Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**	
Title	**Privacy Sublayer Clean up**	
Date Submitted	**2004-11-04**	
Source(s)	Jun Hyuk Song, Yong Chang Samsung Electronics Co., LTD. David Johnston Intel Corporation 2111 NE 25th Ave. Hillsboro 97124	Voice:+82(31)279-3639 [mail:junhyuk.song@samsung.com] Voice: +1 (503)264-3855 [mailto:dj.Johnston@intel.com]
Re:	Re: Sponsor ballot on IEEE P802.16e/D5	
Abstract	Proposal editorial clean up for Privacy Sublayer	
Purpose	Discuss and Adopt as the baseline text	
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>.	

Privacy Sublayer Clean Up

JunHyuk Song, Yong Chang, Samsung Electronics
David Johnston, Intel Corporation

Introduction

Some of the PKMv2 features have been left out in Privacy Sublayer section with some typo and without editorial alignment with existing PKMv1 features ever since a group of companies introduces PKMv2 concept. In this proposal, we provide editorial clean up of the whole Privacy Sublayer section according to previous PKMv2 proposal over the existing 802.16e/D5. The proposed text need to be further updated once group decided to adopt new Key Hierarchy proposal.

7. Privacy Sublayer

Privacy provides subscribers with privacy, authentication or confidentiality across the broadband wireless network. It does this by applying cryptographic transforms to MPDUs carried across connections between SS and BS.

In addition, security provides operators with strong protection from theft of service. The BS protects against unauthorized access to these data transport services by securing the associated service flows across the network. Privacy 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 privacy mechanisms are strengthened by adding digital-certificate based MSS device-authentication to its key management protocol.

7.1 Architecture

Security has two component protocols as follows:

a) An encapsulation protocol for securing packet data across the 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 BS to MSS. Through this key management protocol, MSS and BS synchronize keying data; in addition, the BS uses the protocol to enforce conditional access to network services.

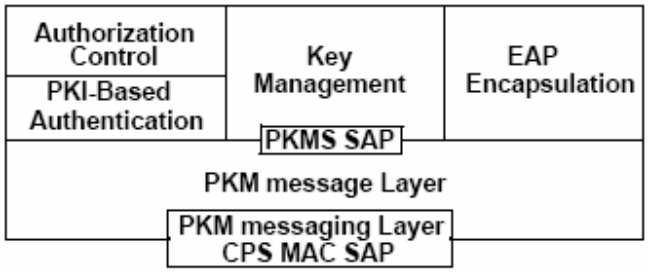

Figure 131—Security Sublayer

7.1.1 Packet Data Encryption

Encryption services are defined as a set 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 ciphersuites; the generic MAC header is not encrypted. All MAC management messages shall be sent in the clear to facilitate registration, ranging, and normal operations of the MAC.

The format of MAC PDUs carrying scured packet data payloads is specified in 6.4.3.6.

7.1.2 Key Management Protocol

The PKM protocol facilitates mutual authentication of the MSS and BS, as well as distribution of traffic keying material from the BS to the MSS. 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] to perform authentication. It uses strong symmetric algorithms to perform key exchanges between MSS and BS.

The PKM's authentication protocol establishes a shared secret (i.e., an AK) between MSS and BS. The shared secret is then used to secure subsequent PKM exchanges of TEK. This two-tiered mechanism for key distribution permits refreshing of TEKs without incurring the overhead of computation-intensive public key operations.

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

The BS associates an MSS'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 establish an authenticated identity of a client MSS and the services (i.e., specific TEKs) the MSS is authorized to access.

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

The traffic-key management portion of the PKM protocol adheres to a client/server model, where the MSS (a PKM "client") requests keying material, and the BS (a PKM "server") responds to those requests, ensuring that individual MSS 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 6.4.2.3. The PKM protocol is defined in detail in 7.2.

7.1.3 Authentication Protocol

An MSS 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 [PKCS #1] (support is mandatory in all devices)
Extensible Authentication Protocol (support is optional as described in xx)

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].

A BS authenticates a client SS during the initial authorization exchange. Each MSS carries a unique X.509 digital certificate issued by the MSS's manufacturer. The digital certificate contains the MSS's Public Key and MSS MAC address. When requesting an AK, an MSS 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 MSS.

