example, changing the key group for all keys associated with one data unit may have the ripple effect of changing the key group for some or all of the keys associated with other data units as well. Likewise, **destroying a key used to encrypt data on one data unit may result in loss of access to data on other data units.** Many KMS 1.x customers used a single encryption key across multiple sites for weeks or months at a time to simplify sharing of data between sites. As a result, a huge amount of critical data is protected with a single key. The inadvertent destruction of this key could result in a staggering data loss. Extreme caution must be exercised when destroying keys.

Furthermore, this ripple effect may not be obvious. Tape media originally encrypted in a KMS 1.x system will not have a data unit ID associated with it until it is first mounted in a KMS 2.0 drive. Therefore, the KMS may report that a given key is associated with only one data unit when, in fact, it was also used to encrypt data on one or more other tape volumes not yet known to the KMS.

Summary

KMS 2.0 is a comprehensive key management platform designed to address the rapidly growing enterprise commitment to storage-based data encryption. It provides

- Security
- Performance
- High capacity
- High availability
- Scalability
- Interoperability
- Central management
- Long-term key retention
- Ease-of-use
- Configurability
- Upgrade path from KMS 1.x

Customers employing KMS 2.0 can be confident that their data is secure from unauthorized access and at the same time highly available to authorized users.

Appendix – KMS 2.0 Operation with HP LTO4 Tape Drive

KMS 2.0 encryption operates somewhat differently when the agent is an HP LTO4 encryption-enabled tape drive. Listed below are some behavior differences:

- The HP LTO4 drive is encryption capable as shipped; no license key is needed to enable encryption.
- Enrollment of an HP LTO4 encryption capable drive into the KMS cluster is a multi-step process that requires the use of the VOP-LTO4 software designed specifically to manage the HP LTO4 drive. The first two steps are preparatory steps that are done through KMS Manager by a user with Operator privileges. The final two steps, the actual agent enrollment, are done using the VOP-LTO4 software.
    1. Create an agent for the drive in a KMA, specifying an agent ID and passphrase.
    2. Assign one or more key groups to the agent and designate one group as its default key group. (When an agent request results in the KMS creating a new write key, the key will be assigned to the agent's default key group.)
    3. Connect to the drive using the VOP-LTO4 software, and click on the Configure Drive tab. In the fields provided, enter the agent ID and passphrase specified in Step 1 and the IP address of a KMA port on the drive network, and click on Commit.
    4. Click on the Diagnose Drive tab, and verify from the log entries that the commit operation succeeded. Return to the Configure Drive tab, and click on Enroll. Again monitor the log entries on the Diagnose Drive tab to verify the success of the enrollment operation. The Encrypt button at the top of the VOP display should now be colored blue.
- The HP LTO4 drive stores at most one encryption key in its memory when it is configured to obtain keys from the Sun KMS. StorageTek drives can store up to 32 keys.
- The HP LTO4 drive does not pre-fetch keys associated with a tape volume when that volume is mounted. Instead, it reads the barcode label and a media identifier from the cartridge memory, sends this information to the KMS and waits for receipt of an I/O request.
- When the application issues a request to read encrypted data from the tape, the drive requests from the KMS the key needed to decrypt that data. As long as the tape volume remains mounted, subsequent requests to read data written with the same key will be completed with no further requests to the KMS.
- Write keys are handled differently. The HP LTO4 drive requests a key from the KMS on any write operation after the tape has been repositioned. In a Veritas NetBackup environment, this means that if multiple backup images are to be written to the same tape volume, the drive requests a key from the KMS at the start of each backup job. If the encryption key last used to write data to the tape volume is still in Protect and Process state, the KMS resends that key to the drive. Otherwise, the KMS creates a new key, associates it with the data unit and sends it to the drive.

  StorageTek drives use the protect-and-process key acquired when the tape volume is mounted to encrypt data as long as the tape volume is mounted. This reduces the key fetch overhead when writing multiple backup jobs to one tape volume.

- When an HP LTO4 drive receives a request to write from BOT, it does not send a disassociate keys request to the KMA. Therefore, all keys used to write data to a data unit continue to be reported by the KMS as In Use By Data Unit even though the data encrypted with some of those keys may have been overwritten.

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA Phone 1-650-960-1300 or 1-800-555-9SUN Web sun.com

Part Number: xxxx 06/30/2008