**IEEE P802.15**

**Wireless Personal Area Networks**

|  |  |  |
| --- | --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) | |
| Title | **LB213 - Proposed Resolutions for Security – Part 1** | |
| Date Submitted | July 2025 | |
| Sources | Rojan Chitrakar, Lei Huang (Huawei)  [rojan.chitrakar@huawei.com](mailto:rojan.chitrakar@huawei.com) |  |
| Re: |  | |
| Abstract |  | |
| Purpose | To propose resolution for “P802.15.4ab™/D01 Draft Standard for Low-Rate Wireless Networks” . | |
| 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. | |

Rev 0: Initial version.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Index#** | **Pg** | **Sub-Clause** | **Ln** | **Comment** | **Proposed Change** | **Disposition** |
| Kivinen, Tero | 64 | 27 | 8.3.4 | 12 | This table duplicate most of the fields of the Table 8-2 SecurityParameterDescriptor, which is already part of the MCPS-DATA.request. | Remove this table, and use security parameters of the standard SecurityParams parameter. | **Reject**  A new table was deliberately added since the parameter valid range are different for Compact frames. |
| Kivinen, Tero | 65 | 28 | 8.3.4 | 1 | The KeySource seems to be described incorrectly. KeySource is normally not the originator of the received frame or to be used transmitting the frame. It is used to find the associated key from the security tables, and it is included in the auxiliary security header, but does not necessarely match the originator of the frame. | If this is trying to be the originator of the frame then use some different name than KeySource. Remove the whole table 2, and CompactSecurityParams, and use standard SecurityParams, which already have SecurityLevel, KeyIndex, and KeySource fields. | **Reject**  The KeySource is used to identify the key to be used for the secured Compact frame and its usage is described in 10.39.11. |
| MAMAN, MICKAEL | 150 | 28 | 8.3.4 | 1 | SecurityLevel 0x00 with no security is missing | change valid range of SecurityLevel to 0x00-0x07 | **Reject**  SecurityLevels 0x00 (None) and 0x04 (Reserved) were intentionally omitted since they are not applicable for Secure Compact frames. |
| VERSO, BILLY | 329 | 28 | 8.3.4 | 1 | Given that SecurityLevel, KeyIndex, KeySource are already present in SecurityParams structure in base standard. It would be more unified approach to reused these for compact frame security. | Integrate the descriptions for compact frame security into the base standard SecurityParams and delete the separate CompactSecurityParams parameter. | **Reject**  A new table was deliberately added since the parameter valid range are different for Compact frames. |
| Kivinen, Tero | 66 | 28 | 8.3.4 | 2 | The KeyIndex has valid range of 0x00-0x07, but I think the Key ID of the Secured Compact Frame is four bit field, thus correct range is 0-15. | Change range to 0x00-0x0f. | **Accept** |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Index#** | **Pg** | **Sub-Clause** | **Ln** | **Comment** | **Proposed Change** | **Disposition** |
| Kivinen, Tero | 69 | 31 | 9.2.1 | 8 | Instead of making special case for securing compact frames it would be better to use standard security processing to protect those frames. | Remove all changes to section 9, and replace the compact frames with alternate format described in the document 15-25-0173, | **Reject**  Standard security processing does not work for Compact frames. |
| VERSO, BILLY | 332 | 31 | 9.2.1 | 8 | Why cannot "macSecurityEnabled" apply to compact frames also? On TX side we have SendFrameType to identify the frame type and on the receive side we have the Compact Frame type and Compact Frame ID of Secured to identify the secured frame. | Remove/don’t amend the first paragraph of 9.2.1, and replace secCompactFrameSecurityEnabled with macSecurityEnabled where it is used | **Reject**  The security procedures for Compact frames are different from those for the legacy frames, and hence the need for a separate PIB for Compact frames. |
| VERSO, BILLY | 333 | 31 | 9.2.1 | 9 | Reading the pre-existing part of this paragraph coming from the base standard, there is a notion of a device that does not implement security…. Do we want 4ab devices without security? If not we should state that the EMDEV shall support security. | Add statement somewhere that the HRP-EMDEV shall support securing of MAC frames, including compact frames as used for MMS UWB control and reporting . | To be discussed. |
