Project	IEEE 802.16 Broadband Wireless Access Working Group <**http://ieee802.org/16**>
Title	Key Hierarchy Text for PKMv2
Date Submitted	2004-04-11
Source(s)	David Johnston Intel Corporation 2111 NE 25th Ave. Hillsboro 97124 Voice: +1 (503) 264-3855 [mailto:dj.johnston@intel.com]
Re:	IEEE 802.16e
Abstract	Proposal to introduce a key hierarchy to PKMv2
Purpose	To create an well defined hierarchy for the storage and derivation of keys in PKMv2
Notice	This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.
Release	The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE's name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE's sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that this contribution may be made public by IEEE 802.16.
Patent Policy and Procedures	The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures <http://ieee802.org/16/ipr/patents/policy.html>, including the statement "IEEE standards may include the known use of patent(s), including patent applications, provided the IEEE receives assurance from the patent holder or applicant with respect to patents essential for compliance with both mandatory and optional portions of the standard." Early disclosure to the Working Group of patent information that might be relevant to the standard is essential to reduce the possibility for delays in the development process and increase the likelihood that the draft publication will be approved for publication. Please notify the Chair <mailto:chair@wirelessman.org> as early as possible, in written or electronic form, if patented technology (or technology under patent application) might be incorporated into a draft standard being developed within the IEEE 802.16 Working Group. The Chair will disclose this notification via the IEEE 802.16 web site <http://ieee802.org/16/ipr/patents/notices>.

PKMv2 Key Hierarchy

David Johnston, Intel; JunHyuk Song, Samsung; Seokheon Cho, ETRI; Sung-cheol Chang, ETRI;
Seong-Choon Lee, KT; YoungMan Park, KT; Donnie Lee, SKT; Jesse Walker, Intel

PKM version 1 defines a set of symmetric keys and their relationships, but a clear definition of the total key hierarchy is not present. Neither is the context in which the AK is stored. There are vulnerabilities associated with this.

This proposal creates a well-defined set of security contexts and a clear key hierarchy for PKMv2. The approach adapts the PMK1 key material and key deriving methodologies to achieve backward compatibility.

Two keying models are assumed. The first is a model where the master key at each base station is different between different base stations. A full secure network entry is required to establish keying material when a mobile station hands over.

The second model is where pre-authentication is used. The AAA server and the MSS share a master key, called the MK. The PMK is derived from the master key, and the identities of the BS and MSS. Thus there is a different AK per {BS,SS} tuple. The AAA server populates each BS with the appropriate {BS,SS} AK, thus making it known to BS. Since the SS can derive the PMK, it does not need to enter into a key establishment procedure with the target BS during a handover.

There is a PAK that may be derived from a certificated RSA exchange.

There is an AK, this is equivalent to the AK in the existing PKMv1 protocol. The AK is derived from the PMK and/or the PAK, whichever is available.

A generic PRF Dot16KDF is defined to be used to derive keys. The definition of Dot16KDF varies based on the state OMAC/HMAC bit from the authentication policy negotiation.

A 160 bit PAK is generated randomly in the BS and transmitted to the SS, encrypted with RSA during the PKMv2 authorization exchange.

If a PKMv2 authorization exchange takes place, followed by an EAP authentication exchange, the EAP messages may be protected using two keys (EEK – EAP Encryption Key and EIK – EAP Integrity Key) derived from the PAK.

EEK || EIK || KDK = Dot16KDF(PAK, SSID || BSID || "EPK+EIK+KDK", 384)

Here the order of SSID and BSID don't matter, but it does matter that they are included.

MSK is generated in the AAA server and the SS as a result of an EAP based authentication exchange when the EAP method used supports key establishment.

The PMK is the first 160 bits of the MSK received from the AAA server. The SS and the BS shall discard the other MSK bits received.

If more keying material is needed for future link ciphers, the keylength of the PMK may be increased.

