802.16 security changes

**2. Normative references**

|  |  |
| --- | --- |
| **802.16t modification Instructions** | **802.16t modified paragraphs** |
| ***[Insert “FIPS 198-1, The keyed HASH Message Authentication Code (HMAC), July 2008” after “FIPS******197, Advanced Encryption Standard (AES).” in Clause 2]*** | FIPS 198-1, The keyed HASH Message Authentication Code (HMAC), July 2008.  |
| ***[Remove “IETF RFC 2104, “HMAC: Keyed-Hashing for Message Authentication,” Krawczyk, H.,******Bellare, M., and Canetti, R., Feb. 1997” from Clause 2]*** | IETF RFC 2104, “HMAC: Keyed-Hashing for Message Authentication,” Krawczyk, H., Bellare, M., and Canetti, R., Feb. 1997.  |

**6. MAC common part sublayer**

|  |  |  |
| --- | --- | --- |
| **Page number as per 802.16-2017** | **802.16t modification Instructions** | **802.16t modified paragraphs** |
| *238-**239* | ***[Replace “PKMv2 ECC-Request” with “PKMv2 RSA/ECC-Request” in the 2nd paragraph after “PKM identifier” in Clause 6.3.2.3.8.]******[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in the 4th paragraph after “PKM identifier” in Clause 6.3.2.3.8]******[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in the 6th paragraph after “PKM identifier” in Clause 6.3.2.3.8]*** | The SS shall increment (modulo 256) the Identifier field whenever it issues a new PKM message. In PKMv1, a “new” message is an Authorization Request or Key Request that is not a retransmission being sent in response to a Timeout event. In PKMv2, a PKMv2 RSA/ECC-Request, PKMv2 SA-TEK-Challenge, or PKMv2 Key-Request message is a “new” message. For retransmissions, the Identifier field shall remain unchanged.…On reception of a PKM-RSP message, the SS associates the message with a particular state machine (the Authorization state machine in the case of Authorization Replies, Authorization Rejects, and Authorization Invalids for the PKMv1, PKMv2 RSA/ECC Reply, PKMv2 RSA/ECC Reject,PKMv2 EAP Transfer, PKMv2 SA-TEK-Challenge, PKMv2 SA-TEK-Response for the PKMv2; a particular TEK state machine in the case of Key Replies, Key Rejects, and TEK Invalids the PKMv1, PKMv2-Key-Reply, PKMv2-Key-Reject, PKMv2 TEK-Invalids, and PKMv2 Group-Key-Update-Command messages for the PKMv2).--In PKMv2, both an SS and a BS shall keep track of their latest ID. An SS shall keep track of the ID of its latest, pending PKMv2 RSA/ECC-Request. The SS shall discard PKMv2 RSA/ECC-Reply and PKMv2 RSA/ECC-Reject messages with Identifier fields not matching that of the pending PKMv2 RSA/ECC-Request. Moreover, a BS shall keep it, pending PKMv2 RSA/ECC-Reply. The BS shall discard PKMv2 RSA/ECC-Acknowledgment messages with Identifier fields not matching that of the pending PKMv2 RSA/ECC-Reply.---An SS shall keep track of the ID of its latest, pending PKMv2 RSA/ECC-Request. The SS shall discard PKMv2 RSA/ECC-Reply and PKMv2 RSA/ECC-Reject messages with Identifier fields not matching that of the pending PKMv2 RSA/ECC-Request. Moreover, a BS shall keep it, pending PKMv2 RSA/ECC-Reply. The BS shall discard PKMv2 RSA/ECC-Acknowledgment messages with Identifier fields not matching that of the pending PKMv2 RSA/ECC-Reply. |
| *239* | ***[Replace “RSA” with “RSA/ECC” in Table 6-69]*** |

|  |  |
| --- | --- |
| 13 | PKMv2 RSA/ECC-Request  |
| 14 | PKMv2 RSA/ECC-Reply  |
| 15 | RSA/ECC-Reject  |
| 16 | RSA/ECC-Acknowledgment  |

 |
| *241* | ***[Replace “RSA public key” with “public key (RSA or ECC)” in 1st paragraph after Table 6-71]*** | The SS-Certificate attribute contains an X.509 SS certificate (see 7.6) issued by the SS’s manufacturer. The SS’s X.509 certificate is a public-key certificate that binds the SS’s identifying information to its public key (RSA or ECC) in a verifiable manner. The X.509 certificate is digitally signed by the SS’s manufacturer, and that signature can be verified by a BS that knows the manufacturer’s public key. The manufacturer’s public key is placed in an X.509 certification authority (CA) certificate, which in turn is signed by a higher level CA. |
| *246* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Clause 6.3.2.3.9.11]******[Replace “RSA-based authorization” with “RSA-based or ECC-based authorization” in Clause******6.3.2.3.9.11]*** | **6.3.2.3.9.11 PKMv2 RSA/ECC-Request message**A client MS sends a PKMv2 RSA/ECC-Request message to the BS in order to request mutual authentication in the RSA-based or ECC-based authorization. |
| *246* | ***[Replace “PKMv2 RSA-Request” with “PKMv2 RSA/ECC-Request” in Table 6-80]******[Replace “RSA signature” with “RSA signature or ECC signature” in Table 6-80]*** | **Table 6-80— PKMv2 RSA/ECC-Request message attributes**

|  |  |
| --- | --- |
| **Attribute** | **Contents** |
| SigSS  | An RSA or ECC signature over all the other attributes in the message |

 |
| *246* | ***[Replace “RSA signature” with “RSA signature or ECC signature” in 2nd paragraph after Table 6-80]*** | The SigSS attribute indicates a RSA signature or ECC signature over all the other attributes in this message, and the MS’s private key is used to make a RSA signature or ECC signature. |
| *246* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Clause 6.3.2.3.9.12]*** | **6.3.2.3.9.12 PKMv2 RSA/ECC-Reply message**Sent by the BS to a client MS in response to a PKMv2 RSA/ECC-Request message, the PKMv2 RSA/ECC-Reply message contains an encrypted pre-primary authorization key (pre-PAK), the key’s lifetime, and the key’s sequence number. The pre-PAK shall be encrypted with the MS’s public key. The MS Random number is returned from the PKMv2 RSA/ECC-Request message, along with a random number supplied by the BS, thus enabling assurance of key liveness. |
| *247* | ***[Replace “PKMv2 RSA-Reply” with “PKMv2 RSA/ECC-Reply” in Table 6-81]******[Replace “RSA-OAEP-Encrypt” with “RSA/ECC-OAEP-Encrypt” in Table 6-81]******[Replace “RSA signature” with “RSA signature or ECC signature” in Table 6-81]*** | **Table 6-81— PKMv2 RSA/ECC-Reply message attributes**

|  |  |
| --- | --- |
| **Attribute** | **Contents** |
| Encrypted pre-PAK | RSA/ECC-OAEP-Encrypt(PubKey(MS), pre-PAK | MS MAC Address) |
| … | … |
| SigBS | An RSA signature or ECC signature over all the other attributes in the message |

 |
| *247* | ***[Replace “RSA signature” with “RSA or ECC signature” in 1st paragraph after Table 6-81]*** | The SigBS attribute indicates a RSA signature or ECC signature over all the other attributes in this message, and the BS’s private key is used to make a RSA signature or ECC signature. |
| *247* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Clause 6.3.2.3.9.13]******[Replace “RSA-based” with “RSA/ECC-based” in Clause 6.3.2.3.9.13]*** | **6.3.2.3.9.13 PKMv2 RSA/ECC-Reject message**The BS responds to an SS’s authorization request with a PKMv2 RSA/ECC-Reject message if the BS rejects the SS’s authorization request. When an MS receives this message, an MS may retransmit the PKMv2 RSA/ECC-Request message or quit RSA/ECC-based mutual authentication. |
| *247* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Table 6-82]******[Replace “RSA signature” with “RSA signature or ECC signature” in Table 6-82]*** | **Table 6-82—PKMv2 RSA/ECC-Reject message attributes**

|  |  |
| --- | --- |
| **Attribute** | **Contents** |
| SigBS | An RSA signature or ECC signature over all the other attributes in the message |

 |
| *247* | ***[Replace “RSA-based” with “RSA/ECC-based” in paragraph 1-2 after Table 6-82]******[Replace “RSA signature” with “RSA signature or ECC signature” in paragraph 1-2 after Table 6-82]*** | The Error-Code and Display-String attributes describe to the requesting MS the reason for the RSA/ECC-based authorization failure. The SigBS attribute indicates a RSA signature or ECC signature over all the other attributes in this message, and the BS’s private key is used to make a RSA signature or ECC signature. |
| *248* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Clause 6.3.2.3.9.14]*** | **6.3.2.3.9.14 PKMv2 RSA/ECC-Acknowledgment message**The MS sends the PKMv2 RSA/ECC-Acknowledgment message to BS in response to a PKMv2 RSA/ECC-Reply message. Only if the value of the Auth Result Code attribute is failure, then the Error-Code and Display-String attributes can be included in this message. |
| *248* | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Table 6-83]******[Replace “RSA signature” with “RSA signature or ECC signature” in Table 6-83]*** | **Table 6-83—PKMv2 RSA/ECC-Acknowledgment message attributes**

|  |  |
| --- | --- |
| **Attribute** | **Contents** |
| SigBS | An RSA signature or ECC signature over all the other attributes in the message |

 |
| *248* | ***[Replace “RSA” with “RSA/ECC” in 1st paragraph after Table 6-83]*** | The SigSS attribute indicates a RSA/ECC signature over all the other attributes in this message, and the SS’s private key is used to make a RSA/ECC signature. |
| *366* | ***[Replace “Bit 2: Reserved” with “Bit 2: 256-bit HMAC/CMAC” from Table 6-173, parameter:* HO\_authorization\_policy\_support*]*** |

