Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**	
Title	**Restructuring of Clause 7 of 802.16e**	
Date Submitted	**[The date the document is contributed, in the format 2003-08-21]**	
Source(s)	David Johnston Intel Corporation Hillsboro, OR USA	Voice: 502 264 3855 Fax: mailto:dj.johnston@ieee.org
Re:	Comments 8007 and 8008 in sponsor recirculation of 802.16	
Abstract	A proposal, with specific text to restructure clause 7, making the editorial directions clear, accomodating both PKMv1 and PKMv2 and making the section numbers correct with respect to IEEE 802.16-2004 and corrigendum 1 so that it is possible to correctly amend the base document with the text into 802.16e.	
Purpose	Consider and adopt this text into the 802.16e draft as a resolution of comments 8007 and 8008.	
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>.	

Restructuring of Clause 7 of 802.16e

David Johnston
Intel Corporation(optional)

The following text represents a complete replacement of the text of clause 7 in 802.16e. This is proposed as a resolution of comments 8007 and 8008 by the Editor, calling for clause 7 to be restructured since the document is inconsistent with the structure of the base document.

It is a restructuring that yields a number of benefits

a) The numbering lines up with the numbering of the base document
b) The editorial instructions are unambiguous
c) All the crypto algorithms are grouped in the same place under 7.5
d) Some errors, such as duplication of text in two places, empty subclauses and PKMv2 sections under the PKMv1 section are corrected by removing the duplication or relocating the incorrectly located text.
e) There is no text in this version for which an editorial instruction cannot be unambiguously identified
f) The document is consistent with corrigendum 1
g) A large number of textual references has been converted to autonumbered cross references, such that the numbering remains correct over future amendements to the document.

This text has been developed from the Framemaker sources of the draft, so there will be no transcription errors and it is a simple task for the editor to include this text.

It is proposed that the text of clause 7 in 802.16e be replaced in its entirety with the text below.

7. Security sublayer

[Change Clause 7 as indicated:]

The ~~S~~security sublayer provides subscribers with privacy, authentication or confidentiality[a] across the ~~fixed~~ broadband wireless network. It does this by applying cryptographic transforms to MPDUs carried across ~~between~~ connections between SS and BS.

In addition, the security sublayer provides operators with strong protection from theft of service. The BS protects against unauthorized access to these data transport services by securing ~~enforcing encryption of~~ the associated service flows across the network. The s~~S~~ecurity sublayer employs an authenticated client/server key management protocol in which the BS, the server, controls distribution of keying material to client SS. Additionally, the basic security ~~transport connection security~~ mechanisms are strengthened by adding digital-certificate-based SS device-authentication to ~~its~~the key management protocol.

[Change 7.1 as indicated:]

7.1 Architecture

Security has two component protocols as follows:

a) An encapsulation protocol for securing~~encrypting~~ packet data across the ~~fixed~~ BWA network. This protocol defines (1) a set of supported *cryptographic suites*, i.e., pairings of data encryption and authentication algorithms, and (2) the rules for applying those algorithms to a MAC PDU payload.
b) A key management protocol (PKM) providing the secure distribution of keying data from the BS to the SS. Through this key management protocol, SS and BS synchronize keying data; in addition, the BS uses the protocol to enforce conditional access to network services.

The protocol stack for the security components of the system are shown in Figure 130j.

[Insert new figure:]

[a]In security parlance, confidentiality = privacy + authenticity

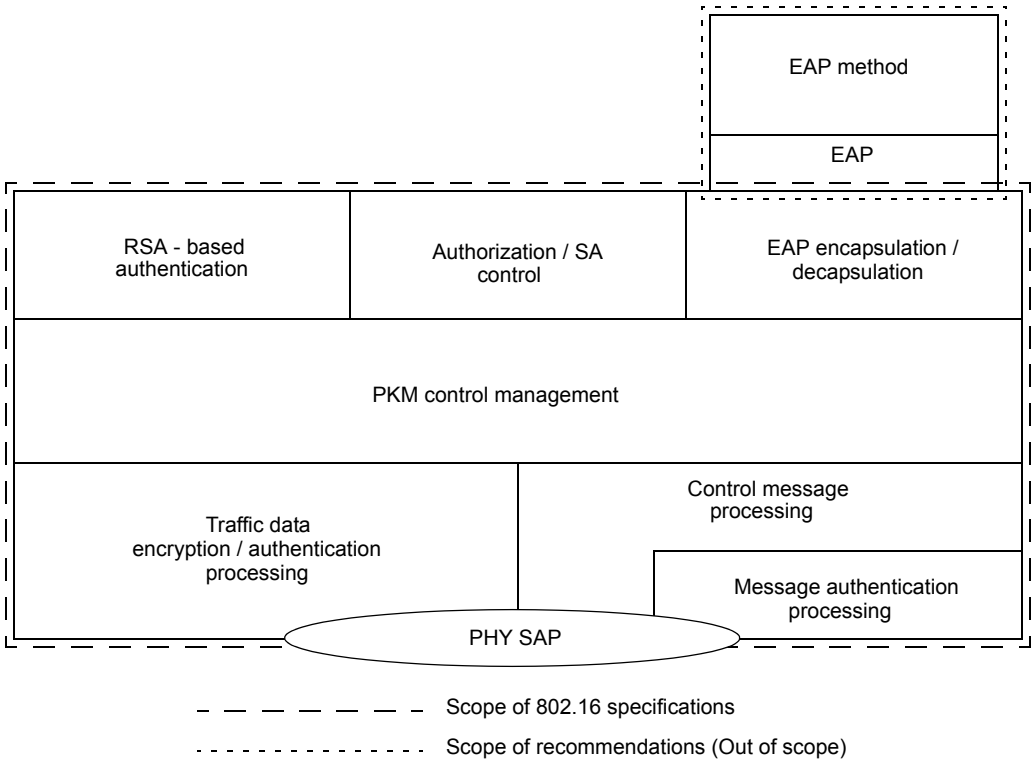

Figure 130j—Security sublayer

[Change 7.1.1 as indicated:]

7.1.1 ~~Packet data encryption~~ Secure encapsulation of MPDUs

Encryption services are defined as a set of capabilities within the MAC Security Sublayer. MAC Header information specific to encryption is allocated in the generic MAC header format.

Encryption is applied to the MAC PDU payload when required by the selected ciphersuite; the generic MAC header is not encrypted. All MAC management messages ~~described n subclause 6.4.2.3~~ shall be sent in the clear to facilitate registration, ranging, and normal operation of the MAC.

[Replace 7.1.2 with the following:]

7.1.2 Key management protocol

The PKM protocol allows for both mutual authentication and unilateral authentication (e.g., where the BS authenticates SS, but not vice versa). It also supports periodic reauthentication/reauthorization and key refresh. The key management protocol uses either EAP [IETF RFC 3748], or X.509 digital certificates [IETF RFC 3280] together with RSA public-key encryption algorithm [PKCS #1] or a sequence starting with RSA authentication and followed by EAP authentication. It uses strong encryption algorithms to perform key exchanges between an SS and BS.

The PKM's authentication protocol establishes a shared secret (called an Authorization Key (AK)) between the SS and the BS. The shared secret is then used to secure subsequent PKM exchanges of TEKs. This two-

tiered mechanism for key distribution permits refreshing of TEKs without incurring the overhead of computation-intensive operations.

A BS authenticates a client SS during the initial authorization exchange. Each SS presents its credentials, which will be a unique X.509 digital certificate issued by the SS's manufacturer (in the case of RSA authentication) or a operator-specified credential (in the case of EAP-based authentication).

The BS associates an SS's authenticated identity to a paying subscriber, and hence to the data services that subscriber is authorized to access. Thus, with the AK exchange, the BS determines the authenticated identity of a client SS and the services (i.e., specific TEKs) the SS is authorized to access.

Since the BS authenticates the SS, it may protect against an attacker employing a cloned SS, masquerading as a legitimate subscriber's SS.

The traffic-key management portion of the PKM protocol adheres to a client/server model, where the SS (a PKM "client") requests keying material, and the BS (a PKM "server") responds to those requests, ensuring that individual SS clients receive only keying material for which they are authorized.

The PKM protocol uses MAC management messaging, i.e., PKM-REQ and PKM-RSP messages defined in 6.3.2.3. The PKM protocol is defined in detail in 7.2.

[Replace 7.1.3 with the following:]

7.1.3 Authentication protocol

An SS uses the PKM protocol to obtain authorization and traffic keying material from the BS, and to support periodic reauthorization and key refresh.

PKM supports two distinct authentication protocol mechanisms:

- RSA protocol [PKCS #1 v2.1 with SHA-1(FIPS 186-2)] (support is mandatory in PKMv1, support is optional in PKMv2)

- Extensible Authentication Protocol (optional unless specifically required)

[Insert new subclause 7.1.3.1:]

7.1.3.1 PKM RSA authentication

The PKM RSA authentication protocol uses X.509 digital certificates [IETF RFC 3280], the RSA public-key encryption algorithm [PKCS #1] that binds public RSA encryption keys to MAC addresses of SSs.

A BS authenticates a client SS during the initial authorization exchange. Each SS carries a unique X.509 digital certificate issued by the SS's manufacturer. The digital certificate contains the SS's Public Key and SS MAC address. When requesting an AK, an SS presents its digital certificate to the BS. The BS verifies the digital certificate, and then uses the verified Public Key to encrypt an AK, which the BS then sends back to the requesting SS.

All SSs using RSA authentication shall have factory-installed RSA private/public key pairs or provide an internal algorithm to generate such key pairs dynamically. If an SS relies on an internal algorithm to generate its RSA key pair, the SS shall generate the key pair prior to its first AK exchange, described in 7.2.1. All SSs with factory-installed RSA key pairs shall also have factory-installed X.509 certificates. All SSs that rely on internal algorithms to generate an RSA key pair shall support a mechanism for installing a manufacturer-issued X.509 certificate following key generation.

[Insert new subclause 7.1.3.2:]

7.1.3.2 PKM EAP authentication

PKM EAP Authentication uses Extensible Authentication Protocol [IETF RFC 3748] in conjunction with an operator-selected EAP Method (e.g. EAP-TLS [IETF RFC 2716]). The EAP method will use a particular kind of credential – such as an X.509 certificate in the case of EAP-TLS, or a Subscriber Identity Module in the case of EAP-SIM.

The particular credentials and EAP methods that are to be used are outside of the scope of this specification. However, the EAP method selected should fulfill the "mandatory criteria" listed in section 2.2 of RFC 4017. Use of an EAP method not meeting these criteria may lead to security vulnerabilities.

During re-authentication, the EAP transfer messages are protected with an HMAC/CMAC tuple. The BS and SS must discard unprotected EAP transfer messages, or EAP transfer messages with invalid HMAC/ CMAC digests during re-authentication.

7.1.4 Mapping of connections to SAs

[No change to existing text in 7.1.4]

7.1.5 Cryptographic Suite

[No change to existing text in 7.1.5]

[Explanatory note : Section 7.2 in IEEE 802-16-2004+Cor1 describes the PKM protocol. In the document as amended by 802.16e, PKM has been renamed PKMv1 and a newer protocol PKMv2 is also defined. This necessitates the renumbering of the subclause numbers in 7.2 to accomodate both protocols. The new subclause numbers are called out explicitly and the text to place under those subclause numbers are incorporated by reference (E.G. 7.2.1.4) to the original subclauses of the IEEE-2004+Cor1 document, except where changes are made and the text is included in standard delta form (E.G. 7.2.1.7.5) . All other new text is below in section 7.2.]

[Replace 7.2 and all its subclauses with 7.2 and following subclauses as indicated:]

7.2 PKM protocol

There are two Privacy Key Management Protocols supported in 802.16e. PKM version 1 and PKMv2 with more enhanced features such as new key hierarchy, AES-CMAC, AES-key-wraps, and MBS.

7.2.1 PKM Version 1

7.2.1.1 Security Associations

A *Security Association* (SA) is the set of security information a BS and one or more of its client SSs share in order to support secure communications across the IEEE Std 802.16 network. Three types of SAs are defined: *Primary, Static, and Dynamic*. Each SS establishes a Pprimary Ssecurity association during the SS initialization process. Static SAs are provisioned within the BS. Dynamic SAs are established and eliminated, on the fly, in response to the initiation and termination of specific service flows. Both Static and Dynamic SAs may be shared by multiple SSs.

An SA's shared information shall include the Cryptographic Suite employed within the SA. The shared information may include TEKs and Initialization Vectors. The exact content of the SA is dependent on the SA's Cryptographic Suite.

SAs are identified using SAIDs.

Each SS shall establish an exclusive Primary SA with its BS. The SAID of any SS's Primary SA shall be equal to the Basic CID of that SS.

Using the PKM protocol, an SS requests from its BS an SA's keying material. The BS shall ensure that each client SS only has access to the SAs it is authorized to access.

An SA's keying material [e.g., Data Encryption Standard (DES) key and CBC Initialization Vector] has a limited lifetime. When the BS delivers SA keying material to an SS, it also provides the SS with that material's remaining lifetime. It is the responsibility of the SS to request new keying material from the BS before the set of keying material that the SS currently holds expires at the BS. Should the current keying material expire before a new set of keying material is received, the SS shall perform network entry as described in 6.3.9.

In certain Cryptographic Suites, key lifetime may be limited by the exhaustion rate of a number space [e.g. the PN (Packet Number) in AES-CCM mode]. In this case, the key ends either at the expiry of the key lifetime or the exhaustion of the number space, which ever is earliest. Note that in this case, security is not determined by the key lifetime.

7.2.1.3 SS authorization and AK exchange overview

SS authorization, controlled by the Authorization state machine, is the process of the BS authenticating a client SS's identity:

a) The BS and SS establishing a shared AK by RSA, from which a key encryption key (KEK) and message authentication keys are derived.

b) The BS providing the authenticated SS with the identities (i.e., the SAIDs) and properties of primary and static SAs the SS is authorized to obtain keying information for.

After achieving initial authorization, an SS periodically reauthorize with the BS; reauthorization is also managed by the SS's Authorization state machine. TEK state machines manage the refreshing of TEKs.

