IEEE P802.11
Wireless LANs

|  |
| --- |
| MLO Multi-Link Setup: Security |
| Date: 2020-09-02 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Duncan Ho | Qualcomm |  |  | dho@qti.qualcomm.com |
| George Cherian |  |  |  |
| Alfred Asterjadhi |  |  |  |
| Abhishek Patil |  |  |  |
| Yanjun Sun |  |  |  |
| Menzo Wentink |  |  |  |

Abstract

We propose the draft specification text for MLO Multi-Link Setup Security to help the creation of TGbe draft D0.1.

Revisions:

* Rev 0: Initial version of the document.

The texts is prepared for the following motions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Layer** | **SFD Topic** | **POC** | **TTT** | **R1/R2** | **Notes** |
| MAC | MLO-Multi-link setup: Security | Duncan Ho | Po-kai Huang, Insun Jang, Yonggang Fang, Liwen Chu, Abhishek Patil, Dibakar Das, Yongho Seok, Jarkko Kneckt, Guogang Huang, Rojan Chitrakar, Chenhe Ji, Yonggang Fang, Yong Liu, Jason Yuchen Guo, Xiaofei Wang, Harry Wang, Gabor Bajko, John Yi | R1 | Motion 71Motion 111, #SP0611-29Motion 112, #SP40Motion 119, #SP130 |

“After multi-link setup between two MLDs, different GTK/IGTK/BIGTK in different links with different PN spaces are used.

* GTK/IGTK/BIGTK in different links can be delivered in one 4-way handshake.”

[Motion 71, [21] and [109]]

Proposed changes for Motion 71, [21] and [109] are located in the following subclauses:

* Create a new subsection 33.3.x Multi-link security to describe the security changes for MLO
* 12.7.2 EAPOL-Key frames (KDEs definitions)
* 12.7.5 Nonce generation
* 12.7.6 4-way handshake

“802.11be supports that after multi-link setup between two MLDs, the same PMK and the same PTK across links are used with the same PN space for a PTKSA.”

[Motion 111, #SP0611-29, [13] and [110]]

Proposed changes for Motion 111, #SP0611-29, [13] and [110] are located in the following subclauses:

* 12.5.3.3.2 PN processing
* 12.5.3.3.7 CCM originator processing
* 12.5.5.3.6 GCM originator processing

“Between two MLDs, 802.11be supports using the MLD MAC addresses to derive PMK under SAE method and PTK in 802.11be SFD.”

[Motion 112, #SP40, [13] and [107]]

* Will only cover “using the MLD MAC addresses to derive PTK” here. The rest is covered by Po-kai

Proposed changes for Motion 112, #SP40, [13] and [107] are located in the following subclauses:

* 12.6 RSNA security association
* 12.6.1.1.2 PMKSA
* 12.6.1.1.6 PTKSA
	+ “For the PTKSA derived as a result of the 4-way handshake, there shall be only one PTKSA per band (see 12.6.19 (Protection of robust Management frames)) with the same Supplicant and Authenticator MAC addresses. For the PTKSA derived as a result of an initial mobility domain association or fast BSS transition, there shall be only one PTKSA with the same STA’s MAC address and BSSID.”
* 12.7.1.1 General
* 12.7.1.3 Pairwise key hierarchy (normative)
* 12.7.6 4-way handshake

“An EHT RSNA STA shall support GCMP-256.”

[Motion 119, #SP130, [3] and [173]]

Proposed changes for Motion 119, #SP130, [3] and [173] are located in the following subclause:

* 12.5.5.1 GCMP overview

**Proposed spec text:**

The baseline for this text is 802.11 REVmd draft 3.4 and 802.11ax D6.1.

33. Extreme High Throughput (EHT) MAC specification

**33.3 Multi-link operation**

***TGbe editor: Add new a subclause 33.3.x (Multi-link Security Procedure) under clause 33.3 as follows:***

**33.3.x Multi-link Security**

After a successful multi-link (re)setup between a non-AP MLD and an AP MLD, a PMK is established and a PTK is derived after 4-way handshake between the non-AP MLD and the AP MLD (see 12.7.6 4-way handshake). The PMK, PTK and the same PN space are used for all the setup links between the non-AP MLD and the AP MLD for the PTKSA. The non-AP MLD and the AP MLD use their MLD MAC addresses to derive the PMK under the SAE method and PTK.

Different GTK/IGTK/BIGTK in different links with different PN spaces are used [need to clarify we did not mean link 1 uses 0-50 and link 2 uses 51-100, etc.]. GTK/IGTK/BIGTK of each setup links are delivered to the non-AP MLD in a single 4-way handshake defined in 12.7.6 (4-way handshake).

***TGbe editor: Modify subclause 12.5.3.3.2 (PN processing) as follows:***

* PN processing(#2720)

The PN is incremented by a positive number for each MPDU. The PN shall be incremented in steps of 1 for constituent MPDUs of fragmented MSDUs and MMPDUs. (11ah)For PV0 MPDUs, the PN shall never repeat for a series of encrypted MPDUs using the same temporal key. (11ah)For PV1 MPDUs, the PN shall never repeat for a series of encrypted MPDUs using the same temporal key and TID/ACI.

If the MPDU is to be transmitted by a STA that is affiliated with an MLD, the space of the PN shall be shared among all the STAs affiliated with the MLD.