|  |  |  |
| --- | --- | --- |
| **HO\_authorization\_policy\_support** | 8 | Bit 0: RSA authorizationBit 1: EAP authorizationBit 2: ~~Reserved~~ 256 bit HMAC/CMACBit 3: HMAC supportedBit 4: CMAC supportedBit 5: 64-bit Short-HMACBit 6: 80-bit Short-HMACBit 7: 96-bit Short-HMAC |

 |
| *382* | ***[Replace “Bit 2: Reserved” with “Bit 2: 256-bit HMAC/CMAC” from Table 6-175, parameter*: HO\_authorization\_policy\_support*]*** |

|  |  |  |
| --- | --- | --- |
| **HO\_authorization\_policy\_support** | 8 | Bit 0: RSA authorizationBit 1: EAP authorizationBit 2: ~~Reserved~~256 bit HMAC/CMACBit 3: HMAC supportedBit 4: CMAC supportedBit 5: 64-bit Short-HMACBit 6: 80-bit Short-HMACBit 7: 96-bit Short-HMAC |

 |

**7. Security sublayer**

|  |  |  |
| --- | --- | --- |
| **Page** | **Instruction** | **Changes** |
| *841* | **[*Replace “*— RSA-based Authentication: This stack performs the RSA-based authentication function using the SS’s X.509 digital certificate and the BS’s X.509 digital certificate, when the RSA-based authorization is selected as an authorization policy between an SS and a BS.” with *“*— RSA/ECC-based Authentication: This stack performs authentication function using the SS’s X.509 digital certificate and the BS’s X.509 digital certificate, using RSA algorithm or ECC algorithm, when the RSA/ECC-based authorization is selected as an authorization policy between an SS and a BS. Note that ECC offers greater cryptographic strength for equivalent key size.” in clause 7.1 on p. 841]*****[Replace “when the RSA-based” with “when the RSA/ECC-based” in the 9th paragraph in Clause 7.1]*** | RSA/ECC-based Authentication: This stack performs the RSA/ECC-based authentication function using the SS’s X.509 digital certificate and the BS’s X.509 digital certificate **using either RSA algorithm or ECC algorithm**, when the RSA/ECC-based authorization is selected as an authorization policy between an SS and a BS. |
| *842* | ***[Replace “RSA” with “RSA/ECC” in Figure 7-1]*** |  |
| *842* | ***[Replace “PKCS #1” with “PKCS #1 v2.2], ECC public-key encryption algorithm [FIPS 186-5]” in 1st******paragraph of Clause 7.1.2]******[Replace “starting with RSA” with “starting with RSA/ECC” in 1st paragraph of Clause 7.1.2]******[Replace “RSA authentication” with “RSA/ECC authentication” in 3rd paragraph of Clause 7.1.2]*** | 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 v2.2], ECC public-key encryption algorithm [FIPS 186-5] or a sequence starting with RSA/ECC authentication and followed by EAP authentication. It uses strong encryption algorithms to perform key exchanges between an SS and BS.…A BS authenticates a client SS during the initial authorization exchange. Each SS presents its credentials, which shall be a unique X.509 digital certificate issued by the SS’s manufacturer (in the case of RSA/ECC authentication) or a operator-specified credential (in the case of EAP-based authentication). |
| *843* | ***[Replace “RSA protocol” with “RSA/ECC protocol, RSA-based” in 3rd paragraph of Clause 7.1.3]******[Replace “RSA protocol” with “RSA/ECC protocol, RSA-based” in 3rd paragraph of Clause 7.1.3]*** | — RSA/ECC protocol, RSA-based [PKCS #1 v2.1 with SHA-1(FIPS 186-2)] (support is mandatory in PKMv1; support is optional in PKMv2, with RSA [PKCS#1 v2.2 with SHA-x, where x is 2 or higher (FIPS 186-5)] or ECC-based [FIPS 186-5] |
| *843* | ***[Replace “PKM RSA authentication” with “PKM RSA/ECC authentication” in 1st paragraph of Clause 7.1.3.1]******[Replace “PKCS #1]” with “PKCS #1, v2.2] or the ECC key encryption [FIPS 186-5]” in the 1st paragraph of Clause 7.1.3.1]*** | The PKM RSA/ECC authentication protocol uses X.509 digital certificates [IETF RFC 3280], the RSA public key encryption algorithm [PKCS #1,v2.2] or the ECC key encryption [FIPS 186-5] that binds public RSA/ECC encryption keys to MAC addresses of SSs.  |
| 843 | ***[Replace “RSA authentication” with “RSA/ECC authentication” in the 3rd paragraph of Clause 7.1.3.1]******[Replace “or provide” with “or ECC private/public key pairs or provide” in the 3rd paragraph of Clause 7.1.3.1]******[Replace “its RSA key pair” with “its public key pair” in the 3rd paragraph of Clause 7.1.3.1]******[Replace “installed RSA key pairs” with “installed RSA key pairs or ECC key pairs” in the 3rd paragraph of Clause 7.1.3.1]*** | All SSs using RSA/ECC authentication shall have factory-installed RSA private/public key pairs or ECC 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 public 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 or ECC key pairs shall also have factory-installed X.509 certificates. All SSs that rely on internal algorithms to generate a public key pair shall support a mechanism for installing a manufacturer-issued X.509 certificate following key generation. |
| 844 | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in the 1st paragraph of Clause 7.1.6.3]*** | Similar to other MAC management messages, all the PKM messages are exchanged between MS/RS and MR-BS. For the PKM messages that are not protected by the message authentication code from the MS/RS (termed as non-authenticated PKM messages, e.g., Auth Request, Auth Reply, PKMv2 RSA/ECC-Request, PKMv2 RSA/ECC-Reply), the following procedure may be applied. For all the other cases, the access RS and the intermediate RSs just simply relay the PKM messages. |
| 861 | ***[Replace paragraph 4 of Clause 7.2.2.1 with “TEK and KEKs is N bits long, where N may be either 128,******192 or 256. SAs employing any ciphersuite with a block size of N bits shall use B-bit TEKs and KEKs. The******name TEK-N is used to denote n-bit TEK. Similarly, KEK-N is used to denote a N-bit KEK.”]*** | TEKs and KEKs is N bits long, where N may be either 128, 192 or 256. SAs employing any ciphersuite with a basic block size of N bits shall use B-bit TEKs and KEKs. The name TEK-N is used to denote a n-bit TEK. Similarly, KEK-N is used to denote a N-bit KEK. |
| 861 | ***[Replace “128” with “N” in the 6th paragraph of Clause 7.2.2.1]*** | For SAs using a ciphersuite employing N bits keys, such as AES-CCM mode, the TEK in the Key Reply is AES encrypted using a N-bit key derived from the AK and a N-bit block size. |
| 862 | ***[Replace “RSA” with “RSA/ECC” in the 2nd paragraph of Clause 7.2.2.2]*** | Since there are two authentication schemes, one based on RSA/ECC and one based on EAP, there are two primary sources of keying material. |
| 862 | ***[Replace “RSA” with “RSA/ECC” in the 3rd paragraph of Clause 7.2.2.2]*** | The keys used to protect management message integrity and transport the TEKs are derived from source key material generated by the authentication and authorization processes. The RSA/ECC-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. |
| 862 | ***[Replace “RSA” with “RSA/ECC” in Clause 7.2.2.2.1]*** | **7.2.2.2.1 RSA/ECC-based authorization**When the RSA/ECC-based authorization is negotiated as authorization policy, the PKMv2 RSA/ECC-Request, the PKMv2 RSA/ECC-Reply, the PKMv2 RSA/ECC-Reject, and the PKMv2 RSA/ECC-Acknowledgment messages are used to share the pre-PAK. |
| 862 | ***[Replace “RSA” with “RSA/ECC” in Clause 7.2.2.2.2]*** | PAK shall be used to generate the AK (see 7.2.2.2.3) if RSA/ECC authorization was used. PAK is 160 bits long. |
| 863 | 7.2.2.2.2 | If a RSA/ECC mutual authorization took place before the EAP exchange, 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 BS and the SS will share the AK which is derived from the PMK (from EAP-based authorization procedure) and/or the PAK (from RSA/ECC-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. |
| 867 | ***[Replace “128) (Used for broadcast MAC), “ with “N) (Used for broadcast MAC), N=128, 192 or 256]” in******the 5th paragraph of Clause 7.2.2.2.9.2]*** | CMAC\_KEY\_GD ⇐ Dot16KDF(GKEK, “GROUP CMAC KEY”,N) (Used for broadcast MAC), N=128, 192 or 256 |
| 867 | ***[Replace “AES-128” with “AES-N” in the 5th paragraph of Clause 7.2.2.2.9.2]*** | Specifically, the preprocessed value of CMAC\_PREKEY\_\* is treated as the Cipher Key of the Advanced Encryption Standard (AES) algorithm AES-N (FIPS197). The CMAC\_KEY\_COUNT is treated as the Input Block Plain Text of this algorithm. The AES-N algorithm is executed once. The Output Block Cipher Text of this algorithm is treated as the resulting CMAC\_KEY\_\*. When CMAC\_KEY\_COUNT is used as an input of AES-N algorithm, 112 zero bits are prepadded before the 16-bit CMAC\_KEY\_COUNT where the |
| 868 | ***[Replace “HMAC\_KEY\_U | HMAC\_KEY\_D | KEK*** ⇐ ***Dot16KDF(AK, SS MAC Address | BSID |******“HMAC\_KEYS+KEK”, 448)” with “HMAC\_KEY\_U | HMAC\_KEY\_D | KEK ? Dot16KDF(AK, SS MAC******Address | BSID | “HMAC\_KEYS+KEK”, M+N)” in Clause 7.2.2.2.9.2 on p. 868]******Insert “The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively. Additionally, the value of N is the length of a KEK key.”******before the paragraph beginning with “Exceptionally, the message authentication keys” in Clause******7.2.2.2.9.2 on p. 868]******[Replace “,* HMAC\_KEY\_U | HMAC\_KEY\_D ⇐ Dot16KDF(EIK, SS MAC Address | BSID |“HMAC\_KEYS”,** ***320)” with “,* HMAC\_KEY\_U | HMAC\_KEY\_D ⇐ Dot16KDF(EIK, SS MAC Address | BSID |“HMAC\_KEYS”,** ***2N)” in Clause 7.2.2.2.9.2 on p. 868]******[Insert “The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively. Additionally, the value of N is the length of a KEK key.” before the paragraph beginning with “Exceptionally, the message authentication keys” in Clause******7.2.2.2.9.2 on p. 868]*** | HMAC\_KEY\_U | HMAC\_KEY\_D | KEK ⇐ Dot16KDF(AK, SS MAC Address | BSID | “HMAC\_KEYS+KEK”,M+N) …The value of M in the Dot16KDF function used for HMAC depends on the specific variant ofSHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, Mequals to 224, 256, 384, and 512, respectively. Additionally, the value of N is the length of a KEK key…HMAC\_KEY\_U | HMAC\_KEY\_D ⇐ Dot16KDF(EIK, SS MAC Address | BSID |“HMAC\_KEYS”, 2M)The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively |
| 868 | ***[Replace “RSA-based” with “RSA/ECC-based” in the 1st paragraph of Clause 7.2.2.2.10”]*** | Figure 7-4 outlines the process to calculate the AK when the RSA/ECC-based authorization process has taken place, but where the EAP based authentication process has not taken place, or the EAP method used has not yielded an MSK |
| 868 | ***[Replace “RSA” with “RSA/ECC” in Figure 7-4]*** | **AK from PAK only (from RSA/ECC-based authorization)** |
| 869 | ***[Replace “RSA” with “RSA/ECC” in Figure 7-5]*** | **AK from PAK and PMK (RSA/ECC-based and EAP-based authorization)** |
| 870 | ***[Replace “384” with “320+N” in Figure 7-7]******[Replace “KEK (128 bits)” with “KEK (N bits)” (two appearances) in Figure 7-7]******[Replace “448” with “2M+N” in Figure 7-7]******[Replace “160” with “M” in Figure 7-7]******[Insert “The value of M in the Dot16KDF function used for HMAC depends on the specific variant of******SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively. Additionally, the value of N is the length of a KEK key.” before “Figure 7-8 outlines the MBS key hierarchies starting from the MAK” in Clause 7.2.2.2.10 on p. 870]*** | The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively. Additionally, the value of N is the length of a KEK key.  |
| 870 | ***[Replace “128” with “128/192/256” in Figure 7-8]*** |  |
| 872 | ***[Replace “, 320)” with “, 2M)” in Clause 7.2.2.2.13 on p. 872]******[Replace “, 160)” with “, 2M)” in Clause 7.2.2.2.13 on p. 872]******[Insert “The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively.” as the final paragraph of Clause 7.2.2.2.13]*** | The security zone keys used for HMAC generation are derived as follows:HMAC\_KEY\_SZU|HMAC\_KEY\_SZD<= Dot16KDF (SZK, “SECURITY\_ZONE\_KEYS”, 2M)The CMAC\_KEY\_SZ used in SZK multicast update mode is derived as follows:CMAC\_KEY\_SZ<= Dot16KDF (SZKEK, “SECURITY\_ZONE\_UPDATE\_KEY,” 128)The HMAC\_KEY\_SZ used in SZK multicast update mode is derived as follows:HMAC\_KEY\_SZ<= Dot16KDF (SZKEK, “SECURITY\_ZONE\_UPDATE\_KEY,” 2M)The value of M in the Dot16KDF function used for HMAC depends on the specific variant of SHA hash function. For SHA-1, M equals 160, while for SHA-224, SHA-256, SHA-284 and SHA-512, M equals to 224, 256, 384, and 512, respectively. |
| 875 | ***[Replace “128” with “N” in paragraph 3-4 of Clause 7.2.2.3.1]******[Replace “128” with “N” in paragraph 4-5 of Clause 7.2.2.3.1]*** | — The KEK, a N-bit key encryption key, derived from the AK.— TEK0 and TEK1, N-bit traffic encryption keys, generated within the BS and transferred from — The MGTEK, a N-bit MBS GTEK, used indirectly to protect MBS traffic. It is updated more frequently than the MAK.— The MTK (MBS Traffic Key) a N-bit key used to protect MBS traffic, derived from the MAK and MGTEK. |
| 876 | ***[Replace “160 or” in Clause 7.2.2.3.4]******[Replace “128” with “128 to 512” in Clause 7.2.2.3.4]*** | The contents of Security Zone SA are as follows:— The SZK, a 128 to 512-bit Security Zone Key, serves as a root for HMAC/CMAC keys derivation.— The SZKEK, a 128 to 512-bit Security Zone Key Encryption Key, serves the same function as SA KEK but for GSA. |
| 877 | ***[Replace “160/128” with “160/128/224/256/512” in Table 7-5 (parameter: HMAC/CMAC\_KEY\_U, HMAC/CMAC\_KEY\_D) in Clause 7.2.2.4.1]******[Replace “If RSA authentication” with “If RSA/ECC authentication” in Table 7-5, parameter PMK Sequence Number]******[Replace “128” with “256” in Table 7-5, parameter KEK]*** |