| VERSO, BILLY | 334 | 31 | 9.2.1 | 10 | Since we have a Compact Frame ID for the Secured frame (it's #17 in Table 10), is the CompactFrameID parameter in the MCPS-DATA.request set to 17 or to the ID of the unsecured frame type ID (#0 to 16, and 30?)… We need to think about and specify how Compact frames that are secured are presented to the MAC in MCPS-DATA.request and delivered in MCPS-DATA.indication. | Add description in in MCPS-DATA.request and in MCPS-DATA.indication, and in clause 9 as to make this interface specification clear | **Revise** |
| Kivinen, Tero | 70 | 31 | 9.2.12 | 27 | What shall the device do if someone tries to secure those frames in table 24? | Add text saying that this procedure shall return error if the compact frame given in input is of one in table 24. | **Revise** |
| VERSO, BILLY | 335 | 31 | 9.2.12 | 27 | Is there something about the non-eligible frames that makes them unsuitable for securing, if not why disallow it, if there is a technical reason then please state it. (I think it is because the slot index is used to give a frame number). | Allow securing for all compact frame types, or, add technical reason why the ineligible frames are ineligible |  |
| VERSO, BILLY | 336 | 31 | 9.2.12 | 27 | Might be worth considering adding some frame counter mechanism to allow securing of the ineligible frames, to avoid rogue elements using them for some type of attack. | Add optional SEC frame counter field into the Secure frame to allow for securing of frames sent before the ranging slot structure is established. |  |
| PANPAN, LI | 258 | 32 | 9.2.12 | 1 | This sentence is general description of the inputs to secure procedure, so no need to give the exact terminology. Why not change "SecurityLevel, KeySource, KeyIndex" into "security level, key source, key index"? Also change in the following three bullets. | As in comment | **Reject**  The inputs are names of the parameter so need to be in the right format. |
| Kivinen, Tero | 71 | 32 | 9.2.12 | 22 | As there are compact frames that allows setting fields used as a frame counter how does the system make sure it never reuses any of the frame counters? | Instead of using block/round/slot indexes to generate frame counter use the same frame counter that is already used by the normal frames. | **Reject**  The combination of block/round/slot indexes do not repeat and hence ensures the uniqueness of the Nonce. |
| Kivinen, Tero | 72 | 32 | 9.2.12 | 31 | What about the Message ID field? It is not included at all, meaning anybody can see it and change it at will. | Include Message ID in the Private Payload Fields. | **Reject**  The Message ID field is included in the a data and hence authenticated. |
| Kivinen, Tero | 73 | 32 | 9.2.12 | 32 | This step does not do the validations done in the step f) in 9.2.3 i.e., that the secAeadAlgorithm is supported and that the length of the key matches what is required by the algorithm. It also does not give the secAeadAlgoithm from the secCompactFrameKeyDescriptor to the encryption process. | Add the missing validations and give the secAeadAlgorithm support. | **Revise** |
| VERSO, BILLY | 337 | 32 | 9.2.13 | 37 | There is only one compact frame type eligible for desecuring and that is the Secured Compact frame, with Compact Frame ID value of 17. | Change the paragraph to say only the Secured Compact frame, with Compact Frame ID value of 17, is eligible for desecuring. | **Revise** |
| Kivinen, Tero | 74 | 32 | 9.2.13 | 38 | The device has know of knowing what the actual compact frame type inside the secured compact frame is before it has processed the security control field. | Remove sentence starting at line 38. | **Revise** |
| VERSO, BILLY | 338 | 33 | 9.2.13 | 3 | Suggest to merge this into first paragraph of the subclause | The input is the Secured Compact frame, with Compact Frame ID value of 17, to be desceured | **Accept** |
| VERSO, BILLY | 339 | 33 | 9.2.13 | 9 | Probably don't need separate variable for compact frame security, indeed maybe we should mandate that security support is required for EMDEV, i.e. perhaps step (a) is not needed beyond that we have received Compact Frame ID value of 17. | Use macSecurityEnabled or always just process Compact Frame ID value of 17. | To be discussed, together with #333 |