(#4031)(#2500)If the PN exceeds the threshold that is defined in dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh, an MLME-PN-EXHAUSTION.indication primitive shall be generated.

NOTE 1—When PN space is exhausted, the choices available to an implementation are to replace the corresponding key or to end communications.(#4522)

NOTE 2—When a group addressed MSDU is retransmitted using GCR, it is concealed from non-GCR capable STAs using the procedures described in 11.22.16.3.5 (Concealment of GCR transmissions). The MPDU containing this concealed A‑MSDU has a different PN from the MPDU that contained the original transmission of the group addressed MSDU.

***TGbe editor: Modify subclause 12.5.3.3.7 (CCM originator processing) as follows:***

* CCM originator processing(#2720)

CCM is a generic authenticate-and-encrypt block cipher mode, and in this standard, CCM is used with the AES block cipher.

There are four inputs to CCM originator processing:

* *Key:* the temporal key (16 octets).
* *Nonce:* the nonce (13 octets) constructed as described in (#2720)12.5.3.3.3 (Construct AAD(#2720))(#4262).
* *Frame body:* the plaintext frame body of the MPDU.
* *AAD:* the AAD ((11ah)16–30 octets) constructed from the MPDU header as described in (#2720)12.5.3.3.3 (Construct AAD(#2720)).

(#2720)CCM originator processing provides authentication and integrity of the frame body and the AAD as well as data confidentiality of the frame body. The output from (#2720)CCM originator processing consists of the encrypted data and an encrypted MIC (see Figure 12-16 (Expanded CCMP MPDU(#4384))).

The PN values sequentially number each MPDU. Each transmitter STA that is not affiliated with an MLD and each MLD shall maintain a single PN (48-bit counter) for each PTKSA and GTKSA(#59). Each transmitter STA that is affiliated with an MLD shall use the PN that is maintained by the MLD for the PTKSA and GTKSA. The PN shall be implemented as a 48-bit strictly increasing integer, initialized to 1 when the corresponding temporal key is initialized or refreshed.

A transmitter shall not use an IEEE 802.11 MSDU or A-MSDU priority if this would cause the total number of priorities used during the lifetime of the SA to exceed the number of replay counters supported by the receiver (for a pairwise SA) or all the receivers (for a group SA) for that SA. The transmitter shall not reorder CCMP protected frames that are transmitted to the same RA within a replay counter, but may reorder frames across replay counters. One possible reason for reordering frames is the IEEE 802.11 MSDU or A‑MSDU priority.

The transmitter shall preserve the order of protected robust Management frames that are transmitted to the same DA without the QMF service. When the QMF service is used, the transmitter shall not reorder robust IQMFs within an AC when the frames are transmitted to the same RA.

A CCMP protected individually addressed robust Management frame shall be protected using the same TK as a Data frame.

***TGbe editor: Modify subclause 12.5.5.1 (GCMP overview) as follows:***

* GCMP overview

Subclause 12.5.5 (GCM protocol (GCMP)) specifies the GCMP, which provides data confidentiality, authentication, integrity, and replay protection. A DMG RSNA STA shall support GCMP-128. An EHT RSNA STA shall support GCMP-256.

***TGbe editor: Modify subclause 12.5.5.3.6 (GCM originator processing) as follows:***

* GCM originator processing

GCM is a generic authenticate-and-encrypt block cipher mode, and in this standard, GCM is used with the AES block cipher.

There are four inputs to GCM originator processing:

* *Key:* the temporal key (16 octets).
* *Nonce:* the nonce (12 octets) constructed as described in 12.5.5.3.4 (Construct GCM nonce).
* *Frame body:* the plaintext frame body of the MPDU.
* *AAD:* the AAD (22-30 octets) constructed from the MPDU header as described in 12.5.5.3.3 (Construct AAD).

(#4386)GCM originator processing provides authentication and integrity of the frame body and the AAD as well as data confidentiality of the frame body. The output from (#4386)GCM originator processing consists of the encrypted data and 16 additional octets of (#4093)MIC (see Figure 12-26 (Expanded GCMP MPDU)).

NOTE—In GCM originator processing, the MIC obtained is not encrypted, unlike in CCM originator processing. Refer to NIST Special Publication 800-38D for details.(#4093)

The PN values sequentially number each MPDU. Each transmitter STA that is not affiliated with an MLD and each MLD shall maintain a single PN (48-bit counter) for each PTKSA and GTKSA(#59). Each transmitter STA that is affiliated with an MLD shall use the PN that is maintained by the MLD for the PTKSA and GTKSA. The PN shall be implemented as a 48-bit strictly increasing integer, initialized to 1 when the corresponding temporal key is initialized or refreshed.

A transmitter shall not use IEEE 802.11 MSDU or A‑MSDU priorities without ensuring that the receiver supports the required number of replay counters. The transmitter shall not reorder GCMP protected frames that are transmitted to the same RA within a replay counter, but may reorder frames across replay counters. One possible reason for reordering frames is the IEEE 802.11 MSDU or A‑MSDU priority.

The transmitter shall preserve the order of protected robust Management frames that are transmitted to the same DA without the QMF service. When the QMF service is used, the transmitter shall not reorder robust IQMFs within an AC when the frames are transmitted to the same RA.

A GCMP protected individually addressed robust Management frame shall be protected using the same TK as a Data frame.