7.2.1.3.1 Authorization via RSA authentication protocol

An SS begins authorization by sending an Authentication Information message to its BS. The Authentication Information message contains the SS manufacturer's X.509 certificate, issued by the manufacturer itself or by an external authority. The Authentication Information message is strictly informative; i.e., the BS may choose to ignore it. However, it does provide a mechanism for a BS to learn the manufacturer certificates of its client SS.

The SS sends an Authorization Request message to its BS immediately after sending the Authentication Information message. This is a request for an AK, as well as for the SAIDs identifying any Static Security SAs the SS is authorized to participate in. The Authorization Request includes:

a) A manufacturer-issued X.509 certificate.

b) A description of the cryptographic algorithms the requesting SS supports; an SS's cryptographic capabilities are presented to the BS as a list of cryptographic suite identifiers, each indicating a particular pairing of packet data encryption and packet data authentication algorithms the SS supports.

c) The SS's Basic CID. The Basic CID is the first static CID the BS assigns to an SS during initial ranging—the primary SAID is equal to the Basic CID.

In response to an Authorization Request message, a BS validates the requesting SS's identity, determines the encryption algorithm and protocol support it shares with the SS, activates an AK for the SS, encrypts it with the SS's public key, and sends it back to the SS in an Authorization Reply message. The authorization reply includes:

a) An AK encrypted with the SS's public key.

b) A 4-bit key sequence number, used to distinguish between successive generations of AKs

c) A key lifetime

d) The identities (i.e., the SAIDs) and properties of the single primary and zero or more static SAs the SS is authorized to obtain keying information for.

While the Authorization Reply shall identify Static SAs in addition to the Primary SA whose SAID matches the requesting SS's Basic CID, the Authorization Reply shall not identify any Dynamic SAs.

The BS, in responding to an SS's Authorization Request, shall determine whether the requesting SS, whose identity can be verified via the X.509 digital certificate, is authorized for basic unicast services, and what additional statically provisioned services (i.e., Static SAIDs) the SS's user has subscribed for. Note that the protected services a BS makes available to a client SS can depend upon the particular cryptographic suites SS and BS share support for.

An SS shall periodically refresh its AK by reissuing an Authorization Request to the BS. Reauthorization is identical to authorization with the exception that the SS does not send Authentication Information messages during reauthorization cycles. Subclause 7.2.1.6's description of the authorization state machine clearly indicates when Authentication Information messages are sent.

To avoid service interruptions during reauthorization, successive generations of the SS's AKs have overlapping lifetimes. Both SS and BS shall be able to support up to two simultaneously active AKs during these transition periods. The operation of the Authorization state machine's Authorization Request scheduling algorithm, combined with the BS's regimen for updating and using a client SS's AKs (see 7.4), ensures that the SS can refresh.

7.2.1.4 TEK exchange overview

[Relocate text of 7.2.2 to here and renumber subclause 7.2.1.4]

7.2.1.4.1 TEK exchange overview for PMP topology

[Relocate text of 7.2.2.1 to here and renumber subclause 7.2.1.4.1]

7.2.1.4.2 TEK exchange overview for Mesh mode

[Relocate text of 7.2.2.2 to here and renumber subclause 7.2.1.4.2]

7.2.1.5 Security capabilities selection

[Relocate text of 7.2.3 to here and renumber subclause 7.2.1.5]

7.2.1.6 Authorization state machine

[Relocate text of 7.2.4 to here and renumber subclause 7.2.1.6]

7.2.1.6.1 States

[Relocate text of 7.2.4.1 to here and renumber subclause 7.2.1.6.1]

7.2.1.6.2 Messages

[Relocate text of 7.2.4.2 to here and renumber as subclause 7.2.1.6.2]

7.2.1.6.3 Events

[Relocate text of 7.2.4.3 to here and renumber as subclause 7.2.1.6.3]

7.2.1.6.4 Parameters

[Relocate text of 7.2.4.4 to here and renumber as subclause 7.2.1.6.4]

7.2.1.6.5 Actions

[Relocate text of 7.2.4.5 to here and renumber as subclause 7.2.1.6.5]

7.2.1.7 TEK state machine

[Relocate text of 7.2.5 here and renumber as subclause 7.2.1.7]

7.2.1.7.1 States

[Relocate text of 7.2.5.1 here and renumber as subclause 7.2.1.7.1]

7.2.1.7.2 Messages

[Relocate text of 7.2.5.2 here and renumber as subclause 7.2.1.7.2]

7.2.1.7.3 Events

[Relocate text of 7.2.5.3 here and renumber as subclause 7.2.1.7.3]

7.2.1.7.4 Parameters

[Relocate text of 7.2.5.4 here and renumber as subclause 7.2.1.7.4]

7.2.1.7.5 Actions

[Relocate text of 7.2.5.5 here and renumber as subclause 7.2.1.7.5]

7.2.2 PKM Version 2

7.2.2.1 TEK exchange overview for PMP topology

If SS and BS decide "No authorization" as their authorization policy, SS and BS shall perform neither SA-TEK handshake nor Key Request/Key Reply handshake. In this case, target SAID value which may be included in DSA-REQ/RSP messages shall be Null SAID.

Upon achieving authorization, an SS starts a separate TEK state machine for each of the SAIDs identified in the Authorization Reply or PKMv2 SA-TEK-RSP message, if data traffic encryption is provisioned for one or more service flows. Each TEK state machine operating within the SS is responsible for managing the keying material associated with its respective SAID. TEK state machines periodically send Key Request messages to the BS, requesting a refresh of keying material for their respective SAIDs.

The BS responds to a Key Request with a Key Reply message, containing the BS's active keying material for a specific SAID.

TEKs and KEKs may be either 64 bits or 128 bits long. SAs employing any ciphersuite with a basic block size of 128 bits shall use 128-bit TEKs and KEKs. Otherwise 64-bit TEKs and KEKs shall be used. The name TEK-64 is used to denote a 64-bit TEK and TEK-128 is used to denote a 128-bit TEK. Similarly, KEK-64 is used to denote a 64-bit KEK and KEK-128 is used to denote a 128-bit KEK.

For SAs using a ciphersuite employing DES-CBC, the TEK in the Key Reply is triple DES (3-DES) (encrypt-decrypt-encrypt or EDE mode) encrypted, using a two-key, 3-DES KEK derived from the AK.

For SAs using a ciphersuite employing 128 bits keys, such as AES-CCM mode, the TEK in the key Reply is AES encrypted using a 128-bit key derived from the AK and a 128-bit block size.

Note that at all times the BS maintains two diversity sets of keying material per SAID. The lifetimes of the two generations overlap such that each generation becomes active halfway through the life of it predecessor and expires halfway through the life of its successor. A BS includes in its Key Replies both of an SAID's active generations of keying material.

For SAs using a ciphersuite employing CBC mode encryption the Key Reply provides the requesting SS, in addition to the TEK and CBC initialization vector, the remaining lifetime of each of the two sets of keying material. For SAs using a ciphersuite employing AES-CCM mode, the Key Reply provides the requesting SS, in addition to the TEK, the remaining lifetime of each of the two sets of keying material. The receiving SS uses these remaining lifetimes to estimate when the BS will invalidate a particular TEK, and therefore when to schedule future Key Requests such that the SS requests and receives new keying material before the BS expires the keying material the SS currently holds. For AES-CCM mode, when more than half the available PN numbers in the 31-bit PN number space are exhausted, the SS shall schedule a future Key Request in the same fashion as if the key lifetime was approaching expiry. The operation of the TEK state machine's Key Request scheduling algorithm, combined with the BS's regimen for updating and using an SAID's keying material (see 7.4), ensures that the SS will be able to continually exchange encrypted traffic with the BS.

A TEK state machine remains active as long as

a) the SS is authorized to operate in the BS's security domain, i.e., it has a valid AK, and
b) the SS is authorized to participate in that particular SA, i.e., the BS continues to provide fresh keying material during rekey cycles.

MAC PDUs sent on connections that belong to an SA that includes data encryption, shall be encrypted. A MAC PDU received on such connections, with the EC bit not set, shall be discarded.

7.2.2.2 Key derivation

The PKMv2 key hierarchy defines what keys are present in the system and how the keys 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.

The keys used to protect management message integrity and transport the traffic encryption keys are derived from source key material generated by the authentication and authorization processes. The RSA-based authorization process yields the pre-Primary AK (pre-PAK) and the EAP based authentication process yields the MSK. 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.

All PKMv2 key derivations are based on the Dot16KDF algorithm as defined in 7.5.4.6.1.

The MSK is the shared “master key” that is derived by the two sides in the course of executing the EAP inner method. The authentication part of the authorization flow (and the involvement of the generic EAP layer) is now complete.

7.2.2.2.1 RSA-based authorization

When the RSA-based authorization is negotiated as authorization policy, the PKMv2 RSA-Request, the PKMv2 RSA-Reply, the PKMv2 RSA-Reject, and the PKMv2 RSA-Acknowledgement messages are used to share the pre-PAK (Primary Authorization Key).

The pre-PAK is sent by the BS to the SS encrypted with the public key of the SS certificate. Pre-PAK is mainly used to generate the PAK. The optional EIK for transmitting authenticated EAP payload (see 7.2.2.2.2) are also generated from pre-PAK:

EIK | PAK = Dot16KDF(pre-PAK, SS MAC Address | BSID | "EIK+PAK", 320)

PAK will be used to generate the AK (see below) if RSA authorization was used. PAK is 160 bits long.

7.2.2.2.2 EAP authentication

If a RSA mutual authorization took place before the EAP exchange or if the first EAP took place during EAP-in-EAP mode, the EAP messages may be protected using EIK - EAP Integrity Key derived from pre-PAK (see 7.2.2.2.1). EIK is 160 bits long.

The product of the EAP exchange which is transferred to 802.16 layer is the Master Session Key (MSK), which is 512 bits in length. This key is derived (or may be equivalent to the 512-bits Master Session Key (MSK)). This key is known to the AAA server, to the Authenticator* (transferred from AAA server) and to the SS. The SS and the authenticator derive a PMK (Pairwise Master Key) and optional EIK by truncating the MSK to 320 bits.

The PMK derivation from the MSK is as follows:

The PMK and EIK derivation from the MSK during first EAP method is as follows:

EIK | PMK = truncate (MSK, 320)

The PMK2 derivation from the MSK2 during second EAP method is as follows:

PMK2:= truncate(MSK2, 160)

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

After successful EAP based authorization, if the SS or BS negotiates authorization policy as "Authenticated EAP after EAP" mode, the authenticated EAP messages shall carry second EAP message. It shall cryptographically bind previous EAP authentication and following EAP authentication session, while protecting second EAP messages. In order to prevent "man-in-the-middle attack", the first and second EAP method should fulfill the "mandatory criteria" listed in section 2.2 of RFC 4017 such as EAP-PSK, EAP-AKA.

If SS and BS negotiate double EAP mode (a.k.a. Authenticated EAP after EAP), SS and BS perform two rounds of EAP as follows:

1) In order to initiate 1st round EAP of double EAP, SS may send PKMv2 EAP Start message with no attribute.
2) SS and BS shall perform 1st round EAP conversation with PKMv2 EAP Transfer message without HMAC/CMAC Digest.
3) During 1st EAP conversation, if BS has to send EAP-Success, BS shall send EAP payload to SS with PKMv2 EAP Complete message signed by newly generated EIK BS shall re-send the PKMv2 _EAP_Complete message by Second_EAP_Timeout. Total number of sending PKMv2_EAP_Complete message is EAP_Complete_Resend. After SS receives the PKMv2 EAP_Complete message which includes EAP-Success payload, SS can possess EIK and PMK. In this case, SS can validate the message. Otherwise, if SS receives EAP-Failure or can not validate the message, SS fails in authentication. After BS transfers the PKMv2 EAP Complete message to SS, BS activates the Second_EAP_Timeout in order to wait PKMv2 Authenticated EAP Start message. When the timer expires, BS shall regard the authentication as failure.
4) After the successful 1st round EAP, SS shall send PKMv2 EAP Start message signed by EIK to initiates 2nd round EAP conversation. If BS validates the PKMv2 EAP Start message by EIK, BS shall initiate 2nd EAP by sending PKMv2 Authenticated EAP message including EAP-Identity/Request to SS. If BS cannot validate the PKMv2 Authenticated EAP Start message, BS shall regard the authentication as failure.
5) SS and BS shall perform 2nd EAP conversation with PKMv2 Authenticated EAP message signed by EIK.
6) If 2nd round EAP succeeds, both SS and authenticator generate AK from PMK and PMK2. SS and BS shall perform SA-TEK 3way handshake.

After the successful initial authentication, SS and BS shall perform reauthentication by PMK/PMK2 lifetime. In performing reauthentication, SS and BS perform double EAP just like initial authentication. Otherwise, SS and BS can perform EAP once.

When SS and BS perform reauthentication with double EAP also, the following procedure shall be performed as follows:

1) In order to initiate reauthentication, SS may send PKMv2 EAP Start message signed by H/CMAC_KEY_U derived from AK.
2) SS and BS shall use PKMv2 EAP Transfer message to carry 1st round EAP conversation
3) BS shall carry EAP-Success or EAP-Failure message with PKMv2 EAP Complete message signed by AK generated from the previous double EAP.
4) After successful 1st round EAP, SS shall initiate 2nd round EAP by sending PKMv2 EAP Start message signed by H/CMAC_KEY_U generated from AK (previous double EAP generated this key).
5) SS and BS shall perform 2nd round EAP conversation with PKMv2 EAP Transfer message signed by AK which is generated by previous double EAP.
6) SS and BS shall perform SA-TEK 3way handshake.

When SS and BS perform reauthentication with double EAP, SS and BS can perform EAP once as follows:

1) In order to initiate reauthentication, SS may send PKMv2 EAP Start message signed by H/ CMAC_KEY_U derive d from AK.
2) SS and BS shall use PKMv2 EAP Transfer message to carry 1st round EAP conversation
3) BS shall carry EAP-Success or EAP-Failure message with PKMv2 EAP Transfer instead of sending PKMv2 EAP Complete signed by AK. It means that BS doesn't want to run 2nd round EAP.

7.2.2.2.3 Authorization Key (AK) derivation

The AK will be derived by the BS and the SS from the PMK (from EAP-based authorization procedure) and/ or the PAK (from RSA-based authorization procedure). Note that PAK and/or PMK can be used according to the value of Authorization Policy Support field included in the SBC-REQ/RSP messages.

