To: IEEE P1619.3 Task Group
From: P1619.3 Architecture Subcommittee, Members:

Mike Witkowski, CipherMax (acting chairman)
Jon Holdman, Sun
Larry Hofer, Emulex
Luther Martin, Voltage Security
Landon Noll, Independent
Glen Jaquette, IBM
Matt Ball, Quantum
Robert Lockhart, Independent
Robert Sussland, Decru
Kevin Marks, Dell
Subhash Sankuratripati, Decru
Gaurav Agarwal, Decru
Christopher Kusek, Network Appliance
Kevin Butt, IBM
Robert Griffin, RSA/EMC
Robert Snively, Brocade
Chris Williams, HP
Subbu Muddappa, Server Engines

Date: December 13, 2007
Purpose: Proposed changes against P1619.3/D1 to create a new clause 5 entitled “Key Management Models” to incorporate new key management related models and supporting material that provides a basis for the remaining sections being developed by the other subcommittees in P1619.3 Task Group.

Introduction

The P1619.3 Architecture Subcommittee has been working on a proposal to create a new Section 5, entitled “Key Management Models” to incorporate new architecture models and supporting identification and high level definitions and explanations for the model components, interfaces, and interactions. An important objective of this new model section is to clearly define what is and is not covered in the scope of this standard. In addition, this standard’s relationship to other industry, national, and international standards will be defined. This new section will be placed between Section 4 entitled “General Overview,” and Section 6 (previously Section 5) entitled “Key Management Operations”. This section will provide several models of key management entities, objects, operations, states, and interfaces that provide a framework for the more detailed description of these aspects of key management described in the following sections of this standard.

Changes to P1619.3/D1

The changes required by this proposal to draft P1619.3/D1 are: There will also be additional references for the Normative References clause, and definitions and acronyms for the Definitions, acronyms, and abbreviations clause.

1) Creation of Clause 5, entitled “Key Management Models”.
2) Addition of references to Normative References clause.
3) Addition and deletion of definitions, acronyms, and abbreviations for to the Definitions, Acronyms, and Abbreviations clause (Clause 3).
4) Addition of an informative annex that specifies the differences between the key management models proposed in the model clause of P1619.3 and the NIST SP 800-57 Part 1 standard and the IEC/ISO 11770-1 standard.

2. Normative References

Add reference to ASN X9.24. Others, TBD.

3. Definitions, acronyms, and abbreviations

The following definitions in P1619.3/D1 must be removed:

- Client

Any replicate definitions, acronyms, and abbreviations in the following subclauses indicate that they are to replace the corresponding definitions, acronyms, and abbreviations specified in P1619.3/D1. Any new, unique definitions, acronyms, and abbreviations should be added to the draft.

3.1 Definitions

3.1.1 Encryption Entity: An entity that is composed of a Cryptographic Unit and a Key Management Client.

3.1.2 Cryptographic Unit: An entity capable of using cryptographic algorithms to encrypt data being written to and decrypt data read from storage media. Cryptographic units implement the cryptographic algorithms specified in IEEE P1619, IEEE P1619.1, and IEEE P1619.2. Cryptographic units lie in both the data plane (where encryption and decryption actually take place) and the control plane (data unit information and encryption key and security parameters are transported between itself and the Key Management Client.

3.1.3 Key Management Client: TBD.

3.1.4 Key Management Server: TBD.

3.1.5 Key Management API: TBD

3.1.6 Key Management Software Library: TBD

3.1.7 Key Management Operations: TBD

3.1.8 Data Plane: In the context of encryption of stored data, data plane operations occur then encryption users write plaintext data through the cryptographic unit where the data is encrypted and deposited as ciphertext to the storage media. Likewise, data plane operations also occur when encryption users read plaintext data through the cryptographic unit which decrypts ciphertext data retrieved from the storage media. Data plane operations tend to be high performance operations that are decoupled from the control plane where key management operations take place.

