**IEEE P802.15**

**Wireless Personal Area Networks**

|  |  |  |
| --- | --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) | |
| Title | Comments and Resolution Work Items re 802.15.4e Draft D03 | |
| Date | March 12, 2011 | |
| Source | René Struik  Struik Security Consultancy  723 Carlaw Avenue  Toronto ON  Canada M4K 3K8 | E-mail: rstruik.ext@gmail.com  Phone US: +1 (415) 690-7363  Phone Toronto: +1 (416) 466-8084  Cell: +1 (647) 867-5658 |
| Re: | IEEE 802.15.4 TGe/Draft D03 | |
| Abstract | This document provides suggested resolutions of (mostly) security-related comments that were submitted with letter ballot on 802.15.4e/D03. | |
| Purpose | Assist with fixing/improving the (mostly) security-relevant portions of the draft 802.15.4e standard. | |
| Notice | This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. | |
| Release | The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15. | |

IMPORTANT NOTE:

The suggested resolutions in this submission are relative to security-related comments on IEEE 802.15.4e/D03. Some of these are address comments that were already discussed in the context of comment resolution on 802.15.4e/D02, but whose resolution did not make it to the next draft (D3.0). This document will be updated, so as to provide detailed resolutions relative to the most recent version of 802.15.4i, rather than 802.15.4-2006, as is currently the case (to our understanding, 802.15.4i is supposed to be the correct baseline for 802.15.4e).

**7.1.1.1 MCS-DATA.request**

**7.1.1.1.1 Semantics of the Service Primitive**

***Change the MCPS-DATA.request primitive as follows:***

MCPS-DATA.request (

SrcAddrMode,

DstAddrMode,

DstPANId,

DstAddr,

msduLength,

msdu,

msduHandle,

TxOptions,

SecurityLevel,

FrameCounterMode,

KeyIdMode,

KeySource,

KeyIndex

)

***Insert the following element in Table 41, directly below the SecurityLevel element***

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| *FrameCounterMode* | Integer | 0x00-0x07 | Representation of the frame counter to be used (see 7.6.2.2.3). This parameter is ignored if the  SecurityLevel parameter is set to 0x00. |

**7.1.1.1.3 Effect on Receipt**

***Change the security-related paragraph as follows:***

If the SecurityLevel parameter is set to a valid value other than 0x00, indicating that security is required for

this frame, the MAC sublayer will set the Security Enabled subfield of the Frame Control field to one. The

MAC sublayer will perform outgoing processing on the frame based on the DstAddr, SecurityLevel,

FrameCounterMode, KeyIdMode, KeySource, and KeyIndex parameters, as described in 7.5.8.2.1. If any error occurs during outgoing frame processing, the MAC sublayer will discard the frame and issue the MCPS-DATA.confirm primitive with the error status returned by outgoing frame processing.

***Editorial note*** – The FrameCounterMode parameter should be added to all outgoing primitives with security-related attributes SecurityLevel, KeyIdMode, KeySource, KeyIndex and text modified accordingly (first occurrence with 7.1.1.1.1 noted above). With the MLME-Start.request (7.1.14.1), one should add both the CoordRealignFrameCounterMode, and the BeaconFrameCounterMode.

Impacted clauses of 802.15.4-2006 are enumerated below (similar changes to TG4e-only primitives):

MCPS-DATA.indication (7.1.1.3), MLME-Associate.request (7.1.3.1), MLME-Associate.indication (7.1.3.2), MLME-Associate.response (7.1.3.3), MLME-Associate.confirm (7.1.3.4), MLME-Disassociate.request (7.1.4.1), MLME-Dissociate.indication (7.1.4.2), MLME-GTS.request (7.1.7.1), MLME-GTS.indication (7.1.7.3), MLME-Orphan.indication (7.1.8.1), MLME-Orphan.response (7.1.8.2), MLME-Scan.request (7.1.11.1), MLME-Comm-Status.indication (7.1.12.1), 7.1.14.1 MLME-Start.request (7.1.14.1), MLME-Sync-Loss.indication (7.1.15.2), MLME-Poll.request (7.1.16.1).

**7.5.8.1 Security-related MAC-PIB attributes**

***Change the Automatic request attributes as follows:***

−Automatic request attributes (*macAutoRequestSecurityLevel*, *macAutoRequestFrameCounterMode*,

*macAutoRequestKeyIdMode*, *macAutoRequestKeySource*, *macAutoRequestKeyIndex*)

***Insert the following attribute at the end of the enumeration:***

−Device time (*macCurrentTime*)

**7.5.8.1.4 Frame counter**

***Change 7.5.8.1.4 as follows:***

The ~~4-octet~~ 6-octet frame counter is used to provide replay protection and semantic security of the cryptographic building block used for securing outgoing frames. The frame counter is included in each secured frame and is one of the elements required for the unsecuring operation at the recipient(s). The frame counter is incremented each time an outgoing frame is secured, as described in the outgoing frame security procedure (see 7.5.8.2.1). When the frame counter is used, it is scaled-down to a 4 ½ octet value for usage with a particular key. When ~~the~~ this scaled frame counter reaches its maximum value of 0xfffffffff, the associated keying material can no longer be used, thus requiring ~~all~~ this particular key~~s~~ ~~associated with the device~~ to be updated. This provides a mechanism for ensuring that the keying material for every frame is unique and, thereby, provides for sequential freshness. The frame counter may be included with the secured frame in a compressed format, thus allowing bandwidth savings in scenarios where the full frame counter value can be faithfully reconstructed from the compressed frame counter value, as contained in an incoming frame, and locally maintained status information.