All MSSs shall have factory-installed RSA private/public key pairs or provide an internal algorithm to generate such key pairs dynamically. If an MSS 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 MSSs 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

7.1.3.2 PKM EAP Authentication

PKM EAP Authentication uses Extensible Authentication Protocol [IETF RFC 3748]~~[IETF RFC 2284bis]~~ in conjunction with a vendor-selected standardized EAP Method (eg. 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 Information 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, but they should be selected with awareness of the security issues described in [IETF RFC 3748 ~~[IETF RFC 2284bis]~~ section 7.

Figure 131a shows the relationship between the lower levels of the 802.16 MAC and the generic EAP components (and the interface between them)

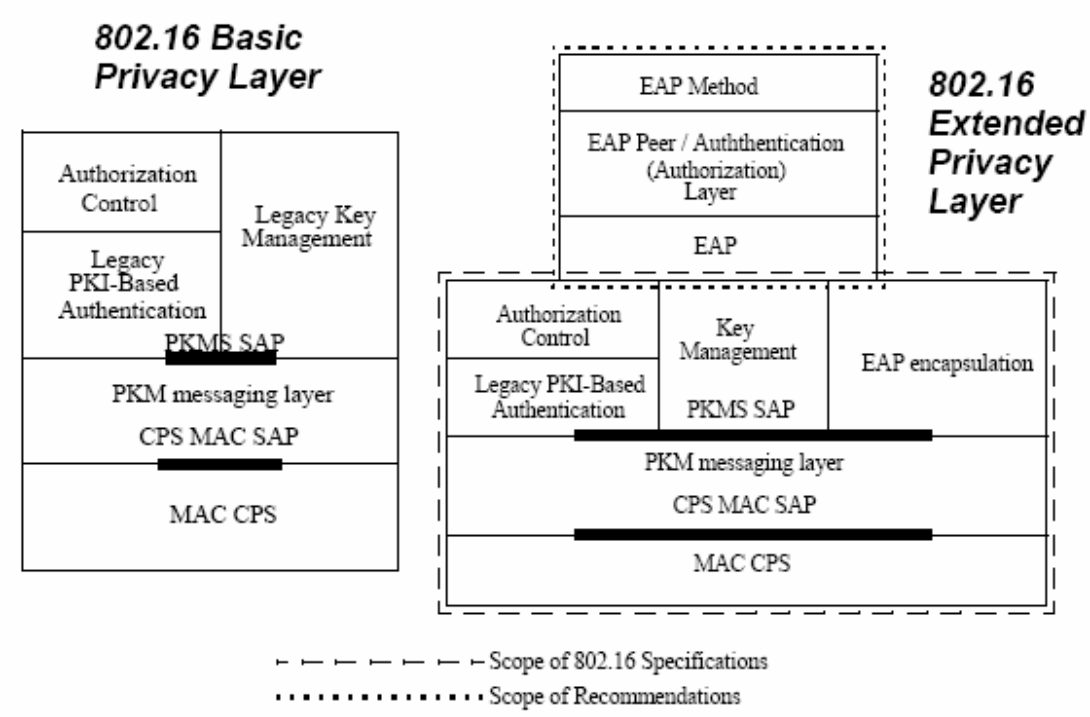

Figure 131a—Comparison of the Basic and Extended Privacy Layers (Control Plane)

7.2 PKM Protocols

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-OMAC, AES-key-wraps, pre-authentication and MBS are supported

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 manageable SS establishes a Primary Security 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 can 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 manageable 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. The PKM protocol specifies how SS and BS maintain key synchronization.

