Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**
Title	802.16e Privacy Key Management (PKM) version 2
Date Submitted	**2004-07-08**
Source(s)	Jeff Mandin jeff@streetwaves-networks.com Streetwaves Networking Amatzia 5 Jerusalem, Israel 93148
Re:	Security Adhoc
Abstract	PKMv2 text
Purpose	Specification of structure and some basic functions for Privacy Key Management version 2
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>.

PKM version 2

Jeff Mandin
Streetwaves Networking

1 Overview

Modifies section 7 to incorporate PKM version 2 functionality. An overview clause in section 7 provides high-level description of functionality common to v1 and v2.

This contribution includes:

- EAP authentication for PKMv2
- EAP keying handshake (strawman for further discussion and analysis)
- PMK naming
- PMK caching
- Post-HO behaviour for case where PMK is already established

Text and placeholders for some other aspects of PKM v2 (eg. key hierarchy, mutual authentication) are marked by clause headings or "TBD".

2 Changes to 802.16e D3 text

[Modify section 7.1.2 as follows:]

7.1.2 PKM Key management protocol

The PKM key management protocol facilitates mutual authentication of the SS and BS, as well as distribution of traffic keying material from the BS to the SS. It also supports periodic reauthentication/reauthorization and traffic key refresh. The ~~key management~~ PKM protocol uses either EAP [IETF RFC 2284], or X.509 digital certificates [IETF RFC 3280] together with RSA public-key encryption algorithm [PKCS #1], to perform authentication. ~~It~~ PKM uses strong symmetric algorithms to perform key exchanges between SS and BS. PKM version 2 mandates mutual authentication, uses AES-based key derivation functions, and has other improvements. The legacy PKM version 1 uses X.509 certificates on the SS side only, and uses SHA-based key derivation functions. See Annex G (Security Considerations) for further details on PKMv1 and PKMv2 design and claims.

~~The PKM's authentication protocol establishes a shared secret (i.e., an AK) between SS and 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 public-key operations.~~

~~1In security parlance, confidentiality = privacy + authenticity~~

~~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 vendor-specific 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 establishes an 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 can 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 6.4.2.3. The PKM protocol is defined in detail in 7.2.

7.1.3 PKM-based Authentication

~~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 [PKCS #1] (support is mandatory in all devices)

- Extensible Authentication Protocol (support is optional as described in xx)

7.1.3.1 RSA Authentication

The PKM RSA authentication protocol uses X.509 digital certificates [IETF RFC 3280], the RSA publickey encryption algorithm [PKCS #1].

In PKMv1, a ~~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,~~ At authentication, an SS presents its digital certificate to the BS. The BS verifies the digital certificate, and upon successful verification will used the included public key for subsequent key management. ~~then uses the verified Public Key to encrypt an Authorization Key (AK) , which the BS then sends back to the requesting SS.~~

All SSs 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.

In PKMv2, the RSA authentication is mutual. Consequently, PKMv2-supporting devices (BS and SS alike) must carry a factory-installed x.509 certificate.

7.1.3.2 EAP Authentication

PKM EAP Authentication uses Extensible Authentication Protocol [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 [IETF draft-haverinen-pppext-eap-sim-13.txt or successor document ~~xxxxx~~].

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 criteria discussed in Annex F. ~~security issues described in [IETF RFC 3748 2284bis] section 7.~~

7.1.3.2.1 Network model for EAP-based Authentication

As described in [IETF RFC 3748], an EAP-based authentication involves three entities:

a) Peer (ie. the MSS in the case of 802.16e)

b) Authenticator (the BS)

c) Backend Authentication Server (though the Authentication Server function can be colocated in the BS, more typically there is an external AAA server that communicates with the BS via an EAP-enabled protocol such as RADIUS or DIAMETER). Operations between the BS and Backend Authentication Server are out of the scope of this specification.

The 802.16e MAC layer supplies a control plane logical interface to an EAP logical entity. The EAP logical entity performs the following functions:

a) sends and receives EAP messages with its peer EAP entity (ie. at the BS or SS) on the other end of the 802.16e air link during authentication

b) Notifies the MAC layer of the success or failure of authentication

c) Supplies keying material to the MAC layer

Figure 130 shows the integrated RSA and EAP control plane network models.

[Insert figure 130o –

Delete the left-hand "802.16 Basic Privacy Sublayer" stack

Remove label "802.16 Extended Privacy Layer"
change label of figure to "802.16 Privacy sublayer (Control Plane)"
change "PKI-based" to "RSA-based"
correct spelling of "authentication"]

7.1.3.3 RSA-Protected EAP Authentication

TBD

[Insert 7.1.4 as follows:]

7.1.4 Authorization

The authencation process associates an SS's authenticated identity to a paying subscriber, and hence to the data services that subscriber is authorized to access. *Authorization* is the term that refers to the sequence that informs the SS of the services to which it is entitled and the establishment of the *Authorization Key* (AK).

[Change current 7.1.4 to 7.1.4.1 as follows:]

7.1.4.1 Security associations

A *Security Association* (SA) is the set of security information shated by a BS and one or more SS. ~~a BS and one or more of its client SSs share inorder 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 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 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~~ receives or establishes 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 6.4.9.~~

~~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.~~

[Insert 7.1.4.2 as follows]

7.1.4.2 PKMv2-specific Security associations

[import SA text from "Key hierarchy" document]

[Rename Section 7.2 to "PKM protocol version 1 "]

[Insert new section 7.3 as follows]

7.3 PKM protocol version 2

PKM version 2 differs from PKM version 1 in the following respects:

- mutual authentication is required with both RSA and EAP
- AES-based key derivation functions
- tbd

7.3.1 PKMv2 Key hierarchy

7.3.2 PKMv2 RSA authentication

7.3.3 PKMv2 EAP-based Authentication

7.3.3.1 EAP Authentication Exchanges on Network Entry

At network entry, the BS and MSS determine capabilities using SBC-REQ/SBC-RSP as described in xx.

