IEEE P802.11
Wireless LANs

|  |
| --- |
| Proposed Resolution for some LB249 CRs |
| Date: June 29, 2020 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Nehru Bhandaru | Broadcom | 250 Innovation Drive, San Jose CA | +1 408 391 2159 | nehru.bhandaru@broadcom.com |

Abstract

This document contains proposed resolution for the CID 3880 against TGaz Draft 2.0 from LB249. Proposed changes are relative to TGmd Draft 3.3, TGba Draft 6.1 and TGaz Draft 2.2.

| CID | Page | Clause | Comment | Proposed Change | Resolution |
| --- | --- | --- | --- | --- | --- |
| 3880 |   | 12.7.1.3 | Tgba Draft 5.0 - also extends the key hierarchies - Pairwise (12.7.1.3), FT (12.7.1.6), FILS (12.12.2) to derive key material for use as WTK (WUR Transient Key). Tgaz 2.0 draft also extends the same key hierarchies and must reconcile with changes forthcoming in TGba. | Either Tgba uses HLTK based derivation for WUR key or Tgaz needs to modify the derivation to take into account the bits, if any, required for WUR operation. |  |

**Discussion**

CID 3880 was based on TGba Draft 5.0 which also modifies PTK derivation for obtaining a WUK pairwise key. TGba Draft 6.1 adopted a different mechanism that is aligned with the scheme used in TGaz viz. deriving a high level transient key (HLTK) as part of PTK derivation.

However, the terminology used in TGba draft is slightly different (called KDK – key derivation key).

Some of the changes are common to the current TGaz draft (2.2) and TGba.

This submission attempts to reconcile the changes in TGaz draft with those in TGba draft with respect to HLTK/KDK derivation.

Summary of changes to TGaz draft

* Replace HLTK with KDK throughout the document. J.13 KDK (HLTK) test vectors can be kept for now and removed when test vectors appear TGba
* Remove HLTK from definitions

**Proposed Changes**

**Tgaz Editor: Replace the word ‘HLTK’ with ‘KDK’ throughout the draft for consistency with TGba.**

**Tgaz Editor: Remove the definition of HLTK from §3.2 definitions and §3.4 abbreviations**

**…p20.21**

**~~Higher Layer Transient Key (HLTK):~~** ~~Input key material derived as a seed to be used for higher layer protection.~~

**…p21.7**

**~~HLTK~~** ~~Higher Layer Transient Key~~

**Tgaz Editor: Remove the following change from TGaz draft - § 12.6.1.1.6, p178.12**

***~~Insert the following text at the end of the clause:~~*** ~~11~~

~~— HLTK, if higher layer security is supported 12~~

~~HLTK shall be derived if dot11SecureLTFImplemented is true and the peer STA has indicated 13 Secure LTF Support capability in its advertised Extended Capabilities.~~

**Tgaz Editor: Change the text at the end of clause § 12.6.1.1.6 (after applying TGba changes) - as follows – TGmd p178.12 – Tgba p90.49**

12.6.1.1.6 PTKSA

…

THE PTKSA consists of the following:

—PTK, where the PTK includes the KDK when any of the following are true

* WUR frame protection is negotiated (#7088)
* dot11SecureLTFImplemented is true and the peer STA has advertised Secure LTF Support capability in its Extended Capabilities

—Pairwise cipher suite selector, and when WUR frame protection is negotiated, the cipher suite selector 00-0F-AC:6 (BIP-CMAC-128) for individually addressed WUR Wake-up frames

…

---

Tgaz Editor: Remove all changes from TGaz draft - § 12.7.1.3 Pairwise key hierarchy p178.25. These changes include modified figure 12-30 that is updated by TGba i.e. keep the figure from TGba. See changes later in this document

Tgaz Editor: Remove all changes from TGaz draft - § 12.7.1.6 FT key hierarchy p179.17. This section is updated by TGba. See changes later in this document.