7.2.1.2 Mapping of Connections to SAs

7.2.1.3 MSS authorization and AK exchange overview

MSS authorization, controlled by the Authorization state machine, is the process of

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

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

7.2.1.3.1 Authorization via PKM RSA Authentication Protocol

An MSS begins authorization by sending an Authentication Information message to its BS. The Authentication Information message contains the MSS 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 MSS.
The MSS 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 MSS 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 MSS supports; an MSS'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 MSS supports
c) the MSS's Basic CID. The Basic CID is the first static CID the BS assigns to an MSS during initial ranging the primary SAID is equal to the Basic CID

In response to an Authorization Request message, a BS validates the requesting MSS's identity, determines the encryption algorithm and protocol support it shares with the MSS, activates an AK for the MSS, encrypts

7.2.1.3.2 Authorization via PKM Extensible Authentication Protocol

The first steps of the authorization flow are as follows:

a) Upon successful completion of ranging (and capabilities exchange), a logical signal (ie. "link activation") is sent upwards on the Logical Control Interface at the BS (ie. the EAP authenticator). This will cause the authenticator to begin the authentication sequence.
b) EAP on the Authenticator sends an EAP-Request message to the supplicant. This Request might be an EAP identity request or the beginning of an EAP method. The message is encapsulated in a MAC management PDU and transmitted.
c) EAP on the supplicant receives EAP-Request, passes it to the local EAP method for processing, and transmits EAP Response.

Steps 2 and 3 (EAP-Request/Response exchange) continue as many times as needed.

After one or more EAP-Request/Response exchanges, the authentication server (whether local to the Authenticator or connected remotely via an AAA protocol) determines whether or not the authentication is successful.

The next steps of the authorization flow are as follows:

d) Upon success, EAP on the authenticator transmits a “success” signal on the logical control interface to fully activate the airlink.
e) EAP on the authenticator transmits EAP-success, which is then encapsulated in a MAC management message and transmitted to the supplicant.
f) EAP on the supplicant transmits a “success” indication on the logical control interface to fully activate the airlink.
g) Both EAPs (authenticator and supplicant) export the AAA-key across the logical control interface. As detailed in [3], the AAA-key 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.

The final steps of the authorization flow:

1) The BS and MSS each derive the EAP Master Key from the AAA-Key. The EAP Master Key is derived simply the taking the 32 lowest order octets of the AAA-Key.
2) BS sends the EAP-Establish-Key-Request PKM message (including a 32-byte nonce) to the MSS. The MSS then generates its own 32-byte nonce, and derives a Transient Key (TK) as follows:

TK = PRF-384(*EAP Master Key*, “Pairwise key expansion”,
Min(*BSId, SSId*) |
Max(*BSId, SSId*) |
Min(*BS-Generated-Nonce, MSS-Generated-Nonce*) |
Max(*BS-Generated-Nonce, MSS-Generated-Nonce*))
where
PRF-384 (K, A, B) :=
for *i* = 0 **to** 3 **do**
R = R | HMAC-SHA-1(K, A | 0 | B | I)
return LeastSignificant-384-bits(R).
and “|” denotes bitstring concatenation.
The MSS then derives Key Confirmation Key (KCK) and Authorization Key (AK) as follows:
KCK = bits 0-127 (ie. lowest order) of the TK
AK = bits 224-383 of the TK
The MSS can attempt to use a cached or handover-transferred Master Key and avoid a full reauthentication. To do this, it sends EAP-Establish-Key-Request specifying the MKID attribute, which identifies by name the Master Key that the MSS should use for AK establishment if it also has the MK cached.

3) MSS sends the EAP-Establish-Key-Reply PKM message (including the 32-byte nonce that it used to derive TK) to the BS. EAP-Establish-Key-Reply includes an HMAC Tuple TLV, which must be calculated using the KCK derived above.

Upon receipt of the EAP-Establish-Key-Reply, the BS computes the TK, KCK, and AK as above.
BS then validates the HMAC Tuple. If the HMAC tuple is incorrect, BS discards the message without responding.

If the MSS elects not to proceed with key establishment (eg. the EAP-Establish-key-request specified an unknown MKID), the MSS sends EAP-Establish-Key-Reject instead.

4) BS sends the EAP-Establish-Key-Confirm PKM message to supply the MSS with its SA information and activate the AK.

