**IEEE P802.15**

**Wireless Personal Area Networks**

|  |  |
| --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) |
| Title | **Text proposal for 15.4ab Secure compressed PSDU**  |
| Date Submitted | September 2023 |
| Sources | Rojan Chitrakar, Lei Huang, Bin Qian, David Xun Yang (Huawei) |  |
| Re: | Contribution to IEEE 802.15.4ab  |
| Abstract |  |
| Purpose | This submission proposes text to for the IEEE Std 802.15.4ab specification framework document.  |
| Notice | This document does not represent the agreed views of the IEEE 802.15 Working Group or IEEE 802.15.4ab Task Group. It represents only the views of the participants listed in the “Sources” field above. It is offered as a basis for discussion and is not binding on the contributing individuals. The material in this document is subject to change in form and content after further study. The contributors reserve the right to add, amend or withdraw material contained herein. |

Baseline for the below text is: IEEE 802.15.4-2020

**8.4.3 MAC PIB attributes**

**8.4.3.1 Overview**

…

*TG4ab editor, add the following entries to the end of Table 8-94 as follows (track changes ON):*

**Table 8-94—MAC PIB attributes**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Type** | **Range** | **Description** | **Default** |
| … |  |  |  |  |
| *macCPSDUSecurityEnabled* | Boolean | TRUE, FALSE | Indication of whether the security of compressed PSDU is enabled. A value of TRUE indicates that security is enabled, while a value of FALSE indicates that security is disabled. | FALSE |

9. Security

9.1 Overview

9.2 Functional description

*TG4ab editor, modify the subclause 9.2.2 as follows (track changes ON):*

9.2.2 Outgoing frame security procedure for frames that are not compressed PSDUs

This procedure shall only be used for outgoing frames that are not compressed PSDUs. For compressed PSDUs, the procedure in 9.2.12 is used instead.

The inputs to this procedure are the frame to be secured and the SecurityLevel, KeyIdMode, KeySource, and KeyIndex parameters. MLME-START.request primitive gives SecurityLevel, KeyIdMode, KeySource and KeyIndex for Beacon frames and Coordinator Realignment commands. If the frame was generated in response to an MLME or MCPS primitive, then the value of SecurityLevel, KeyIdMode, KeySource, and KeyIndex are set to the corresponding values of the primitive parameters. Otherwise, the inputs are as follows:

…

*TG4ab editor, add the following entries to the end of Table 9-1 as follows (track changes ON):*

**Table 9-1—Exceptions to Private Payload field and Open Payload field definitions**

|  |  |  |
| --- | --- | --- |
| **Frame type** | **Private Payload field** | **Open Payload field** |
| Beacon(Frame Version < 2) | Beacon Payload field | All other fields in the MAC Payload field |
|
| MAC Command(Frame Version < 2) | Content field | Command ID field |
| SECURE-REPORT(from initiator) | TurnAroundTime | All other fields in the MessageContent field |
| SECURE-REPORT(from responder) | ReplyTime | All other fields in the MessageContent field |
| SECURE-REPORT(from initiator in one-to-many ranging) | TurnAroundTime | All other fields in the MessageContent field |
| SECURE-REPORT(from responder in one-to-many ranging) | ReplyTime | All other fields in the MessageContent field |

9.2.4 Incoming frame security procedure, Security Enabled field is set to one

*TG4ab editor, modify the subclause as follows:*

This procedure shall only be used for incoming frames that are not compressed PSDUs, and in which the Security Enabled field is set to one. For frames in which the Security Enabled field is set to zero, the procedure in 9.2.5 is used instead. For compressed PSDUs, the procedure in 9.2.13 is used instead.

9.2.5 Incoming frame security procedure, Security Enabled field is set to zero

*TG4ab editor, modify the subclause as follows:*

This procedure shall only be used for incoming frames that are not compressed PSDUs, and in which the Security Enabled field is set to zero. For frames in which the Security Level field is set to one, the procedure in 9.2.4 is used instead. For compressed PSDUs, the procedure in 9.2.13 is used instead.

*TG4ab editor, insert new subclauses 9.2.12 and 9.2.13 after 9.2.11 as follows:*

9.2.12 Outgoing frame security procedure for compressed PSDUs

This procedure shall only be used for outgoing compressed PSDUs. For other frames, the procedure in 9.2.2 is used instead.