3.1.9 Control Plane: TBD

3.1.10 Encryption Users: TBD

3.1.11 Storage Media: TBD

3.1.12 Key Management API: TBD

3.1.13 Key Management API: TBD

3.1.14 Key Management API: TBD

3.1.15 Key Management API: TBD

3.2 Acronyms and abbreviations

CU	Cryptographic Unit
KM	Key Management
KMC	Key Management Client
KMS	Key Management Server
KM API	Key Management Application Programming Interface
KM Ops	Key Management Operations
KM SW Lib	Key Management Software Library

4. General Overview

TBD

5. Key Management Models

This clause contains a number of abstract models that define the high level architecture for environments and systems that provide management for encryption keys and their related security attributes, parameters, and objects used for the protection of stored data. Industry and international standards define the operations and procedures for managing encryption keys and their related security material as “key management,” and thus the objective of this clause is to present a series of key management models that provide an introduction to, and a context for, the detailed key management operations, objects, and protocols presented throughout this remainder of this standard.

Because of the complex nature of key management, several models have been developed to provide various viewpoints into the architecture of a key management environment and describe the entities and objects that make up these environments. The key management models presented in this standard are as follows:

- Key Management Architecture Model – the basic, high level model for key management environments that provides a foundation for the remaining models and detailed information developed in the later sections of this standard.

- Key Management Conceptual Model – provides the basic model for the internal and external organization of the primary components in a key management environment.

- Key Lifecycle Model – the basic state machine model that encompasses the entire set of standard key states and the transitions between these states as maintained by the key management systems that implement this standard.

- Key Management Sequence Models – a series of models that describe the high level interactions between the primary components in the Key Management Conceptual Model.

5.1 Key Management Architecture Model

The protection of stored data using encryption requires the use of a key management environment where the encryption keys and other security related parameters can be adequately managed and protected. This standard defines a basic architecture for key management environment which includes the definitions and relationships between components within this environment.

Figure 1 illustrates the architecture model for the key management environment defined in this standard.