7.2.1.4 TEK exchange overview

[Take existing PKMv1 text]

7.2.1.4.1 TEK exchange overview for PMP topology

[Take existing PKMv1 text]

7.2.1.4.2 TEK exchange overview for PMP mode

[Take existing PKMv1 text]

7.2.1.5 Security Capabilities selection

[Take existing PKMv1 text]

7.2.1.6 Authorization State Machine

[Need new state machine here]

7.2.1.7 TEK State Machine

[Take existing PKMv1 text]

7.2.1.8 Cryptographic methods

[Take existing PKMv1 text]

7.2.2 PKM Version 2

7.2.2.1 Security Associations

[Text needs to be added based on Key Hierarchy decision]

7.2.1.2 Mapping of Connections to SAs

[Text needs to be added based on Key Hierarchy decision]

7.2.2.1 MSS authorization and PAK exchange overview

[Text needs to be added based on Key Hierarchy decision]

7.2.2.1.1 Authorization via PKM RSA Authentication Protocol

[Text needs to be added based on Key Hierarchy decision]

7.2.2.1.2 Authorization via PKM Extensible Authentication Protocol

[Text needs to be added based on Key Hierarchy decision]

7.2.2.1.3 MSS and BS mutual authorization and AK exchange overview

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

a) The BS authenticating a client MSS's identity
b) The MSS authenticating the BS's identity

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

After achieving initial authorization, an MSS periodically seeks reauthorization with the BS; reauthorization is also managed by the MSS's PKMv2 Authorization state machine. An MSS 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 MSS 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 MSS 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 MSS supports; an MSS'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 MSS supports
c) the MSS's Basic CID. The Basic CID is the first static CID the BS assigns to an MSS during initial ranging—the primary SAID is equal to the Basic CID

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

An MSS 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 MSS's AKs have overlapping lifetimes. Both MSS 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 MSS's AKs (see 7.4), ensures that the MSS can refresh TEK keying information without interruption over the course of the MSS's reauthorization periods.

7.2.2.1.4 Pre-Authentication

After a HO-REQ/RSP exchange, an MSS may seek to use pre-authentication to effect a fast handover. An MSS seeking to use pre-authentication shall transmit a PKM_PREAUTH-REQ.

A BS on recipt of a PKM-AUTH-REQ message shall reply with a PKM-PREAUTH-RSP message, or with a PKM_PREAUTH-REJECT message.

A BS may send an unsolicited PKM_AUTH-RSP message.

A PKM-PREAUTH-RSP indicates that the chosen BS is populated with a PMK coupled to the identity of the requesting MSS.

The pre-authenticated MSS may skip the authorization and EAP stages of network entry. The primary keying material available at the BS and MSS shall be the computed PMK as defined in 7.x.x.x key Hierarchy. Therefore the AK computation will be based on the PMK and not the PAK, consistent with the AK computation rules in the PKMv2 key hierarchy.

7.2.2.2 Key Hierarchy

[Text needs to be added based on Key Hierarchy decision]

7.2.2.2.1 Key Derivation Function

[Text needs to be added based on Key Hierarchy decision]

7.2.2.3 MBRA (Multicast & Broadcast Rekeying Algorithm)

If GTEK update exchange method for the multicast service and the broadcast service is identically applied to one for the unicast service, then that multicast and broadcast rekeying is resource inefficient.

Therefore, GTEK refreshment for the multicast service and the broadcast service should be different from one for the unicast service. The new MBRA (Multicast & Broadcast Rekeying Algorithm) to efficiently refresh GTEK is needed. The MBRA is restricted to the multicast service and the broadcast service.

The aims of the MBRA are satisfied with the following:

- Provide efficient rekeying method for multicast group and broadcast group.
- Provide a BS's key push mode to an MSS.
- Provide strong protection for the replay attack.

