Proposal to provide sufficient interoperable key roles for financial applications

Administrative information

Proposal created by: Jon Geater, Thales E-Security

Contributors: Jon Geater, Thales E-Security
Todd Arnold, IBM
Chris Dunn, Safenet

Proposal Version: 1.6

Date: 2009-07-15

Purpose

To a first approximation, in financial crypto all keys are DES keys of some length or another, and policy is defined at the application layer (eg "VerifyPIN" rather than "encrypt" or "decrypt") so basic crypto-level access control does not work: at that level (algorithm, mechanism) all keys are effectively the same. In order to prevent abuse of keys an application layer system of key usage called 'key roles' is employed. By attaching a role to a key it is possible to differentiate it from other keys preventing a PIN validation key from being used as a key encryption key, for example.

Concerns have been raised (most notably by Todd Arnold of IBM, KMIP liaison to ANSI X9F) that the set of financial key roles currently defined in KMIP is insufficient to cover all the needs of financial applications in the field. Augmenting KMIP to cover all the needs of the financial community would be difficult: the world of financial crypto is a complex one with a significant history of regionalization, customization and vendor 'tweaks', making it complex, divergent, and confounding interoperability. However the financial community, under ANSI X9, has defined an interoperable key block for secure key exchange which captures the set of key roles for keys that are commonly transferred between implementations.

While all vendors of financial HSMs/APIs have larger sets of roles with improved security properties or flexibility the workload implications of explicitly supporting all these specializations in the normative document are many. Given that KMIP is an interoperability specification it is deemed sufficient to include only those roles deemed relevant for interchange by the subject matter experts in X9.

Proposal

This proposal completely replaces specification lines 358 (section 3.6) and 1575 (section 9.1.3.2.15). In addition it adds to the definition of Cryptographic Usage Mask in sections 3.12 and 9.1.3.3.1 to support the new roles definitions.

Change 1: Line 358 change to:

Key Role definitions are chosen to match those defined in ANSI X9 "TR-31 2005 Interoperable Secure Key Exchange Key Block Specification for Symmetric Algorithms" and are defined as follows:

BDK	Base Derivation Key (ANSI X9.24 DUKPT key derivation)
CVK	Card Verification Key (CVV/signature strip number validation)
DEK	Data Encryption Key (General Data Encryption)
MKAC	EMV/chip card Master Key: Application Cryptograms
MKSMC	EMV/chip card Master Key: Secure Messaging for Confidentiality
MKSMI	EMV/chip card Master Key: Secure Messaging for Integrity
MKDAC	EMV/chip card Master Key: Data Authentication Code
MKDN	EMV/chip card Master Key: Dynamic Numbers
MKCP	EMV/chip card Master Key: Card Personalization
MKOTH	EMV/chip card Master Key: Other
KEK	Key Encryption or Wrapping Key
MAC16609	ISO16609 MAC Algorithm 1
MAC97971	ISO9797-1 MAC Algorithm 1
MAC97972	ISO9797-1 MAC Algorithm 2
MAC97973	ISO9797-1 MAC Algorithm 3 *(Note this is commonly known as X9.19 Retail MAC)*
MAC97974	ISO9797-1 MAC Algorithm 4
MAC97975	ISO9797-1 MAC Algorithm 5
ZPK	PIN Block Encryption Key
PVKIBM	PIN Verification Key, IBM 3624 Algorithm
PVKPVV	PIN Verification Key, VISA PVV Algorithm
PVKOTH	PIN Verification Key, Other Algorithm

Accredited Standards Committee X9, Inc. - Financial Industry Standards (www.x9.org) contributed to the above table. Key role names and descriptions are derived from material in the Accredited Standards Committee X9, Inc's Technical Report "TR-31 2005 Interoperable Secure Key Exchange Key Block Specification for Symmetric Algorithms" and used with the permission of Accredited Standards Committee X9, Inc. in an effort to improve interoperability between X9 standards and OASIS KMIP. The complete ANSI X9 TR-31 is available at www.x9.org.

Change 2: Line 1575 change to:

9.1.3.2.15 Role Type Enumeration

Role Type	
Name	**Value**
BDK	00000001
CVK	00000002
DEK	00000003
MKAC	00000004
MKSMC	00000005
MKSMI	00000006
MKDAC	00000007
MKDN	00000008
MKCP	00000009
MKOTH	0000000A
KEK	0000000B
MAC16609	0000000C
MAC97971	0000000D
MAC97972	0000000E
MAC97973	0000000F
MAC97974	00000010
MAC97975	00000011
ZPK	00000012
PVKIBM	00000013
PVKPVV	00000014
PVKOTH	00000015
Extensions	8xxxxxxx

Note that while the set and definitions of key types are chosen to match TR-31 there is no necessity to match binary representations.