| Kivinen, Tero | 75 | 33 | 9.2.13 | 15 | The term "frame eligible for security" is not exact. | Specify that if secured compact frame id in the security control field is not listed in table 23, then procedure shall return error. | **Revise** |
| Kivinen, Tero | 76 | 33 | 9.2.13 | 18 | The process in the 10.39.11.1.2.3 might end up in the RPA which is marked resolved, but which do not have extended address associated to it. | Add error case where the source address cannot be obtained because the macIrkDescriptorElement structure does not contain macIrkAssociatedEui. | **Revise** |
| Kivinen, Tero | 77 | 33 | 9.2.13 | 24 | How does this step know what values the ranging slot, ranging round and rounding block have? Are they in some pib attribute? | Specify where the unsecuring process can find the information needed to calculate the nonce. | **Reject**  The indices of the ranging slot, ranging round and ranging block in which the Compact frame is received is known similar to how the ASN is known in the TSCH mode. |
| Kivinen, Tero | 78 | 33 | 9.2.13 | 27 | Again the Message ID is missing. | Add Message ID to the Private Payload fields. | **Reject**  Message ID is part of the a data. |
| Kivinen, Tero | 79 | 33 | 9.2.13 | 28 | This does not take in to account the secAeadAlgorithm in the secCompactFrameKeyDescriptor. | Add text for secAeadAlgorthm. | **Revise** |
| Kivinen, Tero | 80 | 33 | 9.2.13 | 34 | Why use separate CompactSecurityParams when there is already SecurityParameterDescriptor that has same fields. | Remove CompactSecurityParams structure and use standard SecurityParameterDescriptor instead. | **Reject**  A new table was deliberately added since the parameter valid range are different for Compact frames. |
| Kivinen, Tero | 81 | 34 | 9.3.2.4 | 8 | Using this construct will have same issues about the replay protection that TSCH has, i.e., if packet is retransmitted then there is no way to detect that retransmitted frame is same packet than before. | Add similar text than what is done for TSCH in the 9.1 when using compact frames. | **Reject**  Retransmission process is not defined for Compact frames. |
| Kivinen, Tero | 82 | 34 | 9.3.2.4 | 10 | The order of Block Index, Round Index and Slot Index seems to assume little endian format, but section 9.3.1.3 specifies that all integers are sent in most-significant-octet-first order. | Change the order to match the most-significant-octet-first order. | **Revise** |
| Kivinen, Tero | 83 | 34 | 9.3.2.4 | 16 | How is the recipient supposed to decrypt the secured one-to-one poll frame containing the block index and round index field if it does not know what block index and round index is used to encrypt the frame. | There is no way of decrypting the frame unless the nonce is known by the recipient device, thus it can't decrypt the one-to-one compact frame to find indexes to calculate the frame counter to generated the AEAD nonce. Remove all changes to section 9, and replace the compact frames with alternate format described in the document 15-25-0173, which uses standard secured frames and is more efficient than compact frames for unsecured frames, and transmits the frame counter in the frames for secured frames thus does not have this problem. | **Reject**  Block based ranging is built on the premise of synchronization of the ranging block structure between the participating devices. |
| VERSO, BILLY | 340 | 36 | 9.5.11 | 6 | secCompactFrameKeyID has range 0-255 in Table 3, but KeyIndex field in Table 2 has range 0 to 7, while Key ID field within Secured Control field has a size of 4 bits which would cover a range of 0 to 15. I am not sure if these are all related, but I think at least some of them are, so field size and value ranges should be aligned. | Check/align the key ID and key index field sizes and ranges as appropriate. | **Revise**  Resolved by comment #85 |
| Kivinen, Tero | 85 | 36 | 9.5.11 | 7 | Here the range is 0-255, but I think the frames only include four bits for this field. | Change to 0-15. | **Accept** |
| Kivinen, Tero | 86 | 36 | 9.5.11 | 7 | Do not use the same name secAeadAlgorithm than what is already used in other tables. | Change secAeadAlgorithm to secCompactAeadAlgorithm. | **Accept** |