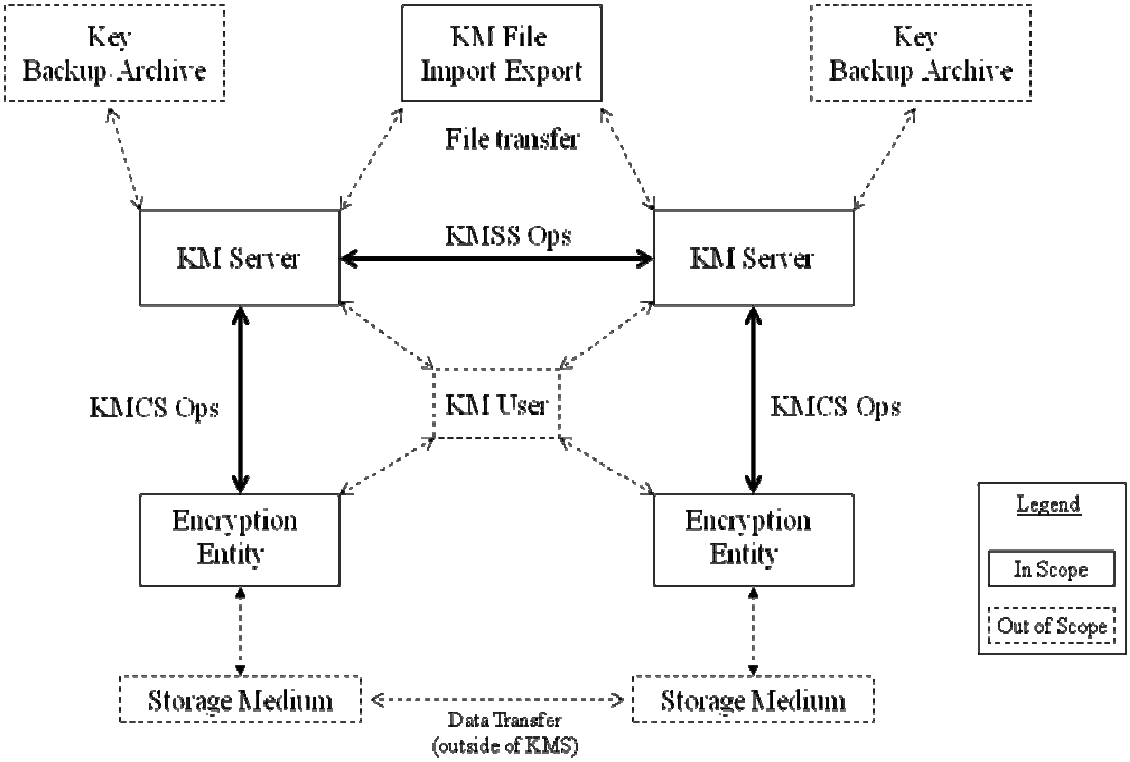

Figure 1— Key Management Architecture Model

Model discussion…

5.2 Key Management Conceptual Model

The key management conceptual model is defined in this standard to provide a more detailed view of the relationship between the critical entities involved in the key management environment. The purpose of the key management conceptual model is fourfold:

- To explicitly identify the differences between data plane operations and control plane operations. The process of actually using an encryption key…
- To identify the entities, components, and their interfaces which are defined as part of this standard.
- To identify the entities and components which are not within the scope of the standard, but need to be understood based on their impact to a key management environment.
- To understand possible embodiments of this standard in actual implementations. The key management conceptual model, coupled with the use cases defined in the informational annex of this standard will aid implementers in their efforts to be compliant to this standard.

Figure 2 illustrates the conceptual model for the key management environment defined in this standard.

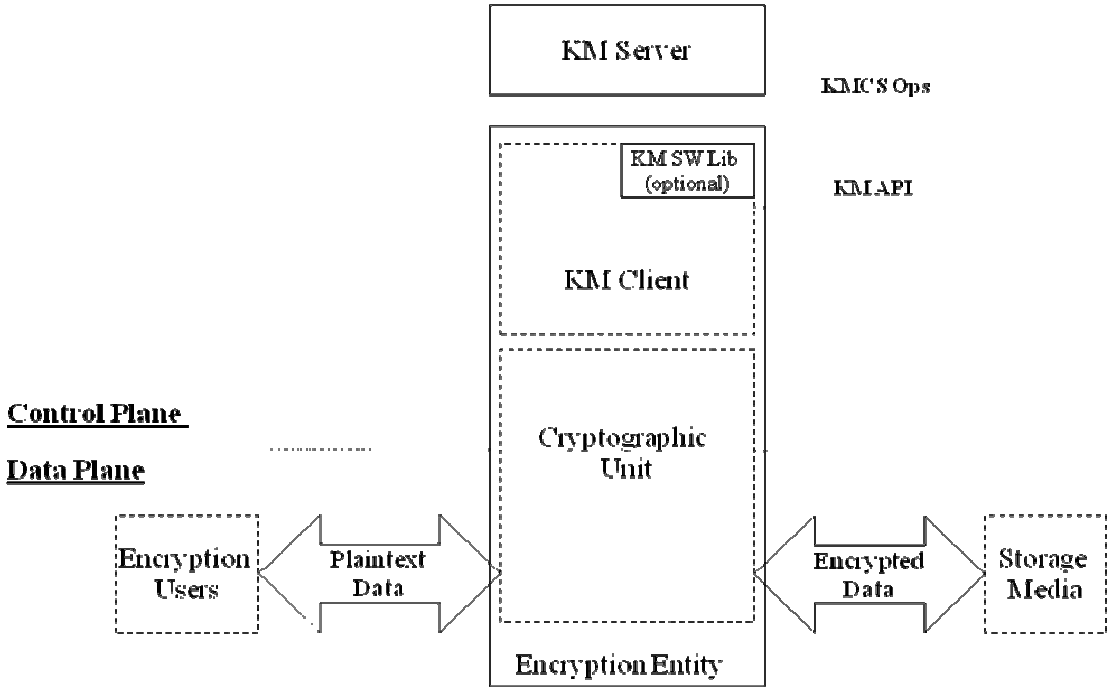

Figure 2— Key Management Conceptual Model

Model discussion…

5.3 Key Lifecycle Model

A crucial aspect of this key management standard is the concise definition for the states of encryption keys and the various transitions that can occur between these states. This definition of states of transitions between states for encryption keys is commonly referred to as the key lifecycle. Several standards bodies have attempted to define general key lifecycle models with their respective key state and transition definitions. These models were typically created in the context of using keys for protecting "data in flight" over communications links. The ephemeral nature of the keys and protected communication "sessions" does not lend itself well to the using these models "as-is" in the protection of stored data, or "data at rest" as it is commonly known.