There are four possibilities for authorization and authentication exchanges during network entry. These possibilities arise from the authorization and authentication stages being individually negotiable. In addition, the EAP method used in an EAP authentication exchange may or may not yield an MK. The following rules for deriving the AK are formulated to accommodate these different cases:

```
If (the authorization exchange has been used yielding a PAK and the EAP authentication exchange has been used, yielding an MK) then
        AK = Dot16KDF(PMK, SSID || BSID || AKID || PAK || "AK", 160);
ElsIf (the authorization exchange has been used yielding PAK and the EAP authentication exchange has been used, but not yielding an MK) then
AK <= PAK
        Elsif (the EAP authentication exchange has been used, yielding an MK) then
                AK <= PMK
                AK = Dot16KDF(PMK, SSID || BSID || AKID || "AK", 160);
Elsif (the authorization exchange has been used) then
                AK <= PAK
Else
                No security mode is selected
End if
```

The keys used for the OMAC tuple in management messages are derived from the AK as follows:

OMAC_KEY_U || OMAC_KEY_D || KEK <= Dot16KDF(AK, SSID || BSID || "OMAC_KEYS+KEK", 384)

The keys used for the HMAC tuple in management messages and the KEK used in the TEK exchange are derived from the AK, the SSID and the BSID as follows:

HMAC_KEY_U || HMAC_KEY_D || KEK <= Dot16KDF(AK, SSID || BSID || "HMAC_KEYS+KEK", 448)

The TEK is generated as a random number in the BS and is encrypted using the corresponding TEK encryption algorithm (e.g. AES_KEY_WRAP for SAs with the TEK encryption algorithm identifier in the cryptographic suite is equal to 0x04) keyed with the KEK and transferred between BS and SS in the TEK exchange.

For the purpose of transferring GTEKs. The GKEK encryption key is derived as follows:
GKEKEK (GKEK Encryption Key) <= Dot16KDF(AK, "GKEKEK", 128)

GKEK is encrypted with GKEKEK.

MAK is encrypted with the MAKEK, integrity checked with the MAKIK transferred to the MBS GSA.
The keys within the MBS GSA are the same as for a GSA, however the MAK is used in place of the AK.

MKEK and MTEK rules are as per the unicast key rules, but MAK and MTEK keys are common and known to all members within an MBS group.
MBS_traffic_key = Dot16KDF(MAK, MTEK, 128)

For MBS services, the MBSGSAID is caused to be the same amongst group members, network side mechanisms that are outside the scope of the standard.

Editor Instructions:
[In the appropriate PKMv2 part of of clause 7, insert:]

7.x.x.x Key Hierarchy

The PKMv2 key hierarchy defines what keys are present in the system how keys they are generated.
Since there are two authentication schemes, one based on RSA and one based on EAP, there are two primary sources of keying material.

Keying material is held within associations. There are four types of association: The Authorized Association (AA) that is the top of the security hierarchy, security associations (SA) that maintain keying material for unicast connections, group security associations (GSA) that hold keying material for multicast groups and MBSGSAs which hold keying material for MBS services.

The keys used to protect unicast and multicast traffic are derived from source key material generated by the authentication and authorization processes. The authorization process yields the Primary AK (PAK) and the EAP based authentication process yields the MSK (Master Session Key). Keys used to protect MBS traffic are derived from the MBSAK, which is supplied by means outside the scope of this specification. These keys form the roots of the key hierarchy.

[Insert in the appropriate place…]

7.x.x.x Authorized Association

The AA (Authorized Association) contains the primary keying material of the PKMv2 protocol. The contents are:

The AAID, a 16 bit identifier for the AA. This AAID shall be unique within a BS.
The Primary AK (PAK). A 160 bit key yielded from the authorization exchange. The PAK is only present if the certificated RSA exchange took place, as a result of the authorization policy negotiation.
The Master Session Key (MSK). A 512 bit key yielded from an EAP method following the successful completion of that EAP method during the EAP exchange and the leftmost truncation of its result. The MSK is only present if the EAP exchange took place, as a result of the authorization policy negotiation.
The Primary Master Key (PMK). A 160 bit key. It is the first 160 bits of the MSK.
The AK. A 160 bit authorization key.
The AK Key lifetime. This shall not exceed the MSK lifetime.
The EAP Encryption Key (EEK), a 128 bit key that may be used to protect EAP traffic.
The EAP Integrity Key (EIK), a 128 bit key that may be used to integrity protect EAP traffic.
The uplink OMAC key (OMAC_KEY_U), a 128 bit key used as the keying material for the generation of uplink OMAC digests.
The downlink OMAC key (OMAC_KEY_D), a 128 bit key used as the keying material for the generation of downlink OMAC digests.
The uplink HMAC key (HMAC_KEY_U), a 160 bit key used as the keying material for the generation of uplink HMAC digests.