The inputs to this procedure are the compressed PSDU to be secured, the SecurityLevel, KeySource, and KeyIndex parameters. The inputs are as follows:

— SecurityLevel shall be set to *secCPSDUSecurityLevel*

— KeySource shall be set to the extended address of the peer device associated with the key

— KeyIndex shall be set to the Key ID field of the compressed PSDU to be secured

The outputs from this procedure are the Status of the procedure and, if this Status is SUCCESS, the secured compressed PSDU.

This procedure involves the following steps:

1. **Is security needed?** If the SecurityLevel parameter is zero, the procedure shall set the secured compressed PSDU to be the compressed PDU to be secured and return with a Status of SUCCESS.
2. **Is security enabled**. If *macCPSDUSecurityEnabled* is set to FALSE, the procedure shall return with a Status of UNSUPPORTED\_SECURITY.
3. **Obtain source address**. Source address shall be set to the *macExtendedAddress*.
4. **Obtain Key.** The procedure shall obtain the matching Key from the *secCPSUKeyList* using the KeySource and KeyIndex. If the Key lookup procedure fails, the procedure shall return with a Status of UNAVAILABLE\_KEY.
5. **Set frame counter.** The frame counter is set as the indices of the slot, round and block in which the compressed PSDU is to be transmitted (see Figure 9-XX2 (Frame Counter field)).
6. **Secure compressed PSDU**. The Private Payload field and Open Payload field shall be set as indicated in the Table 9-1. The procedure shall then use the Private Payload field, the Open Payload field, the source address, the frame counter, and the Key to produce the secured compressed PSDU, according to the transformation process described in 9.3.5.
7. **Finish procedure**. The procedure shall return with a Status of SUCCESS.

**9.2.13 Incoming frame security procedure for compressed PSDUs**

This procedure shall only be used for incoming compressed PSDUs. For other frames, the procedure in 9.2.4 or 9.2.5 is used instead.

The inputs to this procedure are the compressed PSDU to be unsecured and the SecurityLevel. The inputs are as follows:

— SecurityLevel shall be set to *secCPSDUSecurityLevel*

The outputs from this procedure are the status of the procedure and, if the status is SUCCESS the unsecured compressed PSDU, the KeySource, and KeyIndex. The inputs are as follows:

— KeySource shall be set to the extended address of the peer device associated with the key.

— KeyIndex shall be set to the Key ID field of the compressed PSDU to be unsecured

All outputs of this procedure are assumed to be invalid unless and until explicitly set in this procedure.

This procedure involves the following steps:

1. **Check for *macSecurityEnabled***. If *macCPSDUSecurityEnabled* is set to FALSE, the procedure shall return with a Status of UNSUPPORTED\_SECURITY.
2. **Check the SecurityLevel**. If the SecurityLevel is zero, the procedure shall return with a Status of UNSUPPORTED\_SECURITY.
3. **Obtain source address**. Source address shall be set to the extended address of the originator of the compressed PSDU.
4. **Obtain Key.** The procedure shall obtain the matching Key from the *secCPSUKeyList* using the KeySource and KeyIndex. If the Key lookup procedure fails, the procedure shall return with a Status of UNAVAILABLE\_KEY.
5. **Set frame counter.** The frame counter is set as the indices of the slot, round and block in which the compressed PSDU is received (see Figure 9-XX2 (Frame Counter field)).
6. **Unsecure compressed PSDU**. The Private Payload field and Open Payload field shall be set as indicated in the Table 9-1. The procedure shall then use the Private Payload field, the Open Payload field, the source address, the frame counter, and the Key to produce the unsecured compressed PSDU, according to the inverse transformation process described in the security operations, as described in 9.3.6. If the inverse transformation process fails, the procedure shall return with a Status of SECURITY\_ERROR.
7. **Return unsecured compressed PSDU**. The procedure shall return with the unsecured compressed PSDU, KeySource, KeyIndex and a Status of SUCCESS.

9.3 Security Operations

9.3.3 AEAD Nonce

*TG4ab editor, insert the following subclause after 9.3.3.3 (AEAD nonce for Fragment frames):*

9.3.3.4 AEAD Nonce for compressed PSDU

The nonce for compressed PSDU shall be formatted as shown in Figure 9-XX1 (Nonce for compressed PSDU), with the leftmost field in the figure defining the first octets and the rightmost field defining the last octet of the nonce.

|  |  |
| --- | --- |
| Octets: 8 | 5 |
| Source Address | Frame Counter  |

**Figure 9-XX1—Nonce for compressed PSDU**

The Source Address field shall be set to the extended address of the device originating the compressed PSDU.

