CKM Framework

Cross-Reference to Related Documents

Constructive Key Management® or CKM is embodied in numerous standards (X9.69. X9.73, X9.84, X9.96) published by the American National Standards Institute (ANSI) and is being incorporated into ISO 22895 which includes reference to the cited ANSI standards. These standards are also incorporated herein by reference.

ANSI X9.69 – Framework for Key Management Extensions

This standard and other ANSI standards have been implemented for securing financial services in the United States. The framework for Constructive Key Management® (CKM) is identified in the standard. A multi-tier architecture is illustrated that can offer a scalable enterprise solution. The elements of the CKM administration are described in the standard. The Key Establishment portion of key management is defined so that designated groups of user can establish keys for data encryption via a key derivation procedure, thus avoiding key distribution in the usual sense.

X9.73 – Cryptographic Message Syntax

The Standard specifies a Cryptographic Message Syntax (CMS) that can be used to protect financial transactions and other documents from unauthorized disclosure and modification. CKM is defined for message protection that is used in the financial sector. It is encoded in the ASN.1 coding scheme to offer additional efficiency in transmission.

X9.84 - Biometric Information Management and Security for the Financial Services Industry

This Standard specifies the minimum-security requirements for effective management of biometric data. In additional to various aspects of managing of biometrics, the integrity and privacy protection of biometric data is also defined. CKM is recognized as an encryption framework that can be used to protect the biometric data.

X9.96 - XML Cryptographic Message Syntax (XCMS)

This Standard specifies a text based CMS represented using eXtensible Markup Language (XML) 1.0 encoding that can be used to protect financial transactions and other documents from unauthorized disclosure and modification. The message syntax has the following characteristics:

1. Protected messages are represented using the Canonical XML Encoding Rules (cXER), and can be transferred as verbose markup text or in a compact, efficient binary representation using the Basic Encoding Rules (BER) or the canonical subset of BER, the Distinguished Encoding Rules (DER).

2. Messages are protected independently. There is no cryptographic sequencing (e.g., cipher block chaining) between messages. There need not be any real-time connection between the sender and recipient of the message. This makes the syntax suitable for use over store-and-forward systems, e.g. Automated Clearing House (ACH) or Society for Worldwide International Funds Transfer (SWIFT). Standard attributes are defined to allow applications to maintain relationships between messages, if desired.
3. The syntax is algorithm independent. It supports confidentiality, integrity, origin authentication, and non-repudiation services. Only ANSI X9-approved algorithm(s) may be used for message digest, message encryption, digital signature, message authentication, and key management.
4. Support for biometric security, enhanced certificate techniques such as compact-domain certificates and key management extensions such as CKM are provided.
5. Selective field protection can be provided in two ways. First, by combining multiple instances of this syntax into a composite message; and second, by using identifier and type markup tag names to select message components to be protected in a single message, which allows reusable message components to be moved between documents without affecting the validity of the signature.
6. Precise message encoding and cryptographic processing requirements are provided.

These standards are available at http://webstore.ansi.org/ansidocstore/default.asp

Draft ISO 22895 – Financial Services - Security – Cryptographic Syntax Scheme

The standard is based on the combination of inputs from ANSI X9.73 and ANSI X9.96 and is being updated to reflect the current direction in protecting messaging for the international financial community. A cryptographic message schema is defined whose concrete values can be represented using either a compact binary encoding or a human-readable markup language (i.e., XML). CKM is identified as a key management framework that can be applied to protecting content or information.

CKM Overview

The Constructive Key Management® technique (CKM) described in ANSI X9.69, X9.73, and X9.96 is a key management framework that includes architecture for administering keys and for an encryption schema. The framework results in an encrypted message that users share through a common set of keying components.

Security through encryption of information may be done through the transport, or network layer, or encryption may be applied to the information content. There is often a need for a mix between Transport and Content encryption protection.

Typically, transport protection has focused on securing the channel through a Virtual Private Network (VPN). In addition to securing the channel, protection of information or content may also be extended into protecting the content directly.

There is often a need for a mix between a technical encryption solution and a business representation through content. In a client bank exchange, for example, it could be decided to encrypt only some parts of the message depending on the business need for confidentiality and access control. Protecting content can offer more flexibility to combine business models with security through encryption. Signing for non-repudiation can be considered an adjunct to a content protection process.

