Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**
Title	**Corrections for the PKMv2 Key Hierarchy**
Data Submitted	**2005-04-27**

Source(s)	Seokheon Cho Sungcheol Chang Chulsik Yoon, ETRI 161, Gajeong-dong, Yuseong-Gu, Daejeon, 305-350, Korea	Voice: +82-42-860-5524 Fax: +82-42-861-1966 chosh@etri.re.kr

Re:	IEEE P802.16e/D7
Abstract	The existing PKMv2 is somewhat unorganized and insecure security framework. This contribution provides a resolution for secure and uniform key hierarchy in the PKMv2.
Purpose	Adoption of proposed changes into P802.16e/D7
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:chiar@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>.

Corrections for the PKMv2 Key Hierarchy

Seokheon Cho, Sungcheol Chang, and Chulsik Yoon
ETRI

Introduction

The existing PKMv2 is somewhat in disorder and provides unorganized and insecure security framework.
This contribution supports the backward compatibility with the PKMv1 and security framework of the PKMv2.

This contribution provides a resolution for those problems in the PKMv2.

0.1 IEEE P802.16e/D7 Status

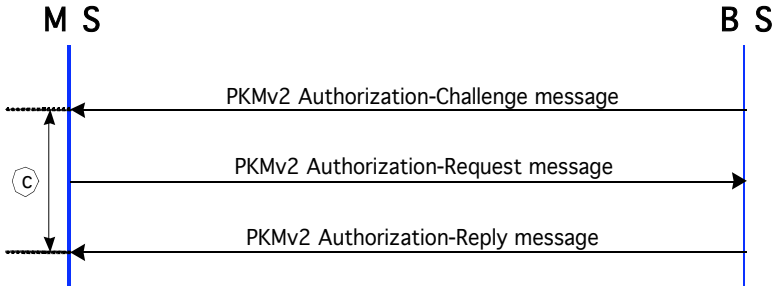
The Key Hierarchy for the PKMv2 is defined. The AK is derived by PAK or/and PMK, SSID, BSID, and so on.

0.2 Problems

- The AK is derived from PAK or/and PMK which are generated and distributed from the BS and Authenticator, respectively. A Nonce from MS as well as BS is necessary to generate AK so as to make more secure key generation mechanism.
- The input key used for AK generation is the only PMK. The PAK should be also used to generate the AK as not input data but an input key.
- The value of EAP session-id is not changed, even though the new AAA-key is refreshed. That is, even if PMK is updated, the value of PMKID (=> hash64(EAP session-id)) and AKID (=> hash64(EAP sessionid|PAKID|BSID)) is not also changed. Therefore, AKID is unsuitable as the identifier or sequence number needed to distinguish new AK from old AK.
- The AK lifetime is computed from the value (=> MIN(PAK lifetime, PMK lifetime)). To maintain AK more secure, however, the AK should be frequently refreshed. Different definition of the AK lifetime is necessary.

0.3 Solutions

a) To derive AK, 3 way handshake procedure is newly provided as follows.
   - The MS_Nonce and the BS_Nonce generated from the MS and the BS respectively. These MS_Nonce and BS_Nonce with PAK or/and PMK shall be used to generate AK.



   i. PKMv2 Authorization-Challenge message: BS_Nonce
   ii. PKMv2 Authorization-Request message: Key Sequence Number (PAK), MS_Nonce, BS_Nonce, Security_Capabilities, SAID, OMAC Digest (from AK)
   iii. PKMv2 Authorization-Reply message: Key Sequence Number (AK), Key Lifetime (AK), BS_Nonce, (one or more) SA-Descriptor(s), OMAC Digest (from AK)
   iv. PKMv2 Authorization-Reject message: Error-Code, Display-String, BS_Nonce, OMAC Digest (from AK)

   - The input key for generating the AK should be both PAK and PMK. The exclusive-or (XOR: ⊕) value of PAK and PMK as input key is used to generated the AK. The generation method of the AK is as follows.

```
If (RSA-based authorization and EAP-based authorization)
        AK <= Dot16KDF (PAK⊕PMK, SS_NONCE|BS_ NONCE|SSID|BSID|“AK”, 160)
Else if (RSA-based authorization)
        AK <= Dot16KDF (PAK, SS_ NONCE|BS_ NONCE|SSID|BSID|“AK”, 160)
Else if (EAP-based authorization)
        AK <= Dot16KDF (PMK, SS_ NONCE|BS_ NONCE|SSID|BSID|“AK”, 160)
```

   - The new key hierarchy is as follows.