The downlink HMAC key (HMAC_KEY_D), a 160 bit key used as the keying material for the generation of downlink HMAC digests.
The Group Key Encryption Key Encryption Key (GKEKEK), a 128 bit key used to encrypt keying material transferred into a GSA.
The SAID list. A list of SAs secured under the AA.
The GSA list. A list of GSAs secured under the AA.
The MBSGSA list. A list of MBSGSAs secured under the AA.

7.x.x.x Security Associations

A security association contains keying material that is used to protect unicast connections. The contents of an SA are:

The SAID, a 16 bit identifier for the SA. The SAID shall be unique within a BS.
The KEK, a 128 bit key encryption key, derived from the AK.
The KIK, a 128 key integrity key, derived from the AK.
TEK0 and TEK1, 64 bit (DES)/ 128 bit (AES) traffic encryption keys, generated within the BS and transferred from the BS to the SS using a secure key exchange.
The TEK Lifetimes TEK0 and TEK1, a key aging lifetime value. These lifetimes shall not extend beyond the end of the lifetime of the superior MSK lifetime.
PN0 and PN1, 32 bit packet numbers for use by the link cipher
RxPN0 and RxPN1, 32 bit receive sequence counter, for use by the link cipher.

7.x.x.x Group Security Association

The Group Security Association (GSA) contains keying material used to secure multicast groups. These are defined separately from SAs since GSA offer a lower security bound that unicast security associations, since keying material is shared between all members of the group, allowing any member of the group to forge traffic as if it came from any other member of the group.

The contents of a GSA are:

The Group Key Encryption Key (GKEK). Serves the same function as an SA KEK but for a GSA
The Group Traffic Encryption Keys (GTEK0 and GTEK1). Serves the same function as an SA TEK but for a GSA.

7.x.x.x MBS Group Security Association

The primary keying material in the MBS Group Security Association is the MAK. This serves the same function as the AK in the Authorized Association, however the MAK is provisioned by an external entity, such as an MBS server. The MAK may be common between members of an MBS group.

The contents of an MBSGSA are:

The MAK, a 160 bit MBS AK, serves the same function as the AK but local to the MBSGSA.
The MGTEK, a 128 bit MBS Group Traffic Encryption Key, used indirectly to protect MBS traffic. It is updated more frequently that the MAK.
The MTK, MBS Traffic Key, a 128 bit key used to protect MBS traffic, derived from the MAK and MGTEK.

7.x.x.x. Key Derivation

All PKMv2 key derivations are based on the Dot16KDF algorithm as defined in 7.x.x.x Dot16KDF.

EEK || EIK || KDK = Dot16KDF(PAK, SSID || BSID || “EEK+EIK+KDK”, 384)

MSK is generated in the AAA server and the SS as a result of an EAP based authentication exchange.

PMK = first 160 bits of the MSK

If the authorization exchange has been used yielding a PAK and the EAP authentication exchange has been used, yielding an MSK then the following AK construction shall be used:
AK = Dot16KDF(PMK, SSID || BSID || AAID || KDK || “AK”, 160);

Else if the authorization exchange has been used yielding a PAK and the EAP authentication exchange has been used, but with an EAP method that does not yield key material then the following AK construction shall be used:
AK = Dot16KDF(0, SSID || BSID || AAID || KDK || “AK”, 160);

Else if only the EAP authentication exchange has been used, yielding an MSK then the following AK construction shall be used:
AK = Dot16KDF(PMK, SSID || BSID || AAID || “AK”, 160);

Else if only the authorization exchange has been used then
AK = Dot16KDF(0, SSID || BSID || AAID || KDK || “AK”, 160);

Keys used by the authentication tuples are generated as follows:
OMAC_KEY_U || OMAC_KEY_D || KEK <= Dot16KDF(AK, SSID || BSID || “OMAC_KEYS+KEK”, 384)
HMAC_KEY_U || HMAC_KEY_D || KEK <= Dot16KDF(AK, SSID || BSID || “HMAC_KEYS+KEK”, 448)

The TEK is passed from the BS to the SS in a TEK exchange. It is encrypted using AES_KEY_WRAP, using KEK as the key.
GKEKEK (GKEK Encryption Key) <= Dot16KDF(AK, “GKEKEK”, 128)
GKEK is encrypted with GKEKEK in group key transfers

The generation and transport of the MAK (MBS AK) is outside the scope of the 802.16 standard. It is provided through means defined at higher layers. However the keying is used in the link cipher, therefore its existence needs to be defined in layer 2.