The message may be protected through the network as in the example of a channel in which there is no direct binding of encryption to the message or may be protected at the content level for which there can be a direct binding of encryption to the content. The CKM schema may apply to a network encryption implementation or to a content protection implementation. For content protection the CKM schema can apply to on-line and off-line communication environments.

The result is to create persistent protection through encryption, resulting with a binding of the encryption and the message throughout the life cycle of the message. Message content may be in the form of data or information. The CKM encryption schema for content protection may be viewed as shifting from a temporal keying design that may be used to establish the communications channel to an ephemeral keying design that may be used to protect the content in transit and in storage.

CKM is exactly what the name implies: a key is constructed per message. A message-encrypting key (or working key) is constructed as needed by the originator of the message, and can only be re-constructed by the appropriate entities as necessitated by their respective roles and relationship to the content.

There are several components that are included in the CKM schema:

1. A random key split for variability,
2. Symmetric split keys for defining the encryption boundaries (or domains) and for updating encryption at a network level without having to change cryptographic controls that are bound to the message, and
3. Asymmetric split keys that bind encryption to controlling access to the message through content. The CKM schema may also include an integrity check on the message content by using a Secure Hash Algorithm (SHA).

Content access may be defined in the form of roles or rules, or other information related representation. The use of a role for controlling the message access offers the potential for multiple persons or events having access, as designated by the role. The role continues to be effective even with changes in persons. A *Credential* or a label represents a role in the CKM schema. The *Credential* remains current with the role or other content designation.

By using roles, the CKM schema is particularly useful where the message flows among groups of users that are unknown to the sender, and where the information must be persistently protected over time. The CKM schema is a closed encryption process, dependent on the creation and life cycle of the key splits through an established administrative process. The *Credential* is bound directly to an asymmetric split key. The asymmetric split key may also be used to enforce read and write access privileges – asymmetric public split key for write and asymmetric private split key for read. The asymmetric split key may be a key establishment (e.g., Diffie-Hellmen) with ephemeral (e.g., elliptical curve).

A combination of these keys splits result in a data or message-encrypting key that is used with a symmetric content encryption algorithm. Inherent with the mix of components and the symmetric algorithm is an ability to view the schema as a set of modules that can offer different authorization or access control results. As an example, by maintaining one set of components and applying the components and the resulting message encrypting key to various symmetric algorithms, the encrypted message cipher will vary with the symmetric algorithm used. In the example, each symmetric algorithm results in an enveloped message that is mathematically independent from another symmetric algorithm result.

The symmetric and asymmetric split keys are pre-placed and positioned through an administrative process. A random split key is generated, to provide entropy, during the process of combining the split keys. Once a message is encrypted, that message may be forwarded to other CKM participants who have the same split key values necessary to decrypt. In this context, only those with appropriate peerage may push an encrypted message into a browser distribution for selective decryption and have the pre-determined split keys, and potential to reconstruct the working key. The symmetric key splits are mathematically included with the asymmetric splits to result in the message-encrypting key. The splits might be exchanged as part of a key management administrative protocol. This can mean that the message-encrypting key is always fully recoverable and the message is always decryptable by the appropriate recipients as well as the message owner and with the proper controls, the owner of the overall enterprise.

The CKM header can be part of a more extensive security encryption header that includes certificates and other code integrity elements. To ensure the integrity of the header, processes should be used to protect sensitive header components.

Once the encryption of the message is completed, the message-encrypting key is destroyed. Only through the construction process can a decryption of the message take place. <u>The CKM schema results in a dynamic creation of a message encrypting key on-the-fly.</u>

The CKM schema can be used for enforcing the separation of data through encryption as an access control functionality while resulting in also protecting the content. Key splits including the associate credentials that represent various categories of access, expressed in the taxonomy of the organization, are combined into a message-encrypting key that is used with a symmetric encryption algorithm to encrypt or decrypt the message. Typically, a number of pre-placed splits are used; these splits are combined with a random split to ensure that a different message-encrypting key is used for each message. The random split is created at the time the message is prepared for encryption. A significant benefit of this approach is the ease of understanding, management, granularity and the overall efficiency of the process compared to traditional static key approaches.