A modified key lifecycle model is required that adequately define the key states and transitions between these states to ensure the highest levels of protection for stored data. The key lifecycle model defined herein is based largely on the key lifecycle models defined in NIST SP800-57 Part 1, and IEC/ISO 11770-1, but with additional states, transitions, and clarifications of these states and transitions in context of key management for the protection of stored data environments. The informative annex of this standard provides a concise comparison of the key lifecycle states and transitions presented in this standard, and compares them to the key lifecycle states and transitions presented in the key lifecycle models of applicable international and national standards.

Figure 3 illustrates the key lifecycle model as defined in this standard.

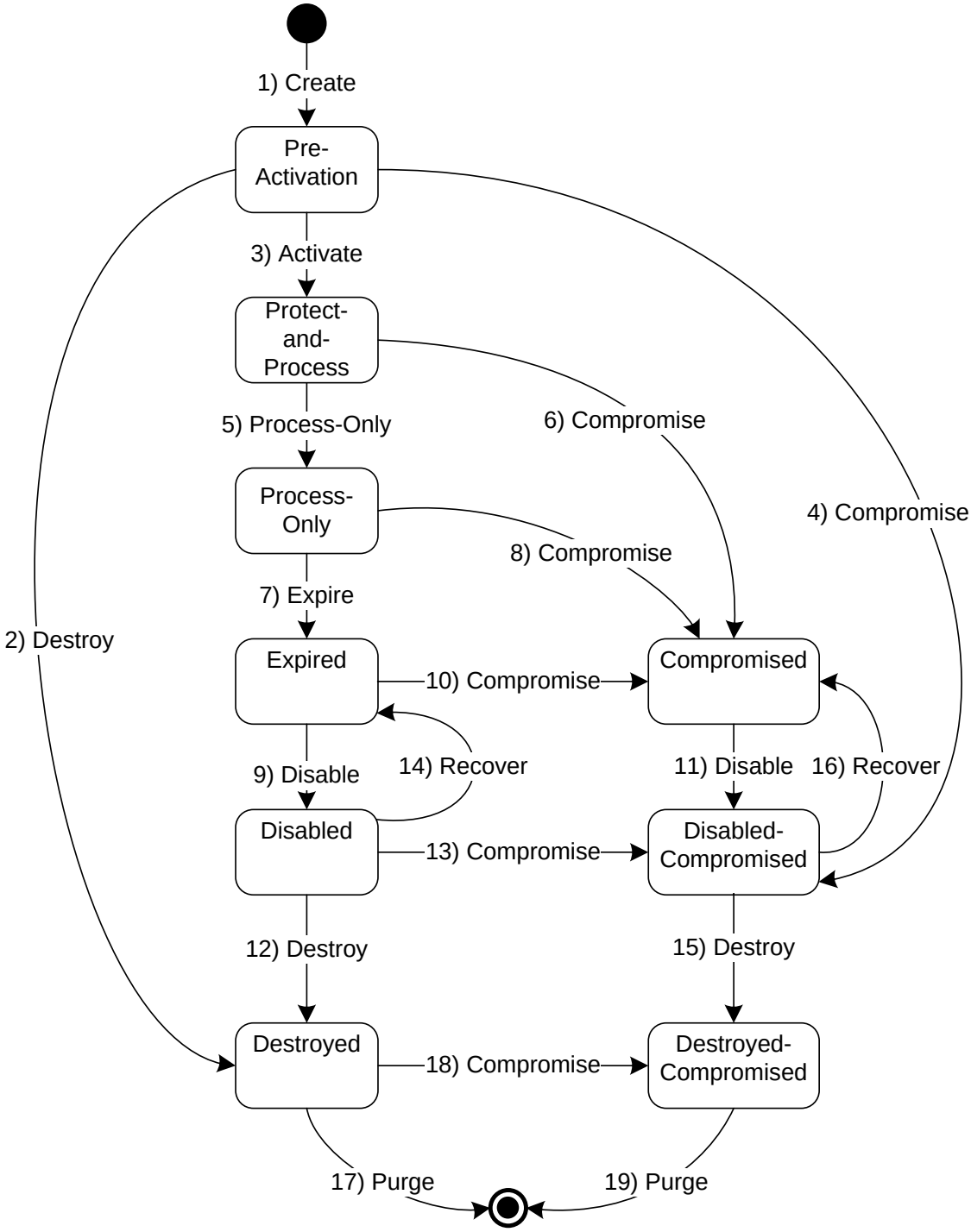

Figure 3— Key Lifecycle Model

The key lifecycle model consists of a set of key states in which depicts the states any particular encryption key can exist in at any particular time. In addition, this model indicates the allowable transitions between states. The definitions of these transitions include the events or actions necessary to cause a key to transition from one state to the next. These events or actions include time out events or user/management controlled actions. To complete the key lifecycle model, the definition of a set of time periods must be

defined that control the automated transition of keys between several of the states in the key lifecycle model.

5.3.1.1 Key State Definitions

The following are the formal descriptions of the key states defined in the key lifecycle model:

— **Pre-Activation**: The key has generated but is not yet in use or distributed to any KM Client or Cryptographic Unit. A key in this state can only exist in the KM Server. Such a key can be distributed to a KM Client.

— **Protect-and-Process**: A key in this state can be used for both encryption (i.e., to protect information) and decryption (to process information). A key is placed into this state when it is initially given to a KM Client, i.e., it is activated. The activation is done when a KM Client requests a new key. If a key is created by a KM Client or Cryptographic Unit and is then given to the KM Server, the key will be immediately placed into Protect-and-Process state. A key will remain in this state until the encryption period passes.