The exclusive-or (XOR: ⊕) value of PAK and PMK is mainly used to generate the AK.

```
If (PAK and PMK)
        AK <= Dot16KDF (PAK ⊕ PMK, SS MAC Address | BSID | PAK | "AK", 160)
Else If (PMK and PMK2)
        AK <= Dot16KDF (PMK ⊕ PMK2, SS MAC Address | BSID | "AK", 160)
Else
        If (PAK)
                AK <= Dot16KDF (PAK, SS MAC Address | BSID | PAK | "AK", 160)
        Else                        // PMK only
                AK = Dot16KDF(PMK, SS MAC Address | BSID |  "AK", 160);
        Endif
Endif
```

7.2.2.2.4 Key Encryption Key (KEK) derivation

The Key Encryption Key or KEK is derived directly from the AK. The KEK is defined in 7.2.2.2.9 with the HMAC/CMAC definition. It is used to encrypt the TEKs, GKEK and all other keys sent by the BS to SS in unicast message.

7.2.2.2.5 Group Key Encryption Key (GKEK) derivation

GKEK is randomly generated at the BS and transmitted to the SS encrypted with the KEK. There is one GKEK per Group Security Association. GKEK is used to encrypt the GTEKs sent in multicast messages by the BS to the SSs in the same multicast group.

7.2.2.2.6 Traffic Encryption Key (TEK)

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 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.

7.2.2.2.7 Group Traffic Encryption Key (GTEK)

The GTEK is used to encrypt multicast data packets and it is shared among all SSs that belongs to the multicast group. There are 2 GTEKs per GSA.

The GTEK is randomly generated at the BS or at certain network node and is encrypted using same algorithms applied to encryption for TEK and transmitted to the SS in multicast or unicast messages. The GTEK in a PKMv2 Key-Request and PKMv2 Key-Reply messages will be encrypted by the KEK. And, the GTEK in a PKMv2 Group Key Update Command message will be encrypted by the GKEK.

7.2.2.2.8 MBS Traffic Key (MTK)

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 key such as the MTK is used in the link cipher, therefore its existence needs to be defined in layer 2.

The MTK is used to encrypt the MBS traffic data. It is defined as follows:

MTK <= Dot16KDF(MAK, MGTEK | “MTK”, 128)

The MGTEK is the GTEK for the MBS. An SS can get the GTEK by exchanging the PKMv2 Key Request message and the PKMv2 Key Reply message with a BS or by receiving the PKMv2 Group Key Update Command message from a BS. The generation and transport of the GTEK is defined as in section 6.3.2.3.9 and 7.9.

7.2.2.2.9 Message authentication keys (HMAC/CMAC) and KEK derivation

MAC (message authentication code) keys are used to sign management messages in order to validate the authenticity of these messages. The MAC to be used is negotiated at SS Basic Capabilities negotiation.

There is a different key for UL and DL messages. Also, a different message authentication key is generated for a multicast message (this is DL direction only) and for a unicast message.

In general, the message authentication keys used to generate the CMAC value and the HMAC-Digest are derived from the AK.

The keys used for CMAC key and for KEK are as follows:

CMAC_KEY_U | CMAC_KEY_D | KEK <= Dot16KDF(AK, SS MAC Address | BSID | "CMAC_KEYS+KEK", 384)

CMAC_KEY_GD <= Dot16KDF(GKEK, "GROUP CMAC KEY",128) (Used for multicast MAC message such as a PKMv2 Group-Key-Update-Command message)

The keys used for HMAC key and for KEK are as follows:

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

HMAC_KEY_GD <= Dot16KDF(GKEK, "GROUP HMAC KEY", 160) (Used for multicast MAC message such as a PKMv2 Group-Key-Update-Command message)

Exceptionally, the message authentication keys for the HMAC/CMAC Digest included in a PKMv2 Authenticated-EAP-Transfer message are derived from the EIK instead of the AK

The keys used for CMAC key and for KEK are as follows:

CMAC_KEY_U | CMAC_KEY_D <= Dot16KDF(EIK, SS MAC Address | BSID | "CMAC_KEYS ", 256)

The keys used for HMAC key and for KEK are as follows:

HMAC_KEY_U | HMAC_KEY_D <= Dot16KDF(EIK, SS MAC Address | BSID | "HMAC_KEYS", 320)

7.2.2.2.10 Key hierarchy

Figure 130k outlines the process to calculate the AK when the RSA-based 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:

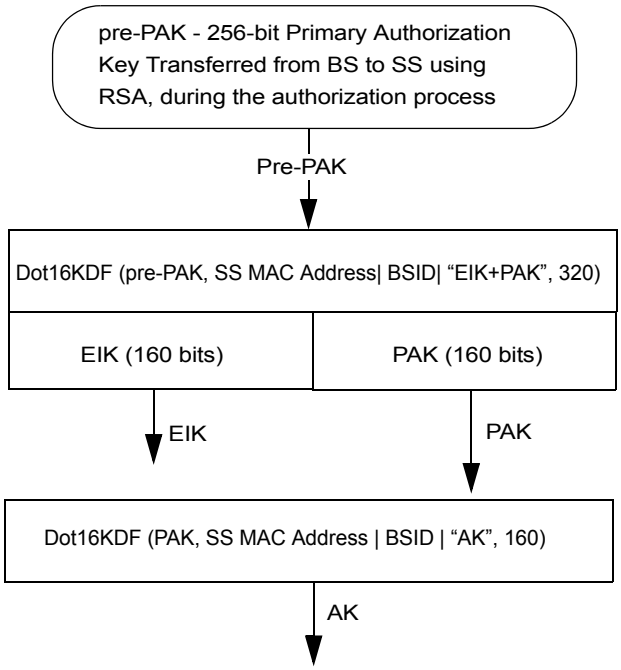

Figure 130k—AK from PAK only (from RSA - based authorization)

Figure 130l outlines the process to calculate the AK when both the RSA-based authorization exchange has taken place, yielding a PAK and the EAP based authentication exchange has taken place, yielding an MSK:

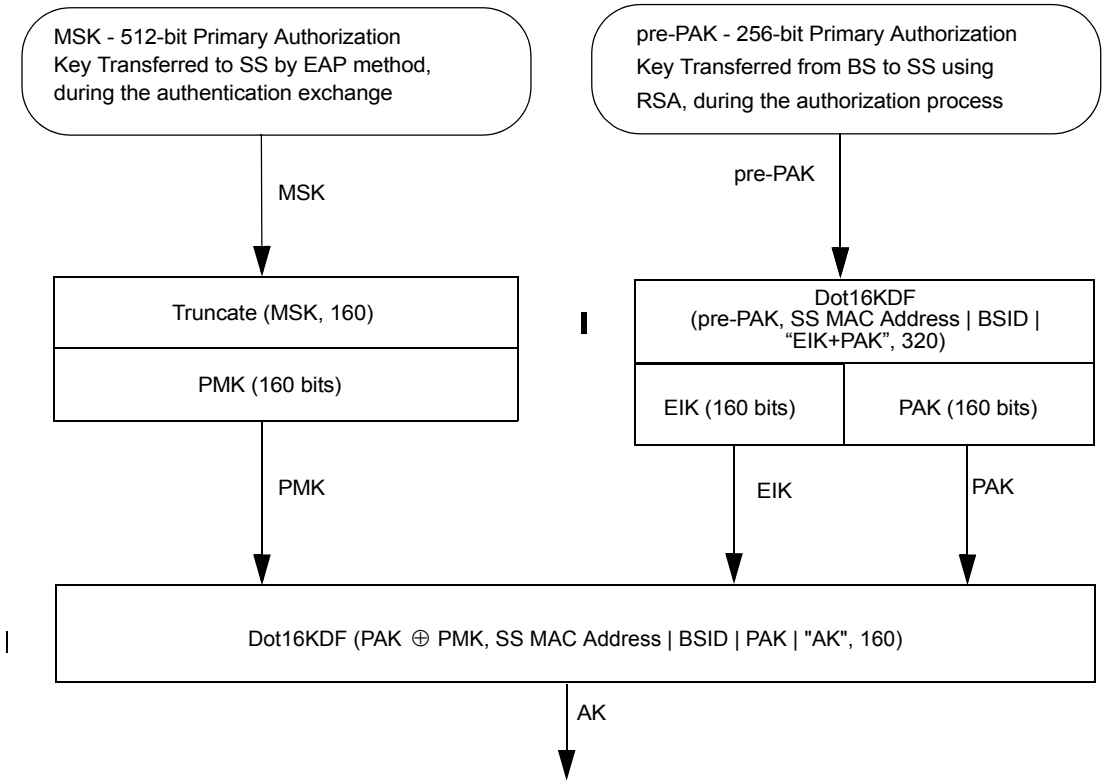

Figure 130l—AK from PAK and PMK (RSA-based and EAP-based authorization)

Figure 130m outlines the process to calculate the AK when only the EAP based authentication exchange has taken place, yielding an MSK:

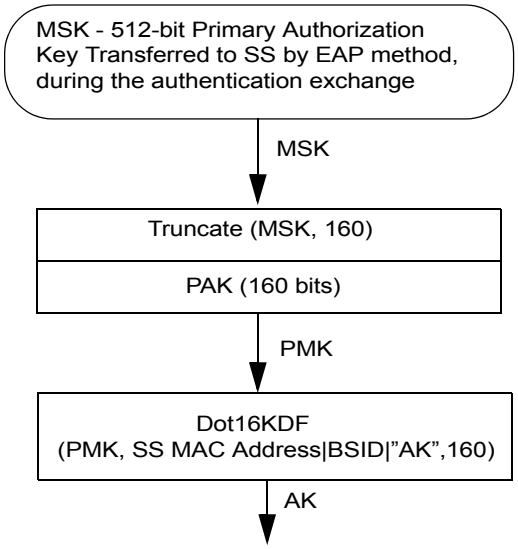

Figure 130m—AK from PMK (from EAP-based authorization)

Figure 130n outlines the process to calculate the AK when EAP in EAP mode authentication exchange has taken place, first EAP yielding EIK and MSK and second EAP yielding MSK2.

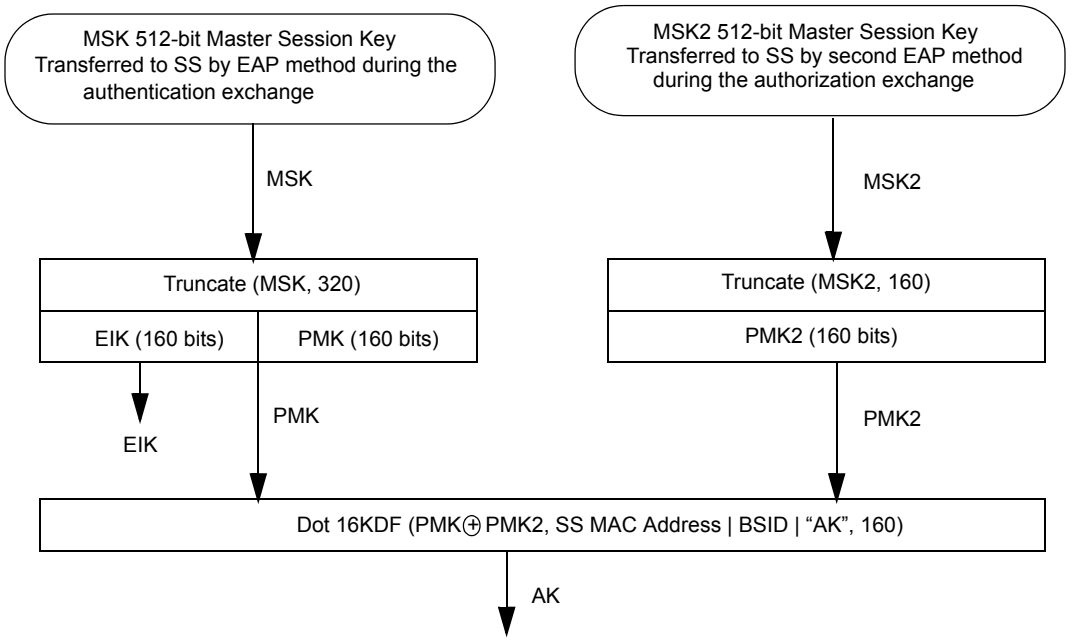

Figure 130n—AK with PMK and PMK2 (EAP-based authorization and Authenticated EAP-based authorization)

Figure 130o outlines the unicast key hierarchy starting from the AK.

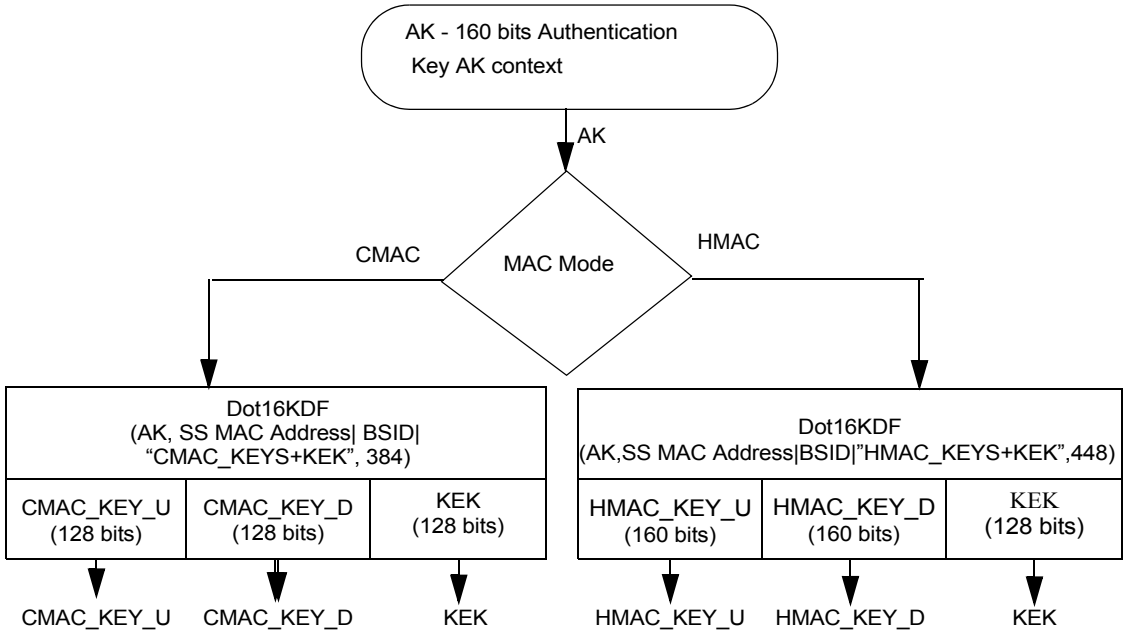

Figure 130o—HMAC/CMAC/KEK derivation from AK

Figure 130p outlines the MBS key hierarchies starting from the MAK:

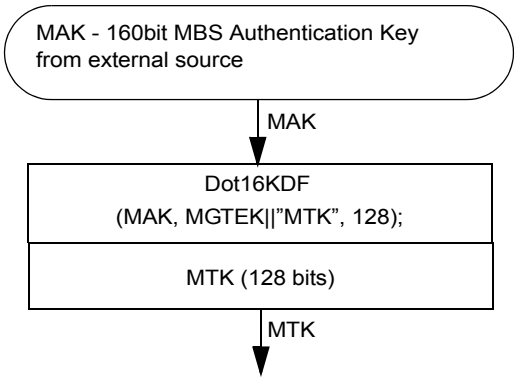

Figure 130p—MTK key derivation from MAK

Figure 130q outlines the process to calculate message authentication keys derived from the EIK. The message authentication keys are used to generate the CMAC value or the HMAC-Digest included in a PKMv2 Authenticated-EAP-Transfer message.