It is possible to create a nesting of multiple message encrypting keys so that the overall content message contains separately encryption-enforced messages. Access to the encrypted messages may differ through the selection of the credentials.

The asymmetric split keys that bind access control encryption to the message through content may also be a symmetric split key. The symmetric key splits can be combined directly, using (for example) a secure hash algorithm. An asymmetric split key has an advantage in that the public key and private key may be used for read and write separation.

Administrative Overview

There are two major administrative functions required to manage the CKM system: the CKM administration (i.e., Enterprise Builder) and Token Distribution. In large organizations, these could be independent of each other. If the two administrative functions are separated, it is possible to establish a scenario for creating the policy and split keys for an organization within the CKM administration and, separately, maintaining the daily use of the split keys with the token distribution. Tokens provide mobility to the user, where appropriate. Tokens also increase the level of assurance associated with the distribution and storage of the *Credentials* issued.

The CKM Administration shall design the overall interconnectivity within the enterprise, establish the policy parameters for maintaining the integrity of the enterprise, create the *Credentials* and asymmetric and symmetric key splits, and establish the read and write privileges for the organizational access needs.

The Token Distribution function shall include the day-to-day management of the system, the maintenance and distribution of *Credential* key splits, the maintenance of current users and their tokens, and the maintenance of the balance of split keys. It is at the Token Distribution level where dual control of key material shall be accomplished, if required by the enterprise. The distribution of split keys shall be accomplished through a secure channel. After the CKM Administration has defined the system and system parameters, and created the split keys, the CKM Administration is normally complete. There may be instances for additional split keys or changes to the split keys and occasional policy changes that need to be done.

KEY CREATION OVERVIEW

There are two fundamental steps for key creation within the overall CKM process. The initial key split value creation is done within the administration process and creates mathematical seeds for selected key splits further used in the CKM combiner process. A second, working key creation is done through a combiner process which uses the split keys from the administration process but adds the additional random split key at the time a message is encrypted. The random key split changes with every session or creation of a message encryption key. The other key splits may change depending on their usage. Once a recipient of an encrypted message has all the key splits, the message may be decrypted.

The user creates a message encryption key at the time of encryption or at the time of decryption by combining the appropriate asymmetric key splits invoked by the credentials associated with each transaction and the symmetric key splits. As a minimum, the symmetric key split must be a random key split. Other symmetric key splits may include a maintenance key split and a domain key split. The maintenance key split can be used as a network level key for cryptographically updating everyone. A change of the maintenance key split effects all in the cryptographic boundaries of the network. A domain split key establishes the boundaries of an information domain and can be used to define virtual domain control that is mathematically aligned with the domain.

After the message encryption key is used, that key is destroyed. Except for the random split key, only pointers to the key splits are retained. The concept of pointers to the actual split keys minimizes an external discovery of the split keys associated with the encryption process. The random split key is also encrypted. Key to encrypt the random split key may be derived from an asymmetric public key associated with a credential or a combination of credentials and their asymmetric public keys, or an enterprise symmetric key.

There may be times when it is more efficient for an application to retain the message encryption key, and there is sufficient integrity included in the protection of this key. Instead of having to recreate message encryption key on demand, a message encryption key with an extensive use or a predefined life cycle, such as a key used to verify Message Authentication Code (MAC), may be retained. At the appropriate time a message can be sent to all appropriate users to activate a specific message encryption key.

The message encryption key is used with a content encryption algorithm. This algorithm can also be used as a secondary access control capability. By applying different symmetric algorithms to a common set of split keys, a strong data separation can be made. An example is having a single message with multiple accesses that need to be enforced by encryption and result in multiple users accessing a common message but with differing access rights.

Encryption as an Information Security Tool

As an information security tool, cryptography can complement changes in information technology. The growth of information systems has been phenomenal. However, today's cryptography and its key management have reached a crossroads as it attempts to adapt to the information system changes. The predominant public key management scheme of the 1980s and 1990s has shortcomings that will constrain the information industry from expanding into greater information sharing applications without a shift in public key application. A new direction in encryption is needed if the distributive enterprise solution, with its myriad information applications, is to be made effective.

By combining what has been learned in the implementations of public key management and pre-1980s key management, an expanded symmetrical core key management technology emerges as the better choice for bridging to the 21st century information applications that include data-at-rest and communications security models.