The Frame Counter field is formatted as illustrated in Figure 9-XX2 (Frame Counter field) and the Slot Index field, the Round Index field and the Block Index field are set as the indices of the slot, round and block in which a compressed PSDU is transmitted or received respectively.

|  |  |  |
| --- | --- | --- |
| Bits: 0-7 | 8-23 | 24-39 |
| Slot Index | Round Index | Block Index |

**Figure 9-XX2—Frame Counter field**

NOTE - To ensure the uniqueness of the nonce, the Key used to secure compressed PSDUs needs to be updated every time the block structure is (re)setup. The same Key cannot be used across multiple block structures.

9.3.4 AEAD prerequisites

*TG4ab editor, modify the subclause as follows (track changes ON):*

Securing a frame involves the use of the AEAD encryption and authentication transformation, as described in B.4.2. Unsecuring a frame involves the use of the AEAD decryption and authentication checking transformation, as described in B.4.3.

The length M of the Authentication field for the AEAD forward transformation and the AEAD inverse transformation is determined from Table 9-6, using *secCPSDUSecurityLevel* for compressed PSDUs and using the Security Level field of the Security Control field of the auxiliary security header of the frame for frames other than compressed PSDUs.

**9.3.5 AEAD transformation data representation**

**9.3.5.3 *a* data and *m* data**

*TG4ab editor, modify the subclause as follows (track changes ON):*

In the AEAD transformation process, the data fields shall be applied as in Table 9-3.

NOTE—The MHR contains the Auxiliary Security Header field, as defined in 7.2 for frames other than compressed PSDUs. For compressed PSDUs, the MHR is composed of the ID field, the RPA\_hash field, the RPA\_prand field if present and the MessageControl field of the compressed PSDU.

**9.3.6 AEAD inverse transformation data representation**

**9.3.6.3 c data and a data**

*TG4ab editor, modify the subclause as follows (track changes ON):*

In the AEAD inverse transformation process, the data fields shall be applied as in Table 9-5.

NOTE—The MHR contains the Auxiliary Security Header field, as defined in 7.2 for frames other than compressed PSDUs. For compressed PSDUs, the MHR is composed of the ID field, the RPA\_hash field, the RPA\_prand field if present and the MessageControl field of the compressed PSDU.

…

**9.5 Security-related MAC PIB attributes**

…

*TG4ab editor, add the following entries to the end of Table 9-8 as follows (track changes ON):*

**Table 9-8—Security-related MAC PIB attributes**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Type** | **Range** | **Description** | **Default** |
| … |  |  |  |  |
| *secCPSDUSecurityLevel* | Integer | As defined in Table 9-6 | The security level negotiated for compressed PSDUs | 0x06 |
| *secCPSUKeyList* | List of secCPSDUKeyDescriptors, as defined in Table 9-17 | - | One or more secCPSDUKeyDescriptors containing keys for compressed PSDUs  | (empty) |

*TG4ab editor, add the following Table after Table 9-16 as follows (track changes ON):*

**Table 9-17—Elements of secCPSDUKeyDescriptors**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Range** | **Description** |
| *secCPSDUKeySource* | 8 Octets | An extended IEEE address | The extended address of the device associated with the *secCPSDUKey* |
| *secCPSDUKeyID* | Integer | 0 or 1 | The Key ID associaited with the *secCPSDUKey* |
| *secCPSDUKey* | 16 octets | - | The value of the key used for compressed PSDU |

**Baseline for the below text is:** **23/412r0 (Consensus NBA-UWB MMS MAC TFD update proposal), 23/401r1 (Consensus One-to-Many Ranging using NBA-MMS TFD) and 23/xxxr0 (XXX-SPN messages)**