— **Process-Only**: A key in this state can be used for decryption but not encryption. When a KM Client/Cryptographic Unit determines that none of the keys available to it (e.g., for a specific tape cartridge or disk volume that is being read or written) are in the protect-and-process state, and it needs to perform encryption, it should create a new key. Keys transition from Protect-and-Process to Process-Only when the Encryption Period for the key expires, or as a result of an administrative action. A key will remain in the Process-Only state until the Crypto Period passes. At that time the key will be expired and move to the Expired state.

— **Expired**: An expired key is a key that has had its Crypto Period expire, but it may still be needed to process (decrypt) information. Keys in this state may be used to process (decrypt) data only. An expired key can be delivered to a KM Client, but the Cryptographic Unit should not use the key to encrypt data. A key will remain in this state until the key's Disable Period expires or as a result of an administrative action. At that time the key will be disabled and it will move to the Disabled state.

  The difference between keys in the Process-Only and Expired states is subtle. As far a KM Clients or Cryptographic Units are concerned, the states are equivalent. The difference is significant only to KM Servers. A key in the Process-Only key is still in the NIST operational phase and would routinely be delivered to KM Clients. A key which has had its Crypto Period expire has moved into the NIST post-operational phase, and may have additional restrictions imposed on its delivery to KM Clients.

— **Disabled**: A disabled key is a key that is still known to the KM Server but it will not be delivered to a KM Client. A disabled key's key material remains intact in the KM Server. A key will remain in the Disabled state until either the key's Destruction Period expires, or administrative action places the key in the Destroyed state, Compromised state, or recovers the key to the Expired state. If the key's Destruction Period expires, the key will be destroyed and will transition to the Destroyed state.

— **Compromised**: Keys are compromised when they are released to or discovered by an unauthorized entity. Compromised keys should not be used to protect information, but may need to be used to process information. The KM Server cannot determine if a key has been compromised. An administrative action is needed to inform a KM server that a key has been compromised. A compromised key will be delivered to a KM Client, but the Cryptographic Unit should not use the key to encrypt data. Just as with Expired keys, a KM Server may impose additional restrictions on the delivery of compromised keys to a KM Client. A key will remain in this state until the Disable Period expires. At that time the key will be disabled and move to the Disabled-Compromised state.