The MBS Transport Key (MTK) is used to protect transport data. It is defined as follows:
MTK <= Dot16KDF(MAK, MGTEK || “MTK”, 128)

Figure xx outlines the process to calculate the AK when the authorization process has taken place, but where the EAP based authentication process hasn’t taken place, or the EAP method used has not yielded an MSK:

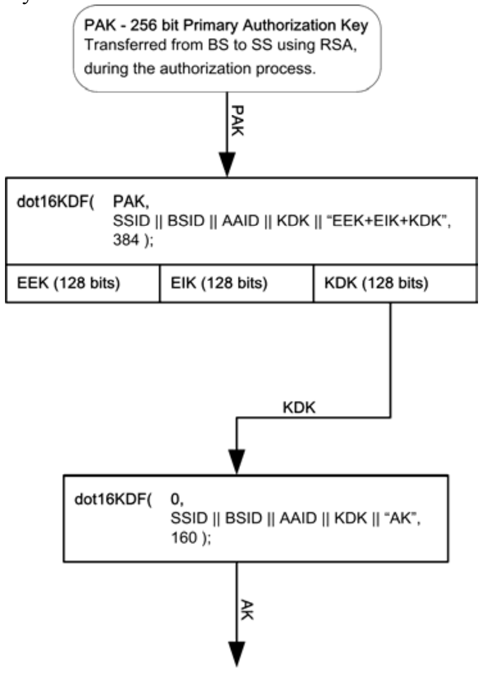

PAK - 256 bit Primary Authorization Key
Transferred from BS to SS using RSA, during the authorization process.
PAK
dot16KDF(PAK, SSID || BSID || AAID || KDK || “EEK+EIK+KDK”, 384);
EEK (128 bits)
EIK (128 bits)
KDK (128 bits)
KDK
dot16KDF(0, SSID || BSID || AAID || KDK || “AK”, 160);
AK

Figure xx outlines the process to calculate the AK when both the authorization exchange has taken place, yielding a PAK and the EAP based authentication exchange has taken place, yielding an MSK:

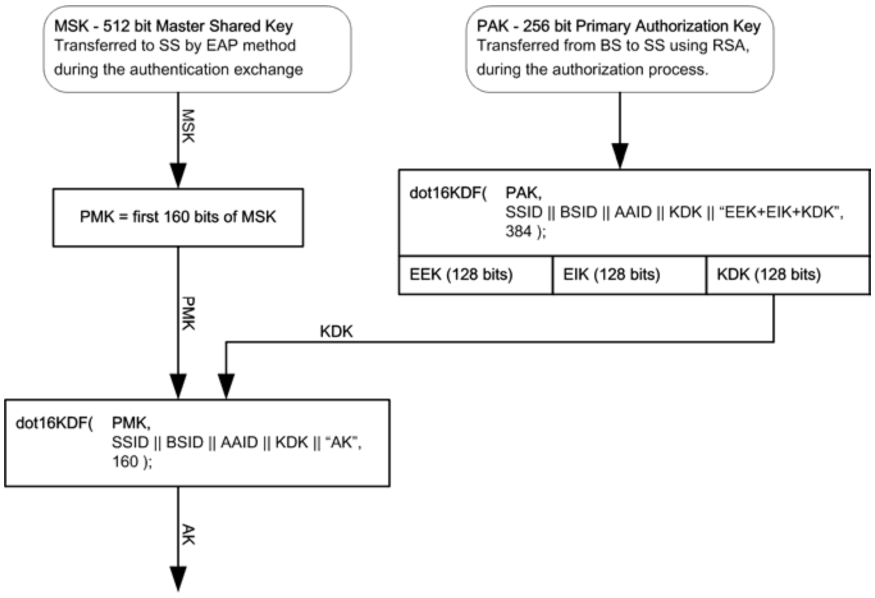

MSK - 512 bit Master Shared Key
Transferred to SS by EAP method
during the authentication exchange
PAK - 256 bit Primary Authorization Key
Transferred from BS to SS using RSA,
during the authorization process.
MSK
PMK = first 160 bits of MSK
dot16KDF(PAK,
SSID || BSID || AAID || KDK || "EEK+EIK+KDK",
384);
EEK (128 bits)
EIK (128 bits)
KDK (128 bits)
PMK
KDK
dot16KDF(PMK,
SSID || BSID || AAID || KDK || "AK",
160);
AK