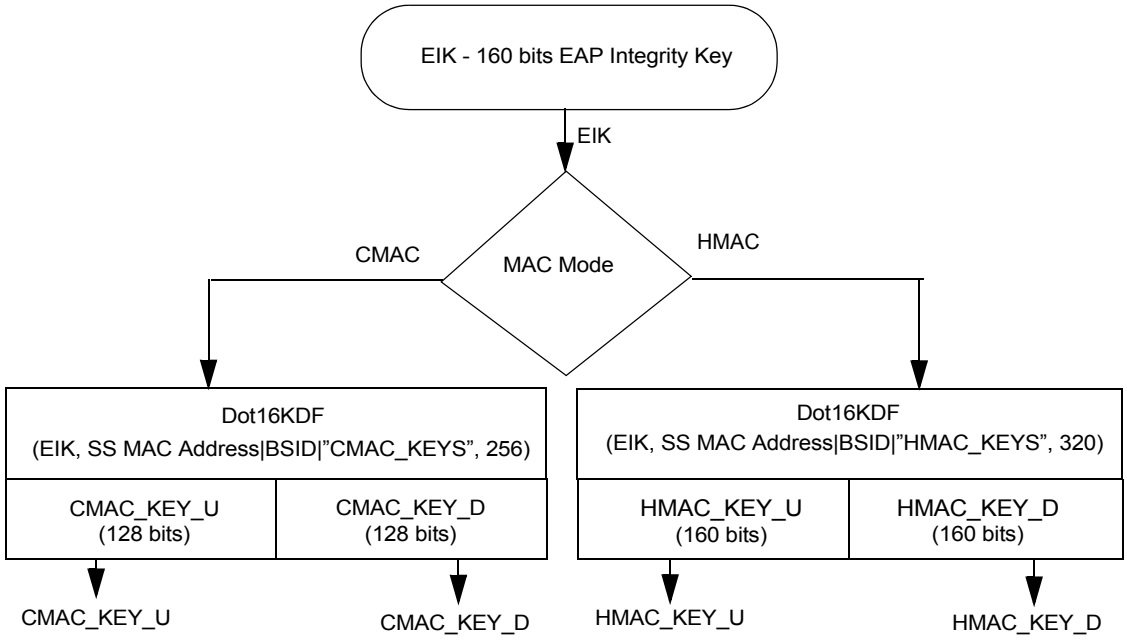

Figure 130q—HMAC/CMAC authentication key derivation from EIK

7.2.2.2.11 Maintenance of PMK and AK

The BS and SS maintain cached PMK and AK as follows:

a) PMK caching

An SS caches a PMK upon successful EAP authentication. An Authenticator caches a PMK upon its receipt via the AAA protocol. Upon caching a new PMK for a particular SS, an Authenticator shall delete any PMK for that SS (as well as all associated AKs).

For the case of reauthentication, deletion of old PMKs at Authenticator and SS is accomplished via the switchover mechanism described in this section using the messages in 6.3.2.3.9.20.

The Authenticator and SS will additionally delete PMKs and/or associated AKs in various situations - including lifetime expiration, reauthentication, and reclamation of memory resources, or as the result of other mechanisms beyond the scope of this specification.

In the case of re-authentication, the older PMK and its AKs shall be deleted by the SS after verifying the HMAC or CMAC of the PKMv2 SA-TEK challenge message and the BS after verifying the HMAC/CMAC of the PKMv2 SA-TEK request message.

b) AK activation and deactivation

Successful completion of the 3-way SA-TEK handshake causes the activation of all the AKs associated with the new PMK (i.e. all AKs on BSs associated with the current authenticator will be active).

If the packet counter belonging to a short HMAC or a CMAC key reaches its maximum value, the associated AK becomes permanently deactivated.
The BS and SS must maintain the AK context (i.e. replay counters etc.) as long as they retain the AK.

7.2.2.2.12 PKMv2 PMK and AK switching methods

Once the PKMv2 SA-TEK 3-way handshake begins, the BS and SS shall use the new AK matching the new PMK context for the 3 way handshake messages. Other messages shall continue to use the old AK until the 3 way handshake completes successfully. Upon successful completion of the 3 way handshake, all messages shall use the new AK.

The old AK matching the old PMK context may be used for receiving packets before the “frame number” attribute specified in PKMv2 SA-TEK-response message.

7.2.2.3 Associations

Keying material is held within associations. There are three types of association: The 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.

If SS and BS decide “No authorization” as their authorization policy, they don't have any security association. In this case, Null SAID shall be used as the target SAID field in DSA-REQ/RSP messages.

7.2.2.3.1 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.

TEK0 and TEK1, 128-bit 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.

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.

A security association is shared between a SS and a BS or, in case of ongoing MDHO(FBSS) between MS and BSs from Diversity Set.

7.2.2.3.2 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 than 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 Key (GTEK). Served the same function as an SA TEK but for a GSA.

7.2.2.3.3 MBS Group Security Association

The primary keying material in the MBS Group Security Association is the MAK. The MAK is provisioned by an external entity, such as an MBS server. The MAK may be common among 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.

The MGTEK is a random number provisioned by the access network such as a BS as an access network authorization key. It is only used for generating MTK together with MAK.

In MBS Group Security Association, the usage of MGTEK is same as that of GTEK.

Key encryption algorithm and key transport mechanism of GTEK shall be also applied for MGTEK.

7.2.2.4 Security context

The security context is a set of parameters linked to a key in each hierarchy that defines the scope while the key usage is considered to be secure.

Examples of these parameters are key lifetime and counters ensuring the same encryption will not be used more than once. When the context of the key expires, a new key should be obtained to continue working.

The purpose of this section is to define the context that belongs to each key, how it is obtained and the scope of its usage.

7.2.2.4.1 AK context

The PMK key has two phases of lifetime: the first begins at PMK creation and the second begins after validation by the 3-way handshake.

The phases ensure that when the PMK is created it will be defined with the PMK or PAK pre-handshake lifetime and after successful 3-way handshake, this lifetime may be enlarged using the PMK lifetime TLV within the 3-way handshake.

If the cached AK and associated context is lost by either BS or SS, no new AKs can be derived from this PMK on handover.

Cached AKs that were derived from the PMK can continue to be used in HO.

Reauthentication is required to obtain a new PMK so as to derive new AKs.

Table 132a—AK Context in PKMv2

Parameter	Size (bits)	Usage
AK	160	The authorization key, calculated as defined in 7.2.2.2.3
AKID	64	AKID = Dot16KDF(AK, AK SN|SS MAC Address|BSID|"AK", 64) The AK_SN in the Dot16KDF function is an 8-bit number which consists of leading 4 zero bits and appending 4-bit AK_SN in MSB first order.
AK Sequence Number	4	Sequence number of root keys (PAK and PMK) for the AK. This value is the most significant 2-bit of PAK sequence number concatenated with the least significant 2-bit of PMK sequence number. If AK = f (PAK and PMK), then AK SN = PAK SN + PMK SN If AK = f (PAK), then AK SN = PAK SN If AK = f (PMK), then AK SN = PMK SN
AK Lifetime	—	This is the time this key is valid; it is calculated AK lifetime = MIN(PAK lifetime, PMK lifetime) - when this expires, re-authentication is needed.
PMK Sequence Number	4	The sequence number of the PMK that this AK is derived from
HMAC/CMAC_KEY_U	160/128	The key which is used for signing UL management messages
HMAC/CMAC_PN_U	32	Used to avoid UL replay attack on the management connection - when this expires re-authentication is needed
HMAC/CMAC_KEY_D	160/128	The key which is used for signing DL management messages
HMAC/CMAC_PN_D	32	Used to avoid DL reply attack on the management connection - when this expires re-authentication is needed
KEK	160	Used to encrypt transport keys from the BS to the SS
EIK	160	EAP Integrity Key for authenticating Authenticated EAP message.

7.2.2.4.2 GKEK context

The GKEK is the head of the group key hierarchy. There is a separate GKEK for each group (each GSA). This key is randomly generated by the BS and transferred to the SS encrypted with KEK. It is used to encrypt group TEKs (GTEK) when broadcasting them to all SSs. The GKEK context is described in Table 132b:

GKEK or KEK can be used for encrypting MGTEK for MBS GSA.

[Insert new subclause 7.2.2.4.3:]

7.2.2.4.3 PMK context

The PMK context includes all parameters associated with the PMK. This context is created when EAP Authentication completes.

Table 132b—GKEK Context

Parameter	Size (bits)	Usage
GKEK	128	Randomly generated by BS and transmitted to SS under KEK
GKEKID	64	Arrives from BS with GKEK
GKEK lifetime	—	Arrives from BS with GKEK; when this expires a new GKEK should be obtained
HMAC_KEY_G/ CMAC_KEY_G	160 or 128	The key which is used for signing group DL GTEK update messages, calculated by KDF(CMAC_PAD,GKEK)
HMAC_PN_G/ CMAC_PN_G	32	Used to avoid DL replay attack on management. When this expires a new GKEK should be obtained
H/CMAC_KEY_U	160 or128	The key which is used for signing UL management messages.

The PMK context is described in Table 132c.

Table 132c—PMK context

Parameter	Size (bits)	Usage
PMK	160	A key yielded from the EAP-based authentication.
PMK sequence number	4	PMK sequence number, when the EAP-based authorization is achieved and a key is generated. The least significant 2 bits are the sequence counter. And the most significant 2 bits set to 0.

7.2.2.4.4 PAK context

The PAK context includes all parameters associated with the PAK. This context is created when RSA Authentication completes.

The PAK context is described in Table 132d.

7.2.5 TEK state machine

The TEK state machine consists of seven states and eleven events (including receipt of messages) that may trigger state transitions. Like the Authorization state machine, the TEK state machine is presented in both a state flow diagram (Figure 131) and a state transition matrix (Table 134). As was the case for the Authorization state machine, the state transition matrix shall be used as the definitive specification of protocol actions associated with each state transition.

Table 132d—PAK context

Parameter	Size (bits)	Usage
PAK	160	A key yielded from the EAP-based authentication.
PAK Lifetime	4	PAK lifetime, from when the RSA-based authorization is achieved. The value of PAK lifetime is initially set to a default value. The 3-way handshake may subsequently change this value
PAK sequence number	4	PAK sequence number, when the RSA-based authorization is achieved and a key is generated. The most significant 2 bits are the sequence counter. And the least significant 2 bits set to 0.

Shaded states in Figure 131 (Operational, Rekey Wait, Rekey Reauthorize Wait, and M&B Rekey Interim Wait) have valid keying material and encrypted traffic may be sent.

The SAID may be replaced by the GSAID for the multicast service or the broadcast service. And, the TEK may be also replaced by the GTEK for the multicast service or the broadcast service.

The Authorization state machine starts an independent TEK state machine for each of its authorized SAIDs. As mentioned in 7.2.2, the BS maintains two active TEKs per SAID.

For the unicast service, the BS includes in its Key Replies both of these TEKs, along with their remaining lifetimes. The BS encrypts downlink traffic with the older of its two TEKs and decrypts uplink traffic with either the older or newer TEK, depending upon which of the two keys the SS was using at the time. The SS encrypts uplink traffic with the newer of its two TEKs and decrypts downlink traffic with either the older or newer TEK, depending upon which of the two keys the BS was using at the time. See 7.4 for details on SS and BS key usage requirements.

For the multicast service or the broadcast service, the BS may include both of GTEKs in its Key Reply messages, when an SS request traffic keying material. And, the BS may include the newer GTEK in the Key Update Command message, when the BS transmits the new traffic keying material in key push mode. The BS encrypts downlink traffic with current GTEK. The SS decrypts downlink traffic with either the older or newer GTEK, depending upon which of the two keys the BS is using at the time. See 7.9 for details on SS and BS key usage requirements.

Through operation of a TEK state machine, the SS attempts to keep its copies of an SAID's TEKs synchronized with those of its BS. A TEK state machine issues Key Requests to refresh copies of its SAID's keying material soon after the scheduled expiration time of the older of its two TEKs and before the expiration of its newer TEK. To accommodate for SS/BS clock skew and other system processing and transmission delays, the SS schedules its Key Requests a configurable number of seconds before the newer TEK's estimated expiration in the BS. With the receipt of the Key Reply, the SS shall always update its records with the TEK Parameters from both TEKs contained in the Key Reply message. TEK Parameters contained in the two Key Update Command messages for the multicast service or the broadcast service.

.