— **Disabled-Compromised**: A key that is both disabled and compromised. Disabled and compromised keys will never be delivered to KM Clients. A key will remain in the disabled compromised state until either the Destruction Period expires or the key is recovered to the Compromised state by

administrative action. If the Destroy Period expires, the key will be destroyed and moved to the Destroyed-Compromised state.

— **Destroyed**: A destroyed key is a key which has had its key material removed from the KM Server. Information or metadata about the key may be retained by the KM Server. Destroyed keys cannot be delivered to a KM Client. Keys can be destroyed by either an administrative action in the KM Server, or by the expiration of the Destruction Period of the key while it is in the Disabled or Disabled-Compromised states.

— **Destroyed-Compromised**: Similar to keys in the Destroyed state, but the key was compromised before or after destruction.

— **Terminal (Purged)**: Purged is the terminal state for keys. A purged key is a key that no longer exists in the KM Server in any form. Neither the key material nor any metadata about the key is known to the KM Server. A purged key cannot be delivered to a KM Client.

5.3.1.2 Key State Transition Definitions

The following are the formal descriptions of the key state transitions defined in the key lifecycle model:

— **(1) Create**: When a key is created by a KM Server, it begins in the pre-activation state. This transition occurs as soon as a key is generated within the KM Server, such as a key being generated by a random number generator (RNG).

Transition 1 applies to newly created keys; the key immediately transitions to Pre-Activation state upon its creation.

— **(2) Destroy**: A key in the Pre-Activation state may be moved directly to the Destroyed state. If the key has never been activated and is no longer required, but there is a requirement to maintain information about the key, the key may be destroyed. This transition can only occur as a result of administrative action.

Transition 2 applies to keys in the Pre-Activation state; the key transitions to the Destroyed state.

— **(3) Activate**: A key transitions from the Pre-Activation state to the Protect-and-Process state when it is available for use. This transition occurs when the key is assigned for use by a KM Client. There is a bit of confusion around the terminology, since a KM Client may view this action as “creating a key” even though the KM Server views this as assigning a previously generated key.

Transition 3 applies to keys in the Pre-Activation state; the key transitions to the Protect-and-Process state.

— **(4) Compromise**: A key transitions to the Compromised state when it has been determined to have been released to or discovered by an unauthorized entity. The KM Server cannot determine that this has happened, so this transition occurs as the result of an administrative action.

Transition 4 applies to keys in the Pre-Activation state; the key transitions to the Compromised state.

— **(5) Process-Only**: A key transitions from the Protect-and-Process state to the Process-Only state when a period of time equal the Encryption Period has expired after the key is activated. This transition may also occur as a result of administrative action. Some KM Clients and Cryptographic Units may be unable to strictly enforce this transition.

Transition 5 applies to keys in the Protect-and-Process state; the key transitions to the Process-Only state.

— **(6) Compromise**: Same actions as for the (4) Compromise transition.

Transition 6 applies to keys in the Protect-and-Process state; the key transitions to the Compromised state.

— **(7) Expire**: A key transitions from active to Expired when its Crypto Period expires. This transition may also occur as a result of administrative action.

Transition 7 applies to keys in the Process-Only state; the key transitions to the Expired state.

— **(8) Compromise**: Same actions as for the (4) Compromise transition. Transition 8 applies to keys in the Process-Only state; the key transitions to the Compromised state.

— **(9) Disable**: This transition will occur after the disable period for a key passes. This transition can also occur as a result of administrative actions. The key material and its metadata will be retained by the KM Server. The key will no longer delivered to KM Clients.

Transition 9 applies to keys in the Expired state; the key will transition to the Disabled state.

— **(10) Compromise**: Same actions as for the (4) Compromise transition.

Transition 10 applies to keys in the Expired state; the key transitions to the Compromised state.

— **(11) Disable**: Same actions as for the (9) Disable transition.

