IEEE P802.11
Wireless LANs

|  |
| --- |
| Comment Resolutions for D6.0 WUR Frame Protection CIDs |
| Date: 2020-03-24 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Rojan Chitrakar | Panasonic |  |  | Rojan.chitrakar@sg.panasonic.com |
| Lei Huang |  |  |  |
| Yoshio Urabe |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Abstract

This submission proposes resolutions of comments received from TGba SA1 comment collection (TGba Draft 6.0).

* CIDs: 7058, ~~7060~~, 7061, ~~7062~~, ~~7063~~, 7064, ~~7088~~, 7115 (8 CIDs)

Revisions:

* Rev 0: Initial version of the document.
* Rev 1: Improved the language of the resolution text for CID 7060. No changes to the spec text itself.
* Rev 2: CIDs 7060, 7062, 7063 & 7088 are deferred for futher discussion. Resolution for CID 7058 was changed from Accepted to Revised as a result of discussion during the call. CIDs 7058, 7061, 7064, 7115 are ready for motion.
1. **Introduction**

Interpretation of a Motion to Adopt

A motion to approve this submission means that the editing instructions and any changed or added material are actioned in the TGba Draft. The introduction and the explanation of the proposed changes are not part of the adopted material.

***Editing instructions formatted like this are intended to be copied into the TGba Draft (i.e. they are instructions to the 802.11 editor on how to merge the text with the baseline documents).***

***TGba Editor: Editing instructions preceded by “TGba Editor” are instructions to the TGba editor to modify existing material in the TGba draft. As a result of adopting the changes, the TGba editor will execute the instructions rather than copy them to the TGba Draft.***

***Part I***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CID | Commenter | Clause  | Page | Line | Comment | Proposed Change | Resolution |
| 7058 | Montemurro, Michael | 12.2.4 | 89 | 19 | The text at the cited location doesn't make sense. "This standard defines one integrity protocol for Management frames and for WUR Wake-up frames: BIP." | I assume the sentence is trying to say that there is one integriy protocol for both Management frames and WUR Wake-up frames. If so,Change"This standard defines one integrity protocol for Management frames and for WUR Wake-up frames: BIP."to"This standard defines the following integrity protocol to protect Management frames and for WUR Wake-up frames: BIP." | **Revised.**TGba editor to change the cited sentence to the following:"This standard defines the following integrity protocol for Management frames and for WUR Wake-up frames: BIP." |
| 7060 | Montemurro, Michael | 29.10 | 123 | 24 | If a WUR STA and a WUR AP negotiate a WIGTKSA, a WUR AP doesn't have an option to not transmit unprotected WUR Wake-up frames. | Change "A WUR AP may transmit" to "A WUR AP shall transmit" | **Revised.**Agree in principle with the comment. However, since AP cannot be mandated to transmit, the cited sentence is reworded to convey that protection is mandatory. TGba editor to make the changes shown in 11-20/0601r1 under all headings that include CID 7060. |
| 7061 | Montemurro, Michael | 29.10 | 123 | 38 | Either all STA associated to the WUR AP use netgotiated protection mechansms after negotiation or not. You cannot have a mixture of STAs. First of all, this sounds more like a client than a note. | Update the not to require all STA to support WUR security, or none do. | **Rejected.**There is no requirement in 802.11ba that mandates that all WUR non-AP STAs associated with a WUR AP shall have the same protection negotiation. It is possible to have a mixture of STAs and thus the NOTE. |
| 7062 | Montemurro, Michael | 29.10 | 123 | 24 | If a WUR STA and a WUR AP negotiate a WIGTKSA, a WUR AP doesn't have an option to not transmit unprotected WUR Wake-up frames. | Change "the WUR AP may transmit" to "the WUR AP shall transmit. | **Revised.**This is a duplicate of CID 7060 and has been resolved by resolution to CID 7060.No action required from the TGba editor. |
| 7063 | Montemurro, Michael | 29.0 | 123 | 47 | A WUR AP cannot determine when WUR STA installs the WTK. | Change"The WUR AP shall not transmit a protected individually addressed WUR Wake-up frame to the WUR non-AP STA until the WTK is installed at the WUR non-AP STA"to"The WUR AP shall not transmit a protected individually addressed WUR Wake-up frame to the WUR non-AP STA until 4-way handshake sucessfully compltetes. | **Revised.**Agree in principle with the comment. However, since the sentence also mentions WIGTK, distribution of key is refered instead of actual key handshake. In addition, the sentence at the end of this paragraph relating to WTK is more suitable for another paragraph that describes non-AP behaviour and has been moved there, while the part about WIGTK has been deleted since it is already covered in that paragraph.TGba editor to make the changes shown in 11-20/0601r1 under all headings that include CID 7063. |
| 7064 | Montemurro, Michael | 12.6.1.1.6 | 90 | 52 | The PTKSA includes a WTK. Howeever there is no requirement on which negotiated cipher suite is used for data confidentiality. BIP-CMAC-128 only provides data integrity and replay detection. | Add requirements to describe how a STA and an AP would communicate to preserve data confidentiality. The negotiated cipher suite could be the one used for unicast transmissions in the BSS. From 12.8.10, it sounds like the cipher is AES-128-CMAC (should be CCMP-128) but its unclear. | **Rejected.**WUR frame protection is only used for integrity protection and hence uses BIP. WTK is used for integrity protection of individually addressed WUR Wake-up frames, while WIGTK is used for integrity protection of broadcast and group addressed WUR Wake-up frames. WUR frame protection does not provide data confidentiality. |
| 7115 | Hamilton, Mark | 29.10 | 123 | 24 | State it clearly that WUR Short Wake-up frames are not protected. | Insert a paragraph, as a new fourth paragraph of 29.10, "WUR Short Wake-up frames are not protected." | **Accepted**. |