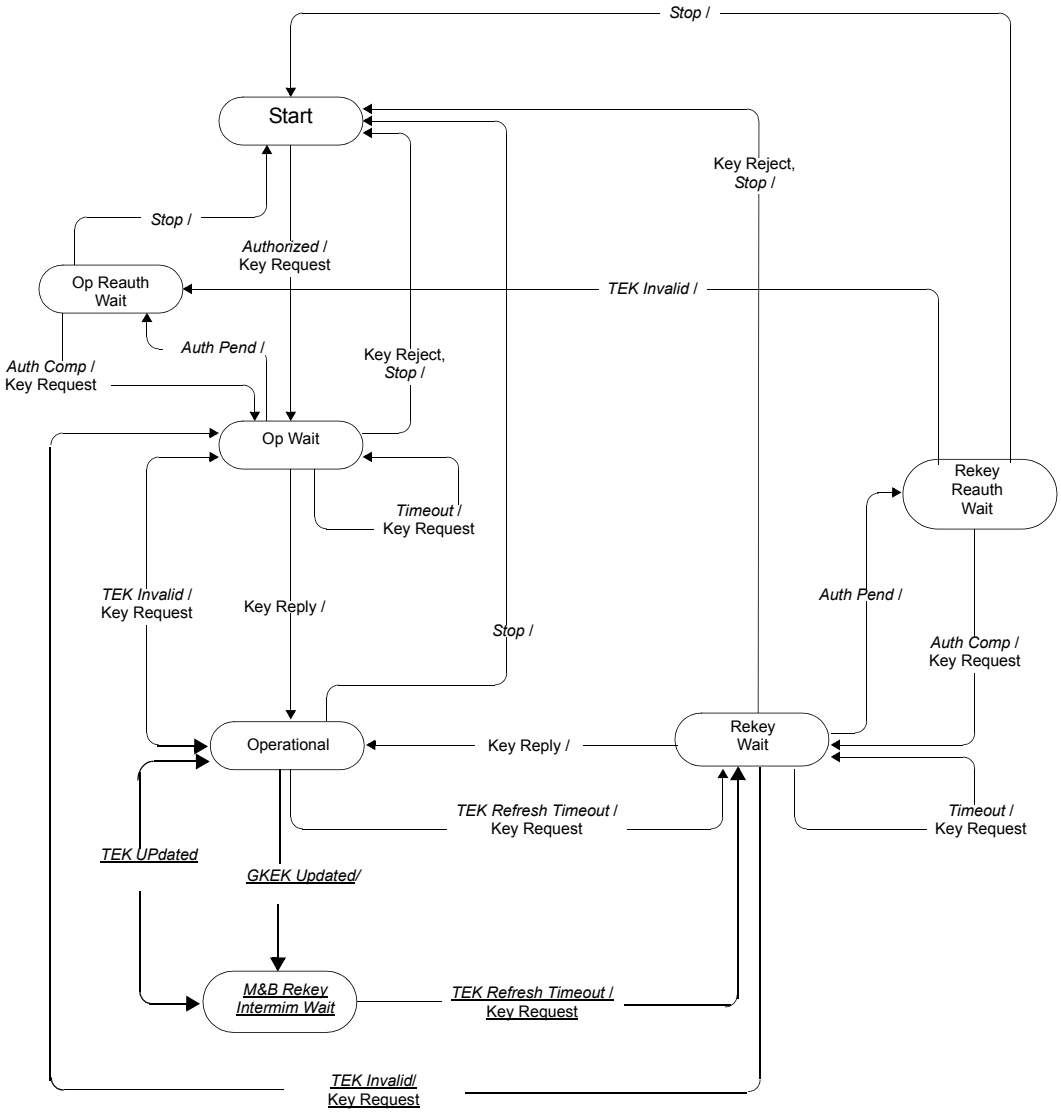

Figure 131—TEK state machine flow diagram

Table 134—TEK FSM state transition matrix

State Event or Rcvd Message	**(A) Start**	**(B) Op Wait**	**(C) Op Reauth Wait**	**(D) Op**	**(E) Rekey Wait**	**(F) Rekey Reauth Wait**	**(G) M&B Rekey Interim Wait**
(1) *Stop*		Start	Start	Start	Start	Start	
(2) *Authorized*	Op Wait						
(3) *Auth Pend*		Op Reauth Wait			Rekey Reauth Wait		
(4) *Auth Comp*			Op Wait			Rekey Wait	
(5) *TEK Invalid*				Op Wait	Op Wait	Op Reauth Wait	
(6) *Timeout*		Op Wait			Rekey Wait		
(7) *TEK Refresh Timeout*				Rekey Wait			Rekey Wait
(8) Key Reply		Opera-tional			Opera-tional		
(9) Key Reject		Start			Start		
(10) GKEK Updated				M&B Rekey Interim Wait			
(11) GTEK Updated							Opera-tional

7.2.5.1 States

Start: This is the initial state of the FSM. No resources are assigned to or used by the FSM in this state—e.g., all timers are off, and no processing is scheduled.

Operational Wait (Op Wait): The TEK state machine has sent its initial request (Key Request) for its SAID's keying material (TEK and CBC initialization vector), and is waiting for a reply from the BS.

Operational Reauthorize Wait (Op Reauth Wait): The wait state the TEK state machine is placed in if it does not have valid keying material while the Authorization state machine is in the middle of a reauthorization cycle.

Operational: The SS has valid keying material for the associated SAID.

Rekey Wait: The TEK Refresh Timer has expired and the SS has requested a key update for this SAID. Note that the newer of its two TEKs has not expired and may still be used for both encrypting and decrypting data traffic.

Rekey Reauthorize Wait (Rekey Reauth Wait): The wait state the TEK state machine is placed in if the TEK state machine has valid traffic keying material, has an outstanding request for the latest keying material, and the Authorization state machine initiates a reauthorization cycle.

M&B Rekey Interim Wait (Multicast & Broadcast Rekey Interim Wait): This state is defined only for the multicast service or the broadcast service. This state is the wait state the TEK state machine is placed in if the TEK state machine has valid traffic keying material and receives the new GKEK from the BS.

7.2.5.2 Messages

Note that the message formats are defined in detail in 6.3.2.3.9.

Key Request: Request a TEK for this SAID. Sent by the SS to the BS and authenticated with keyed message digest. The message authentication key is derived from the AK.

Key Reply: Response from the BS carrying the two diversity sets of traffic keying material for this SAID. Sent by the BS to the SS, it includes the SAID's TEKs, encrypted with a KEK derived from the AK or the GSAID's GTEK, encrypted with a GKEK randomly generated from the BS or the ASA server. The Key Reply message is authenticated with a keyed message digest; the authentication key is derived from the AK.

Key Reject: Response from the BS to the SS to indicate this SAID is no longer valid and no key will be sent. The Key Reject message is authenticated with a keyed message digest; the authentication key is derived from the AK.

TEK Invalid: The BS sends an SS this message if it determines that the SS encrypted an uplink PDU with an invalid TEK, i.e., an SAID's TEK key sequence number, contained within the received PDU's MAC Header, is out of the BS's range of known, valid sequence numbers for that SAID.

Key Update Command: Push a GTEK for this GSAID for the multicast service or the broadcast service. Sent by the BS to the SS and authenticate with keyed message digest. The message authentication key is derived from the AK in the Key Update Command message for the GKEK update mode. The message authentication key is derived from the GKEK in the Key Update Command message for the GTEK update mode.

7.2.5.3 Events

[Insert the following text at the end of 7.2.5.3:]

Stop: Sent by the Authorization FSM to an active (non-START state) TEK FSM to terminate TEK FSM and remove the corresponding SAID's keying material from the SS's key table. See Figure 130k.

Authorized: Sent by the Authorization FSM to a non-active (START state) TEK FSM to notify TEK FSM of successful authorization. See Figure 130k.

Authorization Pending (Auth Pend): Sent by the Authorization FSM to TEK FSM to place TEK FSM in a wait state while Authorization FSM completes re-authorization. See Figure 130k.

Authorization Complete (Auth Comp): Sent by the Authorization FSM to a TEK FSM in the Operational Reauthorize Wait or Rekey Reauthorize Wait states to clear the wait state begun by the prior Authorization Pending event. See Figure 130k.

TEK Invalid: This event is triggered by either an SS's data packet decryption logic or by the receipt of a TEK Invalid message from the BS.

An SS's data packet decryption logic triggers a TEK Invalid event if it recognizes a loss of TEK key synchronization between itself and the encrypting BS. For example, an SAID's TEK key sequence number, contained within the received downlink MAC PDU header, is out of the SS's range of known sequence numbers for that SAID.

A BS sends an SS a TEK Invalid message, triggering a TEK Invalid event within the SS, if the BS's decryption logic recognizes a loss of TEK key synchronization between itself and the SS.

Timeout: A retry timer timeout. Generally, the particular request is retransmitted.

TEK Refresh Timeout: The TEK refresh timer timed out. This timer event signals the TEK state machine to issue a new Key Request in order to refresh its keying material. The refresh timer is set to fire a configurable duration of time (*TEK Grace Time*) before the expiration of the newer TEK the SS currently holds. This is configured via the BS to occur after the scheduled expiration of the older of the two TEKs.

GKEK Updated: This event is triggered when the SS receives the new GKEK through the Key Update Command message for the GKEK update mode.

GTEK Updated: This event is triggered when the SS receives the new GTEK and traffic keying material through the Key Update Command message for the GTEK update mode.

7.2.5.4 Parameters

All configuration parameter values take the default values from Table 343or may be specified in Auth Reply message.

Operational Wait Timeout: Timeout period between sending of Key Request messages from the Op Wait state (see 11.9.19.4).

Rekey Wait Timeout: Timeout period between sending of Key Request messages from the Rekey Wait state (see 11.9.19.5).

TEK Grace Time: Time interval, in seconds, before the estimated expiration of a TEK that the SS starts rekeying for a new TEK. TEK Grace Time takes the default value from Table 343 or may be specified in a configuration setting within the Auth Reply message and is the same across all SAIDs (see 11.9.19.6).

M&B TEK Grace Time (Multicast & Broadcast TEK Grace Time): Time interval, in seconds, before the estimated expiration of an old distributed GTEK.

7.2.5.5 Actions

Actions taken in association with state transitions are listed by <event> (<rcvd message>) --> <state>:

1-B Op Wait (*Stop*) → Start

a) Clear Key Request retry timer
b) Terminate TEK FSM

1-C Op Reauth Wait (*Stop*) → Start

a) Terminate TEK FSM

1-D Operational (*Stop*) → Start

a) Clear TEK refresh timer, which is timer set to go off *"TEK Grace Time"* seconds prior to the TEK's scheduled expiration time
b) Terminate TEK FSM
c) Remove SAID keying material from key table

1-E Rekey Wait (*Stop*) → Start

a) Clear Key Request retry timer
b) Terminate TEK FSM
c) Remove SAID keying material from key table

1-F Rekey Reauth Wait (*Stop*) → Start

a) Terminate TEK FSM
b) Remove SAID keying material from key table

2-A Start (*Authorized*) → Op Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Operational Wait Timeout

3-B Op Wait (*Auth Pend*) → Op Reauth Wait

a) Clear Key Request retry timer

3-E Rekey Wait (*Auth Pend*) → Rekey Reauth Wait

a) Clear Key Request retry timer

4-C Op Reauth Wait (*Auth Comp*) → Op Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Operational Wait Timeout

4-F Rekey Reauth Wait (*Auth Comp*) → Rekey Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Rekey Wait Timeout

5-D Operational (*TEK Invalid*) → Op Wait

a) Clear TEK refresh timer

b) Send Key Request message to BS
c) Set Key Request retry timer to Operational Wait Timeout
d) Remove SAID keying material from key table

5-E Rekey Wait (*TEK Invalid*) → Op Wait

a) Clear TEK refresh timer
b) Send Key Request message to BS
c) Set Key Request retry timer to Operational Wait Timeout
d) Remove SAID keying material from key table

5-F Rekey Reauth Wait (*TEK Invalid*) → Op Reauth Wait

a) Remove SAID keying material from key table

6-B Op Wait (*Timeout*) → Op Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Operational Wait Timeout

6-E Rekey Wait (*Timeout*) → Rekey Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Rekey Wait Timeout

7-D Operational (*TEK Refresh Timeout*) → Rekey Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Rekey Wait Timeout

7-G M&B Rekey Interim Wait (*TEK Refresh Timeout*) -> Rekey Wait

a) Send Key Request message to BS
b) Set Key Request retry timer to Rekey Wait Timeout

8-B Op Wait (Key Reply) → Operational

a) Clear Key Request retry timer
b) Process contents of Key Reply message and incorporate new keying material into key database
c) Set the TEK refresh timer to go off “TEK Grace Time” seconds prior to the newer key’s scheduled expiration

8-E Rekey Wait (Key Reply) → Operational

a) Clear Key Request retry timer
b) Process contents of Key Reply message and incorporate new keying material into key database
c) Set the TEK refresh timer to go off “TEK Grace Time” seconds prior to the newer key’s scheduled expiration

9-B Op Wait (Key Reject) → Start

a) Clear Key Request retry timer
b) Terminate TEK FSM

9-E Rekey Wait (Key Reject) → Start

a) Clear Key Request retry timer
b) Terminate TEK FSM
c) Remove SAID keying material from key table

10-D Operational (*GKEK Updated*) -> M&B Rekey Interim Wait

a) Process contents of Key Update Command message for the GKEK update mode and incorporate new GKEK into key database

11-G M&B Rekey Interim Wait (*GTEK Updated*) -> Operational

a) Clear Key Request retry timer
b) Process contents of Key Update Command message for the GTEK update mode and incorporate new traffic keying material into key database
c) Set the TEK refresh timer to go off "TEK Grace Time" seconds prior to the key's scheduled expiration

[Section 7.3 and its subclauses remain unchanged:]

7.3 Dynamic SA creation and mapping

[Insert text of 7.3 Dynamic SA creation and mapping from IEEE 802.16-2004+cor1, and all its subclauses, here]

7.4 Key Usage

[Insert text of 7.4 Key Usage from IEEE 802.16-2004+cor1, and all its subclauses here]

[Insert new subclause 7.5.1.3:]

7.5.1.3 Data encryption with AES in CTR mode

If the data encryption algorithm identifier in the cryptographic suite of an MBS GSA equals 0x80, data on connections associated with that SA shall use the CTR mode of the US Advanced Encryption Standard (AES) algorithm [NIST Special Publication 800-38A, FIPS 197, RFC 3686] to encrypt the MAC PDU payloads. In MBS, the AES block size and cipher counter block are 128 bits.