Issues that confront future information protection models such as scalar, data separation, or role-based enforcement, system performance, and multiple enterprises authentication for the user or for the workstation can be satisfied by combining enterprise-wide information distribution with information control and access control capabilities while protecting the information.

An evolution in cryptographic technology is taking place. A symmetrical key management model that is particularly well suited for role-based access control, that look to the roles users have within an organization, and to the information access that should be afforded those roles, is being bound to an authentication key management model that incorporates the mathematical models of digital signatures and signed public certificates with physical properties of identification techniques as smart cards. The resultant key management technology is the basis for what will be referred to herein as Constructive Key Management® (CKM).

The age of the Internet and "information warfare" is upon us. The protection of selected information and selected channels of information has become a paramount concern in defense and commerce. While this evolution has been taking place in information processing, cryptography has emerged as a premier protection technology.

Cryptographic keys are an essential part of all encryption schemes. Their management can be the most critical element of any cryptography-based security. The true effectiveness of key management is the ability for keys to be maintained and distributed secretly without penalizing system performance, costs, or user interaction. The management of the keys must be scalar, must be capable of separating information flow, must include interoperability needs, and must be capable of providing information control.

A method of distributing keys predominantly used in the 1980s and 1990s is public key, or asymmetrical, cryptography. In this method, the conversion of information to cipher text and the conversion of basic properties of the public key method include separate encryption and decryption keys, difficulty in deriving one key from another, secret decryption keys, and public encryption keys. The implementation of public key information encrypting keys is the result of the mathematical combination of the encryption and decryption keys.

Public key management was developed for a communications channel requirement to establish cryptographic connectivity between two points, after which a symmetrical cryptogen such as DES was to be executed. Over the years, public key implementations have demonstrated their effectiveness to authenticate between two entities. However, to take the authentication process to a global certificate process has not been successfully done. Stated in other words, public key management is effective in an information model that defines point-to-point communications channels where the information encrypted may not need to be recovered.

Many of the recent implementations of public key management have left users with an option to create their own pair-wise connectivity within the network. This action can leave an organization vulnerable, and in some cases liable, if that user leaves the organization without identifying the keys that were previously used for encrypted files or data. Also, to assure the integrity of the public key from misuse, a third-party infrastructure scheme has surfaced, that is, a certificate authority process is created to mathematically confirm that a particular public key was issued to a specific user. The exchange of certificates with a third party can significantly impact the performance of a network. Further, this raises the legal issue of whether an organization should give a third party control over the validation of corporate correspondence.

A negative aspect of the public key process is a high computation time, which can impact the performance of an information application. In many instances, hardware solutions have compensated for the high computational requirements. Semi-public key architecture historically has been a point-to-point design; moving to a distributive network with group sharing of information can create higher transmission costs and greater network impact. Although the older key management systems of the 80’s and 90’s worked well for point-to-point communications and one-to-one file transfer, they are too time consuming to use in a case wherein a single file is placed on a file server and decrypted by thousands of users. As the trend toward work groups and complex communications infrastructures continues, the need for more efficient information and communications key management technology becomes paramount.

Shared secret keys or symmetrical key is the earliest key management design and pre-dates public key management. The earlier versions of symmetrical designs suffered what was referred to as the “n-squared” problem, in that the number of keys needed was very large as a network expanded, and these designs did not have an effective authentication capability. However, symmetrical encryption has a measurably better system performance than public key implementations.

Cryptographic keys are an essential part of all encryption schemes. Their management is a critical element of any cryptographic-based security. The true effectiveness of key management is the ability to have keys created, distributed, and maintained without requiring user interaction and without penalizing system performance or costs.

Asymmetric, also called public-key, cryptography has received significant attention in recent years. The public-key method includes separate public encryption and private decryption keys that provide a measure of difficulty in deriving the private key from the public key. Public-key management was developed to establish cryptographic connectivity between two points in a communications channel after which a symmetric cryptogen, such as DES (Data Encryption Standard), was to be executed.

Over the years, public-key implementations have demonstrated their effectiveness to authenticate between entities. However, public-key methods have had some shortcomings to handle the requirements of today's global networks.