***TGbe editor: Modify subclause 12.6.1.1.2 (PMKSA) as follows:***

* PMKSA

(#2418)The PMKSA is created by the Authenticator’s SME and Supplicant’s SME when EAP authentication, SAE authentication, or FILS authentication(11ai) completes successfully, or when the PSK is configured.

(#2418)When the negotiated AKM uses PMKID derivation with KCK as a parameter as defined in 12.7.1.3 (Pairwise key hierarchy), the PMKID derived from the KCK during the initial 4-way handshake is not changed during the lifetime of this PMKSA.(M84)

(M117)(M84)A PMKSA association is bidirectional. In other words, both parties use the information in the security association for both sending and receiving. The PMKSA is used to create the PTKSA. PMKSAs have a certain lifetime. For a non-AP MLD that is associated with an AP MLD, the PMKSA association is between the AP MLD and the non-AP MLD. The PMKSA consists of the following:

* PMKID, as defined in 12.7.1.3 (Pairwise key hierarchy) (M117)or 12.7.1.6.3 (PMK-R0). The PMKID identifies the security association.
* Authenticator’s or peer’s MAC address. For multi-band RSNA, the MAC address is associated with the operating band in use when the PMKSA is established. For MLD, the Authenticator’s MAC address is the MLD MAC address of the AP and the Supplicant’s MAC address is the MLD MAC address of the non-AP MLD.
* PMK(M117); or if the PMKSA was established with an AKM suite type for which the Authentication type column includes FT authentication (see Table 9-153 (AKM suite selectors)), MPMK (see 12.7.1.6.3 (PMK-R0)).
* Lifetime, as defined in 12.7.1.3 (Pairwise key hierarchy) (M117)or 12.7.1.6 (FT key hierarchy).
* AKMP.
* All authorization parameters specified by the AS or local configuration. This might include parameters such as the STA’s authorized SSID.
* Cache Identifier, if advertised by the AP in FILS Indication element(11ai).

***TGbe editor: Modify subclause 12.6.1.1.2 (PMKSA) as follows:***

* PTKSA

The PTKSA results from a successful(11ai) 4-way handshake, FT 4-way handshake, FT protocol, (11ai)FT resource request protocol, or FILS authentication(11ai). This security association is also bidirectional. PTKSAs have the same lifetime as the PMKSA or PMK-R1 security Association, whichever comes first. Because the PTKSA is tied to the PMKSA or to a PMK-R1 security association, it only has the additional information from the 4-way handshake,(11ai) or FT Protocol authentication, or FILS authentication(11ai). For the PTKSA derived as a result of the 4-way handshake, there shall be only one PTKSA per band (see 12.6.19 (Protection of robust Management frames)) or per MLD setup (see 33.3.3 Multi-link Setup) with the same Supplicant and Authenticator MAC addresses. For the PTKSA derived as a result of an initial mobility domain association or fast BSS transition, there shall be only one PTKSA with the same STA’s MAC address and BSSID.

During the 4-way handshake defined in 12.7.6.5 (4-way handshake message 4) and the FT 4-way handshake defined in 13.4.2 (FT initial mobility domain association in an RSN), there is state created between message 1 and message 3 of the handshake. This does not create a PTKSA until message 3 is -validated by the Supplicant and message 4 is validated by the Authenticator.

During the FT authentication sequence defined in 13.8 (FT authentication sequence), the PTKSA is validated when message 3 is validated by the R1KH and message 4 is validated by the S1KH.

During the FILS authentication sequence defined in 12.11.2 (FILS authentication protocol), the PTKSA is validated by key confirmation using (Re)Association Request and (Re)Association Response frames(11ai).

The PTKSA consists of the following:

* PTK
* Pairwise cipher suite selector
* Supplicant MAC address or STA’s MAC address
* Authenticator MAC address or BSSID
* Key ID
* If FT key hierarchy is used,
* R1KH-ID
* S1KH-ID
* PTKName

***TGbe editor: Modify subclause 12.7.1.1 (General) as follows:***

* Keys and key distribution
* Key hierarchy
* General

RSNA defines the following key hierarchies:

* Pairwise key hierarchy, to protect individually addressed traffic
* GTK, a hierarchy consisting of a single key to protect group addressed traffic

NOTE—Pairwise key support with enhanced data cryptographic encapsulation mechanisms allows a receiving STA to detect MAC address spoofing and data forgery. The RSNA architecture binds the transmit and receive addresses to the pairwise key. If an attacker creates an MPDU with the spoofed TA, then the decapsulation procedure at the receiver generates an error. GTKs do not have this property.

* Integrity GTK (IGTK), a hierarchy consisting of a single key to provide integrity protection for group addressed robust Management frames
* (#2116)BIGTK, a hierarchy consisting of a single key to provide for integrity protection for Beacon frames
* (M117)FT key hierarchy, to protect individually addressed traffic in an FT environment

The description of the key hierarchies uses the pseudorandom function producing *n* bits of output, PRF‑*n,* defined in 12.7.1.2 (PRF).