|  |  |  |
| --- | --- | --- |
| PMK Sequence Number | 4 | The sequence number of the PAK that this AK is derived from. If RSA/ECC authentication is not used, this value shall be set to zero |
| KEK | 256 | Used to encrypt transport keys from the BS to the SS. |
| HMAC/CMAC\_KEY\_U | 160/128 /224/256 /512 | The key that is used for signing UL management messages. |
| HMAC/CMAC\_KEY\_D | 160/128 /224/256 /512 | The key that is used for signing DL management messages. |

 |
| 878 | ***[Replace “128” with “256” in Table 7-5, parameter KEK]******[Replace “128” with “256” in Table 7-6, parameter KEK]******[Replace “160 or 128” with “160/128/224/256/512” in Table 7-6, (parameter:* HMAC\_KEY\_GD /CMAC\_KEY\_GD*) in Clause 7.2.2.4.2]*** |

|  |  |  |
| --- | --- | --- |
| KEK | 256 | Used to encrypt transport keys from the BS to the SS. |
| HMAC\_KEY\_GD/CMAC\_KEY\_GD | 160/128 /224/256 /512 | The key that is used for signing group DL GTEK update messages, calculated by KDF(CMAC\_PAD, GKEK). |
|  |  |  |

 |
| 878 | ***[Replace “RSA” with “RSA/ECC” in Clause 7.2.2.4.4]*** | The PAK context includes all parameters associated with the PAK. This context is created when RSA/ECC Authentication completes. |
| 879 | ***[Replace “RSA” with “RSA/ECC” in Table 7--8]*** |

|  |  |  |
| --- | --- | --- |
| PAK Lifetime | 32 | PAK lifetime, from when the RSA/ECC-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/ECC-based authorization is achieved and a key is generated. The 2 MSBs are the sequence counter. And the 2 LSBs set to 0. |

 |
| 879 | ***[Replace “160 or 128” with “160/128/224/256/512” in Table 7-9, (parameter:* HMAC\_KEY\_SZU,****CMAC\_KEY\_SZU and parameter: HMAC\_KEY\_SZD,CMAC\_KEY\_SZD*) in Clause 7.2.2.4.5, in addition in the description of parameter:* HMAC\_KEY\_SZU, CMAC\_KEY\_SZU *change DL to UL]*** |

|  |  |  |
| --- | --- | --- |
| HMAC\_KEY\_SZU,CMAC\_KEY\_SZU | 160/128***/224/256/512*** | The key that is used for signing UL management messages. |
| HMAC\_KEY\_SZD, CMAC\_KEY\_SZD | 160/128***/224/256/512*** | The key that is used for signing DL management messages |

 |
| 879 | ***[Replace “160/128” with “160/128/224/256/512” in Table 7-10 (parameters:* HMAC\_KEY\_SZ,****CMAC\_KEY\_SZ*) in Clause 7.2.2.4.6]*** |

|  |  |  |
| --- | --- | --- |
| HMAC\_KEY\_SZU/CMAC\_KEY\_SZU | 160/128 /224/256 /512 | The key that is used for signing UL management messages. |
| HMAC\_KEY\_SZD/CMAC\_KEY\_SZD | 160/128 /224/256 /512 | The key that is used for signing DL management messages. |