**Discussion:** None

**Propose:**

Revised for CIDs 7060, 7063 as per discussion and editing instructions in 11-20/0601r1.

* WUR frame protection (CIDs 7060, 7063 )

***TGba editor: Modify the section as the following (Track Changes ON):***

***…***

If WUR frame protection is negotiated between a WUR AP and a WUR non-AP STA, the WUR AP shall protect all individually addressed WUR Wake-up frames and, if the WIGTK has been distributed to the WUR non-AP STA, all broadcast or group addressed WUR Wake-up frames intended for the WUR non-AP STA; otherwise the AP shall not transmit a protected WUR Wake-up frame to the WUR non-AP STA. If a WUR non-AP STA negotiated WUR frame protection for an association, it shall discard any unprotected WUR Wake-up frames received from the WUR AP associated. If the non-AP STA did not negotiate WUR frame protection for an association, it shall discard any protected WUR Wake-up frames received from the WUR AP. (#7060)

NOTE – If a WUR AP transmits unprotected broadcast or group addressed WUR Wake-up frame where a WUR non-AP STA that has negotiated WUR frame protection with the WUR AP is addressed by the Wake-up frame, then the transmitted uprotected broadcast or group address WUR Wake-up frame is not intended for the WUR non-AP STA.

…

The WUR AP shall not transmit a protected individually addressed WUR Wake-up frame to the WUR non-AP STA until the 4-way handshake has been successfully completed between the WUR AP and the WUR non-AP STA (see 12.7.6 (4-way handshake)), and should not transmit a protected broadcast or group addressed WUR Wake-up frame until the WIGTK has been distributed to all WUR non-AP STAs with which the WUR AP has negotiated WUR frame protection (see 12.7.6 (4-way handshake) or 12.7.7 (Group key handshake)). (#7063)

…

A WUR non-AP STA that installs WTK (see 12.7.6 (4-way handshake)) shall use the WTK to process all subsequently received protected individually addressed WUR Wake-up frames. The WUR non-AP STA shall discard any protected WUR frames received before the WTK is installed. A WUR non-AP STA shall identify the appropriate WIGTK and associated state based on the Key ID subfield of the received protected broadcast or group addressed FL WUR Wake-up frames. If no such WIGTK exists, the WUR non-AP STA shall silently drop the frame and terminate BIP processing for this reception. A WUR non-AP STA shall use the latest installed WIGTK to process all subsequently received protected group addressed VL WUR Wake-up frame. (#7063)