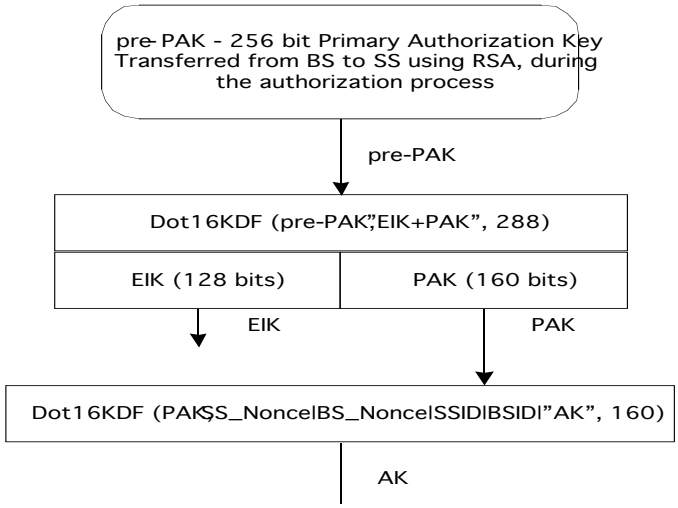


Figure -AK with the only RSA-based authorization process

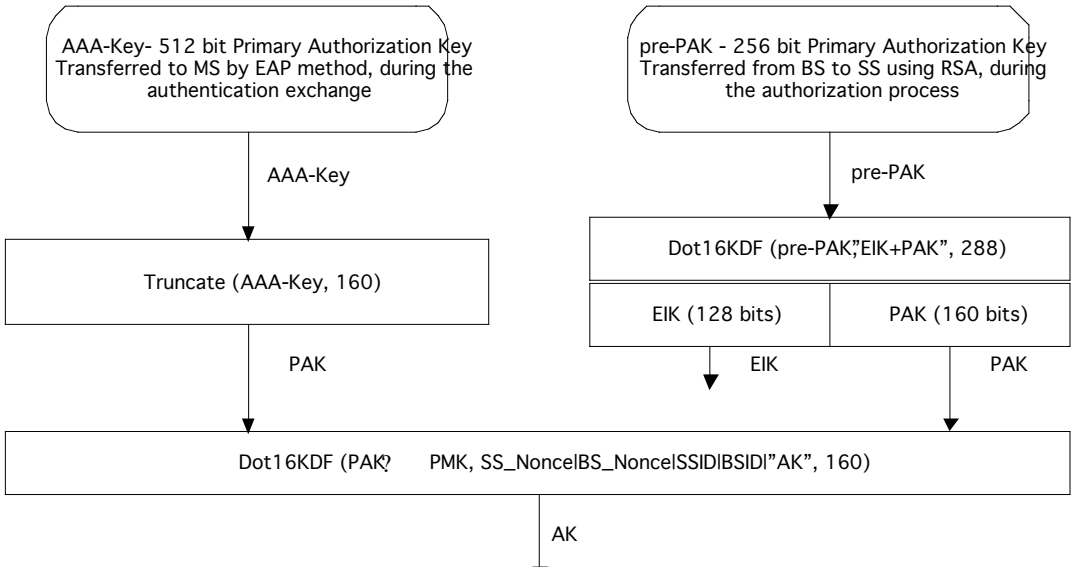


Figure -AK with RSA and EAP authorization process

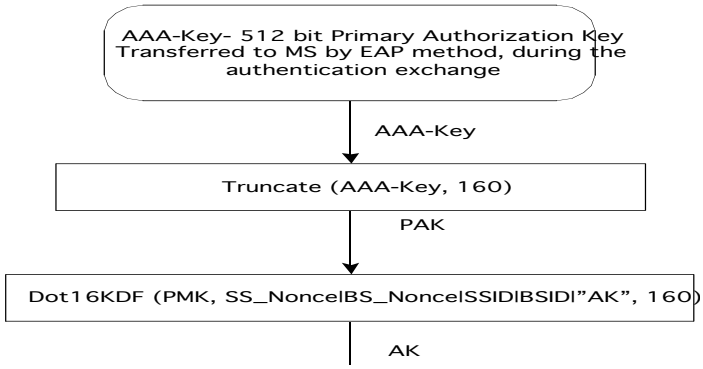


Figure -AK with the only EAP-based authorization process

- To solve the AKID, the AK sequence number as AK identifier is newly defined. The BS generates the AK sequence number and informs it to MS, whenever the AK is updated.
- To maintain AK more secure, the AK has AK lifetime which is assigned from the BS. That is, the MS shall request the new AK, before the old AK expires and after the PAK or the PMK is updated.
- The AK context is as follows.

Table -AK Context in PKMv2

Parameter	Size	Usage
Primary AK (PAK)	160bits	A key yielded from the RSA-based authorization.
PAK Sequence Number	64bits	PAK sequence number, when the RSA-based authorization is achieved.
PAK lifetime		PAK sequence number, when the RSA-based authorization is achieved.
PMK	160bits	A key yielded from the EAP-based authentication (only if EAP protocol generates the AAA-key)..
PMK lifetime		PMK sequence number, when the EAP-based authorization is achieved and the AAA-key is obtained.
AK	160bits	The authorization key, calculated as defined in 7.2.2.2.3
AK Sequence Number	64bits	AK sequence number
AK lifetime		AK lifetime – when this expires, MS's Re-authorization Key process is needed.
H/OMAC_KEY_U	160 bits/128 bits	The key which is used for signing UL management messages.
H/OMAC_PN_U	32 bits	Used to avoid UL replay attack on management messages – when this expires re-authentication is needed.
H/OMAC_KEY_D	160 bits/128 bits	The key which is used for signing DL management messages.
H/OMAC_PN_D	32 bits	Used to avoid DL replay attack on management messages – when this expires re-authentication is needed.