In an infrastructure BSS, the IEEE 802.1X Authenticator MAC address (AA) and the AP’s MAC address are the same, and the Supplicant’s MAC address (SPA) and the STA’s MAC address are equal. (#4568). Between an AP MLD and a non-AP MLD, the the IEEE 802.1X Authenticator MAC address (AA) and the AP MLD’s MAC address are the same, and the Supplicant’s MAC address (SPA) and the non-AP MLD’s MAC address are equal. For the purposes of comparison in this standard, the MAC address is encoded as 6 octets, taken to represent an unsigned integer. The first octet of the MAC address shall be used as the most significant octet. The bit numbering conventions in 9.2.2 (Conventions) shall be used within each octet. This results in a sequence of 48 bits represented such that bit 0 is the first transmitted bit (Individual/Group bit) and bit 47 is the last transmitted bit.(#1300)(#1497)

An RSNA STA shall support at least one pairwise key for any <TA,RA> pair for use with enhanced data cryptographic encapsulation mechanisms. The <TA,RA> identifies the pairwise key, which does not correspond to any WEP key identifier.

An MLD shall support at least one pairwise key for any <transmitter MLD MAC address, receiver\_MLD MAC address> pair for use with enhanced data cryptographic encapsulation mechanisms. The <transmitter\_MLD MAC address, receiver\_MLD MAC address> identifies the pairwise key.

In a a mixed environment, an AP may simultaneously communicate with some STAs using WEP with shared WEP keys and to STAs using enhanced data cryptographic encapsulation mechanisms with pairwise keys. The STAs running WEP use default keys 0–3 for shared WEP keys; the important point here is that WEP can still use WEP default key 0. The AP might be configured to use the WEP key in WEP default key 0 for WEP; if the AP is configured in this way, STAs that cannot support WEP default key 0 simultaneously with a TKIP pairwise key shall specify the No Pairwise subfield in the RSN Capabilities field. If an AP is configured to use WEP default key 0 as a WEP key and a “No Pairwise” STA associates, the AP shall not set the Install bit in the 4-way handshake. In other words, the STA does not install a pairwise temporal key and instead uses WEP default key 0 for all traffic.

NOTE—The behavior of “No Pairwise” STAs is intended only to support the migration of WEP to RSNA.

TKIP STAs in a mixed environment are expected to support a single pairwise key either by using a key mapping key or by mapping to default key 0. The AP uses a pairwise key for individually addressed traffic between the AP and the STA. If a key mapping key is available, the <RA,TA> pair identifies the key; if there is no key mapping key, then the default key 0 is used because the key index in the message is 0.

A STA that cannot support TKIP keys and WEP default key 0 simultaneously advertises this deficiency by setting the No Pairwise subfield in the RSNE it sends in the (Re)Association Request frame to the AP. In response, the AP sets the Install bit to 0 in message 3 of the 4-way handshake to notify the STA not to install the pairwise key. The AP instead sends the WEP shared key to the STA to be plumbed as the WEP default key 0; this key is then used with WEP to send and receive individually addressed traffic between the AP and the STA.

The TKIP STA that has this limitation might not know that it will be forced to use WEP for all transmissions until it has associated with the AP and been given the keys to use. (The STA cannot know that the AP has been configured to use WEP default key 0 for WEP communication.) If this does not satisfy the security policy configured at the STA, the STA’s only recourse is to disassociate and try a different AP.

STAs using enhanced data cryptographic encapsulation mechanisms in a TSN shall support pairwise keys and WEP default key 0 simultaneously. It is invalid for the STA to negotiate the No Pairwise subfield when an enhanced data cryptographic encapsulation mechanism other than TKIP is one of the configured ciphers.

***TGbe editor: Modify subclause 12.7.1.3 (Pairwise key hierarchy) as follows:***

* Pairwise key hierarchy

(M117)Except when preauthentication or FILS authentication(11ai) is used, the pairwise key hierarchy utilizes PRF-384, PRF-512, or PRF-704 to derive session specific(#4663) keys from a PMK, as depicted in Figure 12-30 (Pairwise key hierarchy). When using AKM suite selector 00-0F-AC:12 or 00-0F-AC:15, the length of the PMK, PMK\_bits, shall be 384 bits. When using AKM suite selectors for which the Authentication type column indicates FT authentication (see Table 9-153 (AKM suite selectors)), the FT key hierarchy is used to derive session specific(#4663) keys from an MPMK as defined in 12.7.1.6 (FT key hierarchy). With all other AKM suite selectors, the length of the PMK, PMK\_bits, shall be 256 bits. The pairwise key hierarchy takes a PMK and generates a PTK. The PTK is partitioned into KCK, KEK, and a temporal key, which is used by the MAC to protect individually addressed communication between the Authenticator’s and Supplicant’s respective STAs. PTKs are used between a single Supplicant and a single Authenticator.

For a non-AP MLD associated with an AP MLD, the Supplicant is the non-AP MLD and the Authenticator is the AP MLD. For the rest of this clause and within the context of protecting individually addressed communications between the two MLDs, AA shall be set to the AP MLD MAC address and SPA shall be set to the non-AP MLD MAC address.



(M117)When using IEEE 802.1X authentication, the PMK is derived from the MSK. The PMK shall be computed as the first PMK\_bits bits (bits 0 to PMK\_bits–1) of the MSK: PMK = L(MSK, 0, PMK\_bits). When using SAE or FILS authentication, the PMK is derived per 12.4.5.4 (Processing of a peer’s SAE Commit message) or 12.11.2.5.2 (PMKSA key derivation with FILS authentication), respectively.