***Part II***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CID | Commenter | Clause  | Page | Line | Comment | Proposed Change | Resolution |
| 7088 | Malinen, Jouni | 12.7.1.3 | 93 | 65 | PTK extensions to derive multiple new keys are not very scalable and result in undesired complexity in the pairwise key hierarchy by having multiple conditional components of the PTK being derived. P802.11ba is doing this with the new WTK and P802.11az has a similar need, but with a significantly more generic and scalable design. There is also a similar need in REVmd discussions for generating PMKID values in a manner that protects privacy (which is targeting a design that is compatible with P802.11az need). It would be good for P802.11ba to use a common, extensible mechanism for deriving new keys. The clearest candidate for doing this seems to be the use of a new key derivation key (e.g., HLTK as defined inP802.11az) that is derived as part of PTK as hopefully the last needed extension of PTK while all the new keys could then be derived from this key derivation key using HMAC-Hash() (like P802.11az Secure-LTF-Key-Seed) or HKDF, etc., key derivation function that takes in the new key derivation key and a unique label for each new need. | Replace the WTK key derivation design in P802.11ba with a more generic design in the style of HLTK in P802.11az. This might come in as a part of the REVmd as a baseline for P802.11ba or as an addition in P802.11ba and P802.11az using that same design. | **Revised.**Agree in principle with the comment. Instead of deriving the WTK from the PTK, a Key Derivation Key (KDK) is derived from the PTK. The KDK is then used to derive the WTK using a Hash function.TGba editor to make the changes shown in 11-20/0601r1 under all headings that include CID 7088. |

**Discussion:** Commenter points out that both 11ba and 11az are extending the PTK to derive additional keys, which may not be extensible in the future due to PTK length limitations.

**11ba:**



**11az:**



Suggested solution is to extend the PTK for a common “Key Derivation Key” (KDK) which can then be used by 11ba and 11az and future amendments to derive any additional key for their usage.



Propose to use KDF function as defined in 12.7.1.6.2 (Key derivation function (KDF)) to derive the WTK from KDK:

WTK = KDF-Hash-128 (KDK, “WUR Temporal Key”, context)

**Propose:**

Revised for CIDs 7088 as per discussion and editing instructions in 11-20/0601r1.

3.4 Abbreviations and acronyms

***TGba editor: Insert the following Acryonym (Track Changes ON):***

KDK key derivation key

4.10.3.2 AKM operation with AS

***TGba editor: Modify the cited paragraph as the following (Track Changes ON):***

***…***

—Derive a fresh pairwise transient key (PTK) from the PMK or, in the case of fast BSS transition, from the PMK-R1, the derived PTK including the Key derivation key (KDK) if WUR frame pro­tection is negotiated.

9.4.2.24.2 Cipher suites

***TGba editor: Modify Table 9-150 as the following (Track Changes ON):***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cipher suite selector | GTK | PTK | IGTK or BIGTK |  | WIGTK |
| Use group cipher suite | No | Yes | No |  | No |
| WEP-40 | Yes | No | No |  | No |
| WEP-104 | Yes | No | No |  | No |
| TKIP | Yes | Yes | No |  | No |
| CCMP-128 | Yes | Yes | No |  | No |
| BIP-CMAC-128 | No | No | Yes |  | Yes |
| GCMP-128 | Yes | Yes | No |  | No |
| GCMP-256 | Yes | Yes | No |  | No |
| CCMP-256 | Yes | Yes | No |  | No |
| BIP-GMAC-128 | No | No | Yes |  | No |
| BIP-GMAC-256 | No | No | Yes |  | No |
| BIP-CMAC-256 | No | No | Yes |  | No |