**8.3.4 MCPS-DATA.request [#334**]

***Change the sub-clause as follows (Track changes ON)***

**…**

**Table 1—Elements of the CompactFrameDescriptor**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Valid range** | **Description** |
| CompactFrameID | Unsigned Integer | As per Table 10. | This identifies the Compact frame, as per Table 10, and provides the Compact Frame ID field value. If CompactSecurityEnabled is TRUE, this is set as the Secured Compact Frame ID field of the Secure Compact frame and identifies the Compact frame whose Message Content field is to protected. |
|  | … |  |  |
| CompactSecurityEnabled | Boolean | FALSE, TRUE | When TRUE indicates that the Message Content field of the Compact frame is secured. |
| CompactSecurityParams | Structure | As defined in Table 2 |  |
| SourceIrk | Set of octets | 128-bit identity | This parameter provides an IRK to use to generate the RPA Hash field for those Compact frames for which this is appropriate. |

**9.2.12 Outgoing frame security procedure for Compact frames [#70, #73]**

***Change the sub-clause as follows (Track changes ON)***

**…**

This procedure involves the following steps:

1. **a) Is security needed?** If the SecurityLevel parameter is zero or four, the procedure shall return with 12 a Status of UNSUPPORTED\_SECURITY.
2. **b) Is security enabled?** If *secCompactFrameSecurityEnabled* is set to FALSE, the procedure shall 14 return with a Status of UNSUPPORTED\_SECURITY.
3. **c) Is security applicable?** If the Compact frame is not eligible for security (as listed in Table 24), the procedure shall return with a Status of UNSUPPORTED\_SECURITY.
4. **d) Obtain the source address**. Source address shall be set to the *macExtendedAddress*.

**…**

1. **h) Secure the Compact frame**. The Private Payload field shall be set to the Message Content field, and Open Payload field shall be empty. If the algorithm specified by *secCompactAeadAlgorithm* is not supported, then the procedure shall return with a Status of UNSUPPORTED\_ALGORITHM. If the size of *secCompactFrameKey* does not match the key length requirements of the algorithm specified by *secCompactAeadAlgorithm*, then the procedure shall return with a Status of KEY\_LENGTH\_MISMATCH. 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 Compact frame, according to the transformation process described in 9.3.4.

**9.2.13 Incoming frame security procedure for the Compact frames [#337, #74, #75, #76, #79]**

***Change the sub-clause as follows (Track changes ON)***

**…**

This procedure shall be used for the incoming Secured Compact frames. This procedure does not apply to other Compact frames. For other frame types, the procedures in 9.2.4 or 9.2.5 shall be used.

…

This procedure involves the following steps:

…

**b) Parse the Security Control field**. The procedure shall set SecurityLevel and KeyIndex to the Security Level field and Key ID field, respectively, of the frame to be unsecured. If the SecurityLevel is zero or four, the procedure shall return with a Status of UNSUPPORTED\_SECURITY. If the Secured Compact Frame ID field is not set as the ID of a Compact frame eligible for security (as listed in Table 24), the procedure shall return with a Status of UNSUPPORTED\_SECURITY.

1. **c) Obtain the source address**. Source address and KeySource shall be set to the extended address of the originator of the Compact frame as described in 10.39.11.1.2.3. If the extended address is not found, then the procedure shall return with a Status of UNAVAILABLE\_DEVICE.

…

1. **f) Unsecure the Compact frame**. The Private Payload field shall be set to the Message Content field and Open Payload field shall be empty. If the algorithm specified by *secCompactAeadAlgorithm* is not supported, then the procedure shall return with a Status of UNSUPPORTED\_ALGORITHM. If the size of *secCompactFrameKey* does not match the key length requirements of the algorithm specified by *secCompactAeadAlgorithm*, then the procedure shall return with a Status of KEY\_LENGTH\_MISMATCH. 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 Compact frame, according to the inverse transformation process described in the security operations, as described in 9.3.5. If the inverse transformation process fails, the procedure shall return with a Status of SECURITY\_ERROR.

**9.3.2.4 AEAD Nonce for Compact frames**

***Change the sub-clause as follows (Track changes ON)***

…

|  |  |  |
| --- | --- | --- |
| Octets: 2 | 1 | 2 |
| Slot Index | Round Index | Block Index |

**Figure 2—Frame Counter field for Compact frame nonce**

***Rename Figure 141 as: Security Control field format***