7.2.2.3.1 MBRA (Multicast & Broadcast Rekeying Algorithm)

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

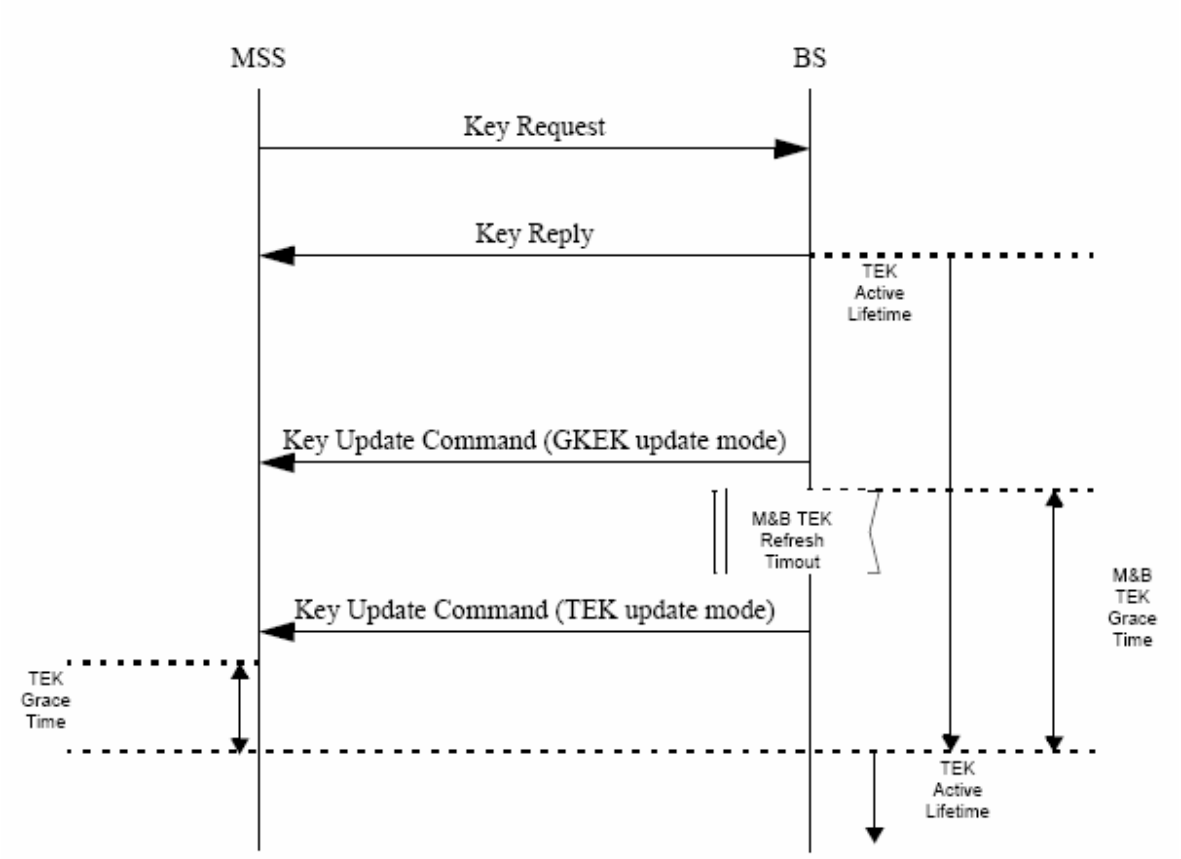

Figure 137b—MBRA management

7.2.2.3.2 BS usage of GTEK

An MSS tries to get the GTEK before an MSS 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, 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 MSS.

A BS distributes updated GTEK by using two Key Update Command messages 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 MSS 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 MSS 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 MSS’s AK. The GKEK can be randomly generated in a BS or an ASA server.

A BS transmits the second Key Update Command message carrying 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.

7.2.2.3.2 MS usage of GTEK

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

7.2.2.3.3 Messages

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

- Key Request
  Refer to 6.3.2.3.9.11.
- Key Reply
  Two subattributes in TEK-Parameters included in Key Reply message. Those subattributes are shown in Table 133a.

Table 133a—TEK-Parameters subattributes

Attribute	Contents
GKEK	GKEK (Group Key Encryption Key), encrypted by the GKEKEK that is derived from the AK.
GTEK	GTEK (Group Traffic Encryption Key), encrypted with the GKEK

Key Reply message includes GKEK as well as GTEK. GKEK and GTEK are encrypted to safely distribute to an MSS. GTEK is encrypted with the GKEK for the multicast service or the broadcast service and GKEK is encrypted with the MSS’s GKEKEK. The lifetime and sequence number of GKEK are identical to ones of GTEK. This message is carried on the primary management connection.

- Key Update Command

A BS transmits Key Update Command message to initiate and push newly updated GKEK and GTEK to an MSS. Attributes of Key Update Command are shown in Table 133b.