12.6.1.1.6 PTKSA

***TGba editor: Modify the cited paragraph as the following (Track Changes ON):***

***…***

—PTK, where the PTK includes the KDK when WUR frame protection is negotiated

12.7.1.1 General

***TGba editor: Modify the cited paragraph as the following (Track Changes ON):***

***…***

RSNA defines the following key hierarchies:

a) Pairwise key hierarchy, to protect individually addressed traffic, where the PTK includes a KDK if WUR frame protection is negotiated, and excludes the KDK otherwise

12.7.1.3 Pairwise key hierarchy

***TGba editor: Modify the cited paragraph as the following (Track Changes ON):***

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, and a temporal key, which is used by the MAC to protect individually addressed communication between the Authenticator's and Supplicant's respective STAs, and if WUR frame protection is negotiated, a KDK which is used to derive a WTK. The WTK 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.

***…***

***TGba editor: Modify the WUR Temporal Key box of Figure 12-30 as the following (Track Changes ON):***

Key Derivation Key

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

(KDK)

***…***

***TGba editor: Modify the cited paragraphs as the following (Track Changes ON):***

The following apply:

—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 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). The value of KDK\_bits is equal to the value of PMK\_bits. 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).

—If WUR frame protection is negotiated, the KDK shall be computed as the next KDK\_bits bits of the PTK:

KDK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits, KDK\_bits)Otherwise, the KDK is not derived.

***…***

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

If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF defined in 12.7.1.6.2 (Key derivation function (KDF)):

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, Min(AA,SPA) || Max(AA,SPA) || Min(ANonce,SNonce) || Max(ANonce,SNonce))

where

— KDF-Hash-Length is the key derivation function as defined in 12.7.1.6.2 (Key derivation function (KDF)) using the hash algorithm identified by the AKM suite selector (see Table 9-151(AKM suite selectors)).

— Length is the total number of bits to derive, i.e., number of bits of the WTK and is equal to 128.

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, and if WUR frame protection is negotiated, the WTK into the STA. The STA uses the temporal key with the pairwise cipher suite; interpretation of this value is cipher-suite dependent(#1408).

***…***

**12.7.1.6 FT key hierarchy**

**12.7.1.6.5 PTK**

***TGba editor: Modify the cited paragraphs as the following (Track Changes ON):***

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

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

The KDK shall be computed as the next KDK\_bits of the PTK:

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

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

If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF defined in 12.7.1.6.2 (Key derivation function (KDF)):

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, SNonce || ANonce || BSSID || STA-ADDR)

where

— KDF-Hash-Length is the key derivation function as defined in 12.7.1.6.2 (Key derivation function (KDF)) using the hash algorithm identified by the AKM suite selector (see Table 9-151(AKM suite selectors)).

— Length is the total number of bits to derive, i.e., number of bits of the WTK and is equal to 128.

The WTK is used to protect individually addressed WUR Wake-up frames, as defined in 29.10 (WUR frame protection).

For vendor-specific cipher suites, the length of the temporal key (and the value of Length) depend on the vendor-specific algorithm.

The temporal key, and if WUR frame protection is negotiated, the WTK is configured into the STA by the SME through the use of the MLME-SETKEYS.request primitive. The STA uses the temporal key with the pairwise cipher suite; interpretation of this value is specific to the cipher suite.

**12.12.2.5.3 PTKSA Key derivation with FILS authentication**

***TGba editor: Modify the cited paragraphs as the following (Track Changes ON):***

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 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); 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 protected 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:

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 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)

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

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

where

ICK\_bits is the length of ICK in bits

KEK\_bits is the length of KEK in bits

FILS-FT\_bits is 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).

X is 640+TK bits, 1124+TK bits, or 1408+TK bits from Table 12-5 (Cipher suite key lengths) depending on the negotiated AKM when WUR frame protection is negotiated; otherwise, 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

If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF defined in 12.7.1.6.2 (Key derivation function (KDF)):

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, SPA || AA || SNonce || ANonce [ ||DHss ])

where

— KDF-Hash-Length is the key derivation function as defined in 12.7.1.6.2 (Key derivation function (KDF)) using the hash algorithm identified by the AKM suite selector (see Table 9-151(AKM suite selectors)).

— Length is the total number of bits to derive, i.e., number of bits of the WTK and is equal to 128.