CKM builds on the advantages, and takes into account the disadvantages, of both public-key and symmetric key implementations. CKM combines an encryption process based on split key capability with access control credentials and authentication process based on public-key techniques. CKM is most effective in modern distributive information models where information flow and control can be defined, where the information encrypted may need to be recovered, and where authentication using public-key technology and a physical token can be implemented. CKM should be considered a key management framework that can be applied protecting content or extended for signing.

Protecting content may also be referred as using encryption for protecting data-at-rest as opposed to data-in-transit. Data-at-rest refers to data encrypted as logical units (objects) and includes the creation, processing, transfer, and storage of these objects. Data-in-transit refers to the stream encryption of data moving through a physical or logical communication channel during a certain period of time. CKM can apply to both types of encryption.

Current CKM technology meets the set of "classical" security objectives.

1. Data confidentiality keeps the content of information from being revealed to those who are not authorized to read it. CKM uses symmetric key cryptography with a robust key management system that provides a new and unique working key for each encryption. The user "selects" the readership or has the readership defined for each encrypted object. An object can be data-at-rest, such as a file, a message, or data-in-transit, such as network traffic.
2. Access control restricts use of encrypted objects to those users specifically given permission to use them. Access control in CKM can be role-based for which permissions are granted and revoked based on that user's responsibility or position within an organization. It currently encompasses the actions of encryption and decryption but may include permissions to use certain programs, certain devices or specific hardware operating modes. Access control may also be extended to data base applications.
3. User Authentication establishes the identity of a user (person or device) to the system. A CKM Signing system can be used for authentication and can be enhanced with other security related technologies.

Smart cards and biometrics provide CKM greater security in meeting the third objective, User Authentication. As well as providing stronger user authentication when used as a token, a smart card can be an excellent hardware platform to implement various levels of CKM technology. The card may be used as a memory only device, or it can be expanded to include processing capability. An advanced smart card, called the SmartToken™ is an enabling technology for CKM. Along with its increased processing and memory, the SmartToken™ can, itself, have additional security enhancements such as a unique radio frequency signature for a random number generation capability.

Another value added security enhancement can be biometrics. Adding biometrics to CKM enhances user authentication and may provide pieces of information for generating the private key part for the asymmetric key cryptographic system that CKM uses for digital signatures.

Inherent in CKM is the means to meet additional objectives:

1. Data separation is the ability to keep data in the same physical space, yet still enforce access controls. Two cryptographic means of separation are used in CKM - separation by algorithm and separation by label. Data Separation can be defined through "Silos" where encryption of CKM enforces hardware memory allocation such as compartmenting smart card memory for multiple application usage while maintaining definable separation among the applications.
2. Key recovery in CKM is the ability to regenerate the keys used to encrypt objects. Key recovery means that within any particular CKM domain (or organization) encrypted objects are not lost with the loss of any individual. Key recovery for export is also possible. Asymmetric key cryptography used for digital signatures offers CKM the means to meet three additional security objectives concerned with message authentication:
    1) Data origin authentication (sometimes called message authentication) corroborates the source of CKM encrypted information.
    2) Data integrity is the ability to prove that a CKM encrypted object has not been altered since being encrypted and digitally signed. If digital signatures are not used, a Message Authentication Code (MAC) or Manipulation Detection Code (MDC) with encryption can provide data integrity.
    3) Non-repudiation proves that the signature on a signed object came from the signatory such that the signatory cannot deny digitally signing the object. The CKM framework includes an infrastructure that digital signatures can be included. The digital signatures may exist from other Public Key Infrastructures and supplement other security functionality of CKM, or the digital signatures may be directly derived from the CKM mathematics and be included in the CKM framework infrastructure.
3. Data separation can be enforced by encryption. The CKM can be used as an encryption tool to enforce data separation by providing an access control mechanism for a token's memory allocation. Data may be separated into compartments that are enforced by encryption. The assurance that the compartments remain intact and enforceable is through the access control functionality of CKM. Each compartment may include separate application data. By having the organizational owner of the token cryptographically separated from the application owner of the stored data, liability and privacy can be addressed. The user of the token becomes under the organizations control of the organizational owner of the token while each of a separate suite of applications can maintain their cryptographic and authentication models with the user separate of the token owner. A tagged environment such as used by an XML infrastructure can be mapped to a CKM header to further embed an encryption process with an information data protocol. Further integrity of the resultant embedded tagged data may require a coding schema.