 |
| 880 | ***[Replace “128” with [“N” in paragraph 3-4 of Clause 7.2.2.7]*** | For SAs using a ciphersuite employing DES-CBC, the AK in the AK Transfer message is triple DES (3-DES) encrypted, using a two-key, 3-DES KEK derived from the Access RS AK. For SAs using a ciphersuite employing N bits keys, such as AES-CCM mode, the TEK in the AK Transfer message is AES encrypted using a N-bit key derived for the Access RS AK and a N-bit block size. |
| 899 | ***[Replace “PKMv1 or PKMv2 RSA-based” with “PKMv1 RSA-based authentication or PKMv2 RSA/ECC-based” in Clause 7.4.2]*** | In PKMv1 RSA-based authentication or PKMv2 RSA/ECC-based authentication, the SS is responsible for sustaining authorization with its BS and maintaining an active AK. In PKMv2 EAP-based authentication, reauthorization can be initiated by either BS or SS to refresh the AK. An SS shall be prepared to use its two most recently obtained AKs according to the manner described in 7.4.2.1 through 7.4.2.3. |
| 899 | ***[Replace “RSA” with “RSA/ECC” in Clause 7.4.2.1]*** | AKs have a limited lifetime and shall be periodically refreshed. In PKMv1, an SS refreshes its AK by reissuing an Auth Request to the BS. The Authorization state machine (7.2.1.5) manages the scheduling of Auth Requests for refreshing AKs. In PKMv2 RSA/ECC-based authentication, the SS refreshes its AK by issuing a PKMv2 RSA/ECC-Request message. In PKMv2 EAP-based authentication, reauthorization can be initiated by either BS or SS to refresh the AK. The SS initiates reauthorization by issuing PKMv2 EAP-Start message to the BS. The BS initiates reauthorization by issuing PKMv2 EAP-Transfer message encapsulating EAP request/identity to the SS. The authorization state machine for PKMv2 EAP-based authentication is described in 7.2.2.8. |
| 906 | ***[Replace “128 bits” with “N bits, where N is 128, 192 or 256” in Clause 7.5.1.3]*** | 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 AES algorithm (NIST Special Publication 800-38A, FIPS 197, IETF RFC 3686) to encrypt the MAC PDU payloads. In MBS, the AES block size and cipher counter block are N bits, where N is 128, 192 or 256. |
| 907 | ***[Replace “repeated four times to construct the 128-bit counter block required by the AES cipher (initial******counter = NONCE | NONCE | NONCE | NONCE). When incremented, this 16-byte counter” to repeated******four/six/eight times to construct the N-bit counter block required by the AES cipher (initial counter = [N/32] \* NONCE, e.g., NONCE | NONCE | NONCE | NONCE...). When incremented, this 16/24/32-byte counter” in the 3rd paragraph of Clause 7.5.1.3]*** | A 32-bit nonce NONCE = n0 | n1 | n2 | n3 (n0 being the most significant byte and n3 the least significant byte) is made of ROC and 24 bits 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/six/eight times to construct the N-bit counter block required by the AES cipher. (initial counter = [N/32] x NONCE, e.g., NONCE|NONCE|NONCE|NONCE…). When incremented, this 16/24/32-byte counter shall be treated as a big endian number. |
| 907 | ***[Replace “128” with “N” in the 5th paragraph of Clause 7.5.1.3]*** | A different N-bit counter value is used to encrypt each N-bit block within a PDU. |
| 910 | ***[Replace “with RSA” with “with RSA or ECC” in Clause 7.5.2.2]******[Replace “PKCS #1 v2.0” with “PKCS #1 v2.2” in Clause 7.5.2.2]******[Replace “equal to 0x02” with “equal to 0x02 (1024 bits RSA), 0x05 (2048 bits RSA) or 0x06 (4096 bits******RSA)” in Clause 7.5.2.2]******[Append new text “The ECC method of encrypting the TEK (FIPS 186-5) shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic s*** REG-RSP ***uite equal to 0x09 (224-bits ECC) or 0x10 (256 bits ECC). When the ECC algorithm is in use for TEK encryption algorithm, the TEK is encrypted with SS’ public key using the ECC algorithm. In this case, KEK is not used.” after the sentence “In this case,******KEK is not used.” in Clause 7.5.2.2]*** | **7.5.2.2 Encryption of TEK with RSA or ECC**The RSA method of encrypting the TEK (PKCS #1 v2.2) shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x02 (1024 bits RSA), 0x05 (2048 bits RSA) or 0x06 (4096 bits RSA). When the RSA algorithm is in use for TEK encryption algorithm, the TEK is encrypted with SS’s public key using the RSA algorithm. In this case, KEK is not used.The ECC method of encrypting the TEK (FIPS 186-5) shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x0B (224-bits ECC) or 0x0C (256 bits ECC). When the ECC algorithm is in use for TEK encryption algorithm, the TEK is encrypted with SS’s public key using the ECC algorithm. In this case, KEK is not used. |
| 910 | ***[Replace “TEK-128” with “TEK-N” in Clause 7.5.2.3]******[Replace “128-bit” with “N-bit” in Clause 7.5.2.3]******[Replace the first occurrence of “0x03” with “0x03 (128-bit key length), 0x07 (192-bit key length) or 0x08 (256-bit key length)” in Clause 7.5.23]******[Insert “Where N is 128, 192 or 256 bit” before “Subclause 7.5.4” in Clause 7.5.2.3]******[Replace the second occurrence of “0x03” with “0x03, 0x07 or 0x08” in Clause 7.5.2.3]*** | **7.5.2.3 Encryption of TEK-N with AES**This method of encrypting the TEK-N shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x03 (128 bit key length), 0x07 (192 bit key length) or 0x08 (256 bit key length). The BS encrypts the value fields of the TEK-N in the Key Reply messages it sends to client SS. This field is encrypted using N-bit AES in ECB mode.**Encryption: C = Ek1[P]****Decryption: P = Dk1[C]****P = Plaintext N-bit TEK****C = Ciphertext N-bit TEK****k1 = the N-bit KEK****E[ ] = N-bit AES ECB mode encryption**D[ ] = N-bit AES ECB decryptionWhere N is 128, 192 or 256 bit.Subclause 7.5.4 describes how the KEK is derived from the AK This method of encrypting the TEK-N shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x03, 0x07 or 0x08. |
| 910 | ***[Replace “128” with “N” in Clause 7.5.2.4]******[Replace “0x04” with “0x04 (128-bit key length), 0x09 (192-bit key length) or 0x10 (256-bit key length)”******in Clause 7.5.2.4]******[Insert “Where N is 128, 192, or 256” after “Dk[] = AES Key Wrap decryption with key k” in Clause******7.5.2.4]*** | **7.5.2.4 Encryption of TEK-N with AES key wrap**This method of encrypting the TEK-N shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x04 (128 bit key length), 0x09 (192 bit key length) or 0x10 (256 bit key length). The BS encrypts the value fields of the TEK-N 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 N-bit TEK****C = Ciphertext N-bit TEK****I = Integrity Check Value****k = the N-bit KEK****Ek[ ] = AES Key Wrap encryption with key k****Dk[ ] = AES Key Wrap decryption with key k****Where N is 128, 192 or 256** |
| 911 | ***[Replace “SHA-1 (FIPS 180-1)” with “SHA-2 (FIPS-180-4)” in Clause 7.5.3]*** | The calculation of the keyed hash in the HMAC Digest attribute and the HMAC Tuple shall use the HMAC (IETF RFC 2104) with the secure hash algorithm SHA-2 (FIPS 180-4). The DL authentication key HMAC\_KEY\_D shall be used for authenticating messages in the DL direction. The UL authentication key HMAC\_KEY\_U shall be used for authenticating messages in the UL direction. UL and DL message authentication keys are derived from the AK (see 7.5.4 for details). The HMAC Sequence number in the HMAC Tuple shall be equal to the AK Sequence Number of the AK from which the HMAC\_KEY\_x was derived. |
| 911 | ***[Replace “RSA” with “RSA/ECC” in Clause 7.5.4]*** | The BS generates AKs, TEKs, and IVs. A random or pseudo-random number generator shall be used to generate AKs and TEKs. A random or pseudo-random number generator may also be used to generate IVs. Regardless of how they are generated, IVs shall be unpredictable. Recommended practices for generating random numbers for use within cryptographic systems are provided in IETF RFC 1750 [B32]. In case of using RSA/ECC algorithm, KEK is not used. |
| 912 | ***[Replace “128” with “N” in Clause 7.5.4.2]******[Replace “the full 128 bits of the KEK” with “the HASH function is SHA-256 and the full N-bits of the******KEK” in Clause 7.5.4.2]*** | The construction of the KEK for use with TEK-N keys shall be the same as for 3-DES KEKs except that the HASH functions is SHA-256 and the full N bits of the KEK are used directly as the N-bit AES key, instead of the KEK being split into two 64-bit DES keys. |
| 914 | ***[Replace “128” with “N” in Clause 7.5.4.5.1]*** | The construction of the KEK for use with TEK-N keys shall be the same as for 3-DES KEKs as described in 7.5.4.2 except that the full N bits of the KEK are used directly as the N-bit AES key, instead of the KEK being split into two 64-bit DES keys. |
| 915 | ***[Replace “v2.1” with “v2.2” in Clause 7.5.4.5.2.2]******[Replace “0x02” with “0x02 (1024-bit key length), 0x05 (2048-bit key length) or 0x06 (4096-bit key length)” in Clause 7.5.4.5.2.2]*** | The RSA method of encrypting the GKEK (PKCS #1 v2.2, 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 (1024-bit key length), 0x05 (2048-bit key length) or 0x06 (4096-bit key length).  |
| 915 | ***[Replace “0x03” with “0x03 (128 bit), 0x07 (192 bit) or 0x08 (256 bit)” in Clause 7.5.4.5.2.3]******[Replace “128” with “N” in Clause 7.5.4.5.2.3]******[Insert “Where N=128, 192 or 256” after “D [] = 128-bit AES ECB mode decryption” in Clause******7.5.4.5.2.3]*** | 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 (128 bit), 0x07 (192 bit) or 0x08 (256 bit). The BS encrypts the value fields of the GKEK in the PKMv2 Group-Key-Update-Command messages (for the GKEK update mode) it sends to client SS. This field is encrypted using N-bit AES in ECB mode.**Encryption: C = Ek1[P]****Decryption: P = Dk1[C]****P = Plaintext N-bit GKEK****C = Ciphertext N-bit GKEK****k1 = the N-bit KEK****E [ ] = N-bit AES ECB mode encryption****D [ ] = N-bit AES ECB mode decryption****Where N=128, 192 or 256** |
| 915 | ***[Replace “0x04” with “0x04 (128 bit), 0x09 (192 bit) or 0x10 (256 bit)” in Clause 7.5.4.5.2.4]******[Replace “128” with “N” in Claue 7.5.4.5.2.4]******[Insert “Where N=128, 192 or 256” after “D [] = AES Key Wrap decryption with key k” in Clause******7.5.4.5.2.4]*** | 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 (128 bit), 0x09 (192 bit) or 0x10 (256 bit).The BS encrypts the value fields of the GKEK in the PKMv2 Group-Key-Update-Command messages (for the GKEK update mode) it sends to client SS. This field is encrypted using N-bit AES key wrap algorithm. This N-bit AES key wrap algorithm is defined only for PKM version 2.**Encryption: C,I = Ek[P]****Decryption: P,I = Dk[C]****P = Plaintext N-bit GKEK****C = Ciphertext N-bit GKEK****k = the N-bit KEK derived from the AK****Ek[ ] = AES Key Wrap encryption with key k****Dk[ ] = AES Key Wrap decryption with key k****Where N=128, 192 or 256** |
| 917 | ***[Replace “1024 bits” with “M” bits, where M is 1024, 2048 or 4096” in Clause 7.5.8.******[Replace “PKCS #1” with “PKCS#1, v2.2” in Clause 7.5.8]******[Replace “SHA-1” with “SHA-256” in Clause 7.5.8]*** | AKs in Auth Reply messages shall be RSA public-key encrypted, using the SS’s public key. The protocol uses 65537 (0x010001) as its public exponent and a modulus length of M bits, where M is 1024, 2048 or 4096. The PKM protocol employs the RSAES-OAEP encryption scheme (PKCS #1v2.2). RSAES-OAEP requires the selection of a hash function, a mask-generation function, and an encoding parameter string. The default selections specified in PKCS #1v2.2 shall be used when encrypting the AK. These default selections are SHA-256 for the hash function, MGF1 with SHA-256 for the mask-generation function, and the empty string for the encoding parameter string. |
| 917 | ***[Replace “PKCS #1) with SHA-1 (FIPS 186-2)” with “PKCS #1 v2.2) with SHA-2 or SHA-3 (FIPS 186-5)” for Clause 7.5.9]******[Replace “2048” with “4096” for Clause 7.5.9]*** | The Protocol employs the RSA Signature Algorithm (PKCS #1 v2.2) with SHA-2 or SHA-3 (FIPS 186-5) for both of its certificate types.As with its RSA encryption keys, Privacy uses 65537 (0x010001) as the public exponent for its signing operation. Manufacturer CAs shall employ signature key modulus lengths of at least 1024 bits and no greater than 4096 bits. |
| 918 | ***[Replace “SHA-1” with “SHA-2 or SHA-3” for the 1st paragraph after Table 7-13]******[Replace “PKCS #1” with “PKCS #1, v2.2” for the 1st paragraph after Table 7-13]******[Replace “FIPS 180-1” with “FIPS 180-4” in the 1st paragraph after Table 7-13]*** | All certificates described in this specification shall be signed with the RSA signature algorithm using SHA-2 or SHA-3 as the one-way hash function. The RSA signature algorithm is described in PKCS #1 v2.2; SHA-2 and SHA-3 is described in FIPS 180-4. Restrictions posed on the certificate values are described in 7.6.1.1 through 7.6.1.8. |
| 918 | ***[Replace “All certificates” with “All PKMv1 certificates” in Clause 7.6.1.3]******[Replace “The RSA signature algorithm is described in PKCS #1; SHA-1 is described in FIPS 180-1” with “PKMv2 certificates shall be signed with either RSA signature algorithm, using SHA-2 or SHA-3 as the one-way hash function or by the ECC signature algorithm, as described in FIPS 180-5” in Clause******7.6.1.3]******[Replace “sha-1” with “sha-x” in Clause 7.6.1.3]*** | All PKMv1 certificates described in this specification shall be signed with the RSA signature algorithm, using SHA-1 as the one-way hash function. PKMv2 certificates shall be signed with either RSA signature algorithm, using SHA-2 or SHA-3 as the one-way hash function or by the ECC signature algorithm, as described in FIPS 180-5. The ASN.1 OID used to identify the “SHA-x with RSA” signature algorithm is sha-xWithRSAEncryption OBJECT IDENTIFIER ::={ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5}When the sha-xWithRSAEncryption OID appears within the ASN.1 type AlgorithmIdentifier, as is the case with both tbsCertificate.signature and signatureAlgorithm, the parameters component of that type is theASN.1 type NULL. |
| 721 | ***[Replace “contains the RSA (with SHA-1” with “contains the RSA or ECC (with SHA-2 or SHA-3” in******Clause 7.6.1.8]*** | In all three PKM certificate types, the signatureValue contains the RSA or ECC (with SHA-2 or SHA-3) signature computed over the ASN.1 DER encoded tbsCertificate. The ASN.1 DER encoded tbsCertificate is used as input to theRSA signature function. The resulting signature value is ASN.1 encoded as a bit string and included in the Certificate’s signatureValue field. |
| 721 | ***[Replace “RSA private/public key pairs shall also have” with “RSA private/public key pairs or ECC******private/public key pairs shall also have” in Clause 7.6.2]*** | Manufacturer-issued SS certificates shall be stored in SS permanent, write-once memory. SSs that have factory-installed RSA private/public key pairs or ECC private/public key pairs shall also have factory-installed SS certificates. SSs that rely on internal algorithms to generate an RSA key pair shall support a mechanism for installing a manufacturer issued SS certificate following key generation.  |
| 924 | ***[Replace “BS and SS RSA mutual authentication and AK exchange overview” with “BS and SS mutual******authentication” in Clause 7.8.2]******[Replace “two modes” with “three modes” in Clause 7.8.2]******[Replace “only mutual authentication is used” with “only RSA/ECC mutual authentication is used” in Clause 7.8.2]******[Replace “In the other mode, the mutual authentication” with “In the second mode, the RSA/ECC******mutual authentication” in Clause 7.8.2]******[Insert “In the third mode, the mutual authentication is performed with EAP-TLS, and a certificate authority authenticates both SS and BS certificates. 7.8.2.1 BS and SS RSA/ECC mutual authentication and AK exchange overview.” after the sentence ending with “if authentication is needed in reentry.” In Clause 7.8.2]******[Replace “SS mutual authorization” with “SS RSA/ECC mutual authorization” in Clause 7.8.2]******[Replace “RSA signature” with “RSA signature or ECC signature” in the last paragraph of Clause 7.8.2]******[Replace “RSA-Reply” with “RSA/ECC-Reply” in Clause 7.8.2]******[Replace “RSA-based” with “RSA/ECC-based” in the 2nd paragraph of Clause 7.8.2]******[Replace “bind RSA” with “bind RSA or ECC” in the 2nd paragraph of Clause 7.8.2]*** | **7.8.2 BS and SS mutual authentication and AK exchange overview**The BS mutual authentication can take place in one of three modes of operation. In one mode, only RSA/ECC mutual authentication is used. In the second mode, the RSA/ECC 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 if authentication is needed in reentry. In the third mode, the mutual authentication is performed with EAP-TLS, and a certificate authority authenticates both SS and BS certificates. …SS RSA/ECC mutual authorization, controlled by the PKMv2 Authorization state machine, is the process of…— The RSA signature or ECC signature over all the other attributes in the auth-reply message by BS, used to assure theauthenticity of the above PKMv2 RSA/ECC-Reply messages.…After successful RSA/ECC based authorization either EAP based authorization or Authenticated EAP based authorization maybe supported according to the value of Authorization policy negotiated in the SBCREQ/RSP messages. It shall cryptographically bind RSA or ECC and further EAP authentication |
| 1554 | ***[Replace “PKMv2 RSA” with “PKMv2 RSA/ECC” in Table 10-3]*** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MS | Authorize WaitTimeout | PKMv2 RSA/ECC-Requestretransmission interval fromAuth Wait state | 2 s | 10 s | 30 s |
| MS | Reauthorize WaitTimeout | PKMv2 RSA/ECC-Requestretransmission interval fromReauth Wait state | 2 s | 10 s | 30 s |
| MS | Authorize RejectWait Timeout | Delay before resending PKMv2 RSA/ECC-Request after receiving PKMv2 RSA/ECC-Reject | 10 s | 60 s | 10 min(600 s) |