7.5.1.3.1 Encrypted MBS PDU payload format

Counter mode requires unique initial counter and key pair across all messages. This section describes the initialization of the 128-bit initial counter, constructed from the 24-bit PHY synchronization field or frame number and a new 8-bit Rollover counter (ROC).

NOTE—When we start to deal with a new PDU we have a new frame number and therefore re-initialize the counter. When the frame number reaches 0x000000 (from 0xFFFFFF), we increment ROC.

The PDU payload for AES-CTR encryption shall be prepended with the 8-bit ROC, i.e., the ROC is the 8 MSBs of the 32-bit nonce. The ROC shall not be encrypted.

Any tuple value of {AES Counter, KEY} shall not be used more than once for the purposes of encrypting a block. SS and BS shall ensure that a new MGTEK is requested and transferred before the ROC reaches 0xFF.

A 32 bit nonce NONCE = n0 | n1 | n2 | n3 (n0 being the MSByte and n3 the LSByte) is made of ROC and 24bits frame number in the following way: n0=ROC and n1, n2, n3 are the byte representation of frame-number in MSB first order. NONCE shall be repeated four times to construct the 128-bit counter block required by the AES-128 cipher. (initial counter = NONCE|NONCE|NONCE|NONCE). When incremented, this 16-byte counter will be treated as a Big Endian number.

This mechanism can reduce per-PDU overhead of transmitting the full counter. At the most 2^32 PDUs can be encrypted with a single MTK.

The plaintext PDU shall be encrypted using the active MBS_Traffic_key (MTK) derived from MAK and MGTEK, according to CTR mode specification. A different 128-bit counter value is used to encrypt each 128-bit block within a PDU.

The processing yields a payload that is 8 bits longer than the plaintext payload.

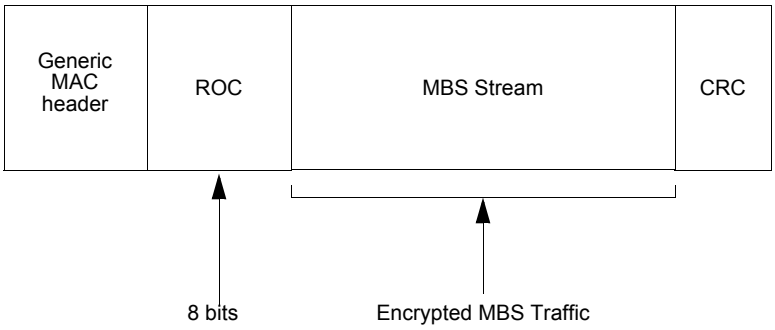

Figure 137a—MBS MAC PDU Ciphertext payload format

7.5.1.4 Data encryption with AES in CBC mode

If the data encryption algorithm identifier in the cryptographic suite of an SA equals 0x03, data on connections associated with that SA shall use the CBC mode of the US Advanced Encryption Standard (AES) algorithm [NIST Special Publication 800-38A, FIPS 197] to encrypt the MAC PDU payloads.

Residual termination block processing shall be used to encrypt the final block of plaintext when the final block is less than the cipher block size. Given a final block having n bits, where n is less than the cipher block size m, the next-to-last ciphertext block shall be divided into two parts. One of the two parts is n bits, the other part is m-n bits. The former will be sent to receiver as the final block ciphertext. Padding the final short block to obtain a complete plaintext block, then encrypt it with AES algorithm in CBC mode. The encryption and decryption procedure is depicted in Figure 137b.

In the special case when the payload portion of the MAC PDU is less than the cipher block size, the most significant n bits of the generated CBC-IV, corresponding to the number of bits of the payload, shall be XORed with the n bits of the payload to generate the short cipher block.

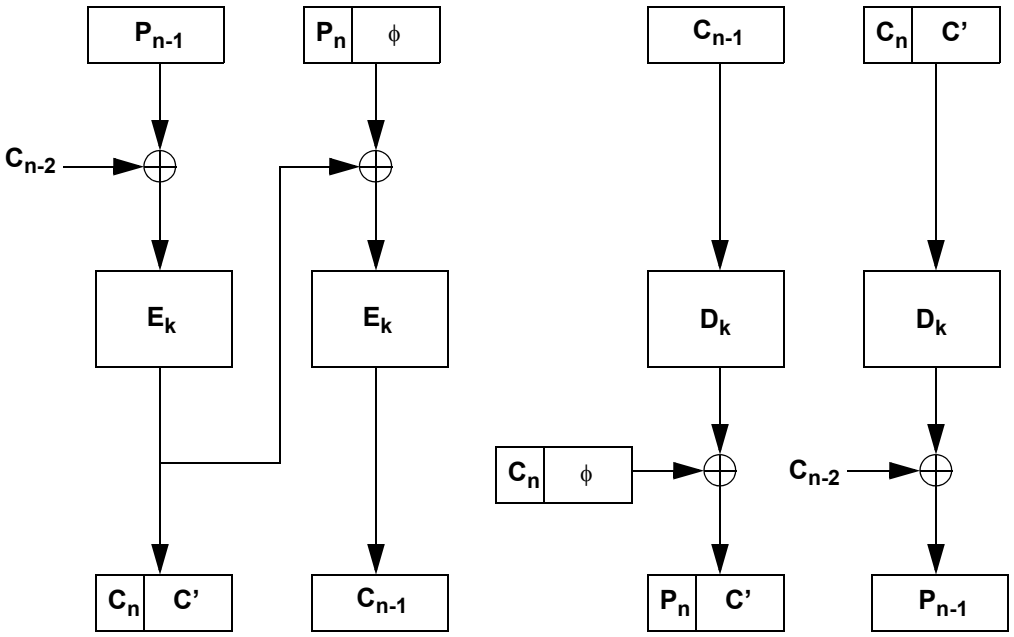

Figure 137b—Residual termination block processing with CTS

7.5.1.4.1 CBC IV generation

The Zero Hit Counter is initialized into zero when the Key Reply message is received, and updates whenever either the PHY Frame number is zero or MAC PDU is received in a frame. The Zero Hit Counter increases by one if the previous PHY Frame number is equal to or greater than the current PHY Frame number.

The CBC IV is generated as the result of the AES block ciphering algorithm with the key of TEK. Its plain text for the CBC IV generation is calculated with the exclusive-or (XOR) of (1) the CBC IV parameter value included in the TEK keying information, and (2) the 128-bits content which is a concatenation of the 48-bit MAC PDU header, the 32-bit PHY Synchronization value of the MAP that a data transmission occurs, and the XOR value of the 48-bit SS MAC address and the Zero Hit Counter.

The CBC IV shall be updated every MAC PDUs.

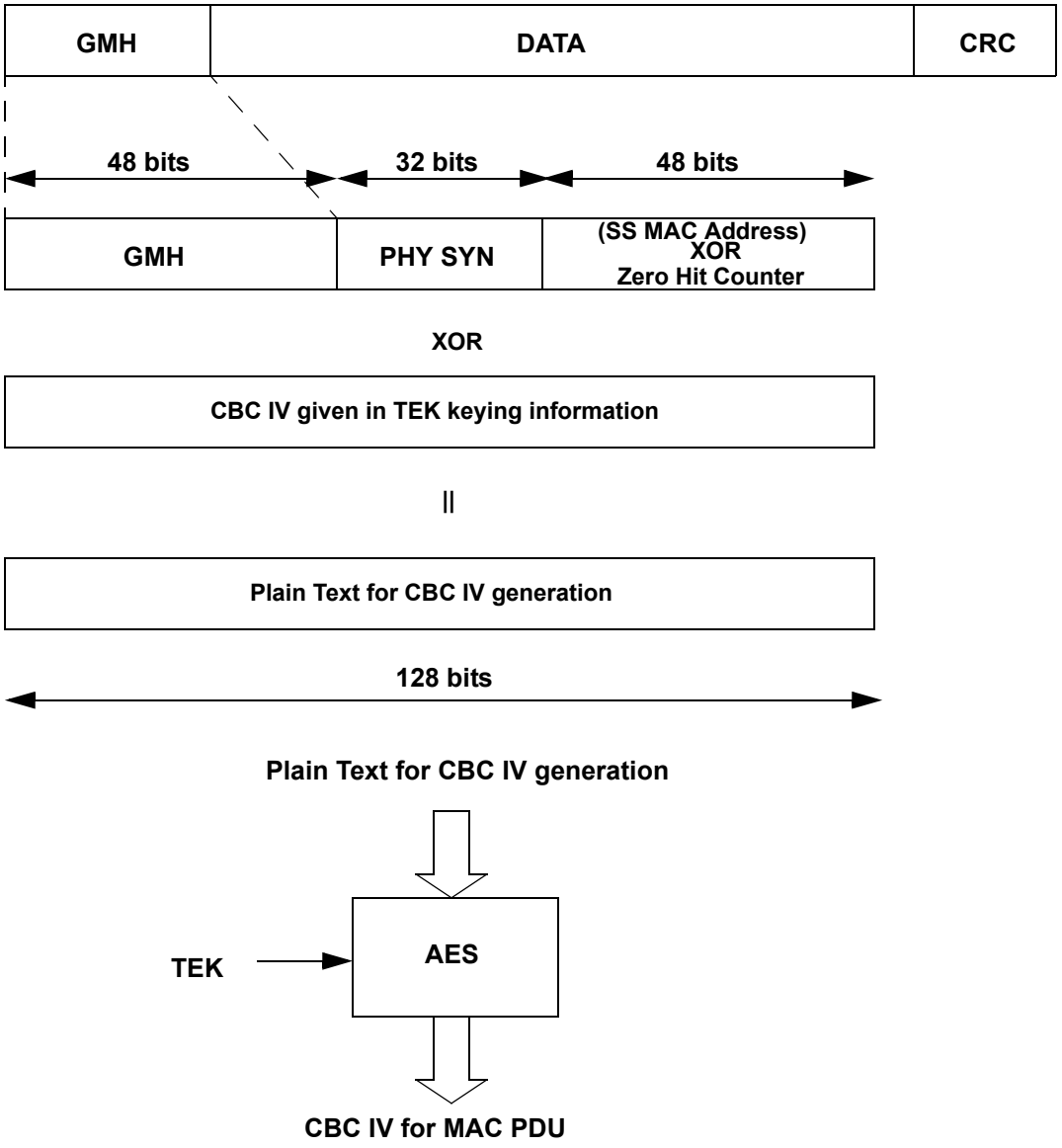

Figure 137c—CBC IV generation

If the MAC PDU is decoded from several channel coded blocks transmitted at different frames in HARQ operation, the MAC PDU payload must be decrypted with the CBC IV value which are generated from the PHY Synchronization value of the MAP when spid=0.

7.5.1.2.4 Receive Processing Rules

[Insert at the end of 7.5.1.2.4:]

The receiver shall maintain a PN window whose size is specified by the PN_WINDOW_SIZE parameter for SAs and management connections as defined in 11.9.36. Any received PDU with a PN lower than the beginning of the PN window shall be discarded as a replay attempt. The receiver shall track PNs within the PN window. Any PN that is received more than once shall be discarded as a replay attempt. Upon reception of a PN which is greater than the end of the PN window, the PN window shall be advanced to cover this PN.

[Insert new subclause 7.5.2.4]

7.5.2.4 Encryption of TEK-128 with AES Key Wrap

This method of encrypting the TEK-128 shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x04.

The BS encrypts the value fields of the TEK-128 in the Key Reply messages it sends to client SS. This field is encrypted using the AES Key Wrap Algorithm.

encryption: C,I = Ek[P]
decryption: P,I = Dk[C]
P = Plaintext 128-bit TEK
C = Ciphertext 128-bit TEK
I = Integrity Check Value
k = the 128-bit KEK
Ek[ ] = AES Key Wrap encryption with key k
Dk[ ] = AES Key Wrap decryption with key k

The AES key wrap encryption algorithm accepts both a ciphertext and an integrity check value. The decryption algorithm returns a plaintext key and the integrity check value. The default integrity check value in the NIST AES Key Wrap algorithm shall be used.

7.5.3 Calculation of HMAC-Digests

[Replace the last sentence of the section with the following sentence:]

~~The digest shall be calculated over the entire MAC Management message with the exception of the HMAC-Digest and HMAC Tuple attributes.~~

The HMAC sequence number in the HMAC tuple or Short-HMAC tuple shall be equal to the AK sequence number of the AK from which the HMAC_KEY_x was derived.

In the case of PKMv2, Short-HMAC Digest Calculations shall include the HMAC_PN_* that should be concatenated after the MAC Management message

[Insert new sub-clause 7.5.4.4 as follows:]

7.5.4.4 Cipher-based MAC (CMAC)

A BS or SS may support management message integrity protection based on Cipher-based MAC - together with the AES block cipher. The CMAC construction as specified in Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication: May 2005 shall be used.

7.5.4.4.1 Calculation of CMAC Value

The calculation of the keyed hash value contained in the CMAC-Digest attribute and the CMAC Tuple shall use the CMAC Algorithm with AES. The downlink authentication key CMAC_KEY_D shall be used for authenticating messages in the downlink direction. The uplink authentication key CMAC_KEY_U shall be used for authenticating messages in the uplink direction. Uplink and downlink message authentication keys are derived from the AK (see 7.5.4 below for details).

For authentication multicast messages (in the DL only) a CMAC_KEY_GD shall be used (one for each group), group authentication key is derived from GKEK

The CMAC-Digest and CMAC Tuple attributes shall be only applicable to the PKM version 2. In the PKM version 2 protocol, the CMAC key sequence number in the CMAC tuple shall be equal to the 4-bit AK sequence number of the AK from which the CMAC_KEY_x was derived.

The CMAC Packet Number Counter (CMAC_PN_*) is a 4 byte sequential counter that is incremented in the context of UL messages by the SS, and in the context of DL messages by the BS,. The BS will also maintain a separate CMAC_PN_* for multicast packets per each GSA and increment that counter in the context of each multicast packet from the group. For MAC messages that have no CID e.g. RNG-REQ message, the CMAC_PN_* context will be the same as used on the basic CID. If basic CID is unknown (e.g. in network reentry situation) then CID 0 should be used.

