**IEEE P802.11
Wireless LANs**

|  |
| --- |
| Key derivation extensions for 802.11az measurement security |
| Date: 2018-07-10 |
| Author(s): |
| Name | Company | Address | Phone | Email |
| Nehru Bhandaru | Broadcom Ltd. | 250 Innovation Drive, San Jose CA 95134 USA | 408-391-2159 | nehru.bhandaru@broadcom.com |
| Thomas Derham | Broadcom Ltd. | 16340 West Bernardo, San Diego,California,US,92127 |  | thomas.derham@broadcom.com |

**Abstract**

This submission contains proposed 802.11az amendment text to provide key derivation extensions to support ranging measurement security. Secure HEz ranging requires key material that is derived from (and thus bound to) the security state (e.g. PMKSA) between the peers to generate a protected LTF sequence. The input key material to generate the LTF sequence is identified by a sequence authentication code (SAC) that is provided by HEz responder via HEz negotiation protocol. This document provides the rationale and mechanism for deriving the secret input to the LTF generation mechanism for a given SAC.

Proposed changes are relative to *IEEE P802.11-REVmd™/D1.2, May 2018 [1] and IEEE P802.11-az/D0.3, July 2018[2]*

# Document History

r0 – Initial Revision

# References

[1] Draft P802.11REVmd D1.2 - <http://www.ieee802.org/11/private/Draft_Standards/11md/Draft%20P802.11REVmd_D1.2.pdf>

[2] Draft P802.11az D0.3 - <http://www.ieee802.org/11/private/Draft_Standards/11az/Draft%20P802.11az_D0.3.pdf>

[3] TGaz Functional Requirements – <https://mentor.ieee.org/802.11/dcn/16/11-16-0424-11-00az-proposed-802-11az-functional-requirements.docx>

[4] TGaz Specification Framework - <https://mentor.ieee.org/802.11/dcn/17/11-17-0462-15-00az-11-az-tg-sfd.doc>

# Document Conventions

Suggested changes are specified as follows

* Red for editorial instructions
* Strikethrough for text to be deleted
* Underlined and Blue for new proposed text
* Figures or changes to existing figures are described black and white or any other color.
* Black for existing text
* Prefix changes with

**<draft> Editor: [Add, Change, Delete, Replace] <description>(<section>, p<page>, <line>)**

Where **<draft>** is **TGaz** for TGaz draft [2] changes or empty for base specification [1] changes

Existing clauses are identified by section, page and line numbers.

# Discussion - Topics to Cover

* Extend pairwise key hierarchy to include HLTK (Higher layer transient key seed)
* 12.7.1.3 pairwise key hierarchy
* 12.7.1.6.5 FT pairwise key derivation
* 12.12.2.5.3 PTKSA Key derivation with FILS authentication
* PASN PTKSA derivation
* Derivation based on MIB/Feature support

# Discussion – General

Secure HEz ranging requires key material that is derived from (and thus bound to) the security state (e.g. PMKSA) between the peers to generate a protected LTF sequence. The input key material to generate the LTF sequence is identified by a sequence authentication code (SAC) that is provided by HEz responder via HEz negotiation protocol. Relevant references from section 6 Security of SFD [4] are:

 (6) …field used for measurement shall be derived from random sequences known to I-STA and R-STA only

 (12) Security keys for 11az Management Frame Protection and range measurement field protection are derived based on security negotiation

(14) (a) for VHTz and 17 (c) for HEz- The keys or cipher sequence (if needed) for LTF sequence generation are the result of the FTM negotiation

 (18) (d) An adversary that does not know the SAC is unable to predict it.

It would be nice to have the same mechanism apply to pre-association as well as post-association security. From a security perspective, the options (number of symbols, streams etc.) could be handled by deriving a possibly different number of bits using the KDF.

# Discussion – where does the material come from

It is possible that input bits to generate LTF can use an RNG that is not bound to the SA between peers and conveyed by the responder or initiator to ranging the peer securely using PMF. The number of LTFs in a given HEz measurement frame (NDP) can be fairly large (~96) due to MU, repetition for reliability and security consistency checks, LTFs for multi-NSS support, and random CSD shift. In addition, the number of required input bits required would also vary based on BW and modulation scheme used for the measurement frame. Consequently, it is prudent for the ranging peers to independently derive the input bits using a derivation function over the established security state between the peers.