Table 133b—Key Update Command attributes

Attribute	Contents
Key-Sequence-Number	Authorization key sequence number
GSAID	Security Association ID
Key Push Modes	Usage code of Key Update Command message
Key Push Counter	Counter one greater than that of older generation for replay attack
TEK-Parameters	"Newer" generation of key parameters relevant to GSAID
GKEK	GKEK, encrypted by the GKEKEK that is derived from the AK
GTEK	GTEK, encrypted with the GKEK
Key-Lifetime	GTEK Remaining Lifetime
Key-Sequence-Number	GTEK Sequence Number
CBC-IV	Cipher Block Chaining (CBC) Initialization Vector
HMAC-Digest	Keyed SHA message digest

There are two types of Key Update Command message, GKEK update mode and GTEK update mode. Key Push Modes indicates the usage code of Key Update Command message.

Key Push Counter is used to protect for replay attack. This value is one greater than that of older generation.

Key Update Command message contains only newer generation of key parameters, because this message inform an MSS of next key materials.

7.2.2.3.4 Encryption of GKEK

The 160-GKEK used to encrypt GTEK is encrypted using 128 bit AES KEY WRAP.

A BS encrypts the value fields of the 128-GKEK in the first Key Update Command messages (GKEK update mode) and sends to each MSS served.

Encryption: C = AES_KEY_WRAP_ENCRYPT(k1, P)
Decryption: P = AES_KEY_WRAP_DECRYPT(k1,C)
P = Plaintext GKEK 160-bit
C = Ciphertext GKEK 160-bit
k1 = GKEKEK
I = AES_KEY_WRAP_DECRYPT(k1, C)
I: AES Key Wrap Integrity Value

7.2.2.4 MBS (Multicast Broadcast Service) support

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

7.2.2.4.1 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 bind to any MSS authorization state in order to support secure and access controlled MBS content reception across the IEEE Std 802.16 network. Each MBS capable MSS may establish a MBS security association during the MSS 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 MSS shall establish one or more MBS GSA with its serving BS.

Using the PKMv2 protocol, an MSS receives or establishes an MBS GSA's keying material. The BS and MBS content server shall ensure that each client MSS 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 MSS, it also provides the MSS with that material's remaining lifetime. It is the responsibility of the MSS to request new keying material from the MBS server or BS before the set of keying material that the MSS currently holds expires at the MBS Server or BS.

7.2.2.4.2 MBS Key Management

7.2.2.4.2.1 MBS Authorization Key (MAK) establishment

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

7.2.2.4.2.2 MGTEK establishment

See 7.x.x.x. MBS Group Security Association and PKMv2 7.x.x.x Key Derivation.

7,2,2,4,2,3 MBS Traffic Key establishment

See 7.x.x.x PKMv2 Key Derivation.

7.2.2.5 Cryptographic methods

7.2.2.5.1 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 MSS. 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

7.2.2.5.2 Calculation of OMAC-Digest

The calculation of the keyed hash in the OMAC-Digest attribute and the OMAC Tuple shall use the OMAC Algorithm with AES. The downlink authentication key OMAC_KEY_D shall be used for authenticating messages in the downlink direction. The uplink authentication key OMAC_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).

In the PKM version 2 protocol, The OMAC Sequence number in the OMAC Tuple shall be equal to the 48 bit AK Sequence Number of the AK from which the OMAC_KEY_x was derived. In the PKM version 1 protocol, The 4 least significant bits of the OMAC Sequence number in the OMAC Tuple shall be equal to the 4 bit AK Sequence Number and the 44 most significant bits shall be equal to 0.

The digest shall be calculated over a field consisting of the OMAC key sequence number followed by the frame number, expressed as an unsigned 32 bit number, followed by the 16 bit connection ID on which the message is sent followed by the entire MAC management message with the exception of the OMAC-Digest but including the OMAC Tuple attributes.

The least significant bits of the digest shall be truncated to yield a 64 bit length digest.

I.E.:

OMAC digest <= Truncate64(OMAC(OMAC_KEY_*, OMAC sequence number | Frame number | CID | MAC_Management_Message | OMAC_TLV_Attributes))

If the message is included in an MPDU that has no CID, E.G. A RNG-REQ message, the CID used shall take the value 0.