---

**Tgaz Editor: Change clause § 12.7.1.3 Pairwise key hierarchy as follows – updated by TGba p93.5**

…

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 keys from a PMK, as depicted in Figure 12-30 (Pairwise key hierarchy). When using AKM suite selector 00-0F-AC:12, the length of the PMK, PMK\_bits, shall be 384 bits. 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, a temporal key, and optionally a KDK.  ~~if WUR frame protection is negotiated; otherwise the PTK is partitioned into KCK, KEK, and a temporal key,~~  A KDK is derived *if and only if* any of the following are true

* WUR frame protection is negotiated
* dot11SecureLTFImplemented is true and the peer STA has advertised Secure LTF Support capability in its Extended Capabilities

The temporal key which is used by the MAC to protect individually addressed communication between the Authenticator's and Supplicant's respective STAs. If WUR frame protection is negotiated, the KDK is used to derive a WTK, which is used by the MAC of the WUR AP to protect and by the MAC of the WUR non-AP STA to validate individually addressed WUR Wake-up frames. PTKs are used between a single Supplicant and a single Authenticator.(#7088)

…

The following apply when not using FILS authentication:

—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 + KDK\_bits~~, if WUR frame protection is being negotiated~~

 ~~; otherwise, Length = KCK\_bits + KEK\_bits + TK\_bits.~~ The values of KCK\_bits and KEK\_bits are AKM suite dependent

and are listed in Table 12-8 (Integrity and key-wrap algorithms). The value of TK\_bits is cipher-suite dependent and is defined in Table 12-5 (Cipher suite key lengths). If a KDK is derived, the ~~The~~ value of KDK\_bits is equal to the value of PMK\_bits; otherwise the value of KDK\_bits shall be 0 . 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).(#7088)

~~—If WUR frame protection is being negotiated, the KDK shall be computed as the next KDK\_bits bits of the PTK:(#7088)~~

* If a KDK is derived, it shall be computed as the next KDK\_bits bits of the PTK:

KDK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits, KDK\_bits)(#7088)

~~Otherwise, the KDK is not derived.(#~~

…

**Tgaz Editor: Change clause § 12.7.1.6 FT key hierarchy as follows – updated by TGba p94.40**

**12.7.1.6 FT key hierarchy**

**12.7.1.6.5 PTK**

***Change the last bullet of the 2nd paragraph as follows:***

Using the KDF defined in 12.7.1.6.2 (Key derivation function (KDF)), the PTK derivation is as follows:

PTK = KDF-Hash-Length(PMK-R1, "FT-PTK", SNonce || ANonce || BSSID || STA-ADDR)

where

—Length is the total number of bits to derive, i.e., number of bits of the PTK. The length is dependent on the negotiated cipher suites and AKM suites as defined by Table 12-5 (Cipher suite key lengths) in 12.7.2 (EAPOL-Key frames) and Table 12-8 (Integrity and key-wrap algorithms) in 12.7.3 (EAPOL-Key frame construction and processing~~), and whether WUR frame protection is negoti­ated.~~  , and whether a KDK is derived. A KDK shall be derived *if and only if* any of the following are true

* WUR frame protection is negotiated
* dot11SecureLTFImplemented is true and the peer STA has advertised Secure LTF Support capability in its Extended Capabilities

***Change the 3rd paragraph as follows:***

Except when ~~WUR frame protection is negotiated~~ a KDK is derived, each PTK has five component keys, KCK, KEK, a tem­poral key, KCK2, and KEK2 derived as follows:

**TGba p95.1**

***Insert the following after five component key are described as follows:***

The KEK2 is used to provide data confidentiality for certain fields in the FT authentication sequence, as defined in 13.8 (FT authentication sequence).

When ~~WUR frame protection is negotiated~~ a KDK is derived, each PTK has six component keys, KCK, KEK, a temporal key, KCK2, KEK2, and a KDK derived as follows:(#7088)

The KCK, KEK, temporal key, KCK2, and KEK2 shall be computed in the same way as when a KDK is not derived ~~WUR frame protection is not negotiated~~.