4. Access to a database can be enforced through the encryption of a cell, a row, or a column. Of course, there is a performance consideration on which choice is selected. Another factor in data base encryption is: should encryption be leveraged for access to the metadata associated with the database. The security architecture parameters would identify an emphasis on protecting access to the metadata, or protecting access directly to the database, or a combination of the metadata and the database.

The CKM Framework

CKM is a framework for generating and regenerating cryptographic keys, and managing those keys within an organization. For data encryption, a cryptographic working key is generated immediately before an object is encrypted or decrypted. It is used to initialize a cryptographic algorithm for encryption or decryption. The working key is discarded after the encryption process is completed.

The working key is built from pieces of information within a combiner process. To be a participant in the system, a user must have the pre-placed pieces necessary to build the key; otherwise encryption and decryption cannot take place. A CKM Administration (Enterprise Builder) creates these pieces, which may be referred as cryptographic key splits. A subset of these splits is distributed to each user in the organization. One of the subsets for mapping information sharing and information control that each user receives is specific to that person and is defined through labels (credentials or permissions) for which the individual may use to encrypt (known as write credential) and which the credentials of that individual may use to decrypt (known as read credential). Several user authentication techniques are used to verify a user to the CKM system before that user is allowed access to this information.

To build a key, other key splits are needed: a variable system wide split, called the organization (domain) split and a variable system wide split, called the maintenance split are used. To this are added a random number, which is called the random split, and the user selected credential splits. The random split ensures that a unique working key is created for each use. User selected permission splits define the "readership" of the CKM encrypted object, i.e. which users will be able to decrypt the object. All of these splits are input to a process known as the CKM combiner process. The output of the combiner process is a unique number that is used as the basis for a working or session key.

CKM uses a hierarchical infrastructure to manage the distribution of data necessary for CKM enabled software to construct cryptographic keys. This infrastructure also provides a method of user certificate and public key distribution for asymmetric key cryptography so that digital signatures may be used.

Further CKM Administration

CKM Administration is structured so that it is a multiple tiered, hierarchical system. A top tier is identified as the Policy Manager. This process enables the closed cryptographic system to generate key splits. The key splits become the basis for the cryptographic process and once an information control relationship is established, these key splits are used in a combiner process with added random and other internal encryption elements for a working key or session key. The working key becomes the on-the-fly key to encrypt data.

From the Policy Manager is the creation of a working relationship for establishing information control through a Credential Manager. The next tier for access control, CKM can establish an information control paradigm through credentials that are directly related to one set of the key splits. Also, alternative methodologies are available for controlling information through other key splits such as the domain key split or the maintenance key split.

The Credential Manager process is given a subset of Credentials, specific algorithms, and enterprise policies from the Policy Manager. Individuals are allocated use of specific *Credentials* and algorithms from the Credential manager's subset. Organizational policies and system parameters generated by the Policy Manager are added to these *Credentials* forming an individual's credentials. A user's credentials are encrypted and distributed to that user as a *"Token".* The token may be in the form of a software token application or in the form of a hardware token application. The smart card can be an example of such a token. The process of *Credential* and algorithm allocation by the Credential manager allows an organization to implement a "role-based" system of access to information.

User authentication and its elements may include a password, a biometric, and/or a smart card. The authentication portion of the framework is bound to the CKM framework. A multiple authentication process is possible that combines the authentication elements to create a key that is used to encrypt data on the smart card or some other token (the choice of the biometric process may be included within the mathematics of the overall process if the biometric event can be replicated into a subkey). Biometric data is physiological or behavioral information that is unique to each individual. Authentication also can include a digital signature.

The CKM infrastructure can be used to provide the means to distribute public keys, which give the CKM framework an ability to manage cryptographic bound digital signatures. A digital signature may be sourced from an external PKI infrastructure. A resultant architecture with the CKM framework and a public key entry can be adjusted to the desired information assurance or comms assurance needs. A digital signature can also be developed from a CKM combiner math embedded with a digital signature math. The use of a Digital Certificate to develop a trust model is also an architectural system option. The Digital Certificate model would entail a periodic call to the CKM Administration to ensure that the trust establishment for authentication is maintained. An alternative for a slightly different trust model could be the creation of a digital signature, a multiple authentication process, and the CKM framework that would establish the user identity and maintain the identity for future access, but not have to periodically reestablish what was initially defined in a trust linkage. CKM is a closed encryption framework that once a linkage can be done, the trust parameters can be enforced through the encryption process. Of course, it is necessary to establish a positive identification for the user and a positive auditing to ensure further trust integrity. The CKM framework is to reinforce the identification for an access control or signing architecture.