Key material may be directly derived from PMK similar to PTK derivation or PTK derivation be extended to derive an application specific key seed that each higher layer (HL) entity that requires protection can use for its application specific security. In this case the application is HEz ranging.

It is better from a security perspective, possibly a requirement, to provide a HLTK to higher layer rather than providing the higher layer with the PMK to use for its key derivation.

It is reasonable to derive HLTK as part of PTK derivation. PTK derivation is different for different AKMs; the AKMs also dictate the hash functions used. Minor extension to Pairwise, FT, and FILS key hierarchies would be needed.

Key material whether derived or generated using a RNG can be precomputed if necessary. Deriving keys would have additional advantages

* The key material for different STAs (on an AP) would be decoupled. Better from a security perspective
* Less data to exchange during negotiation
* Simple devices may not have an strong RNG – they simply use a KDF to derive.
* Even if a HW RNG is present, recommend mechanisms (NIST SP 800 90C) still need a derivation so as to not leak RNG state.

**Discussion – HLTK and key material lifetime**

Since HLTK derived as part of PTKSA, it is destroyed along with the PTKSA. The spec does not seem to have any normative text for PTKSA lifetime except that it would be deleted when PMKSA expires. PASN PTKSA may have a negotiated lifetime.

**Discussion – HLTK key derivation capability**

This support is optional. HLTK is derived if higher layer requires it. It can possibly be tied to any MIB support related to higher layer (feature) – e.g. secure LTF. No additional capability bits would be needed.

**Discussion – How is HLTK used by HEz and VHTz**

Before any secure FTM negotiation, HEz/VHTz derives a LTF protection key from HLTK using key derivation – Secure LTF-Key-Seed = HMAC-Hash(HLTK, “Secure LTF key seed”)). A monotonically increasing 48-bit counter Secure-LTF-Counter that is initialized to 0.

At the start of each secure FTM negotiation and at each measurement

* Increment Secure LTF counter by 1
* SAC is generated using the counter along with secure bits. SAC is sent in the protocol.

The secure (pseudo-random) bits to protect LTF(s) between peers is derived from Secure-LTF-Key-Seed as follows

The responder derives the SAC and the LTF protection bits using

 SAC || Secure-LTF-bits = KDF-Hash-Length(Secure LTF-Key-Seed, “Secure LTF Expansion” || Secure-LTF-Counter)

where Length is the number of bits required to protect \*all\* of the LTFs in a given measurement frame from the responder.

The initiator derives the LTF protection bits using

 Secure-LTF-bits = KDF-Hash-Length(Secure LTF-Key-Seed, “Secure LTF Expansion” || SAC || Secure-LTF-Counter)

The above construction binds SAC to initiator generated bits, and allows the initiator to and responder to choose the number of bits required easily (than using a single construction and splitting the bits).

This allows the HLTK derived from PTKSA negotiation to be reused for multiple HEz/VHTz negotiations. Note that the Secure-LTF-Counter is not reset when a ranging session is torn town. It is reset only when a new PTK for a PTKSA is derived.

**Discussion – How many bits are needed and how are they taken from the security data**

Generate the bits needed using a KDF.

**Discussion – based on some feedback**

* Does SAC need to be transmitted in LMR in addition to the counter?
	+ Per the current proposal, SAC can be derived on both sides along with the bits used to protect LTFs. It would be useful for the receiver of LMR to cross check SAC – e.g. in case there is a key mismatch, for example.
	+ Propose to leave SAC in LMR for now. Further consideration for removing later.
* How many secure bits
	+ As many as needed by the LTFs
* Need to secure VHTz also
	+ azD0.4 describes Secure LTF Parameters etc. as applicable to both VHTz and HEz. The common description applies to derived SAC and secure LTF input bits.
	+ In section 11.22.6.3.1 (11.22.6.3.2 in 11az D0.4), the key derivation uses ‘Secure LTF key seed’ etc. to denote this applies to both.
* Does the ranging session need an ID that is bound to the key seed, in addition to the counter? It might be good to do this if we had a session ID, but is not necessary. That would provide control of the lifetime of the session key material (not valid once the session completes)