 |
|  |  |  |
| 1562 | ***[Replace “Length: 21” with “Length: 17, 29, 33” in Table 11-2]*** | Length: 17, 29, 33  |
| 1563 | ***[Replace “Reserved” with “Digest Length” in Table 11-3, and for the same row, add the following text for the Notes field “0b0001: 224 bits 0b0010: 256 bits 0b0011: 384 bits 0b0100: 512 bits”]******[Replace “160 bits” with “N bits” in Table 11-3, and in the same row, replace “SHA-1” in the Notes field with “SHA-2 or SHA-3”]*** |

|  |  |  |
| --- | --- | --- |
| *Digest Length* | 4 bits | 0b0001: 224 bits0b0010: 256 bits0b0011: 384 bits0b0100: 512 bits |
| HMAC Key Sequence Number | 4 bits |  |
| HMAC Digest | N bits | HMAC with SHA-2 or SHA-3 |

 |
| 1563 | ***[Replace “Length: 13, or 19” with “Length: 13, 17, 19, 29 or 33” in Table 11-4]*** | Length: 13, 17, 19, 29 or 33 |
| 1563 | ***[Replace “160 bits” with “N bits” in Table 11-5]******[Replace “AES-1” with AES-N” in Table 11-5]*** |

|  |  |  |
| --- | --- | --- |
| CMAC Value | N bits | CMAC with AES-N |

 |
| 1564 | ***[Replace “2-Truncate to 12 bytes” with “2-Truncate to 12 bytes******3-Truncate to 28 bytes******4-Truncate to 32 bytes” in Table 11-7, parameter* Short-HMAC Digest, *in Clause 11.1.2.3]*** |

|  |  |  |
| --- | --- | --- |
| Short-HMAC Digest | *variable* | 0—Truncate HMAC to 8 bytes in Short-HMAC Tuple1—Truncate to 10 bytes2—Truncate to 12 bytes3-Truncate to 28 bytes4-Truncate to 32 bytes |

 |
| 1671 | ***[Replace “RSA” with “RSA/ECC” in first paragraph of Clause 11.8.4.2]******[Replace “Bit 2: Reserved; shall be set to 0” with “Bit 2: ECC-based authorization at the initial network******entry” in Clause 11.8.4.2]******[Replace “Bit 6: Reserved; shall be set to 0” with “Bit 6: ECC-based authorization at reentry” in Clause******11.8.4.2]*** | If this field is omitted, then both SS and BS shall use the IEEE 802.16 security, constituting X.509 digital certificates andthe RSA/ECC public key encryption algorithm, as authorization policy.