REVOCATION WITHIN THE CKM FRAMEWORK

The CKM Framework is a closed system design. Revocation can take two avenues to ensure the integrity of the system. One avenue is to change a system level key split; thereby, causing a change to the working key, and a second avenue is to cancel access to a token or to an element external to the CKM framework. Of course, there has to be a means to connect the user to an active network with a level of processing. Neither revocation schemes can affect already accessed information that was protected by the CKM framework.

The maintenance key split can be stepped to create a new key split that is distributed to all the users, or there can be an additional use of the maintenance key split to represent a time allocation function – the key split is only good for a defined period of time.

The token such as a smart card can be administratively cancelled, and it being the host for the CKM framework, cancels the use of the framework capabilities.

Scalability with the CKM Framework

The subject of scalability can be easy to misinterpret.

Since the CKM framework results in a key-on-the fly while building on the traditional asymmetric and symmetric encryption algorithms, the CKM framework can be viewed as a modular design that can be adapted to specific requirements. If an export control requirement is central to the requirement, appropriate algorithms and key spaces (e.g., Suite B with the appropriate elliptic curves) can be included. The point is that the CKM framework is an adaptable architecture for designated encryption algorithms.

Current encryption schemas can include a focus on a public key only architecture that can result in a static key model. The seed key of the CKM framework may be viewed as an equivalent to the operational key of a static model. A public key rollover can be facilitated through the CKM framework when that combiner function is mated to a digital signature.

The multiple tiered CKM Administration has been demonstrated in other encryption architectures to be an effective means for scaling to larger enterprises. By having a separation between the administration of the enterprise policies and the enterprise key distribution, the administrative workload is facilitated. And, having the infrastructure support the token with its authentication model also offers a broad distribution capability.

The Role of the Credential

Credentials are associated with key splits, and an essential part of the administrative action. The use of the term 'labels' is interchangeable with the term 'credentials'. *Credentials* may be viewed in varying context. *Credentials* may be understood as information categories and sub-categories of addressees. *Credentials* may also be roles, or can be rules, or simply identifiers for information. Business information that is relevant to a banking transaction can be a trigger for controlling the flow of information, and the trigger is the *Credential*. The action associated with the *Credential* may give access to an electronic file, may activate a physical device, or differentiate access to portions of a single document, or function associated with the document e.g., print or copy. In many applications *Credentials* are humanly readable, so they are comprehensible to the sender who is determining the recipients of the encrypted content (or object). A *Credential* may also be defined through a collection of credentials (or a category) that relate to access privileges for a common event. In some instances, users can have all of the credentials from a category, or there are instances where some users may not be given all credentials or within all categories. It is possible to configure access rights to content through combinations of credentials and categories. *Credentials* may be associated with information contained in an XML tag.

The *Credential* may be personal to each user or user group. *Credentials* may be maintained on a (physical or logical) token that is distributed by the CKM administration. The token shall define the user's privileges. Access to the token may be through a biometric access or a pin or some combination for multi-factor authentication (the process that would be used is outside of the scope of this standard).

Every *Credential* points to a unique asymmetric key split value that is part of a CKM combiner process. The *Credential* establishes the access parameters that are further combined with other key split values to create the message encryption key (or working key). An administrative action can establish a policy for read or write access privileges that are enforced through the asymmetric key split’s public key mathematics.

User credentials, contained in computer files, include a user's permission set, i.e., the label splits, their associated label names and indices that can be used for encryption (write permission) and decryption (read permission), and the permissions to algorithms that may be used. In addition, the organization name and associated split, maintenance level and associated split, header encryption split and certain parameters to be used by the organization are contained in a user's credentials. Policies, such as minimum password length, are also included in the user's credentials. When digital signatures are used, a copy of all the organization's *Credential* manager's public keys are included, as well as the user's signed certificate, if applicable.