* + - 1. Compressed PSDU messages

*TG4ab editor, modify the compressed PSDU messages as follows (track changes ON):*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** | **Message Name** | **Octet 0 (Msg ID)** | **Octets 1-N [Len]** | **Description** |
| Control | POLL | 0x04 | [RPA\_hash[3], RPA\_prand[3], MessageControl[1],MessageContent[],CRC16] | A qualifying poll message. MessageControl=0x00:MessageContent={0x00, 0x00}MessageControl=0x10:MessageContent={Request Bitmap[1],Presence Bitmap[1],If Bit 0 of Presence Bitmap == 1 then {NB Channel Select[2]},If Bit 1 of Presence Bitmap == 1 then {NB PHY Config[1]},If Bit 2 of Presence Bitmap == 1 then {NB MAC Config[7]},If Bit 3 of Presence Bitmap == 1 then {UWB PHY Config[3]},If Bit 4 of Presence Bitmap == 1 then {UWB MAC Config[2]},If Bit 5 of Presence Bitmap == 1 then {BlockIndex[2], RoundIndex[2]}} |
| POLL (one-to-many) | 0x10 | [RPA\_hash[3], RPA\_prand[3],MessageControl[1],MessageContent[],CRC16] | MessageControl=0x00:MessageContent={0x00, 0x00} This is the POLL message for ranging sub-rounds that are not the first one.MessageControl=0x10: MessageContent={Numberof Responders[1], SlotsPerResponder[1], List of Responder Address[3]}MessageControl = 0x20: MessageContent={Number of Responders[1], List of {Responder Address[3], StartSlotIndex[2], EndSlotIndex[2]}}MessageControl = 0x30: Same as Message Control = 0x10, but both Initiator and Responder send the measurement reportMessageControl = 0x40: Same as MessageControl = 0x20, but both Initiator and Responder send the measurement reportMessageControl = 0x50: MessageContent={NumberOfSubRounds[1], SizeOfSubRounds[1]}MessageControl = 0x60: Same as MessageControl = 0x50, but the Response frame and Poll frame in NB is switchedMessageControl = 0x70: MessageContent={Number of Responders[1], SlotsPerResponder[1], Request Bitmap[1],Presence Bitmap[1],List of {Responder Address[3],If Bit 0 of Presence Bitmap == 1 then {NB Channel Select[2]},If Bit 1 of Presence Bitmap == 1 then {NB PHY Config[1]},If Bit 2 of Presence Bitmap == 1 then {NB MAC Config[7]},If Bit 3 of Presence Bitmap == 1 then {UWB PHY Config[3]},If Bit 4 of Presence Bitmap == 1 then {UWB MAC Config[2]},If Bit 6 of Presence Bitmap == 1 then {BlockIndex[2], RoundIndex[2]}}MessageControl=0x80:MessageContent={Number of Responders[1],Request Bitmap[1],Presence Bitmap[1],List of {Responder Address[3],If Bit 0 of Presence Bitmap == 1 then {NbaChannelMap[6]},If Bit 1 of Presence Bitmap == 1 then {NB PHY Config[1]},If Bit 2 of Presence Bitmap == 1 then {NB MAC Config[7]},If Bit 3 of Presence Bitmap == 1 then {UWB PHY Config[3]},If Bit 4 of Presence Bitmap == 1 then {UWB MAC Config[2]},If Bit 5 of Presence Bitmap == 1 then {StartSlotIndex[2], EndSlotIndex[2]}},If Bit 6 of Presence Bitmap == 1 then {BlockIndex[2], RoundIndex[2]}}MessageControl = 0x90: MessageContent={Number of Responders[1], List of {Responder Address, StartSlotIndex[2], TimeShiftIndication[1]}MessageControl = 0xa0: Same as Message Control = 0x90, but both Initiator and Responder send the measurement reportMessageControl = others: reserved |
|  |  |  |  |
| Report | SECURE-REPORT(from initiator) | 0x10 | [RPA\_hash[3],MessageControl[1],MessageContent[],MIC[] | Secured version of the REPORT(from initiator)MessageControl=0x00:MessageContent={PTDataLength[1],PTData[PTDataLength],TurnAroundTime[5]}, where PTDataLength and PTData fields are optionally present and represent pass through data to higher layers.Note - The frame content is same as that of the REPORT(from initiator) except that that the MSB of the PTDataLength indicates the Key ID and CRC16 field is replaced with the MIC and the TurnAroundTime field is situated at the end of the MessageContent field and is encrypted if a security level with encryption is negotiated. |