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 25.2 | 1 | Bit 0: RSA-based authorization at the initial network entryBit 1: EAP-based authorization at the initial network entryBit 2: ECC-based authorization at the initial network entryBit 3: Set to 0Bit 4: RSA-based authorization at reentryBit 5: EAP-based authorization at reentryBit 6: ECC-based authorization at reentryBit 7: *Reserved*; shall be set to 0 |

 |
| 1672 | ***[Replace “shall be used in the RSA-based authorization procedure” with “shall be used with RSA encryption i n the RSA-based authorization procedure and with ECC encryption in the ECC-based authorization procedure” in the 1st paragraph of Clause 11.8.4.2]*** | The PKMv2 Auth-Request/Reply/Reject/Acknowledgment messages shall be used with RSA encryption in the RSA-based authorization procedure and with ECC encryption in the ECC-based authorization procedure. |
| 1672 | ***[Replace “Bits 4-5” with “Bits 4-6” for the 3rd paragraph of Clause 11.8.4.2]*** | Bits 4–6 are only applied to the SBC-REQ message.  |
| 1672 | ***[Replace “RSA” with “RSA/ECC” in second table of Clause 11.8.4.2]*** |

|  |  |  |
| --- | --- | --- |
| Value | Description | Scope |
| Bit 0/4 | Bit 1/5 |
| 0 | 0 | No Authorization (MS cannot support any authorization) | SBC-REQ,PKM-REQ |
| 0 | 1 | Only EAP-based authorization |
| 1 | 0 | Only RSA/ECC-based authorization |
| 1 | 1 | Only RSA/ECC-based authorization orOnly EAP-based authorization orEAP-based authorization after RSA/ECC-based authorization |

 |
| 1672 | ***[Replace “RSA” with “RSA/ECC” in third table of Clause 11.8.4.2]*** |

|  |  |  |
| --- | --- | --- |
| Value | Description | Scope |
| Bit 0 | Bit 1 |
| 0 | 0 | No Authorization  | SBC-RSP,PKM-RSP |
| 0 | 1 | Only EAP-based authorization |
| 1 | 0 | Only RSA/ECC-based authorization |
| 1 | 1 | EAP-based authorization after RSA/ECC-based authorization |

 |
| 1673 | ***[Replace the value of the Length field in the table in Clause 11.8.4.3 of p. 1673 with “2”]******[Insert “Bit 8: 224-bit HMAC******Bit 9: 256-bit HMAC******Bit 10: 384-bit HMAC******Bit 11: 512-bit HMAC******Bit 12: 224-bit Short HMAC (SHA 512/224)******Bit 13: 256-bit Short HMAC (SHA 512/256)******Bit 14-15*: *Reserved.* Set to 0*” in the last line in the Value field of the table in Clause******11.8.4.3]*** |

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 25.3 | 2 | If length is one byte, then only Bit 0-7 are relevant.Bit 0: HMACBit 1: *Reserved*Bit 2: 64-bit Short-HMACaBit 3: 80-bit Short-HMACaBit 4: 96-bit Short-HMACaBit 5: CMACBit 6–7: *Reserved.* Set to 0Bit 8: 224-bit HMACBit 9: 256-bit HMACBit 10: 384-bit HMACBit 11: 512-bit HMACBit 12: 224-bit Short HMAC (SHA 512/224)Bit 13: 256-bit Short HMAC (SHA 512/256)Bit 14-15: Reserved. Set to 0. |

 |
| 1683 | ***[Replace first paragraph in Clause 11.9.2 with “The AK (AUTH-Key) is an N byte quantity, from which a KEK, and two message authentication keys (one for UL requests, and a second for DL replies) are derived. This attribute contains an N byte quantity containing the AK RSA-encrypted or ECC encrypted with the SS’s 8N bit RSA public key or 8N ECC public key. Details of the RSA encryption procedure are given in 7.5. Details on the ECC-ECDSA encryption procedure is given in RFC 6605. The ciphertext produced by the RSA or ECC algorithms shall be the length of the RSA or ECC modulus, i.e., N bytes.*“*]******[Replace “128” with “N“ in length column of table in Clause 11.9.2]*** ***[Replace “128-byte quantity representing an RSA-encrypted AK.” with “N-byte quantity representing an RSA-encrypted AK with 8N bit key length. Where N=128 (1024 bit RSA), 256 (2048 bit RSA) or 512 (4096 bit RSA). or ECC-encrypted AK with 8N bit key length. Where N=32 (256 bits ECDSA), 48 (384 bit ECDSA) or 64 (512 bit ECDSA)” in value column of table in Clause 11.9.2]*** | The AK (AUTH-Key) is an N byte quantity, from which a KEK, and two message authentication keys (one for UL requests, and a second for DL replies) are derived. This attribute contains an N byte quantity containing the AK RSA-encrypted or ECC encrypted with the SS’s 8N bit RSA public key or 8N ECC public key. Details of the RSA encryption procedure are given in 7.5. Details on the ECC-ECDSA encryption procedure is given in RFC 6605. The ciphertext produced by the RSA or ECC algorithms shall be the length of the RSA or ECC modulus, i.e., N bytes.

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 7 | N | N-byte quantity representing an RSA-encrypted AK with 8N bit key length. Where N=128 (1024 bit RSA), 256 (2048 bit RSA) or 512 (4096 bit RSA).or ECC-encrypted AK with 8N bit key length.Where N=32 (256 bits ECDSA), 48 (384 bit ECDSA) or 64 (512 bit ECDSA) |

 |
| 1684 | ***[Replace “AES” with “AES-128” ror the 2nd row of the TEK TLV in Clause 11.9.3]******[Replace “AES” with “AES-192 or for AES-128” for the 3rd row of the TEK TLV in Clause 11.9/3]******[Append the following rows after the last row of the TEK TLV******“{Type} [blank], {Length} 256, {Value} Encrypted TEK for RSA-2048******{Type} [blank], {Length} 512, {Value} Encrypted TEK for RSA-4096”]*** |

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 8 | 81624283240128256512 | Encrypted TEK for DESEncrypted TEK for AES-128Encrypted TEK for AES-192 or for AES-128 key wrap Encrypted TEK for ECC-224Encrypted TEK for AES-256 or for ECC-256Encrypted TEK for AES-256 key wrap Encrypted TEK for RSA-1028 or for ECC-512Encrypted TEK for RSA-2048Encrypted TEK for RSA-4096 |

 |
|  | ***[Replace the text “0x03 the length shall be 16” with “0x03, 0x07 or 0x08 the length shall be 16, 24 or 32 in correspondence” in Clause 11.9.3]******[Replace the text “0x04the length shall be 16” with “0x04, 0x09 or 0x0A the length shall be 16, 24, or 32 in correspondence” in Clause 11.9.3]******[Add the text “When the TEK encryption algorithm identifier in the SA is 0x0B or 0x0C the length shall be 24, and the TEK shall be encrypted with the Eliptic Curve Diffie Hellman (ECDHE) according to the procedure in RFC 8422.” after the text “according to the procedure in 7.5.2.4.” of Clause 11.9.3]*** | When the TEK encryption algorithm identifier in the SA is 0x03, 0x07 or 0x08 the length shall be 16, 24 or 32 in correspondence and the TEK shall be encrypted with AES in ECB mode according to the procedure in 7.5.2.3.When the TEK encryption algorithm identifier in the SA is 0x04, 0x09 or 0x0A the length shall be 16, 24, or 32 in correspondence and the TEK shall be encrypted with the AES key wrap algorithm according to the procedure in 7.5.2.4.When the TEK encryption algorithm identifier in the SA is 0x0B or 0x0C the length shall be 24, and the TEK shall be encrypted with the Eliptic Curve Diffie Hellman (ECDHE) according to the procedure in RFC 8422. |
| 1684 | ***[Replace the last row in Table 11-41 of Clause 11.9.3 with the following rows******Value field: “5” Description field: “RSA with 2048-bit key”******Value field: “6” Description field: “RSA with 4096-bit key”******Value field: “7” Description field: “GCM mode AES with 192-bit key”******Value field: “8” Description field: “GCM mode AES with 256-bit key”******Value field: “9” Description field: “AES-CBC key wrap with 192-bit key”******Value field: “10” Description field: “AES-CBC key wrap with 256-bit key”******Value field: “11” Description field: “ECC with 224-bit key”******Value field: “12” Description field: “ECC with 256-bit key”******Value field: “13-255” Description field: “Reserved”]*** |

|  |  |
| --- | --- |
| Value | Description |
| 0 | *Reserved* |
| 1 | 3-DES EDE with 128-bit key |
| 2 | RSA with 1024-bit key |
| 3 | ECB mode AES with 128-bit key |
| 4 | AES key wrap with 128-bit key |
| 5 | RSA with 2048-bit key |
| 6 | RSA with 4096-bit key |
| 7 | ECB mode AES with 192-bit key |
| 8 | ECB mode AES with 256-bit key |
| 9 | AES key wrap with 192-bit key |
| 10 | AES key wrap with 256-bit key |
| 11 | ECC with 224-bit key |
| 12 | ECC with 256-bit key |
| 13–255 | *Reserved* |

 |
| 1685 | ***[Replace “20 bytes” with the following text in the Length field of the last row of the table in Clause 11.9.6:******“20 bytes******28 bytes******32 bytes******48 bytes******64 bytes”]******[Replace “A 160-bit (20 byte) keyed SHA hash” with the following text in the Value (string) field of the******last row of the table in Clause 11.9.6:******“A 160-bit (20 byte) keyed SHA-1 hash******A 224-bit keyed SHA-224 or SHA-512/224******A 256-bit keyed SHA-256 or SHA-512/256******A 384-bit keyed SHA-384******A 512-bit keyed SHA-512”]*** | The HMAC Digest attribute contains a keyed hash used for message authentication. The HMAC algorithm is defined in FIPS PUB 198.