In assigning a permission set to a user, the *Credential* manager looks to that user's role and its related responsibilities and privileges within the organization. Role templates and role hierarchies in the *Credential* manager software aid the *Credential* manager in this job. An individual's role may change; hence, credentials may be reissued with different labels, or may even be revoked altogether for an individual who has left the organization.

When a smart card is used, a random number can be stored on the smart card. This has the effect of tying the user and the smart card to the credentials file. In this case the credentials file cannot be decrypted without the smart card.

When biometrics are used, the biometric reading offers another piece of information from which to derive the credentials file encryption key if the reading can be reproduced exactly each time. This further ties the user to the credentials file. However, if the biometric reading cannot be reproduced exactly each time it must be compared to a stored baseline template for variance calculation purposes. In this case the template is not used in the encryption of the credentials. Instead, it is used for authentication and is carried in the credentials where it is used to compare to each biometric reading.

The credentials file carries an expiration date. Beyond this date the credentials file is useless. Each CKM encrypted object can contain a time stamp in its header. Objects encrypted by others beyond the expiration date of the credentials cannot be decrypted. The maximum time-out value -the time from credentials issuance to credential expiration -is set by the Policy Manager. A *Credential* manager may further restrict the time-out but cannot extend the time-out value when issuing credentials to a user. To use CKM after credentials have expired, a user must have credentials reissued by that user's *Credential* manager.

THE CKM HEADER

Effective movement of information or content can be viewed as an encapsulated envelope such as a file for which an electronic version is shared. Some form of a header will be needed to locate a point of presence. The header can be viewed as another container associated with the encrypted information and containing selected encryption framework components. In the context of a fundamental CKM, these components are found in the combiner. The encryption process for content may be viewed as a collection of components beyond those needed by the CKM combiner to further manage and control the flow of information.

To protect content or access to content with an encryption framework, it is necessary to include a header with the encrypted content. A message may be viewed as content. Protecting content may be limited to data signing or include data encryption or a combination of both signing and encryption.

For CKM, a resultant secure content envelope that includes the header can be used as adjudicating access to content without compromising the content and while the content is in transit or at rest. The header must contain sufficient information to provide information associated with the data and to provide information for a selected encryption schema.

Content encryption or object encryption results in the encryption process becoming inherent with the life cycle of the data. Data is created, encrypted, transmitted, and stored (or at rest) – at each stage of this cycle, the data is encrypted. While the data remains encrypted, the associated header can be used to make decisions at a router, intrusion detection device, or similar network security entities. Data spillage from a security device becomes a non-concern since the data is always encrypted. The header can also be used for a pre-search of content for data-at-rest scenarios or a verification process before opening the content for plaintext review. As a function contained in the header, components (or attributes) associated with the intent of readership or peerage to content can be viewed in different capacities – as examples: 1) a component that is associated with the operating system such that invoking the component can result in denying access to the print function; or 2) a component that is associated with providing peerage to the content through a credential or other naming convention. The header may also define additional data separation that could result from a single secure content envelope or message being nested into another secure message to effectively have the content encryption enforce banking policies such as privacy, liability and enforcement of the business process sequence that may affect one message separate of the second message, yet the overall content application is the same. The header becomes the vehicle for defining the components associated with the information in regards to how the information is to be controlled. For CKM, pointers in the header may refer to the key split components; a pointer is not used for the encrypted random key split. The pointers offer additional integrity to the CKM encryption framework.

Protection Associated with the Header

The integrity and security of the header includes separate encryption methods to protect the random split key and to protect the balance of the components. The header may be signed with a digital signature. The header may also be protected from modification through the use of an HMAC computed on the header contents. Splits are never seen nor transferred as part of a transaction.

Summary

To implement an encryption schema, the fundamental challenge is: how to maintain a secret associated with that schema so that vastly dispersed usage can be accomplished. In sum, how a balance is reached is to distribute the secret in the form of a key to many users. The mechanism is a framework for a key management architecture.

CKM is built on a key management framework: the architecture is designed for a large scale enterprise solution or for even personal use. The CKM schema results in the dynamic creation of a message encrypting key on-the-fly. There are multiple inherent integrity checks; and the schema has been certified and is contained in national and international standards.