**--**

**Editor: Add the definition for HLTK to definition, acronyms, and abbreviations(3, p158.59)**

**hidden station (STA):** A STA whose transmissions are not detected using carrier sense (CS) by a second

STA, but whose transmissions interfere with transmissions from the second STA to a third STA

**higher layer transient key (HLTK):** Input key material derived as a seed to be used for higher layer protection.

**Editor: change PTKSA contents as follows (12.6.1.1.6, p2509.7)**

**12.6.1.1.6 PTKSA**

…

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

— HLTK, if higher layer security is supported

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

**Editor: Replace** Figure 12-30—Pairwise key hierarchy **with the figure below (12.7.1.3 p2533)**



**Editor: Change PTK derivation description as follows (12.7.1.3 p2534.30)**

 — 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)

 — The Higher Layer Transient Key (HLTK) shall be computed as the next HLTK\_bits of the PTK:

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

 Where HLTK\_bits has the value 256 if higher layer security is supported and 0 otherwise.

**Editor: Change the FT PTK derivation as follows (12.7.1.6.5 p2541.13)**

(#102) The KEK2 is used to provide data confidentiality for certain fields in the FT authentication sequence,

as defined in 13.8 (FT authentication sequence).

In addition, an HLTK may be derived to be provided as a key seed for protecting higher layer exchanges, such as secure timing measurement. HTLK shall be derived as follows:

HLTK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits+KCK2\_bits+KEK2\_bits, HLTK\_bits)

Where HLTK\_bits has the value 256 if higher layer security is supported and 0 otherwise.

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

**Editor: Change FILS PTKSA derivation as follows (12.12.2.5, p2607.1)**

**12.12.2.5.3 PTKSA Key derivation with FILS authentication**

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(#1389) 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(#1389) 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(#1389) is 00-0F-AC:16, FILS-FT is 256 bits; when the negotiated AKM(#1389) is 00-0F-AC:17,

FILS-FT is 384 bits; otherwise, FILS-FT is not derived. If HLTK is desired, additional 256 bits (HLTK\_bits) of HLTK shall be derived, otherwise HLTK\_bits are not derived. The total amount of bits extracted from the KDF

shall therefore be 512+TK bits+HLTK\_bits, 896+TK bits+HLTK\_bits, or 1280+TK bits+HLTK\_bits depending on the negotiated AKM(#1389),

where TK\_bits are determined from Table 12-4:

(#114)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)

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

Where HLTK\_bits has the value 256 if higher layer security is supported and 0 otherwise.

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

**TGaz Editor: Add the following at the end of Subclause 11.22.6.3.1 Secure LTF measurement setup (11.22.6.3.1, p53.4)**

When dot11SecureLTFImplemented is true, prior to generating new LTF Sequence generation information for a given PTKSA, the RSTA initializes a monotonically increasing 48-bit counter Secure-LTF-Counter to 0. The RSTA also derives a Secure-LTF-Key-Seed as follows

 Secure-LTF-Key-Seed = HMAC-Hash(HLTK, “Secure LTF key seed”)

Where HLTK is derived as part of PTKSA establishment, Hash is the hash determined by the AKM and used to derive the PTK.

Similarly, when dot11SecureLTFImplemented is true, an ISTA also derives the same Secure-LTF-Key-Seed prior to initiating a secure FTM negotiation.

For each secure FTM negotiation, prior to transmission of iFTM, and for each measurement within an FTM session, an RSTA shall increment the Secure-LTF-Counter by 1. The Secure-LTF-Counter is included as part of LTF sequence generation information conveyed to the ISTA. An ISTA determines the Secure-LTF-Counter to use for a measurement using the sequence generation information and SAC received from the RSTA in a protected iFTM or LMR frame. The Secure-LTF-Counter shall be reset when a new HLTK is derived as part of a new PTK derivation and it shall continue, and not be reset, for each secure FTM negotiation using a given HLTK.