Transition 11 applies to keys in the Compromised state; the key transitions to the Disabled-Compromised state.

— **(12) Destroy**: A key can be destroyed when it is no longer needed. When the Destruction Period for a key expires, it will be destroyed. This transition can also occur as a result of administrative action. Active keys (keys in the Protect-and-Process or Protect-Only states) cannot be destroyed; only keys that have been previously activated and are now in the Disabled or Disabled-Compromised states can be destroyed. When a key is destroyed, the key material is removed from the KM Server. Metadata about the key is retained in the KM Server.

Transition 12 applies to keys in the Disabled state; the key transitions to Destroyed state.

— **(13) Compromise**: Same actions as for the (4) Compromise transition.

Transition 13 applies to keys in the Disabled state; the key transitions to Disabled-Compromised state.

— **(14) Recover**: When a disabled key is required to be delivered to KM Clients, it can be recovered. This occurs as a result of an administrative action.

Transition 14 applies to key in the Disabled state; the key transitions to Expired state.

— **(15) Destroy**: Same actions as for the (12) Destroy transition.

Transition 15 applies to key in the Disabled-Compromised state; the key transitions to the Destroyed-Compromised state.

— **(16) Recover**: Same actions as for the (14) Recover transition.

Transition 20 applies to keys in the Disabled-Compromised state; the key transitions to the Compromised state.

— **(17) Purge**: The action of purging a key removes all information about the key from the KM Server's key records. Neither the key material nor the key value will be retained by the KM Server. Note, however, that audit logs or other indirect information in the KM Server may still contain information about the key.

Transition 17 applies to keys in the Destroyed state; the key transitions to the Purged state. However, since the key does not exist in the KM Server, there is no record for the key showing the Purged state.

— **(18) Compromise**: Same actions as for the (4) Compromise transition.

Transition 18) applies to keys in the Destroyed state; the key transitions to the Destroyed-Compromised state.

— **(19) Purge**: Same actions as for the (17) Purge transition.

Transition 19 applies to keys in the Destroyed-Compromised state; the key transitions to the Purged state. However, since the key does not exist in the KM Server, there is no record for the key showing the purged state.

5.3.1.3 Key Time Period Definitions

Many of the key states and transitions defined in the key lifecycle model are dependent on one or more time related attributes associated with the keys. These time related attributes are typically categorized as timer values or timeout periods that are activated when keys enter certain states and their expiration cause transitions to certain states.

The key life cycle is based on four time periods:

— **Encryption Period**: This is the period of time after a key is activated that it can be used to encrypt data.

— **Crypto Period**: This is the period of time after a key is activated that it can be used for routine decryption. This time period should always be as long as or longer than the Encryption Period.

— **Disable Period**: This is the period of time after a key is activated before a key will be disabled and become unavailable to KM Clients. A key may still be delivered to a KM Client after the Crypto Period ends but before the Disable Period ends, however, this may require administrative action on the KM Server. This case would be for a non-routine usage of the key. This time period should always be as long as or longer than the Crypto Period.

— **Destruction Period**: This is the period of time after a key is activated before the key is destroyed by the KM Server. This time period should always be as long as or longer than the Disable Period.

All key related time periods start when a key is activated; this occurs when a key is given to a KM Client by a KM Server, or when a key generated by a KM Client is given to the KM Server. Any time period can range from zero to forever, with the limitation that each time period must be at least as long as its predecessor as specified above.

Figure 4 illustrates the key related time periods and their relationship to the keys states of the key lifecycle model.