KEK	160 bits	Used to encrypt TEK or GKEK from the BS to the SS.

Proposed Changes into IEEE P802.16e/D7

[Change 7.2.2.2: as follows]

7.2.2.2.1 ~~Certificated RSA authorization~~ 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.

The pre-PAK (Primary Authorization Key) is sent by the BS to the MS encrypted with the public key from the certificate. Pre-PAK is mainly used to generate the PAK. The optional EIK for ~~EAP exchange~~ the PKMv2 Authenticated EAP-Transfer message (see 7.2.2.2.2) are also generated from pre-PAK:

~~|~~EIK | PAK = Dot16KDF(pre-PAK, ~~SSID |~~ "EIK+PAK", 288)

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~~ EAP-based authorization

There are two kinds of EAP-based authorization; only EAP exchange way (using the PKMv2 EAP-Transfer message) and EAP exchange way based on RSA exchange (using the PKMv2 Authenticated EAP-Transfer message).

In case of the only EAP exchange way, the MS's user authentication is achieved by transferring only EAP payload between a MS and the BS.

Contrary to the only EAP exchange way, in case of the EAP exchange way based on RSA exchange, the MS's user authentication is executed by exchanging PKMv2 Authenticated EAP-Transfer messages. ~~If a mutual authorization took place before the EAP exchange, the EAP messages~~ These messages may be protected using EIK~~ - EAP Integrity Key~~ (EAP Integrity Key) derived from pre-PAK (see 7.2.2.2.1). EIK ~~and EEK are~~ is 128 bits long.

The product of the EAP exchange which is transferred to ~~802.16~~ MAC privacy sub-layer is the AAA-key. 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 MS. The MS and the authenticator (the serving BS or certain network node) derive a PMK (Pairwise Master Key) by truncating the AAA-key after 160 bits.