For a given secure measurement frame (e.g. NDP), the SAC and secret (pseudo-random) bits to protect all of the LTFs in the frame from the RSTA are derived as follows

 SAC || Secure-LTF-bits = KDF-Hash-Length(Secure-LTF-Key-Seed, “Secure LTF Expansion”, Secure-LTF-Counter)

Similarly, for a given secure measurement frame (e.g. NDP), the secret (pseudo-random) bits to protect all of the LTFs in the frame from the ISTA for a given SAC are derived as follows

Secure-LTF-bits = KDF-Hash-Length(Secure-LTF-Key-Seed, “Secure LTF Expansion”, SAC || Secure-LTF-Counter)

Where KDF and Hash are the key derivation function and hash function determined by the AKM used to derive the PTKSA, and Length is the length in bits required for the SAC concatenated with the LTF sequence generation input.

Integer to octet string conversion (MSB first) specified in 12.4.7.2.2 shall be used to encode the Secure-LTF-Counter input to the KDF as well as in the transmitted LTF sequence information. It shall be padded with leading (MSB) 0s to be exactly 6 octets. The SAC transmitted and used in deriving Secure-LTF-bits shall also be of exactly 2 octets in length.

Note: A 6 octet sequence counter is sufficient because a unicast protected management frame that uses a 6 octet packet number is used to convey the LTF sequence information that carries the counter.

With the preceding construction, an attacker not knowing Secure-LTF-Key-Seed, would not be able to predict the SAC that would be used for given measurement.

Secure LTF measurement may require a variable number of input bits, say M, based on parameters negotiated for a ranging measurement. A maximal number Secure-LTF-bits are derived and the first M bits of Secure-LTF-bits are used for the measurement. When multiple components of a given measurement frame use Secure-LTF-bits, the bits are assigned left to right to various components in the order in which the components are ordered in the frame.

Note that in a DL NDP frame used for range measurement, LTFs assigned to each of the recipient STAs would use key material derived from their corresponding Secure-LTF-bits.

Secure-LTF-Counter is maintained for the lifetime of a PTKSA used for Secure LTF measurements. It shall not be reset between measurements and shall not be reset for multiple FTM negotiations using the same PTKSA. The Secure-LTF-Counter value used for each measurement using the HLTK derived for a given PTKSA shall be unique.

**TGaz Editor: Change the following in PTKSA derivation for PASN (12.13.6, p80.24)**

**12.13.6 PTKSA derivation with PASN authentication**

For PTKSA key derivation, the inputs to the PRF are the PMK of the PMKSA, a constant label and a concatenation of non-AP STA’s MAC address, AP’s BSSID and the DH shared secret from the ephemeral exchange.

KCK || TK || HLTK = KDF-HASH-NNN (PMK, “PASN PTK Derivation”, SPA || BSSID || DHss)

where

KCK is the key confirmation key of length 32 octets.

TK is the transient key whose length is the same as a key for the pairwise cipher in RSNE provided by the AP in the second PASN frame. This length is 16 octets for all ciphers, except for the ciphers 00-0F-AC:9 and 00-0F-AC:10 for which it is 32 octets.

Where HLTK is of length HLTK\_bits which has the value 256 if higher layer security is supported and 0 otherwise.

KDF-HASH-NNN is the key derivation function defined in 12.7.1.6.2 (Key derivation function (KDF)) using the hash algorithm defined for the Base AKM (see Table 9-144 (AKM suite selectors)). When there is no Base AKM, the hash algorithm is selected based on the pairwise Cipher Suite provided in the RSNE provided by the AP in the second PASN frame. SHA-256 is used as the hash algorithm, except for the ciphers 00-0F-AC:9 and 00-0F-AC:10 for which SHA-44 384 is used.

DHss is the shared secret derived from the PASN ephemeral key exchange encoded as an octet string (12.4.7.2.2 (Integer to octet string conversion))

and NNN is the bits required for KCK and TK which is either 384 + HLTK\_bits or 512 + HLTK\_bits depending on the pairwise cipher.

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

…

**TGaz Editor: Change the TBDs for LTF Sequence generation information and SAC in Figure 9-610c D0.4 as follows**



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ElementID | Length | Element ID Extension | LTF Sequence Generation Information | LTF Generation SAC | Measurement ResultSAC |

Octets 1 1 1 6 2 2

 Figure 9-610c Secure LTF Parameters element format

…

**TGaz Editor: Remove Figure 9-610d LTF Sequence Generation Information Field Format – the field is now of fixed length)**