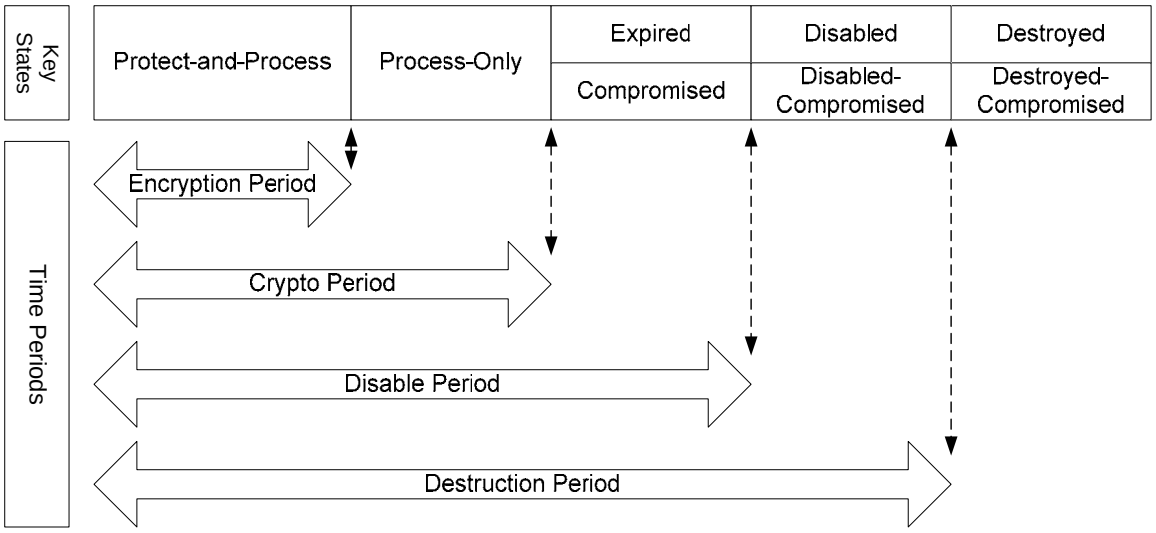

Figure 4— Key Time Periods Compared to Key States

6. Key Management Operations

[Any changes required will be coordinated with Operations and Object subcommittee's proposals.]

7. Key Management Transport

[Any changes required will be coordinated with Transport subcommittee proposals.]

8. Key Management Messaging

[Any changes required will be coordinated with Messaging subcommittee proposals.]

Annex A

(informative)

Bibliography

[List all bibliographic material here]

Annex B

(informative)

Example Use Cases

[Objects & operations or use cases should add the appropriate use cases here]

Annex C

(informative)

XML and TLV Schema Definitions

C.1 XML Schema

[Additional messaging group information for selected XML syntax goes here]

C.2 TLV Schema

[Additional messaging group information for selected TLV schema goes here]

Annex D

(informative)

Comparison of P1619.3 Key Lifecycle Model with Other Standards

The following table compares the mappings from the P1619.3 key states to the key states defined in the NIST SP 800-57 lifecycle model and the IEC/ISO 11770-1 lifecycle model:

P1619.3 Key State	NIST SP 800-57 Key State	IEC/ISO 11770-1 Key State	Notes
Pre-Activation	Pre-activation	Pending Active	Notes 1 & 2
Protect-and-Process	Active	Active	Notes 3 & 4
Process-Only	Active	Active	Notes 3 & 4
Expired	Deactivated	Deactivated	Notes 3 & 4
Disabled	Deactivated	Deactivated	Notes 3 & 4
Compromised	Compromised	None	Notes 3 & 5
Disabled-Compromised	Compromised	None	Notes 3 & 5
Destroyed	Destroyed	Destroyed (Implied)	Notes 1 & 6
Destroyed-Compromised	Destroyed Compromised	None	Notes 1 & 5
Purged	None	None	Notes 7 & 5

Note 1: IEEE P1619.3 key state is identical to the key state defined in the NIST SP 800-57 standard.
Note 2: IEEE P1619.3 key state is identical to the key state defined in the IEC/ISO 11770-1 standard.
Note 3: IEEE P1619.3 key state is a substate of the key state defined in the NIST SP 800-57 standard.
Note 4: IEEE P1619.3 key state is a substate of the key state defined in the IEC/ISO 11770-1 standard.
Note 5: IEEE P1619.3 key state is a new state not defined in the IEC/ISO 11770-1 standard.
Note 6: IEEE P1619.3 key state is the same as the state implied in the IEC/ISO 11770-1 standard.
Note 7: IEEE P1619.3 key state is a new state not defined in the NIST SP 800-57 standard.

[Need to add comparison of key state transitions as well.]