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 11 | 20 bytes28 bytes32 bytes48 bytes64 bytes | A 160-bit (20 byte) keyed SHA-1 hashA 224 bit keyed SHA-224 or SHA-512/224A 256 bit keyed SHA-256 or SHA-512/256A 384 bit keyed SHA-384 A 512 bit keyed SHA-512 |

 |
| 1689 | ***[Replace the last 3 rows in Table 11-45 of Clause 11.9.14 with the following rows******Value field: “4” Description field: “CCM mode, 192-bit AES”******Value field: “5” Description field: “CBC mode, 192-bit AES”******Value field: “6” Description field: “GCM mode, 192-bit AES”******Value field: “7” Description field: “OCB mode, 192-bit AES”******Value field: “8” Description field: “CCM mode, 192-bit AES”******Value field: “9” Description field: “CBC mode, 256-bit AES”******Value field: “10” Description field: “GCM mode, 256-bit AES”******Value field: “11” Description field: “OCB mode, 256-bit AES”*** |

|  |  |
| --- | --- |
| Value | Description |
| 0 | 0 No data encryption |
| 1 | CBC mode, 56-bit DES |
| 2 | CCM mode, 128-bit AES |
| 3 | CBC mode, 128-bit AES |
| 4 | CCM mode, 192-bit AES |
| 5 | CBC mode, 192-bit AES |
| 6 | GCM mode, 192-bit AES |
| 7 | OCB mode, 192-bit AES |
| 8 | CCM mode, 256-bit AES |
| 9 | CBC mode, 256-bit AES |
| 10 | GCM mode, 256-bit AES |
| 11 | OCB mode, 256-bit AES |
| 12-127 | *Reserved* |
| 128 | CTR mode, 128-bit AES for MBS with 8-bit ROC |
| 129 | CTR mode, 192-bit AES for MBS with 8-bit ROC |
| 130 | CTR mode, 256-bit AES for MBS with 8-bit ROC |
| 131–255 | *Reserved* |

 |
| 1689 | ***[Replace the last row in Table 11-46 of Clause 11.9.14 with the following rows******Value field: “2” Description field: “CCM mode, 192-bit AES”******Value field: “3” Description field: “CCM mode, 256-bit AES”******Value field: “4” Description field: “CBC mode, 128-bit AES”******Value field: “5” Description field: “CBC mode, 192-bit AES”******Value field: “6” Description field: “CBC mode, 256-bit AES”******Value field: “7” Description field: “OCB mode, 128-bit AES”******Value field: “8” Description field: “OCB mode, 192-bit AES”******Value field: “9” Description field: “OCB mode, 256-bit AES”******Value field: “10” Description field: “GCM mode, 128-bit AES”******Value field: “11” Description field: “GCM mode, 192-bit AES”******Value field: “12” Description field: “GCM mode, 256-bit AES”******Value field: “13-255” Description field: “Reserved”]*** |

|  |  |
| --- | --- |
| Value | Description |
| 0 | 0 No data authentication |
| 1 | CCM mode, 128-bit AES |
| 2 | CCM mode, 192-bit AES |
| 3 | CCM mode, 256-bit AES |
| 4 | CBC mode, 128-bit AES |
| 5 | CBC mode, 192-bit AES |
| 6 | CBC mode, 256-bit AES |
| 7 | OCB mode, 128-bit AES |
| 8 | OCB mode, 192-bit AES |
| 9 | OCB mode, 256-bit AES |
| 10 | GCM mode, 128-bit AES |
| 11 | GCM mode, 192-bit AES |
| 12 | GCM mode, 256-bit AES |
| 13-255 | *Reserved* |

 |
| 1689 | ***[Replace the last row in Table 11-47 of Clause 11.9.14 with the following rows******Value field: “5” Description field: “RSA with 2048-bit key”******Value field: “6” Description field: “RSA with 4096-bit key”******Value field: “7” Description field: “CBC mode AES with 256-bit key”******Value field: “8” Description field: “OCB mode AES with 256-bit key”******Value field: “9” Description field: “GCM mode AES with 256-bit key”******Value field: “10” Description field: “ECC with 224-bit key”******Value field: “11” Description field: “ECC with 256-bit key”******Value field: “12” Description field: “ECC with 384-bit key”******Value field: “13-255” Reserved”]*** |

|  |  |
| --- | --- |
| Value | Description |
| 0 | *Reserved* |
| 1 | 3-DES EDE with 128-bit key |
| 2 | RSA with 1024-bit key |
| 3 | ECB mode AES with 128-bit key |
| 4 | AES key wrap with 128-bit key |
| 5 | RSA with 2048-bit key |
| 6 | RSA with 4096-bit key |
| 7 | CBC mode AES with 256-bit key |
| 8 | OCB mode AES with 256-bit key |
| 9 | GCM mode AES with 256-bit key |
| 10 | ECC with 224 bit key |
| 11 | ECC with 256 bit key |
| 12 | ECC with 384 bit key |
| 13-255 | *Reserved* |

 |
| 1690 | ***[Insert before the third row in Table 11-48 of Clause 11.9.14 the following rows:******Value field: “0x090000” Description field: “CBC mode 256-bit AES, no data authentication, no key******encryption”******Value field: “0x090007” Description field: “CBC mode 256-bit AES, no data authentication, CBC mode******AES with 256-bit key”******Value field: “0x090607” Description field: “CBC mode 256-bit AES, CBC mode 256-bit AES, CBC mode******AES with 256-bit key”******[Insert before the third to last row in Table 11-48 of Clause 11.9.14 the following rows******Value field: “0x0A0C0B” Description field: “GCM mode 256-bit AES, BCM mode 256-bit, ECC with******256-bit key”******Value field: “0x0A0C0C” Description field: “GCM mode 256-bit AES, BCM mode 256-bit, ECC with******384-bit key”]*** |

|  |  |
| --- | --- |
| Value | Description |
| 0x000000 | No data encryption, no data authentication, no key encryption |
| 0x010001 | CBC mode 56-bit DES, no data authentication and 3-DES,128 |
| 0x090000 | CBC mode 256-bit AES, no data authentication, no key encryption |
| 0x090007 | CBC mode 256-bit AES, no data authentication, CBC mode AES with 256 bit key |
| 0x090607 | CBC mode 256-bit AES, CBC mode 256 bit, CBC mode AES with 256 bit key |
| 0x000002 | No data encryption, no data authentication and RSA, 1024 |
| 0x010002 | CBC mode 56-bit DES, no data authentication and RSA, 1024 |
| 0x020003 | CCM mode AES, no data authentication and AES 128 |
| 0x020103 | CCM mode 128-bit AES, CCM mode, 128-bit, ECB mode AES with 128-bit key |
| 0x020104 | CCM mode 128bits AES, CCM mode, AES key wrap with 128-bit key |
| 0x030003 | CBC mode 128-bit AES, no data authentication, ECB mode AES with 128-bit key |
| 0x0A0C0B | GCM mode 256-bit AES, GCM mode 256 bit, ECC with 256-bit key |
| 0x0A0C0C | GCM mode 256-bit AES, GCM mode 256 bit, ECC with 38-bit key |
| 0x800003 | MBS CTR mode 128 bits AES, no data authentication, AES ECB mode with 128-bit key |
| 0x800004 | MBS CTR mode 128 bits AES, no data authentication, AES key wrap with 128-bit key |
| All remaining values  | *Reserved* |

 |
| 1693 | ***[Add the following text to the end of the text in the Length field of the last row of the table in Clause 11.9.19 “12 16”]******[Add the following text to the end of the text in the Length field of the last row of the table in Clause 11.9.19******“GCM mode AES******CBC mode AES”]*** |

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 29 | 41216 | Randomly generated valueGCM mode AESCBC mode AEs |

 |
| 1694 | ***[Replace the text “This pre-primary authorization key (PAK) is a 256 bit quantity, from which an AK, a KEK, two MAC message authentication keys, and two EAP message protection keys are derived. This attribute contains a 128 byte quantity containing the PAK RSA-encrypted with the MS’s 1024-bit RSA public key. Details of the RSA encryption procedure are given in 7.2.2.2. The ciphertext produced by the RSA algorithm shall be the length of the RSA modulus, i.e., 128 bytes.” with*** ***“This pre-primary authorization key (PAK) is a quantity, from which an AK, a KEK, two MAC message authentication keys, and two EAP message protection keys are derived. This attribute contains quantity containing the PAK RSA or ECC encrypted with the MS’s RSA or ECC public key. Details of the ECC/RSA encryption procedure are given in 7.2.2.2. The ciphertext produced by the RSA algorithm shall be the length of the RSA modulus, i.e., 128 bytes. The ciphertext produced by the ECC encryption algorithm shall be the length = size of symmetric key + size of ECC key + overhead + length of plaintext + padding.” in Clause 11.9.22]******[Replace the Length field text in the last row of the table in Clause 11.9.22 with the following text******“Variable”]******[Replace the text “128-byte” in the Value field of the last row in the table in Clause 11.9.22 with the******following text “A”]******[Replace the text “RSA encrypted” in the Value field of the last row in the table Clause 11.9.22 with the******following text “ECC or RSA-encrypted”]*** | This pre-primary authorization key (PAK) is a quantity, from which an AK, a KEK, two MAC message authentication keys, and two EAP message protection keys are derived. This attribute contains quantity containing the PAK RSA or ECC encrypted with the MS’s RSA or ECC public key. Details of the ECC/RSA encryption procedure are given in 7.2.2.2. The ciphertext produced by the RSA algorithm shall be the length of the RSA modulus, i.e., 128 bytes. The ciphertext produced by the ECC encryption algorithm shall be the length = size of symmetric key + size of ECC key + overhead + length of plaintext + padding.