Upon negotiation of PKMv2, the EAP entity on the BS begins sending EAP messages to the MSS in order to authenticate its identity (except in the case where the BS has a PMK cached for the MSS, as described in 7.3.1.4).

The Privacy sublayer at the BS encapsulates the messages it receives from its EAP entity into EAP-Transfer-Reply PKM-RSP PDUs and transmits them to the MSS on the Basic CID. Upon receipt of the PDUs, the privacy sublayer on the MSS unpacks the EAP messages that it receives and passes them to its EAP entity. Finally, the MSS encapsulates EAP messages sent by its EAP entity into EAP-Transfer-Request PKM-REQ PDUs for transmission on the Basic CID.

EAP messages are passively carried by the Privacy Sublayer and do not affect MAC state.

The exchange of encapsulated EAP messages continues until either:

- EAP-based authentication is successful
- EAP-based authentication fails due to inadequate credentials or other errors

7.3.1.2.1 Successful Authentication

Successful authentication has occurred when the EAP entity on the BS sends an EAP-Success message to the SS. Upon successful authentication, both EAP entities (ie. at BS and MSS) supply the local Privacy Sublayer with a 256-bit Shared Secret called the **Pairwise Master Key** (PMK). The PMK is computed or established by the two EAP entities during the execution of the EAP method.

7.3.1.2.1.1 PMK Naming

Upon reception of the PMK from the EAP entity, the 802.16 Privacy Layer must compute a “handle” for the PMK.

The handle is called the **PMKId**, and is derived using the following formulas:

If SS and BS have negotiated use of AES-based key derivation functions:

PMKId = Truncate128(OMAC(PMK, "PMK Name" || BSId || SSId))

If SS and BS have **not** negotiated use of AES-based key derivation functions:

PMKId = HMAC-SHA1-128 (PMK, "PMK Name" || BSId || SSId)

where || denotes string concatenation

7.3.1.2.2 Authentication Failure

When the EAP entity on the BS determines that authentication has failed (eg. due to an invalid credential), it instructs the MAC layer to deactivate the CIDs associated with the MSS. Hence the MSS must reinitiate Ranging if it wishes to try again to enter the network.

7.3.1.3 EAP-based Authorization in PKMv2

Authorization is the process of the BS and MSS (who already share a PMK) reconfirming each other's identity, and establishing a 160-bit **Authorization Key** (AK).

The Authorization exchange is as follows:

1) BS sends the EAP-Establish-Key-Request PKM message (including the PMKId and a 32-byte nonce) to the SS. The SS then generates its own 32-byte nonce, and derives a Transient Key (TK) as follows:

TK = PRF-384(*PMK*, "Pairwise key expansion",
Min(*BSId, SSId*) |
Max(*BSId, SSId*) |
Min(*BS-Generated-Nonce, SS-Generated-Nonce*) |
Max(*BS-Generated-Nonce, SS-Generated-Nonce*))

1)

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.

2) The SS 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

3) SS 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.

4) 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.

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

7.3.1.4 PMK Caching

A BS may maintain a cache of PMKs and their associated SS_IDs. The BS should cache PMKs associated with MSSes that are expected to arrive due to handover, and may also cache the PMKs belonging to MSSes that have recently disconnected.

7.3.1.5 Fast MSS Authorization

Upon successful ranging by an MSS, a BS may determine that it has a PMK associated with the claimed MAC address in its PMK cache. It may then attempt to use the cached PMK to establish an AK, and avoid a full authentication.

To do this, the BS initiates *Fast MSS authorization* by sending **EAP-Establish-Key-Request** including the PMKID attribute, which identifies by name the Pairwise Master Key that the SS should use for authorization if it also has the PMK cached: If the MSS does not have the PMK cached, it responds with **EAP-Establish-Key-Reject**, and the BS should initiate full authentication.

7.3.2 Authentication and Authorization Pairwise State Machine

The State Machine is pairwise – ie. a BS or MSS can support multiple state machines, each of which pertains to its authentication/authorization state to a particular communication partner.

7.3.2.1 States

1. Initializing
2. unauthenticated
3. Authenticated
4. AuthenticationRejected
5. AuthorizationInProgress
6. Authorized
7. Departed

7.3.2.2 Messages

These are the PKM msgs.

7.3.2.3 Events

- AuthenticationSuccess signal from EAP entity
- AuthenticationFailure signal from EAP entity

7.3.2.4 Parameters

- PartnerId (ie. BS_Id or MSS_Id)
- PMK
- Authorization exchange timers

7.3.2.5 Actions

These accompany state transitions:

1) Initializing --> unauthenticated
    a. Send “connection up” indication to EAP entity

2) Unauthenticated --> departed
    a. Send “connection down” indication to EAP entity

3) Unauthenticated --> AuthenticationRejected
    a. Take down the connection/CID etc.

[Add Annex F, G]

Annex F – EAP method requirements

EAP methods used for 802.16e should be selected with awareness of the security issues described in [IETF RFC 3748] section 7. The considerations described in [IETF Draft "EAP Method Requirements for Wireless LANs"] apply for an 802.16e environment also.

Annex G – Security Claims

TBD, Includes:

- Security claims
- Description of security improvements of v2 over v1