The KDK shall be computed as the next KDK\_bits bits of the PTK:(#7088)

KDK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits+KCK2\_bits+KEK2\_bits, KDK\_bits)(#7088)

The value of KDK\_bits is equal to the value of PMK\_bits (see 12.7.1.3 (Pairwise key hierarchy)).(#7088)

…

**Tgaz Editor: Change clause § 12..11.2.5.3 PTKSA key derivation with FILS as follows – updated by TGba p100.57**

**12.11.2.5.3 PTKSA Key derivation with FILS authentication**

***Change the first paragraph as follows:***

For PTKSA key generation, the inputs to the PRF are the PMK of the PMKSA, a constant label, and a con­catenation of the STA's MAC address, the AP's BSSID, the STA's nonce, and the AP's nonce. When the negotiated AKM is 00-0F-AC:14 or 00-0F-AC:16, the length of KEK shall be 256 bits, and the length of the

ICK shall be 256 bits. When the negotiated AKM is 00-0F-AC:15 or 00-0F-AC:17, the length of the KEK shall be 512 bits, and the length of ICK shall be 384 bits. When the negotiated AKM is 00-0F-AC:16, FILS-FT is 256 bits; when the negotiated AKM is 00-0F-AC:17, FILS-FT is 384 bits; otherwise, FILS-FT is not derived; when a KDK is derived ~~WUR frame protection is negotiated~~, the length of KDK is equal to the value of PMK\_bits (see 12.7.1.3 (Pairwise key hierarchy)); A KDK shall be derived *if and only if* any of the following are true

* WUR frame protection is negotiated
* dot11SecureLTFImplemented is true and the peer STA has advertised Secure LTF Support capability in its Extended Capabilities

~~otherwise, the KDK is not derived~~. The total amount of bits extracted from the KDF shall therefore be 640+TK bits, 1124+TK bits, or 1408+TK bits depending on the negotiated AKM when WUR frame protection(#Ed) is negotiated, otherwise, shall be 512+TK bits, 896+TK bits, or 1280+TK bits depending on the negotiated AKM, where TK\_bits are determined from Table 12-4:(#7088)

FILS-Key-Data = PRF-X(PMK, "FILS PTK Derivation", SPA || AA || SNonce || ANonce [ || DHss ])

ICK = L(FILS-Key-Data, 0, ICK\_bits)

KEK = L(FILS-Key-Data, ICK\_bits, KEK\_bits)

TK = L(FILS-Key-Data, ICK\_bits + KEK\_bits, TK\_bits)

When doing FT initial mobility domain association using FILS authentication,

FILS-FT = L(FILS-Key-Data, ICK\_bits + KEK\_bits + TK\_bits, FILS-FT\_bits)

When ~~WUR frame protection is negotiated~~ a KDK is derived while doing FT initial mobility domain association using FILS authentication,

KDK = L(FILS-Key-Data, ICK\_bits + KEK\_bits + TK\_bits + FILS-FT\_bits, KDK\_bits)(#7088)

When ~~WUR frame protection is negotiated~~ a KDK is derived while not doing FT initial mobility domain association using FILS authentication,

KDK = L(FILS-Key-Data, ICK\_bits + KEK\_bits + TK\_bits, KDK\_bits)(#7088)

where

ICK\_bitsis the length of ICK in bits

KEK\_bitsis the length of KEK in bits

FILS-FT\_bitsis the length of FILS-FT in bits when doing FT initial mobility domain associa­tion using FILS authentication

KDK\_bits is the length of KDK in bits and is equal to the value of PMK\_bits (see 12.7.1.3 (Pairwise key hierarchy)).(#7088)

X is 512+TK bits+KDK\_bits, 768+TK bits+KDK\_bits, or 896+TK bits+KDK\_bits or 1280+TK bits+KDK\_bits from Table 12-5 (Cipher suite key lengths) depending on the negotiated AKM when a KDK is derived ~~WUR frame protection is negotiated~~; otherwise, X is 512+TK\_bits, 768+TK bits, 896+TK bits, or 1280+TK bits from Table 12-5 (Cipher suite key lengths) depending on the negotiated AKM