The PTK shall not be used longer than the PMK lifetime as determined by the minimum of the PMK lifetime indicated by the AS, e.g., Session-Timeout + dot1xAuthTxPeriod or from dot11RSNAConfigPMK--Lifetime. When RADIUS is used and the Session-Timeout attribute is not in the RADIUS Accept message, and if the key lifetime is not otherwise specified, then the PMK lifetime is -infinite.

NOTE 1—If the protocol between the Authenticator (or AP) and AS is RADIUS, then the MS-MPPE-Recv-Key attribute (-vendor-id = 17; see Section 2.4.3 in IETF RFC 2548 [B29]) is available to be used to transport the first 32 octets of the MSK to the AP, and the MS-MPPE-Send-Key attribute (vendor-id = 16; see Section 2.4.2 in IETF RFC 2548 [B29]) is available to be used to transport the remaining 32 octets of the MSK.

NOTE 2—When reauthenticating and changing the pairwise key, a race condition might occur when using TKIP. If a frame is received while MLME-SETKEYS.request primitive is being processed, the received frame might be decrypted with one key and the MIC checked with a different key. Two possible options to avoid this race condition are as follows: the frame might be checked against the old MIC key, and the received frames might be queued while the keys are changed.

NOTE 3—If the AKMP is RSNA-PSK, then a 256-bit PSK might be configured into the STA and AP or a pass-phrase might be configured into the Supplicant or Authenticator. The method used to configure the PSK is outside this standard, but one method is via user interaction. If a pass-phrase is configured, then a 256-bit key is derived and used as the PSK. In any RSNA-PSK method, the PSK is used directly as the PMK. Implementations might support different PSKs for each pair of communicating STAs.

(M117)(#2454)The following apply when not using FILS authentication:

* SNonce is a random or pseudorandom value contributed by the Supplicant; its value is taken when a PTK is instantiated and is sent to the PTK Authenticator.
* ANonce is a random or pseudorandom value contributed by the Authenticator.
* The PTK shall be derived from the PMK by

PTK = PRF-Length(PMK, “Pairwise key expansion”, Min(AA,SPA) || Max(AA,SPA) || Min(ANonce,SNonce) || Max(ANonce,SNonce))

where Length = KCK\_bits + KEK\_bits + TK\_bits. The values of KCK\_bits and KEK\_bits are AKM suite dependent and are listed in Table 12-10 (Integrity and key-wrap algorithms(#102)(#1188)). The value of TK\_bits is cipher-suite dependent and is defined in Table 12-7 (Cipher suite key lengths). The Min and Max operations for IEEE 802 addresses are with the address converted to a positive integer treating the first transmitted octet as the most significant octet of the integer. The nonces are encoded as specified in 9.2.2 (Conventions).

NOTE 4(11ai)—The Authenticator and Supplicant normally derive a PTK only once per association. A Supplicant or an Authenticator use the 4-way handshake or FILS authentication(11ai) to derive a new PTK. Both the Authenticator and Supplicant create a new nonce value for each 4-way handshake or FILS authentication(11ai) instance.

* The KCK shall be computed as the first KCK\_bits bits (bits 0 to KCK\_bits–1) of the PTK:

KCK= L(PTK, 0, KCK\_bits)

The KCK is used by IEEE Std 802.1X-2010 to provided data origin authenticity in the 4-way handshake and group key handshake messages.

* The KEK shall be computed as the next KEK\_bits bits of the PTK:

KEK = L(PTK, KCK\_bits, KEK\_bits)

The KEK is used by the EAPOL-Key frames to provide data confidentiality in the 4-way handshake and group key handshake messages.

* The temporal key (TK) shall be computed as the next TK\_bits bits of the PTK:

TK = L(PTK, KCK\_bits+KEK\_bits, TK\_bits)

(M117)When using FILS authentication, the PTK is derived as defined in 12.11.2.5.3 (PTKSA Key derivation with FILS authentication).

The EAPOL-Key state machines (see 12.7.9 (RSNA Supplicant key management state machine) and 12.7.10 (RSNA Authenticator key management state machine)) use the MLME-SETKEYS.request primitive to configure the temporal key into the STA. The STA uses the temporal key with the pairwise cipher suite; interpretation of this value is cipher-suite dependent(#1408).

(M117)When the negotiated AKM is (#114)00-0F-AC:5 or 00-0F-AC:6, the PMK identifier is defined as

(#2599)PMKID = Truncate-128(HMAC-SHA-256(PMK, “PMK Name” || AA || SPA))

When the negotiated AKM is 00-0F-AC:11, the PMK identifier is defined as

(#2599)PMKID = Truncate-128(HMAC-SHA-256(KCK, “PMK Name” || AA || SPA))

When the negotiated AKM is 00-0F-AC:12, and the PMK identifier is defined as

PMKID = (#2599)Truncate-128(HMAC-SHA-384(KCK, “PMK Name” || AA || SPA))

(M117)When the negotiated AKM is 00-0F-AC:14:

* When IEEE 802.1X authentication is used, the PMK identifier is defined as:

(#2599)PMKID = Truncate-128(HMAC-SHA-256(PMK, “PMK Name” || AA || SPA))

* When FILS authentication is used, the PMK identifier is derived as defined in 12.11.2.5 (Key establishment with FILS authentication).

(M117)When the negotiated AKM is 00-0F-AC:15:

* When IEEE 802.1X authentication is used, the PMK identifier is defined as:

(#2599)PMKID = Truncate-128(HMAC-SHA-384(PMK, “PMK Name” || AA || SPA))

* When FILS authentication is used, the PMK identifier is derived as defined in 12.11.2.5 (Key establishment with FILS authentication).