The PMK derivation from the AAA-key is as follows:

PMK = truncate (AAA-key, 160)

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

7.2.2.2.3 Authorization Key (AK) derivation

The AK will be derived by the authenticator and the MSS from the PMK (from EAP exchange) and the PAK (from RSA exchange). ~~Note that PAK can be used only in initial network entry. In cases of HO and re-authentication: Only EAP keys are applicable.~~ Note that PAK or/and PMK can be used according to the value of Authorization Policy Support field included in the SBC-REQ/RSP messages. The authorization policy shall be negotiated between MS and BS before achieving the authorization procedure, irrespective of case of initial network entry, reentry, and HO.

The exclusive-or (XOR: ⊕) value of PAK and PMK is mainly used to generate the AK. The only PAK is used to derive the AK in case of achieving RSA-based authorization procedure. On the contrary, the only PMK is used in case of executing EAP-based authorization procedure.

~~If (PAK and PMK)~~
~~AK <= Dot16KDF (PMK, SSID|BSID|PAK|"AK", 160)~~
~~Else~~
~~If (PAK)~~
~~AK <= Dot16KDF (0, SSID|BSID|PAK|"AK", 160)~~
~~Else~~
~~AK <= Dot16KDF (PMK, SSID|BSID| "AK", 160);~~
~~Endif~~

~~Endif~~

If (RSA-based authorization and EAP-based authorization)
AK <= Dot16KDF (PAK⊕PMK, SS_NONCE|BS_ NONCE|SSID|BSID|"AK", 160)
Else if (RSA-based authorization)
AK <= Dot16KDF (PAK, SS_ NONCE|BS_ NONCE|SSID|BSID|"AK", 160)
Else if (EAP-based authorization)
AK <= Dot16KDF (PMK, SS_ NONCE|BS_ NONCE|SSID|BSID|"AK", 160)

7.2.2.2.7 Group Traffic Encryption Key (GTEK)

The GTEK is used to encrypt multicast data packets and it is shared between all MSSs that belong to the multicast group. There are 2 GTEKs per GSA.

The GTEK is randomly generated at the BS and is encrypted using ~~AES_KEY_WRAP~~ same algorithms applied to TEK encryption and transmitted to the MS in multicast or unicast messages. ~~In multicast the message will be encrypted by the GKEK. In unicast, it will be encrypted by the KEK.~~ The GTEK will be encrypted by the GKEK.

7.2.2.2.10 Key Hierarchy

Figure 131 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 AAA-key:

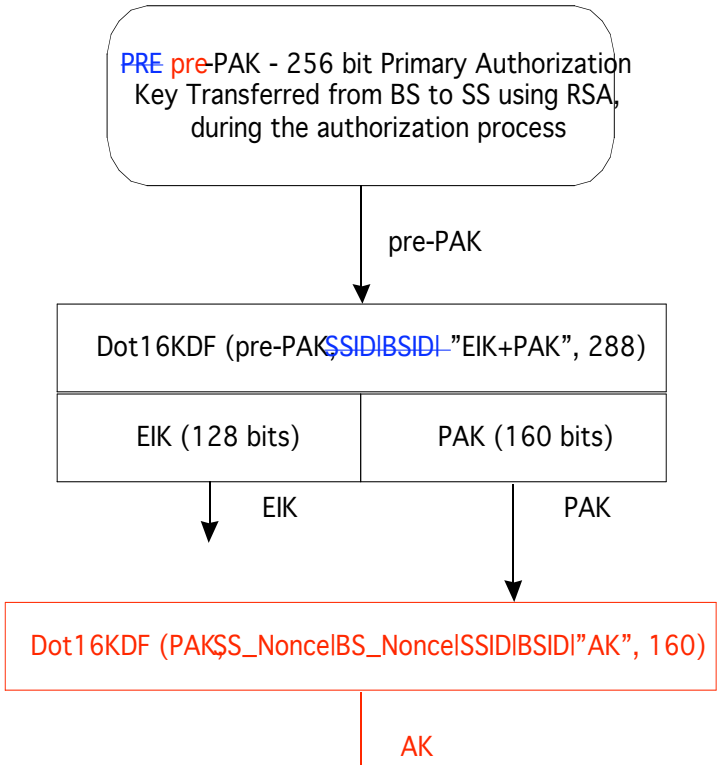


Figure 131-AK with the only RSA-based ~~only~~ authorization process

Figure 132 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 AAA-key:

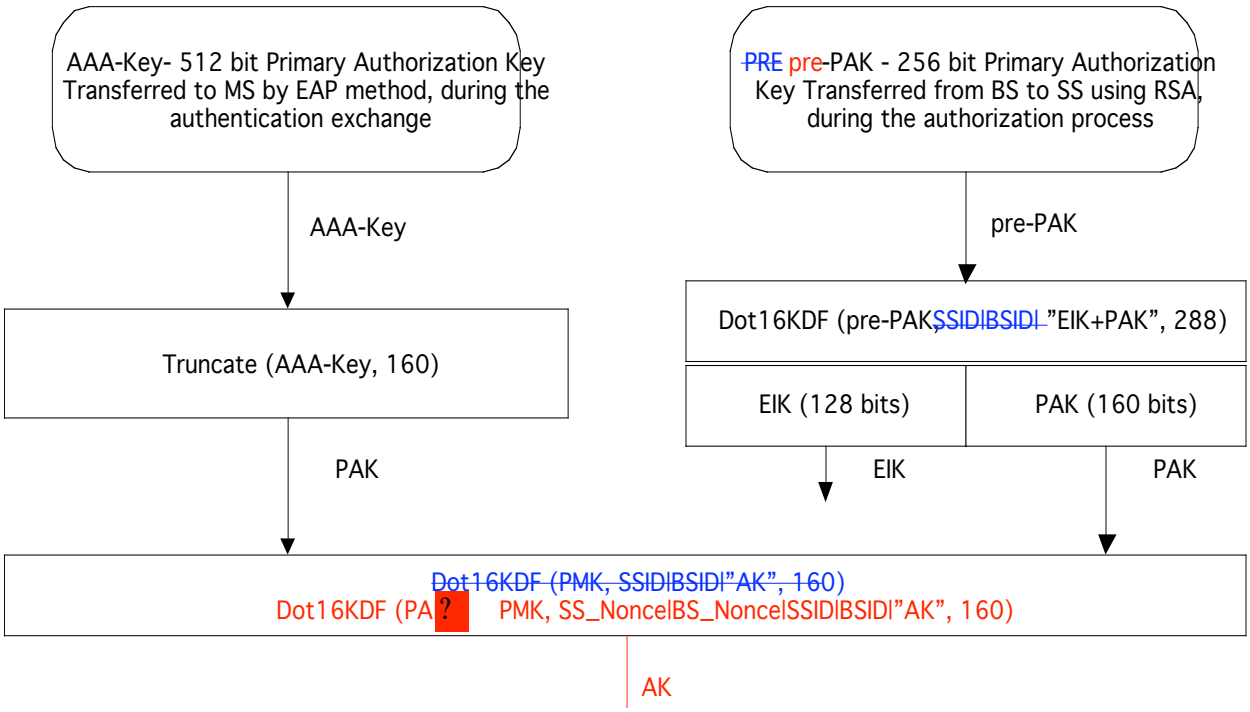


Figure 132-AK with RSA and EAP authorization process

Figure 133 outlines the process to calculate the AK when only the EAP based authentication exchange has taken place, yielding an AAA-key:

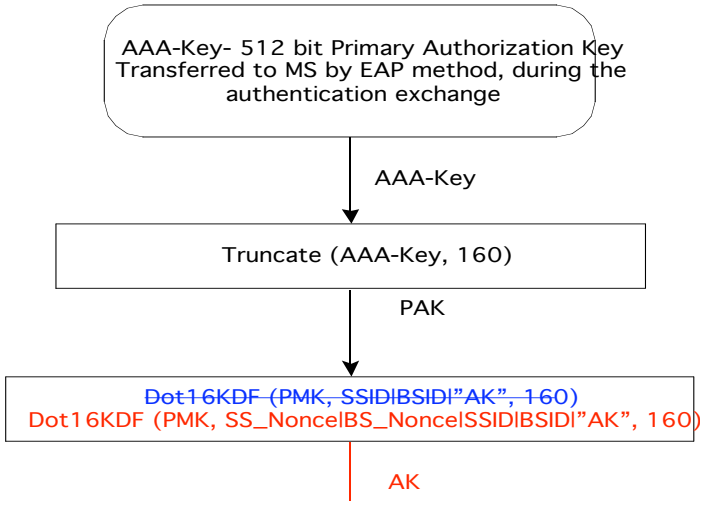


Figure 133-AK with the only EAP-based ~~only authentication~~ authorization process

[Change 7.2.2.4.1 as follows]

7.2.2.4.1 AK Context

The context of AK includes all the parameters connected to AK and keys derived directly from it.

When one parameter from this context expires, a new AK should be obtained in order to start a new context.

Obtaining of new AK means re-authentication - doing ~~the whole EAP and/or PAK~~ the RSA-based authorization procedure or/and

the EAP-based authorization procedure due to ~~the authorization policies~~ the value of the Authorization Policy Support field negotiated between the MS and BS until obtaining a new PMK and/or PAK which AK may be derived from.

Derivation of AK after HO is done separately in the MS and network from ~~a common~~ PMK, PAK, SS_Nonce, BS_Nonce, SSID, and BSID. ~~The PMK and/or PAK may be used to derive keys to several BSs sharing the same PMK and/or PAK.~~ The same PAK or/and PMK can be shared among several BSs.

In HO scenario, if the MS was previously connected to the TBS, the derived AK will be identical to the last one, as long as the PAK or PMK stays the same. In order to maintain security in this scenario: the context of the AK must be cached by both sides and to be used from the point it stopped, if context lost by one side, re-authentication is needed to establish new PAK, PMK and new AK context.

The AK context is described in the table:

Table 133-AK Context in PKMv2

Parameter	Size	Usage
Primary AK (PAK)	160bits	A key yielded from the ~~mutual authorization exchange~~ RSA-based authorization. ~~Only present at initial network entry and only if the certificated RSA exchange took place, as a result of the mutual authorization policy negotiation.~~
~~PAKID~~	~~64bits~~	~~Derived from the mutual authorization, present when PAK is present.~~
PAK Sequence Number	8bits	PAK sequence number, when the RSA-based authorization is achieved.
PAK lifetime		~~Derived from the mutual authorization, present when PAK is present.~~ PAK lifetime, when the RSA-based authorization is achieved.
PMK	160bits	A key yielded from the EAP-based authentication (only if EAP protocol generates the AAA-key)..
PMK lifetime		~~The lifetime of PMK derived from EAP.~~ PMK sequence number, when the EAP-based authorization is achieved and the AAA-key is obtained.
~~PMKID~~	~~64bits~~	~~hash 64(EAP session-id)~~
AK	160bits	~~The authentication key, calculated as f(PAK,PMK), if only EAP, AK=f(PMK).~~ The authorization key, calculated as defined in 7.2.2.2.3
~~AKID~~	~~64bits~~	~~Calculated according to the keys that contributed to AK:~~ ~~If AK=f(PMK,PAK) then AKID=hash 64(EAP sessionid | PAKID | BSID)~~ ~~If AK=f(PMK) then AKID=hash 64(EAP session-id | BSID)~~ ~~If AK=PAK then AKID = PAKID~~
AK Sequence Number	8bits	AK sequence number
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.~~ AK lifetime – when this expires, MS's Re-authorization Key process is needed.
H/OMAC_KEY_U	160 bits/128 bits	The key which is used for signing UL management messages.
H/OMAC_PN_U	32 bits	Used to avoid UL replay attack on management messages – when this expires re-authentication is needed.
H/OMAC_KEY_D	160 bits/128 bits	The key which is used for signing DL management messages.
H/OMAC_PN_D	32 bits	Used to avoid DL replay attack on management messages – when this expires re-authentication is needed.
KEK	160 bits	Used to encrypt ~~transport keys~~ TEK or GKEK from the BS to the SS.