The CMAC Packet Number Counter, CMAC_PN_*, is part of the CMAC security context and must be unique for each MAC management message with the CMAC tuple or digest. Any tuple value of {CMAC_PN_*, AK} shall not be used more than once. The reauthentication process should be initiated (by BS or SS) to establish a new AK before the CMAC_PN_* reaches the end of its number space.

The digest shall be calculated over a field consisting of the CMAC key sequence number followed by the CMAC Packet Number Counter, expressed as an unsigned 32-bit number, followed by the 16-bit Connection ID on which the message is sent, followed by 16-bit of zero padding (for the header to be aligned with AES block size) and followed by the entire MAC management message with the exception of the CMAC-TLV.

The least significant bits of the digest shall be truncated to yield a 64-bit length digest. The CMAC key sequence number shall be equal to the 4-bit AK sequence number of the AK from which the CMAC_KEY_x was derived.

i.e.:

CMAC value <= Truncate64 (CMAC (CMAC_KEY_*, CMAC key sequence number | CMAC_PN | CID |16-bit zero padding | MAC_Management_Message))

If the digest is included in an MPDU that has no CID, e.g. A RNG-REQ message, the CID used shall take the value of the basic CID. If basic CID is unknown (e.g. in network reentry situation) then CID 0 should be used.

[Insert new subclause 7.5.4.5:]

7.5.4.5 Derivation of TEKs, KEKs, message authentication keys and GKEKs in PKMv2

[Insert new subclause 7.5.4.5.1:]

7.5.4.5.1 AES KEKs in PKMv2

The construction of the KEK for use with TEK-128 keys shall be the same as for 3-DES KEKs as described in 7.5.4.2 except that the full 128 bits of the KEK are used directly as the 128-bit AES key, instead of the KEK being split into two 64-bit DES keys.

[Insert new subclause 7.5.4.5.2:]

7.5.4.5.2 Encryption of GKEK in PKMv2

The BS encrypts the value fields of the GKEK in the Key Update Command message for the GKEK update mode and sends the encrypted GKEK to each SS served with the specific multicast service or the broadcast service. The following options for encryption of GKEK may be used. The encryption algorithm is determined according to the value of cryptographic suite. And, the value of cryptographic suite for GKEK encryption is identical to the one for GTEK encryption.

[Insert new subclause 7.5.4.5.2.1:]

7.5.4.5.2.1 Encryption of GKEK with 3-DES in PKMv2

This method of encrypting the GKEK shall be used for SAs with the TEK (or GTEK) encryption algorithm identifier in the cryptographic suite equal to 0x01.

The BS encrypts the value fields of the GKEK in the Key Update Command messages (for the GKEK update mode) it sends to client SS. This field is encrypted using two-key 3-DES in the EDE mode:

Encryption: C = Ek1[Dk2[Ek1[P]]]

Decryption: P = Dk1[Ek2[Dk1[C]]]

P = Plaintext 128-bit GKEK

C = Ciphertext 128-bit GKEK

k1 = leftmost 64 bits of the 128-bit KEK

k2 = rightmost 64 bits of the 128-bit KEK

E [ ] = 56-bit DES ECB mode encryption

D [ ] = 56-bit DES ECB mode decryption

[Insert new subclause 7.5.4.5.2.2:]

7.5.4.5.2.2 Encryption of GKEK with RSA in PKMv2

The RSA method of encrypting the GKEK (PKCS #1 v2.1, RSA Cryptography Standard, RSA Laboratories, June 2002) shall be used for SAs with the TEK (or GTEK) encryption algorithm identifier in the cryptographic suite equal to 0x02.

[Insert new subclause 7.5.4.5.2.3:]

7.5.4.5.2.3 Encryption of GKEK with ECB mode AES in PKMv2

This method of encrypting the GKEK shall be used for SAs with the TEK (or GTEK) encryption algorithm identifier in the cryptographic suite equal to 0x03.

The BS encrypts the value fields of the GKEK in the Key Update Command messages (for the GKEK update mode) it sends to client SS. This field is encrypted using 128-bit AES in ECB mode.

Encryption: C = Ek1[P]

Decryption: P = Dk1[C]

P = Plaintext 128-bit GKEK

C = Ciphertext 128-bit GKEK

k1 = the 128-bit KEK

E [ ] = 128-bit AES ECB mode encryption

D [ ] = 128-bit AES ECB mode decryption

[Insert new subclause 7.5.4.5.2.4:]

7.5.4.5.2.4 Encryption of GKEK with AES Key Wrap in PKMv2

This method of encrypting the GKEK shall be used for SAs with the TEK (or GTEK) encryption algorithm identifier in the cryptographic suite equal to 0x04.

The BS encrypts the value fields of the GKEK in the Key Update Command messages (for the GKEK update mode) it sends to client SS. This field is encrypted using 128-bit AES Key Wrap Algorithm. This 128-bit AES Key Wrap Algorithm is defined only for PKM version 2.

Encryption: C,I = Ek[P]

Decryption: P,I = Dk[C]

P = Plaintext 128-bit GKEK

C = Ciphertext 128-bit GKEK

k = the 128-bit KEK derived from the AK

Ek[ ] = AES Key Wrap encryption with key k

Dk[ ] = AES Key Wrap decryption with key k

[Insert new subclause 7.5.4.6:]

7.5.4.6 Key derivation functions for PKMv2

7.5.4.6.1 Dot16KDF for PKMv2

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/CMAC setting in the MAC (Message Authentication Code) mode is set to CMAC, the algorithm is defined as:

```
Dot16KDF(key, astring, keylength)

{

        result = null;

        Kin = Truncate (key, 128);

        for (i = 0;i <= int((keylength-1)/128); i++) {

                result <= result | Truncate (CMAC(Kin, i | astring | keylength), 128);

        }

        return Truncate (result, keylength);

}
```

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

```
Dot16KDF(key, astring, keylength)

{

        result = null;

        Kin = Truncate (key, 160);

        For (i=0; i <= int( (keylength-1)/160 ); i++) {

                result <= result | truncate (SHA-1( i| astring | keylength | Kin), 160);

        }

        return Truncate (result, keylength);

}
```

The key is a cryptographic key that is used by the underlying digest algorithm (SHA-1 or CMAC-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 $y \leqq x$.

[Insert new subclause 7.6.1.4.3:]

7.6.1.4.3 BS certificate

countryName=<Country of Operation>
organizationName=< Name of Infrastructure Operator>
organizationalUnitName=<WirelessMAN>
commonName=<Serial Number>
commonName=<BS Id>

The BS Id field shall contain the operator-defined BSId.[b] It is expressed as six pairs of hexadecimal digits separated by colons (:), e.g., "00:60:21:A5:0A:23." The Alpha HEX characters (A-F) shall be expressed as uppercase letters.

The attributes listed above shall be included.

[Change 7.6.1.6 as follows:]

7.6.1.6 tbsCertificate.issuerUniqueID and tbsCertificate.subjectUniqueID

The issuerUniqueID and subjectUniqueID fields shall be omitted for all ~~both~~ of the PKM's certificate types.

[Insert new subclause 7.7:]

7.7 Pre-Authentication

In anticipation of a handover, an MS may seek to use pre-authentication to facilitate an accelerated reentry at a particular target BS.

Pre-authentication results in establishment of an Authorization Key (with a unique AK Name) in the MS and target BS. The specific mechanism for Pre-authentication is out of the scope of this specification.

[Insert new subclause 7.8 and its subclauses:]

7.8 PKMv2

7.8.1 PKMv2 SA-TEK 3-way handshake

The AK can be derived in one of three different ways depending on the authentication scheme used as documented in 7.2.2.2.3. Before the 3-way handshake begins, the BS and SS shall both derive a shared KEK and HMAC/CMAC keys as per 7.2.2.2.

[b]The BSId is an operator-defined value, consequently the BS certificate is typically issued by the Operator, who must ensure that the BS ID is unique within the operator's network.

The PKMv2 SA-TEK 3-way handshake sequence proceeds as follows:

1. During initial network entry or reauthorization, the BS shall send PKMv2 SA-TEK-Challenge (including a random number BS_Random) to the SS after protecting it with the HMAC/CMAC Tuple. If the BS does not receive PKMv2 SA-TEK-Request from the SS within SAChallengeTimer, it shall resend the previous PKMv2 SA-TEK-Challenge up to SAChallengeMaxResends times. If the BS reaches its maximum number of resends, it shall initiate another full authentication or drop the SS.

2. If HO Process Optimization bit #1 is set indicating that PKM Authentication phase is omitted during network re-entry or handover, the BS begins the 3-way-handshake by appending the SA Challenge Tuple TLV to the RNG-RSP. If the BS does not receive PKMv2 SA-TEK-Request from the MS within SaChallengeTimer (suggested to be several times greater than the length of SaChallengeTimer), it may initiate full re-authentication or drop the MS. If the BS receives an initial RNG-REQ during the period that PKMv2 SA-TEK-Request is expected, it shall send a new RNG-RSP with another SaChallenge TLV.

3. The SS shall send PKMv2 SA-TEK-Request to the BS after protecting it with the HMAC/CMAC. If the SS does not receive PKMv2 SA-TEK-Response from the BS within SATEKTimer, it shall resend the request. The SS may resend the PKMv2 SA-TEK-Request up to SATEKRequestMaxResends times. If the SS reaches its maximum number of resends, it shall initiate another full authentication or attempt to connect to another BS. The SS shall include, through the Security Negotiation Parameters attribute, the security capabilities that it included in the SBC-REQ message during the basic capabilities negotiation phase.

4. Upon receipt of PKMv2 SA-TEK-Request, a BS shall confirm that the supplied AKID refers to an AK that it has available. If the AKID is unrecognized, the BS shall ignore the message. The BS shall verify the HMAC/CMAC. If the HMAC/CMAC is invalid, the BS shall ignore the message. The BS shall verify that the BS_Random in the SA TEK Request matches the value provided by the BS in the SA Challenge message. If the BS_Random value does not match, the BS shall ignore the message. In addition, the BS must verify the SS's security capabilities encoded in the Security Negotiation Parameters attribute against the security capabilities provided by the SS through the SBC-REG message. If security negotiation parameters do not match, the BS should report the discrepancy to higher layers.

5. Upon successful validation of the PKMv2 SA-TEK-Request, the BS shall send PKMv2 SA-TEK-Response back to the SS. The message shall include a compound TLV list each of which identifies the Primary and static SAs, their SA identifiers (SAID) and additional properties of the SA (e.g., type, cryptographic suite) that the SS is authorized to access. In case of HO, the details of any Dynamic SAs that the requesting MS was authorized in the previous serving BS are also included. In addition, the BS must include, through the Security Negotiation Parameters attribute, the security capabilities that it wishes to specify for the session with the SS (these will generally be the same as the ones insecurely negotiated in SBC-REQ/RSP).

Additionally, in case of HO, for each active SA in previous serving BS, corresponding TEK, GTEK and GKEK parameters are also included. Thus, SA_TEK_Update provides a shorthand method for renewing active SAs used by the MS in its previous serving BS. The TLVs specify SAID in the target BS that shall replace active SAID used in the previous serving BS and also "older" TEK-Parameters and "newer" TEK-Parameters relevant to the active SAIDs. The update may also include multicast/broadcast Group SAIDs (GSAIDs) and associated GTEK-Parameters pairs.

In case of unicast SAs, the TEK-Parameters attribute contains all of the keying material corresponding to a particular generation of an SAID's TEK. This would include the TEK, the TEK's remaining key lifetime, its key sequence number and the cipher block chaining (CBC) initialization vector. The TEKs are encrypted with KEK.

In case of group or multicast SAs, the TEK-Parameters attribute contains all of the keying material corresponding to a particular generation of a GSAID's GTEK. This would include the GTEK, the GKEK, the

GTEK's remaining key lifetime, the GTEK's key sequence number, and the cipher block chaining (CBC) initialization vector. The type and length of the GTEK is equal to ones of the TEK. The GKEK should be identically shared within the same multicast group or the broadcast group. Contrary Key-Update Command, the GTEKs and GKEKs are encrypted with KEK because they are transmitted as a unicast here.

Multiple iterations of these TLVs may occur suitable to re-creating and re-assigning all active SAs and their (G)TEK pairs for the SS from its previous serving BS. If any of the Security Associations parameters change, then those Security Associations parameters encoding TLVs that have changed will be added.

The HMAC/CMAC shall be the final attribute in the message's attribute list.

6. Upon receipt of PKMv2 SA-TEK-Response, an SS shall verify the HMAC/CMAC. If the HMAC/CMAC is invalid, the SS shall ignore the message. Upon successful validation of the received PKMv2 SA-TEK-Response, the SS shall install the received TEKs and associated parameters appropriately. Verification of HMAC/CMAC is done as per sections 7.5.3 and 7.5.4.4.

The SS also must verify the BS’s security negotiation parameters TLV encoded in the Security Negotiation Parameters attribute against the security negotiation parameters TLV provided by the BS through the SBC-RSP message. If security capabilities do not match, the SS should report the discrepancy to upper layers. The SS may choose to continue the communication with the BS. In this case, the SS may adopt the security negotiation parameters encoded in SA-TEK-Response message.

7.8.2 BS and SS RSA mutual authentication and AK exchange overview

The BS mutual authentication can take place in one of two modes of operation. In one mode, only mutual authentication is used. In the other mode, the mutual authentication is followed by EAP authentication. In this second mode, the mutual authentication is performed only for initial network entry and only EAP authentication is performed in the case that authentication is needed in re-entry.

SS mutual authorization, controlled by the PKMv2 Authorization state machine, is the process of

a) The BS authenticating a client SS's identity
b) The SS authenticating the BS's identity
c) The BS providing the authenticated SS with an AK, from which a key encryption key (KEK) and message authentication keys are derived
d) The BS providing the authenticated SS with the identities (i.e., the SAIDs) and properties of primary and static SAs the SS is authorized to obtain keying information for.

After achieving initial authorization, an SS periodically seeks reauthorization with the BS; reauthorization is also managed by the SS's PKMv2 Authorization state machine. An SS must maintain its authorization status with the BS in order to be able to refresh aging TEKs and GTEKs. TEK state machines manage the refreshing of TEKs. The SS or BS may run optional authenticated EAP messages for additional authentication.