(M117)(#114)When the negotiated AKM is 00-0F-AC:20(#2209), the PMK identifier is defined as

(#2599)PMKID = Truncate-128(HMAC-SHA-384(PMK, “PMK Name” || AA || SPA))

When the negotiated AKM is 00-0F-AC:8, the PMK identifier is derived as defined in 12.4.5.4 (Processing of a peer’s SAE Commit message).(#2208)

(M117)When the negotiated AKM is a suite type for which the Authentication type column indicates FT authentication (see Table 9-153 (AKM suite selectors)), the PMKID (used for PMKSA caching in FT Initial Mobility Domain Association, see 12.6.10.3 (Cached PMKSAs and RSNA key management)) and PMKR0Name are derived as defined in 12.7.1.6.3 (PMK-R0) and PMKR1Name is derived as defined in 12.7.1.6.4 (PMK-R1).(#2210)

Otherwise, the PMK identifier is defined as

(#2599)PMKID = Truncate-128(HMAC-SHA-1(PMK, “PMK Name” || AA || SPA))

In all these cases, “PMK Name” is treated as an ASCII string.

When the PMKID is calculated for the PMKSA as part of preauthentication, the AKM has not yet been negotiated. In this case, the HMAC-SHA-1 based derivation is used for the PMKID calculation.

***TGbe editor: Modify subclause 12.7.2 (EAPOL-Key frames) as follows:***

* EAPOL-Key frames

IEEE Std 802.11 uses EAPOL-Key frames to exchange information between STAs’ Supplicants and Authenticators. These exchanges result in cryptographic keys and synchronization of security association state. EAPOL-Key frames are used to implement three different exchanges:

* 4-way handshake, to confirm that the PMK between associated STAs is the same and live and to transfer the GTK to the STA.
* Group key handshake, to update the GTK at the STA.(#59)

[…]

Table 12-9 (KDE selectors(#2501)) lists the KDE selectors defined by this standard.

|  |
| --- |
| * KDE selectors(#2501)
 |
| OUI | Data type | Meaning |
| 00-0F-AC | 0 | Reserved |
| 00-0F-AC | 1 | GTK KDE |
| 00-0F-AC | 2 | Reserved |
| 00-0F-AC | 3 | MAC address KDE |
| 00-0F-AC | 4 | PMKID KDE |
| 00-0F-AC | 5 | Reserved(#59) |
| 00-0F-AC | 6 | Nonce KDE |
| 00-0F-AC | 7 | Lifetime KDE |
| 00-0F-AC | 8 | Error KDE |
| 00-0F-AC | 9 | IGTK KDE |
| 00-0F-AC | 10 | Key ID KDE |
| 00-0F-AC | 11 | Multi-band GTK KDE |
| 00-0F-AC | 12 | Multi-band Key ID KDE |
| 00-0F-AC | 13(Ed) | OCI KDE(M58) |
| 00-0F-AC | 14 | BIGTK KDE(#2116) |
| 00-0F-AC | 15 | MLO GTK KDE |
| 00-0F-AC | 16 | MLO IGTK KDE |
| 00-0F-AC | 17 | MLO BIGTK KDE |
| 00-0F-AC | 18–255 | Reserved |
| Other OUI or CID | Any | Vendor specific |

[…]

The following EAPOL-Key frames are used to implement the three different exchanges:

* **4-way handshake message 1** is an EAPOL-Key frame with the Key Type subfield equal to 1. (M117)Use of the Key Data field to indicate a PMKID when a cached PMKSA is being used in this key derivation is defined in 12.6.10.3 (Cached PMKSAs and RSNA key management). When a cached PMKSA is not being used, inclusion of the PMKID (if derived) is optional. The Key Data field need not be encrypted.
* **4-way handshake message 2** is an EAPOL-Key frame with the Key Type subfield equal to 1. The Key Data field shall contain an RSNE(#2715), may contain an RSNXE, and need not be encrypted.

An ESS Supplicant’s SME shall insert the RSNE it sent in its (Re)Association Request frame(#2715), and shall insert the RSNXE it sent in its (Re)Association Request frame if the RSNXE is present in the (Re)Association Request frame it sent. The RSNE and the RSNXE are included as transmitted in the Management frame. On receipt of message 2, the Authenticator’s SME shall validate the selected security configuration against the RSNE received in (#2715)the (Re)Association Request frame, and shall validate the RSNXE included in message 2 against the RSNXE received in the (Re)Association Request frame from the Supplicant.

An IBSS Supplicant’s SME shall insert an RSNE containing a selected pairwise cipher suite. The Authenticator’s SME shall validate that the pairwise cipher suite selected is one of its configured cipher suites and that the group cipher suite and AKM are consistent.

* **4-way handshake message 3** is an EAPOL-Key frame with the Key Type subfield equal to 1. The Key Data field shall contain one or two RSNEs(#2715), and may contain an RSNXE. If a group cipher has been negotiated, this field shall also include a GTK. This field shall be encrypted if a GTK is included. When the Authenticator is an AP MLD and the Supplicant is a non-AP MLD, this field shall additionally include one MLO GTK for each setup link (see 33.3.2 (Multi-link (re)setup procedure)).