Figure xx outlines the process to calculate the AK when only the EAP based authentication exchange has taken place, yielding an MSK:

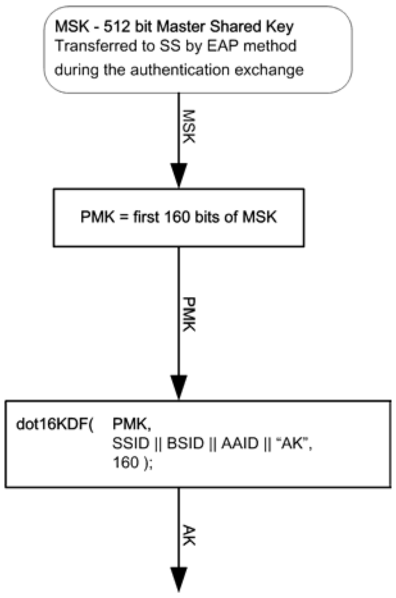

MSK - 512 bit Master Shared Key
Transferred to SS by EAP method
during the authentication exchange
MSK
PMK = first 160 bits of MSK
PMK
dot16KDF(PMK,
SSID || BSID || AAID || "AK",
160);
AK

Figure xx outlines the unicast key hierarchy starting from the AK:

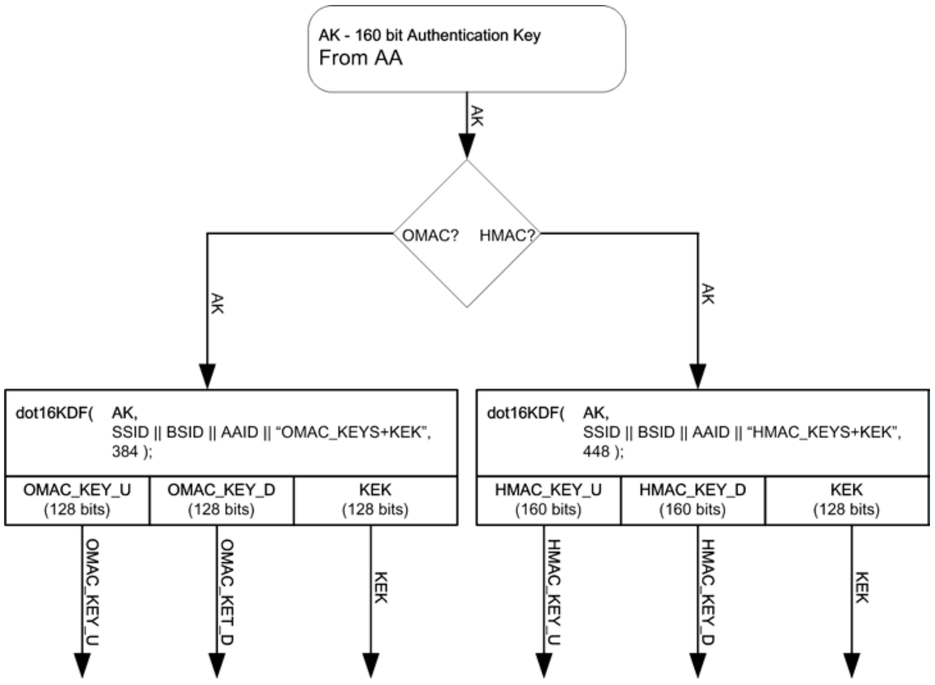
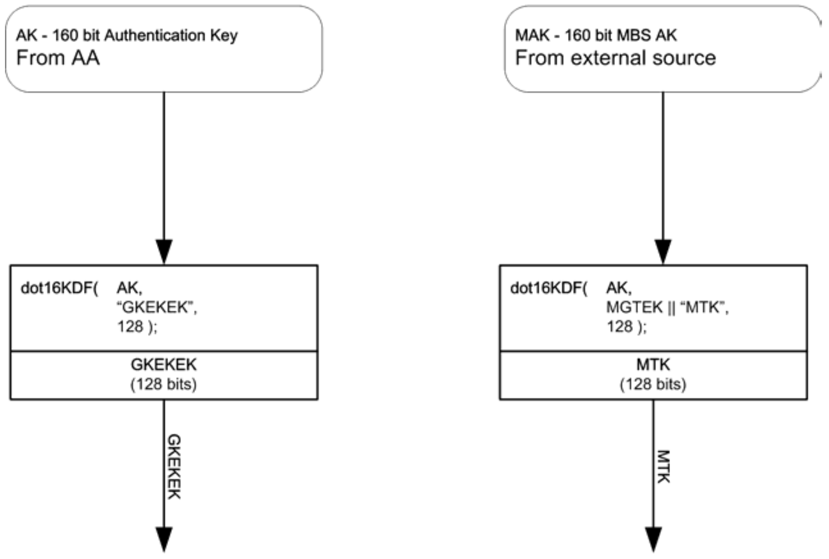