The SS sends an Authorization Request message to its BS immediately after sending the Authentication Information message. This is a request for an AK, as well as for the SAIDs identifying any Static Security SAs the SS is authorized to participate in. The Authorization Request includes (see 6.3.2.3.9.19):

a) A manufacturer-issued X.509 certificate.
b) A description of the cryptographic algorithms the requesting SS supports; an SS's cryptographic capabilities are presented to the BS as a list of cryptographic suite identifiers, each indicating a particular pairing of packet data encryption and packet data authentication algorithms the SS supports.

c) The SS's Basic CID. The Basic CID is the first static CID the BS assigns to an SS during initial ranging-the primary SAID is equal to the Basic CID.

d) A 64-bit random number generated in the SS.

In response to an Authorization Request message, a BS validates the requesting SS's identity, determines the encryption algorithm and protocol support it shares with the SS, activates an AK for the SS, encrypts it with the SS's public key, and sends it back to the SS in an Authorization Reply message. Random numbers are included in the exchange to ensure liveness. The Authorization Reply includes (see 6.3.2.3.9.20)

a) The BS's X.509 certificate, used to verify the BS's identity.

b) A pre-PAK encrypted with the SS's public key.

c) A 4-bit PAK sequence number, used to distinguish between successive generations of AKs.

d) A PAK lifetime.

e) The identities (i.e., the SAIDs) and properties of the single primary and zero or more static SAs the SS is authorized to obtain keying information for.

f) The 64-bit random number generated in the SS.

g) A 64-bit random number generated in the BS, used to ensure key of liveness along with the random number of SS.

h) The RSA signature over all the other attributes in the auth-reply message by BS, used to assure the the authenticity of the above PKMv2 RSA-Reply messages.

An SS shall periodically refresh its AK by reissuing an Authorization Request to the BS. Reauthorization is identical to authorization. To avoid service interruptions during reauthorization, successive generations of the SS's AKs have overlapping lifetimes. Both SS and BS shall be able to support up to two simultaneously active AKs during these transition periods. The operation of the Authorization state machine's Authorization Request scheduling algorithm, combined with the BS's regimen for updating and using a client SS's AKs (see 7.4), ensures that the SS can refresh TEK keying information without interruption over the course of the SS's reauthorization periods.

After successful RSA based authorization either EAP based authorization or Authenticated EAP based authorization maybe supported according to the value of Authorization policy negotiated in the SBC-REQ/RSP messages. It shall cryptographically bind RSA and further EAP authentication.

7.8.3 Multicast Broadcast Service (MBS) support

MBS is an efficient and power saving mechanism that requires PKMv2 to send multimedia broadcast information. It provides subscribers with strong protection from theft of service across broadband wireless mobile network by encrypting broadcast connections between SSs and BSs.

7.8.3.1 MBS security associations

In addition to existing three Security Association, MBS requires a MBS Group Security Association. It is the set of security information that multiple BS and one or more of its client SSs share but not bound to any MS authorization state in order to support secure and access controlled MBS content reception across the IEEE Std 802.16 network. Each MBS capable MS may establish a MBS security association during the MS initialization process. MBS GSAs shall be provisioned within the BS. A MBS GSA’s shared information shall include the Cryptographic Suite employed within the GSA and key material information such as MAKs (MBS Authorization Key) and MGTEKs (MBS Group Traffic Encryption Key). The exact content of the MGSA is dependent on the MGSA’s Cryptographic Suite. As like any other Unicast SAs, MBS GSA is also identified using 16bits SAIDs. Each MS shall establish one or more MBS GSA with its serving BS.

Using the PKMv2 protocol, an MS receives or establishes an MBS GSA's keying material. The BS and MBS content server shall ensure that each client MS only has access to the MGSAs it is authorized to access.

An SA's keying material [e.g., MAK and MGTEK] has a limited lifetime. When the MBS content server or BS delivers MBS SA keying material to an MS, it also provides the MS with that material's remaining life-time. It is the responsibility of the MS to request new keying material from the MBS server or BS before the set of keying material that the MS currently holds expires at the MBS Server or BS.

7.8.3.2 MBS Key Management

7.8.3.2.1 MBS Authorization Key (MAK) establishment

The MAK establishment procedure in MS and BS is outside of scope of this specification.

7.8.3.2.2 MGTEK establishment

See 7.2.2.3.3 MBS Group Security Association and PKMv2 Key Derivation (7.2.2.2).

7.8.3.2.3 MBS Traffic Key establishment

See 7.2.2.2 PKMv2 Key Derivation.

[Insert new subclause 7.9:]

7.9 Optional multicast and broadcast rekeying algorithm (MBRA)

When MBRA is supported, the MBRA shall be used to refresh traffic keying material efficiently not for the unicast service, but for the multicast service or the broadcast service.

7.9.1 MBRA flow

The MBRA overall flow is shown in the Figure 137d.

An SS may get the traffic keying material before an SS is served with the specific multicast service or the broadcast service. The initial GTEK request exchange procedure is executed by using the Key Request and Key Reply messages that are carried on the Primary Management connection. The GTEK (Group Traffic Encryption Key) is the TEK for multicast or broadcast service. Once an SS shares the traffic keying material with a BS, an SS doesn't need to request the new traffic keying material. A BS updates and distributes the traffic keying material periodically by sending two Key Update Command messages.

A BS manages the M&B (Multicast & Broadcast) TEK Grace Time for the respective GSA-ID in itself. The GSA-ID (Group Security Association Identifier) is the SA-ID for multicast or broadcast service. This M&B TEK Grace Time is defined only for the multicast service or the broadcast service. This parameter means time interval (in seconds), before the estimated expiration of an old distributed GTEK. In addition, the M&B TEK Grace Time is longer than the TEK Grace Time managed in an SS.

A BS distributes updated traffic keying material by sending two Key Update Command messages before old distributed GTEK is expired. The usage type of these messages is distinguished according to the Key Push Modes included in the Key Update Command message.

A BS transmits the PKMv2 Group Key Update Command message for the GKEK update mode to each SS served with the specific multicast service or the broadcast service before the M&B TEK Grace Time starts. The purpose of the Key Update Command message for the GKEK update mode is to distribute the GKEK

(Group Key Encryption Key). The Key Update Command message for the GKEK update mode is carried on the Primary Management connection. A BS intermittently transmits the Key Update Command message for the GKEK update mode to each SS in order to reduce the BS's load in refreshing traffic key material. The GKEK is needed to encrypt the new GTEK. The GKEK may be randomly generated in a BS or an ASA server.

A BS transmits the PKMv2 Group Key Update Command message for the GTEK update mode carried on the broadcast connection after the M&B TEK Grace Time starts. The aim of the Key Update Command message for the GTEK update mode is to distribute new GTEK and the other traffic keying material to all SSs served with the specific multicast service or the broadcast service. This GTEK is encrypted with already transmitted GKEK.

An SS shall be capable of maintaining two successive sets of traffic keying material per authorized GSA-ID. Through operation of its GTEK state machines, an SS shall check whether it receives new traffic keying material or not. If an SS gets new traffic keying material, then its TEK Grace Time is not operated. However, if it doesn't have that, then an SS shall request a new set of traffic keying material a configurable amount of time, the TEK Grace Time, before the SS's latest GTEK is scheduled to expire.

If an SS receives the valid two Key Update Command messages and shares new valid GKEK and GTEK with a BS, then that SS doesn't need to request a new set of traffic keying material.

If an SS doesn't receive at least one of two Key Update Command messages, then that SS sends the Key Request message to get a new traffic keying material. A BS responds to the Key Request message with the Key Reply message. In other words, if an SS doesn't get valid new GKEK or GTEK, then the GTEK request exchange procedure initiated by a SS is executed.

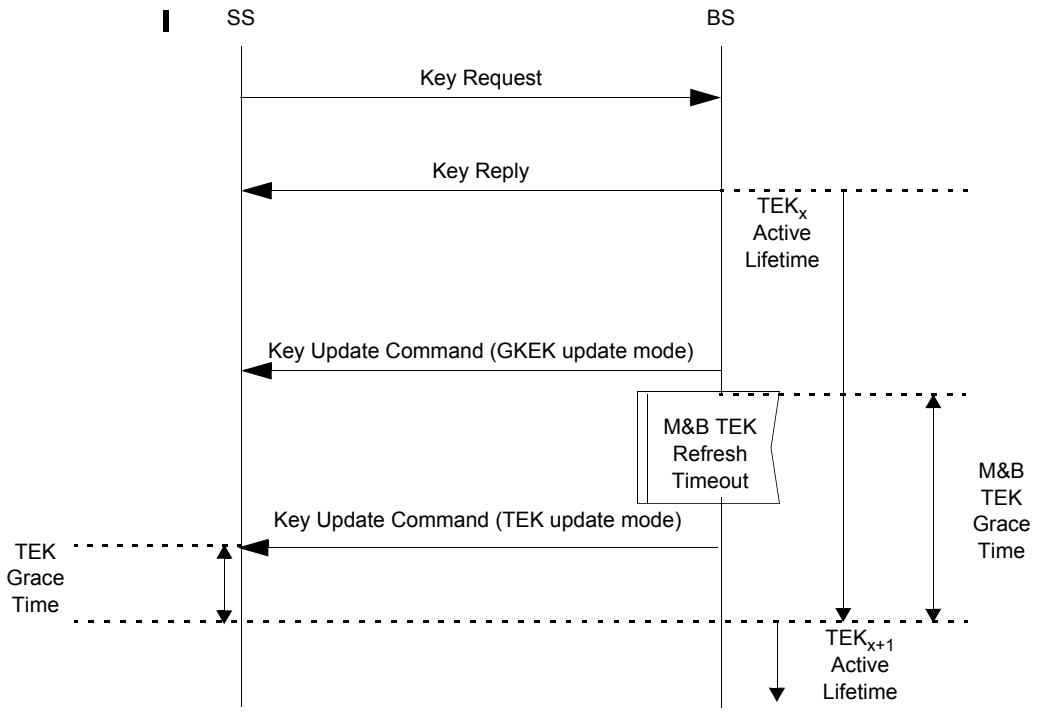

Figure 137d—MBRA management

7.9.1.1 BS usage of GTEK

An SS tries to get the GTEK before an SS is served with the specific service. The initial GTEK request exchange procedure is executed by using the Key Request and Key Reply messages that are carried on the primary management connection.

A BS shall be capable of maintaining two successive sets of traffic keying material per authorized GSAID. That is, when GKEK has been changed a BS manages the M&B (Multicast & Broadcast) TEK Grace Time for the respective GSA-ID in itself. Through operation of its M&B TEK Grace Time, a BS shall push a new set of traffic keying material. This M&B TEK Grace Time is defined only for the multicast service or the broadcast service in a BS. This parameter means time interval (in seconds) before the estimated expiration of an old distributed GTEK. That is, the M&B TEK Grace Time is longer than the TEK Grace Time managed in an SS.

A BS distributes updated GTEK by using two Key Update Command messages when the GKEK has been changed, or by using one (the second) Key Update Command message otherwise, around the M&B TEK Grace Time, before the already distributed GTEK is expired. Those messages are distinguished according to a parameter included in that message, “Key Push Modes.”

A BS transmits the first Key Update Command message to each SS served with the specific service before the M&B TEK Grace Time. The first Key Update Command message is carried on the primary management connection. A BS intermittently transmits the first Key Update Command message to each SS in order to reduce the BS’s load for key refreshment. The purpose of the first Key Update Command message is to distribute the GKEK (Group Key Encryption Key). This GKEK is needed to encrypt the updated GTEK. The GKEK is also encrypted with the SS’s KEK. The GKEK may be randomly generated in a BS or an ASA server.

A BS transmits the PKMv2 Group Key Update Command message carried on the broadcast connection after the M&B TEK Grace Time. The aim of the second Key Update Command message is to distribute the GTEK to the specific service group. This GTEK is encrypted with transmitted GKEK before the M&B TEK Grace Time.

7.9.1.2 SS usage of GTEK

An SS shall be also capable of maintaining two successive sets of traffic keying material per authorized GSAID. Through operation of its GTEK state machines, an SS shall check whether it receives new traffic keying material or not. If an SS get new traffic keying material, then its TEK Grace Time is not operated. However, if it doesn’t has that, then an SS shall request a new set of traffic keying material a configurable amount of time, the TEK Grace Time, before the SS’s latest GTEK is scheduled to expire.

7.9.2 Messages

Messages used in the MBRA are the Key Request, Key Reply, and Key Update Command messages.

- Key Request
  An SS may request the traffic keying material with the Key Request message in the initial GTEK request exchange procedure or the GTEK refresh procedure. Refer to 6.3.2.3.9.5.

- Key Reply
  A BS responds to the Key Request message with the Key Reply message including the traffic keying material. Key Reply message includes GKEK as well as GTEK. The GTEK is the TEK for the multicast or broadcast service. GKEK and GTEK are encrypted to safely distribute to an SS. GTEK is encrypted with the GKEK for the multicast service or the broadcast service. The GKEK is encrypted with the

KEK. See 7.5.4.5.2 and 7.9.3 for details. This message is carried on the primary management connection. Refer to 6.3.2.3.9.6.

- Key Update Command
  A BS transmits Key Update Command message to initiate and push newly updated GKEK and GTEK to every SSs served with the specific multicast or broadcast service.

7.9.3 Encryption of GKEK

The BS encrypts the value fields of the GKEK in the Key Update Command message for the GKEK update mode and sends the encrypted GKEK to each SS served with the specific multicast service or the broadcast service. This field is encrypted using several algorithms. See 7.5.4.5.2 for details.

7.9.4 Message authentication keys for the Key Update Command message

One of the HMAC-Digest attribute or the CMAC-Digest attribute is used for Key Update Command message authentication.

Input key used to generate HMAC authentication keys of Key Update Command message is different according to the value field of the Key Push Modes. The AK shall be used for generation of HMAC-Digest included in the Key Update Command message for the GKEK update mode and the GKEK shall be used for generation of HMAC-Digest included in the Key Update Command message for the GTEK update mode. See 7.2.2.2.9 for details. The CMAC_KEY_GD and HMAC_KEY_GD should be recomputed when a new GKEK is used.