(#1552)(#2599)An Authenticator’s SME shall insert the RSNE it sent in its Beacon or Probe Response frame(#2715), and shall insert the RSNXE it sent in its Beacon or Probe Response frame if the RSNXE is present in the Beacon or Probe Response frame it sent. When this message 3 is part of a fast BSS transition initial mobility domain association or an association started through the FT protocol, the PMKR1Name is added in the PMKID List field of the RSNE. The Supplicant’s SME shall validate the selected security configuration against the RSNE received in message 3(#2715), and shall validate the RSNXE included in message 3 against the RSNXE received in the Beacon or Probe Response frame from the Authenticator. If the second optional RSNE is present, the STA shall either use that cipher suite with its pairwise key or deauthenticate. (#2715)In any of these cases, if the values do not match, then the receiver shall consider the RSNE or the RSNXE modified and shall use the MLME-DEAUTHENTICATE.request primitive to break the association. A security error should be logged at this time.

It may happen, for example, that a Supplicant selects a pairwise cipher suite which is advertised by an AP, but which policy disallows for this particular STA. An Authenticator may, therefore, insert a second RSNE to overrule the STA’s selection. An Authenticator’s SME shall insert the second RSNE, after the first RSNE, only for this purpose. The pairwise cipher suite in the second RSNE included shall be one of the ciphers advertised by the Authenticator. All other fields in the second RSNE shall be identical to the first RSNE.

A GTK shall be included and the unencrypted length of the GTK is six less than the length of the GTK KDE in octets. The entire Key Data field shall be encrypted as specified by the Key Descriptor Version.

* **4-way handshake message 4** is an EAPOL-Key frame with the Key Type subfield equal to 1. The Key Data field can be empty.
* **Group key handshake message 1** is an EAPOL-Key frame with the Key Type subfield equal to 0. The Key Data field shall contain a GTK KDE and shall be encrypted.
* **Group key handshake message 2** is an EAPOL-Key frame with the Key Type subfield equal to 0. The Key Data field can be empty.

[…]

The format of the MLO GTK KDE is shown in Figure 12-36 (GTK KDE format)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Key ID | Tx | Reserved | Reserved | GTK | LinkID | KeyRSC |
| Bits: | 2 | 1 | 4 | 8 | (Length - 4) × 8 | TBD | 48 |
| * MLO GTK KDE format
 |

If the value of the Tx field is 1, then the IEEE 802.1X componentshall configure the temporal key derived from this KDE into its IEEE 802.11 MAC for both transmission and reception.

If the value of the Tx field is 0, then the IEEE 802.1X componentshall configure the temporal key derived from this KDE into its IEEE 802.11 MAC for reception only.

The LinkID field contains the link identifier that corresponds to the STA for which this GTK applies.

The KeyRSC field contains the Key RSC field that corresponds to the STA for which this GTK applies (see Table 12-5 for the description of the Key RSC field).

The format of the MLO IGTK KDE is shown in Figure 12-42 (MLO IGTK KDE format).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Key ID | IPN | IGTK | LinkID |
| Octets: | 2 | 6 | (Length – 12) | TBD |

The LinkID corresponds to the link identity of the STA for which this IGTK applies.

* MLO IGTK KDE format

The format of the MLO BIGTK KDE is shown in Figure 12-47 (MLO BIGTK KDE (#2116)).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Key ID | BIPN | BIGTK | LinkID |
| Octets: | 2 | 6 | (Length - 12) | TBD |
| MLO BIGTK KDE (#2116) |

The BIPN corresponds to the BIPN value that was carried in the MME of the last protected Beacon frame corresponding to the LinkID and it is used by the receiver as the initial value for the BIP replay counter for the BIGTK.

***TGbe editor: Modify subclause 12.7.5 (Nonce generation) as follows:***

* Nonce generation

The following is (M138)a description of nonce(#1406) generation.

All STAs contain a global key counter, which is 256 bits in size. It should be initialized at system boot-up time to a fresh cryptographic-quality random number. Refer to J.5 (Suggestions for random number generation) on random number generation. It is recommended that the counter value is initialized to the following:

PRF-256(Random number, “Init Counter”, Local MAC Address || Time)

The local MAC address should be AA on the Authenticator and SPA on the Supplicant. When the Authenticator is an AP MLD and the Supplicant is a non-AP MLD, the AA shall be the AP MLD MAC address and the SPA shall be the non-AP MLD MAC address.

The random number is 256 bits in size. Time should be the current time from Network Time Protocol (NTP) or another time in NTP format whenever possible. This initialization causes different initial key counter values to occur across system restarts regardless of whether a real-time clock is available. The key counter can be used as additional input data for nonce generation. A STA derives a random nonce for each new use.