| SECURE-REPORT(from responder) | 0x11 | [RPA\_hash[3],MessageControl[1],MessageContent[],MIC[] | Secured version of the REPORT(from responder).MessageControl=0x00:MessageContent={PTDataLength[1],PTData[PTDataLength], ReplyTime[5]}, where PTDataLength and PTData fields are optionally present and represent pass through data to higher layers.MessageControl=0x01:MessageContent={Presence Bitmap[1],PTDataLength[1],PTData[PTDataLength],If Bit 0 of Presence Bitmap == 1 then {NB Channel Select[2]},If Bit 1 of Presence Bitmap == 1 then {NB PHY Config[1]},If Bit 2 of Presence Bitmap == 1 then {NB MAC Config[7]},If Bit 3 of Presence Bitmap == 1 then {UWB PHY Config[3]},If Bit 4 of Presence Bitmap == 1 then {UWB MAC Config[2]}, ReplyTime[5]}} where PTDataLength and PTData fields are optionally present and represent pass through data to higher layers; and at least one of NbaChannelMap, NB PHY Config, NB MAC Config, UWB PHY Config and UWB MAC Config fields shall be present. Note - The frame content is same as that of the REPORT (from responder) except that that the MSB of the PTDataLength indicates the Key ID and the CRC16 field is replaced with the MIC, the ReplyTime field is situated at the end of the MessageContent field and is encrypted if a security level with encryption is negotiated. |
| SECURE-REPORT (from initiator in one-to-many ranging) | 0x12 | [RPA\_hash[3],MessageControl[1],MessageContent[],MIC[] | Secured version of the report message for one-to-many ranging (from initiator).MessageControl=0x00:MessageContent={PTDataLength[1],PTData[PTDataLength], TurnAroundTime[5]}, where PTDataLength and PTData fields are optionally present and represent pass through data to higher layers.Note - The frame content is same as that of the REPORT (from initiator in one-to-many ranging) except that that the MSB of the PTDataLength indicates the Key ID, the CRC16 field is replaced with the MIC and the TurnAroundTime field is situated at the end of the MessageContent field and is encrypted if a security level with encryption is negotiated. |
| SECURE-REPORT (from responder in one-to-many ranging) | 0x13 | [RPA\_hash[3],MessageControl[1],MessageContent[],MIC[] | Secured version of the report message for one-to-many ranging (from responder).MessageControl=0x00:MessageContent={PTDataLength[1],PTData[PTDataLength], ReplyTime[5]}, where PTDataLength and PTData fields are optionally present and represent pass through data to higher layers.MessageControl=0x10:MessageContent={Presence Bitmap[1],PTDataLength[1],PTData[PTDataLength],If Bit 0 of Presence Bitmap == 1 then {NB Channel Select[2]},If Bit 1 of Presence Bitmap == 1 then {NB PHY Config[1]},If Bit 2 of Presence Bitmap == 1 then {NB MAC Config[7]},If Bit 3 of Presence Bitmap == 1 then {UWB PHY Config[3]},If Bit 4 of Presence Bitmap == 1 then {UWB MAC Config[2]},ReplyTime[5]}, where at least one of NbaChannelMap, NB PHY Config, NB MAC Config, UWB PHY Config and UWB MAC Config fields shall be present.Note - The frame content is same as that of the REPORT (from responder in one-to-many ranging) except that that the MSB of the PTDataLength indicates the Key ID, the CRC16 field is replaced with the MIC and the ReplyTime field is situated at the end of the MessageContent field and is encrypted if a security level with encryption is negotiated. |
|  |  |  |  |  |
|  |  |  |  |  |

* + - 1. Compressed PSDU message fields

*TG4ab editor, add the following to the end of the table (track changes ON):*

|  |  |  |
| --- | --- | --- |
| **Field name** | **Length in bits** | **Description** |
| MIC | 32 or 64 or 128 | Either the MIC-32 or the MIC-64 or the MIC-128 as described in 9.3 (Security Operations). |
| BlockIndex | 16 | Index of the current ranging block |
| RoundIndex | 16 | Index of the current ranging round |
| Key ID | 1 | Key index that allows unique identification of two keys with the same originator. |

References

[1] 15-23-0216-01-04ab-secured-compressed-frames-for-mms-ranging

[2] 23/412r0 (Consensus NBA-UWB MMS MAC TFD update proposal)

[3] 23/401r1 (Consensus One-to-Many Ranging using NBA-MMS TFD)