Change 3.1: Section 3.12 modify as:

[…]

~~448~~ • CRL Sign
449 • Generate Cryptogram
450 • Validate Cryptogram
451 • Translate Encrypt
452 • Translate Decrypt
453 • Translate Wrap
454 • Translate Unwrap

~~449~~ 455 This list takes into consideration values which may appear in the Key Usage extension in an
[…]

Change 3.2: Line 1591 change to:

9.1.3.3.1 Cryptographic Usage Mask Values

Cryptographic Usage Mask	
Name	**Value**
Sign	00000001
Verify	00000002
Encrypt	00000004
Decrypt	00000008
Wrap	00000010
Unwrap	00000020
Export	00000040
MAC	00000080
Derive Key	00000100
Content Commitment (Non Repudiation)	00000200
Key Agreement	00000400
Certificate Sign	00000800
CRL Sign	00001000
MAC Verify	00002000
Generate Cryptogram	00004000
Validate Cryptogram	00008000
Translate Encrypt	00010000
Translate Decrypt	00020000
Translate Wrap	00040000
Translate Unwrap	00080000
Extensions	XXX00000

Change 3.3: Usage guide explanation of asymmetric concepts with symmetric keys

Asymmetric concepts with symmetric keys

The 'Cryptographic Usage' field is intended to adequately support asymmetric concepts using symmetric keys. This is fairly common practice in established crypto systems: the MAC is an example of an operation where a single symmetric key is used at both ends, but policy dictates that one end can only generate cryptographic tokens using this key (the MAC) and the other end can only verify tokens. Security of the system fails if the verifying end is able to use the key to perform generate operations.

In these cases it is not sufficient to describe the usage policy on the keys in terms of cryptographic primitives like "encrypt" vs. "decrypt" or "sign" vs. "verify". There are two reasons why this is the case.

- In some of these operations, such as MAC generate and verify, the same cryptographic primitive is used in both of the complementary operations. MAC generation involves computing and returning the MAC, while MAC verification involves computing that same MAC and comparing it to a supplied value to determine if they are the same. Thus, both generation and verification use the "encrypt" operation and the two usages cannot be distinguished by considering only "encrypt" vs. "decrypt".

- Some operations which require separate key types use the same fundamental cryptographic primitives. For example, encryption of data, encryption of a key, and computation of a MAC all use the fundamental operation "encrypt", but in many applications securely differentiated keys must be used for these three operations. Simply looking for an attribute that permits "encrypt" is not sufficient.

Allowing use of these keys outside of their specialized purposes can compromise security. Instead, specialized application-level permissions are required to control the use of these keys. KMIP provides several pairs of such permissions in the Cryptographic Usage Mask (3.12), such as:

MAC MAC VERIFY	For cryptographic MAC operations. Although it is possible to compose using a series of encrypt calls, the security of the MAC relies on the operation being atomic and specific.
GENERATE CRYPTOGRAM VALIDATE CRYPTOGRAM	For composite cryptogram operations such as financial CVC or ARQC. To specify exactly which cryptogram the key is used for it is also necessary to specify a *role* for the key (see section 3.6 "Cryptographic Parameters" in the normative specification).

TRANSLATE ENCRYPT TRANSLATE DECRYPT TRANSLATE WRAP TRANSLATE UNWRAP	To accommodate secure routing of traffic and data. In many areas that rely on symmetric techniques (notably but not exclusively financial networks), information is sent from place to place encrypted using shared symmetric keys. When encryption keys are changed it is desirable for the change to be an atomic operation, otherwise distinct unwrap-wrap or decrypt-encrypt steps risk leaking the plaintext data in the middle. *TRANSLATE ENCRYPT/DECRYPT* are used for data encipherment. *TRANSLATE WRAP/UNWRAP* are used for key wrapping.

98

99 In order to support asymmetric concepts using symmetric keys in a KMIP system the server
100 implementation needs to be able to differentiate between clients for generate operations and clients for
101 verify operations. As indicated by section 3 ("Attributes") of the normative specification there will be a
102 single key object in the system to which all relevant clients refer, but when a client requests that key the
103 server is able to choose which attributes (permissions) to send with it based on the identity and
104 configured access rights of that specific client. There is thus no need to maintain and synchronize
105 distinct copies of the symmetric key: just a need to define access policy for each client or group of
106 clients.

107 The internal implementation of this feature at the server end is a matter of choice for the vendor:
108 storing multiple key blocks with all necessary combinations of attributes or generating key blocks
109 dynamically are both acceptable approaches.