***TGbe editor: Modify subclause 12.7.6.4 (4-way handshake message 3) as follows:***

* 4-way handshake message 3

Message 3 uses the following values for each of the EAPOL-Key frame fields:

Descriptor Type **=** N – see 12.7.2 (EAPOL-Key frames)

Key Information:

Key Descriptor Version = 1 (ARC4 encryption with HMAC-MD5) or 2 (NIST AES key wrap with HMAC-SHA-1-128) or 3 (NIST AES key wrap with AES-128-CMAC), in all other cases 0 – same as message 1

Key Type = 1 (Pairwise) – same as message 1

(#59)Reserved = 0

Install = 0/1 – For PTK generation, 0 only if the AP does not support key mapping keys, or if the STA has the No Pairwise bit (in the RSN Capabilities field) equal to 1and only the group key is used.(#59)

Key Ack = 1

Key MIC = 0 when using an AEAD cipher or 1 otherwise(11ai)

Secure = 1 (keys installed)

Error = 0 – same as message 1

Request = 0 – same as message 1

Encrypted Key Data = 1

Reserved = 0 – unused by this protocol version

Key Length = Cipher-suite dependent(#1408); see Table 12-7 (Cipher suite key lengths)

Key Replay Counter = *n+1*

Key Nonce = ANonce – same as message 1

EAPOL-Key IV = 0 (Version 2) or random (Version 1)

Key RSC = For PTK generation, starting TSC or PN that the Authenticator’s STA uses in MPDUs protected by GTK.(#59)

Key MIC = Not present when using an(Ed) AEAD cipher; or otherwise(11ai), MIC(KCK, EAPOL) or MIC(SKCK, EAPOL) – MIC computed over the body of this EAPOL-Key frame with the Key MIC field first initialized to 0

(#2440)Key Data Length = length of Key Data field in octets

Key Data =

* For PTK generation for the current operating band, the AP’s Beacon/Probe Response frame’s RSNE for the current operating band, and, optionally, a second RSNE that is the Authenticator’s pairwise cipher suite assignment for the current operating band, and, if a group cipher has been negotiated, the GTK and the GTK’s key identifier (see 12.7.2 (EAPOL-Key frames)) for the current operating band, and if management frame protection is negotiated, the IGTK KDE, and if beacon protection is enabled, the BIGTK KDE(#2116), and when this message 3 is part of a fast BSS transition initial mobility domain association or an association started through the FT protocol, the PMKR1Name calculated according to the procedures of 12.7.1.6.4 (PMK-R1) in the (#2205)PMKID List field of the RSNE and the FTE with the same contents as in the (Re)Association Response frame, the MDE with the same contents as in the (Re)Association Response frame, the reassociation deadline timeout set to the minimum of dot11FTReassociationDeadline and the key lifetime in the TIE[ReassociationDeadline], and the PTK lifetime in the TIE[KeyLifetime]; or
* For PTK generation for a supported band other than the current operating band, the Authenticator’s Beacon/DMG Beacon/Announce/Probe Response/Information Response frame’s Multi-band element associated with the supported band, and optionally a second Multi-band element that indicates the Authenticator’s pairwise cipher suite assignment for the supported band, and, if group cipher for the supported band is negotiated, the Multi-band GTK KDE for the supported band if dot11MultibandImplemented is true, or;
* For generating a single PTK for all involved bands, the Authenticator’s Beacon/DMG Beacon/Announce/Probe Response/Information Response frame’s RSNE and Multi-band element(s), and optionally, additional RSNE and Multi-band element(s) that indicate the Authenticator’s assignment of one pairwise cipher suite for all involved bands; if a group cipher for all involved bands is negotiated, the GTK and the GTK’s key identifier for all involved bands, if dot11MultibandImplemented is true and both the Authenticator and the Supplicant use the same MAC address in the current operating band and the other supported band(s), or;
* For generating different PTKs for the current operating band and other supported band(s), the Authenticator’s Beacon/DMG Beacon/Announce/Probe Response/Information Response frame’s RSNE and Multi-band element(s), and optionally, additional RSNE and Multi-band elements that are the Authenticator’s pairwise cipher suite assignments for one or more involved bands; if group ciphers for the involved bands are negotiated, the Multi-band GTK KDEs for the involved bands, if dot11MultibandImplemented is true and the Joint Multi-band RSNA subfield is 1 for both the Authenticator and Supplicant, and either the Authenticator or the Supplicant uses different MAC addresses for different bands.(#59), or;
* For generating a single PTK between a non-AP MLD associated with an AP MLD, the AP’s Beacon/Probe Response frame’s RSNE for the current operating band, and, optionally, a second RSNE that is the Authenticator’s pairwise cipher suite assignment for the current operating band, and, if a group cipher has been negotiated, the GTK and the GTK’s key identifier (see 12.7.2 (EAPOL-Key frames)) of each setup link (see (see 33.3.2 (Multi-link (re)setup procedure), and if management frame protection is negotiated, the IGTK KDE of each setup link, and if beacon protection is enabled, the BIGTK KDE(#2116) for each setup link, and when this message 3 is part of a fast BSS transition initial mobility domain association or an association started through the FT protocol, the PMKR1Name calculated according to the procedures of 12.7.1.6.4 (PMK-R1) in the (#2205)PMKID List field of the RSNE and the FTE with the same contents as in the (Re)Association Response frame, the MDE with the same contents as in the (Re)Association Response frame, the reassociation deadline timeout set to the minimum of dot11FTReassociationDeadline and the key lifetime in the TIE[ReassociationDeadline], and the PTK lifetime in the TIE[KeyLifetime]
* (#2329)Additionally, contains an OCI KDE when dot11RSNAOperatingChannelValidationActivated is true on the Authenticator.(M58)
* (#2715)The RSNXE that the Authenticator sent in its Beacon or Probe Response frame, if this element is present in the Beacon or Probe Response frame that the Authenticator sent.

**Straw Poll: Do you support to incorporate the proposed draft text in this document 11-20/1445r0 to the TGbe Draft 0.1?**

**Result: Yes/No/Abstain**