The frame number in which a message containing an OMAC tuple may be fragmented and so be transmitted in more than one frame number. In this case, the frame number used in the OMAC calculation shall take the value of the frame number of the frame in which the first fragment is transmitted.

7.2.2.5.3 Data Encryption with AES in CTR mode

The PDU payload shall be prepended with a 32-bit nonce. Each base station in the MBS zone shall use the same nonce.The nonce shall be transmitted in little endian byte order. The nonce shall not be encrypted.

The nonce shall be repeated four times to construct 128bits nonce. (Ex. NONCE|NONCE|NONCE|NONCE)

The plaintext PDU shall be encrypted using the active MBS_Traffic_key (MTK) derived from MAK and MGTEK, according to CTR specification.

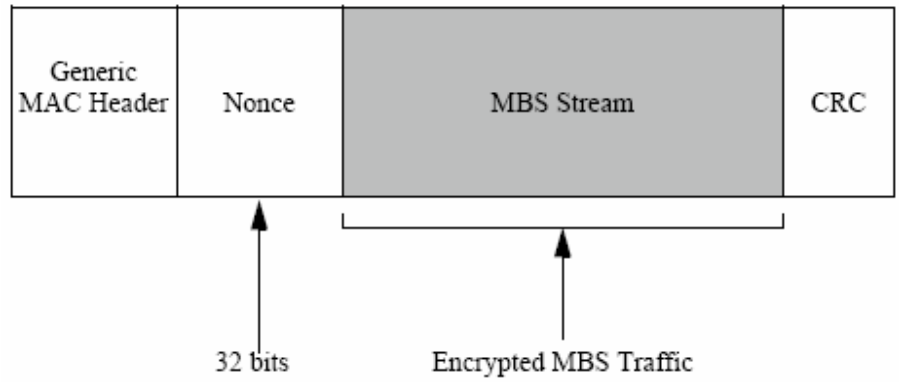

Figure 137a—MBS MAC PDU Ciphertext Payload Format

7.3 Dyanmic SA creation and mapping

[Text needs to be added based on Key Hierarchy decision]

7.4 Key Usage

[Text needs to be added based on Key Hierarchy decision]

7.5 Certificate Profile

7.5.1 Certificate Format

This subclause describes the X.509 [IETF RFC 2459] Version 3 certificate format and certificate extensions used in IEEE 802.16-compliant SSs. Table 112 below summarizes the basic fields of an X.509 Version 3 certificate.

Table 120—Basic fields of an X.509 Version 3 certificate

X.509 v3 field	Description
tbsCertificate.version	Indicates the X.509 certificate version. Always set to v3 (value of 2)
tbsCertificate.serialNumber	Unique integer the issuing CA assigns to the certificate.
tbsCertificate.signature	Object identifier (OID) and optional parameters defining algorithm used to sign the certificate. This field shall contain the same algorithm identifier as the signatureAlgorithm field below.
tbsCertificate.issuer	Distinguished Name of the CA that issued the certificate.
tbsCertificate.validity	Specifies when the certificate becomes active and when it expires.
tbsCertificate.subject	Distinguished Name identifying the entity whose public key is certified in the subjectpublic key information field.
tbsCertificate.subjectPublicKeyInfo	Field contains the public key material (public key and parameters) and the identifier of the algorithm with which the key is used.
tbsCertificate.issuerUniqueID	Optional field to allow reuse of issuer names over time.
tbsCertificate.subjectUnique ID	Optional field to allow reuse of subject names over time.
tbsCertificate.extensions	The extension data.
signatureAlgorithm	OID and optional parameters defining algorithm used to sign the certificate. This field shall contain the same algorithm identifier as the signature field in tbsCertificate.
signatureValue	Digital signature computed upon the ASN.1 DER encoded tbsCertificate.

All certificates described in this specification shall be signed with the RSA signature algorithm using SHA-1 as the one-way hash function. The RSA signature algorithm is described in PKCS #1; SHA-1 is described in FIPS 180-1. Restrictions posed on the certificate values are described below:

7.5.1.1 tbsCertificate.validity.notBefore and tbsCertificate.validity.notAfter