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 35 | *Variable* | A quantity representing an ECC or RSA encrypted pre-PAK, which generates PAK |

 |
| 1694 | ***[Replace the text “RSA signature” with the text “RSA or ECC signature” in Clause 11.9.24]******[Replace the text “PKMv2 RSA Reply message” with the text “PKMv2 RSA/ECC Reply message” in Clause 11.9.24]******[Replace the text “PKMv2 RSA Reject message” with the text “PKMv2 RSA/ECC Reject message” in Clause 11.9.24]******[Add the text “in the case of RSA or SHA-256 in the case of ECC” to the end of the text in the Value field of the last row of the table in Clause 11.9.24]*** | The SigBS attribute contains a RSA or ECC signature computed over the PKMv2 RSA/ECC Reply message or the PKMv2 RSA/ECC Reject message with the BS’s private key.

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 38 | *Variable* | An RSA or ECC signature computed overall attributes included in the PKMv2 RSA/ECC Reply message or the PKMv2 RSA/ECC Reject message with the BS’s private key. This value is calculated usingPKCS #1 OAEP 1.5 signing algorithm withSHA-1 hash in the case of RSA or SHA-256 in the case of ECC. |

 |
| 1697 | ***[Replace the text “RSA signature” with “RSA or ECC signature” in Clause 11.9.30]******[Replace the text “PKMv2 RSA Request” with the text “PKMv2 RSA/ECC Request” in Clause 11.9.30]******[Replace the text “PKMv2 RSA Acknowledgment” with the text “PKMv2 RSA/ECC Acknowledgment” in Clause 11.9.30]******[Replace the text in the Length field of the last row of the table in Clause 11.9.30 with the text “Variable”]******[Replace the text “RSA signature” with “RSA or ECC signature” in the Value field of the last row of the table in Clause 11.9.30]******[Replace the text “PKMv2 RSA Acknowledgment” with the text “PKMv2 RSA/ECC Acknowledgment” in the Value field of the last row of the Table in Clause 11.9.30]******[Add the text “in the case of RSA or SHA-256 in the case of ECC” to the end of the text in the Value field of the last row of the table in the Value field of the last row of the table in Clause 11.9.30]*** | The SigSS attribute contains an RSA or ECC signature computed over the PKMv2 RSA/ECC Request message or the PKMv2 RSA/ECC Acknowledgment message with the SS’s private key

|  |  |  |
| --- | --- | --- |
| Type | Length | Value |
| 44 | *Variable* | An RSA or ECC signature computed over all attributes included in the PKMv2 RSA/ECC Requestmessage or the PKMv2 RSA/ECC Acknowledgment message with the SS’s private key. This value is calculated using RSASSA-PKCS-v1\_5-Sign algorithm with SHA-1 hash in the case of RSA or SHA-256 in the case of ECC. |

 |
| 1697 | ***[Repace the text “RSA-based” with “RSA/ECC-based” in Clause 11.9.33]*** | The Auth-Result-Code attribute contains the result code of the RSA/ECC-based authorization (only for PKMv2). |
| 1909 | ***[Replace the last line in Clause 13.2.3 of p. 1909 (the text “aes128BitCtrMode(128) }”) with the following******text******“aes128BitCbcMode(3),******aes192BitCcmMode(4),******aes192BitCbcMode(5),******aes192BitGcmMode(6),******aes192BitOcbMode(7),******aes256BitCcmMode(8),******aes256BitCbcMode(9),******aes256BitGcmMode(10),******aes256BitOcbMode(11),******aes128BitCtrMode(128),******aes192BitCtrMode(129),******aes256BitCtrMode(130)}”]*** | WmanIf2DataEncryptAlgId ::= TEXTUAL-CONVENTIONSTATUS currentDESCRIPTION"Data encryption algorithm identifiers."REFERENCE"Table 597"SYNTAX INTEGER {none(0),des56BitCbcMode(1),aes128BitCcmMode(2),aes128BitCbcMode(3),aes192BitCcmMode(4),aes192BitCbcMode(5),aes192BitGcmMode(6),aes192BitOcbMode(7),aes256BitCcmMode(8),aes256BitCbcMode(9),aes256BitGcmMode(10),aes256BitOcbMode(11),aes128BitCtrMode(128),aes192BitCtrMode(129),aes256BitCtrMode(130)} |
| 1910 | ***[Replace 8th line in Clause 13.2.3 on p. 1910 (the text “aes128BitCcmMode(2) }” with the following text******“aes128BitCcmMode(1),******aes192BitCcmMode(2),******aes256BitCcmMode(3),******aes128BitCbcMode(4),******aes192BitCbcMode(5),******aes256BitCbcMode(6),******aes128BitOcbMode(7),******aes192BitOcbMode(8),******aes256BitOcbMode(9),******aes128BitGcmMode(10),******aes192BitGcmMode(11),******aes256BitGcmMode(12) }”]*** | WmanIf2DataAuthAlgId ::= TEXTUAL-CONVENTIONSTATUS currentDESCRIPTION"Data authentication algorithm identifiers."REFERENCE"Table 598"SYNTAX INTEGER {noDataAuthentication(0),aes128BitCcmMode(1),aes192BitCcmMode(2),aes256BitCcmMode(3),aes128BitCbcMode(4),aes192BitCbcMode(5),aes256BitCbcMode(6),aes128BitOcbMode(7),aes192BitOcbMode(8),aes256BitOcbMode(9),aes128BitGcmMode(10),aes192BitGcmMode(11),aes256BitGcmMode(12)} |
| 1910 | ***[Replace the line “aes128BitKeyWrap(4) }” in Clause 13.2.3 on p. 1910 with the following text******“aes128BitKeyWrap(4),******rsa2048BitKey(5),******rsa4096BitKey(6),******aes256BitKeyCbcMode(7),******aes256BitKeyOcbMode(8),******aes256BitKeyGcmMode(9),******ecc224BitKey(10),******ecc256BitKey(11),******ecc384BitKey(12) }”]*** | WmanIf2TekEncryptAlgId ::= TEXTUAL-CONVENTIONSTATUS currentDESCRIPTION"TEK encryption algorithm identifiers."REFERENCE"Table 599"SYNTAX INTEGER {tripleDes128BitKey(1),rsa1024BitKey(2),aes128BitKeyEcbMode(3),aes128BitKeyWrap(4),rsa2048BitKey(5),rsa4096BitKey(6),aes256BitKeyCbcMode(7),aes256BitKeyOcbMode(8),aes256BitKeyGcmMode(9),ecc224BitKey(10),ecc256BitKey(11),ecc384BitKey(12)} |
| 2293 | ***[Replace the text “RSA-based” in the clause title of Clause 14.2.2.2 with the text “RSA/ECC-based”]******[Replace the text “RSA-based’ in Clause 14.2.2.2 with the text “RSA/ECC-based” in the first paragaph]******[Replace the text “RSA-based’ in Clause 14.2.2.2 with the text “RSA/ECC-based” in the third paragaph]******[Replace the text “PKMv2 RSA-Request message” in Clause 14.2.2.2 with the text “PKMv2 RSA/ECCRequest******message”]*** | **RSA/ECC-based authentication procedure**When an SS tries to initiate an RSA/ECC-based authentication or reauthentication procedure with a BS, it sends PKM-REQ messages with Auth Info, Auth Request or PKMv2 RSA/ECC-Request message type. When an NCMS(SS) sends a C-SM-REQ/Certificate\_Information primitive to an IEEE 802.16 entity (SS), the SS sends a PKM-REQ message with Auth Info message type, which includes a CA’s certificate to the IEEE 802.16 entity (BS), and the IEEE 802.16 entity (BS) informs the NCMS(BS) by a C-SM-REQ/ Certificate\_Information primitive. The NCMS(BS) verifies the CA’s certificate if it has no information about the CA and keeps the certificate.When an NCMS(SS) sends a C-SM-REQ/Certificate\_Verification primitive to the IEEE 802.16 entity(SS) to authenticate the SS and the IEEE 802.16 entity (SS) sends a PKM-REQ message with Auth Request or PKMv2 RSA-Request message type, the IEEE 802.16 entity (BS) notifies the NCMS(BS) by a C-SM-REQ/ Certificate\_Verification primitive. The NCMS(BS) verifies the SS’s certificate through asking to a CA and an OCSP (Online Certificate Status Protocol) server. The NCMS returns the result of verification to the IEEE 802.16 entity (BS) whether the SS is authenticated or not as a C-SM-RSP/Certificate\_Verificationprimitive. The IEEE 802.16 entity (BS) sends the result of authentication and security information to theIEEE 802.16 entity (SS) including security key information and the IEEE 802.16 entity (SS) returns the result as a C-SM-RSP/Certificate\_Verification primitive.to the NCMS(SS).Figure 14-5 shows an RSA/ECC-based authentication procedure between an IEEE 802.16 entity and an NCMS on the MS side and the BS side as follows: |
| 2294 | ***[Replace the text “RSA-based” in the caption to Figure 14-5 of Clause 14.2.2.2 with the text “RSA/ECCbased”]******[Replace the text “PKMv2 RSA” with the text “PKMv2 RSA/ECC” in Figure 14-5]*** | **RSA/ECC-based authentication procedure** |