Figure xx outlines the group and MBS key hierarchies starting from the AK and MAK respectively:


AK - 160 bit Authentication Key
From AA
MAK - 160 bit MBS AK
From external source
dot16KDF(AK, "GKEKEK", 128);
GKEKEK (128 bits)
dot16KDF(AK, MGTEK || "MTK", 128);
MTK (128 bits)
GKEKEK
MTK

[Insert under the cryptographic algorithms section]

7.x.x.x.x Dot16KDF

The Dot16KDF algorithm is a CTR mode construction that may be used to derive an arbitrary amount of keying material from source keying material.

In the case that the HMAC/OMAC setting in the authentication policy bits is set to OMAC, the algorithm is defined as:

Dot16KDF(key, astring, keylength) is
{

```
  result = null;
  Kin = Truncate (key, 128);
  for (i = 0;i <= int((keylength-1)/128); i++)
          {
            result <= result | Truncate (OMAC(Kin, i | astring | keylength), 128);
          }
  return Truncate (result, keylength);
}
```

In the case that the HMAC/OMAC setting in the authentication policy bits is set to HMAC, the algorithm is defined as:

```
Dot16KDF(key, astring, keylength) is
{
  Kin = Truncate (key, 160);
  return Truncate (SHA-1(astring | key), keylength);
}
```

In both cases, Truncate(*a*, *n*) returns the first *n* bits of string *a* and destroys any remaining bits.

The key is a cryptographic key that is used by the underlying digest algorithm (SHA-1 or OMAC-AES). 'astring' is an octet string used to alter the output of the algorithm. 'keylength' is used to determine the length of key material to generate and is used in the digest input data to prevent extension attacks. Truncate(x,y) is the rightmost y bits of a value x only if y <= x.

[Change 11.9.3 as below to add in support for the AES Key Wrap algorithm, unless it is fixed in the .16-2004 correigendum]

11.9.3 TEK

Description: This attribute contains a quantity that is a TEK key, encrypted with a KEK derived from the AK.

Type	Length	Value (String)
8	8, 16 or 24	Encrypted TEK

When the TEK encryption algorithm identifier in the SA is 0x01, the length shall be 8 and the TEK shall be encrypted with 3DES in EDE mode according to the procedure defined in 7.5.2.1.

When the TEK encryption algorithm identifier in the SA is 0x03, the length shall be 16 and the TEK shall be encrypted with AES in ECB mode according to the procedure in 7.5.2.3

When the TEK encryption algorithm identifier in the SA is 0x04, the length shall be 24 and the TEK shall be encrypted with the AES Key Wrap algorithm according to the procedure in 7.5.2.4

[Insert an entry for AES Key Wrap into table 375a]

Table 375a—TEK encryption algorithm identifiers

Value	Description
0	Reserved
1	3-DES EDE with 128-bit key
2	RSA with 1024-bit key
3	ECB mode AES with 128-bit key
4	AES Key Wrap with 128 bit key
~~3~~5-255	Reserved