***Insert the following subclause right at the end of 7.5.8.1 (right after 7.5.8.1.7 – PAN coordinator address):***

**7.5.8.1.8 Device time**

The 6-octet device time, if available, is used both to enable compression of frame counters (thus, realizing overhead reduction) and to enhance the security functionality, by providing the capability to detect whether a received frame was purportedly sent relatively recently. This functionality is implemented by correlating the frame counter with the device time, when securing outgoing frames and unsecuring incoming frames. Device time is expressed with granularity of 16kHz. When the device time reaches its maximum value 0xffffffffffff, security can no longer be used. Frame counter compression and detection of so-called stale frames is only offered if device time is indeed available. If device time is not available, it is assumed to be set to the fixed integer value zero.

**7.5.8.2.1**

***Change the first sentence of 7.5.8.2.1 as follows:***

The inputs to this procedure are the frame to be secured and the SecurityLevel, FrameCounterMode, KeyIdMode, KeySource, and KeyIndex parameters from the originating primitive or automatic request PIB attributes.

***Change the following steps in 7.5.8.2.1:***

***Insert the following step right before Step a):***

a0) The procedure shall set *macCurrentTime* to the current absolute device time, measured in 16 kHz granularity. If

this procedure fails, the procedure shall set this parameter to the integer value zero.

***Replace Step f) by the following two steps, to be inserted right after Step g):***

***Editorial note – the order has to be changed, since Step f1 below depends on availability of the KeyDescriptor.***

f1) The procedure shall set the frame counter to the maximum value of the *macFrameCounter*and the

*macCurrentTime* attributes and shall set the scaled frame counter to the difference between the

frame counter and the KeyOffset element of the KeyDescriptor.

f2) If the scaled frame counter value is negative, the procedure shall return with a status of KEY\_ERROR. If the scaled frame counter is greater than or equal to 0xfffffffff, the procedure shall return with a status of COUNTER\_ERROR.

***Change Step h) as follows:***

***Insert the following step right after Step h)2):***

2a) The frame counter mode subfield of the security control field shall be set to the Frame-

CounterMode parameter.

***Change Step h)3) as follows:***

3) The Frame Counter field shall be set to the representation of the frame counter indicated by the Frame Counter mode parameter.

If the FrameCounterMode is set to 0x00, the FrameCounter subfield of the auxiliary security header shall be set to value of the frame counter (modulo 232).

If the FrameCounterMode is set to 0x01, the FrameCounter subfield of the auxiliary security header shall be set to value of the frame counter (modulo 256).

If the FrameCounterMode is set to 0x02, the FrameCounter subfield of the auxiliary security header shall be set to the empty string.

***Change the first sentence of Step i) as follows:***

The procedure shall then use *aExtendedAddress*, the scaled frame counter, the security level, and the ~~k~~Key Element of the KeyDescriptor to produce the secured frame according to the transformation process known as CCM\* [or the extension of CCM, which is the combined counter with CBC-MAC (i.e., cipher block chaining message authentication code) mode of operation] that is described in the security operations (see 7.6.3.4).

**7.5.8.2.2 Outgoing frame key retrieval procedure**

***Change the second sentence as follows:***

The outputs from this procedure are a passed or failed status and, if passed, a KeyDescriptor~~key~~.

***Remove Step d).***

***Change Step e) as follows:***

e) The procedure shall return with a passed status, having obtained the KeyDescriptor~~key identifier and the key~~.

**7.5.8.2.3 Incoming frame security procedure**

***Change the following steps in 7.5.8.2.3:***

***Insert the following step right before Step a):***

a0): The procedure shall set *macCurrentTime* to the current absolute device time, measured in 16 kHz granularity. If this procedure fails, the procedure shall set this parameter to the integer value zero.

***Change the first sentence of Step c) as follows:***

c) If the Security Enabled subfield of the Frame Control field of the frame to be unsecured is set to one, the procedure shall set the security level, the frame counter mode, and the key identifier mode to the corresponding subfields of the Security Control field of the auxiliary security header of the frame to be unsecured and shall set the key source and key index to the corresponding subfields of the Key Identifier field of the auxiliary security header of the frame to be unsecured, if present.

***Replace Step j) by the following three steps (in order):***

j1) The procedure shall reconstruct the frame counter from the FrameCounter field of the auxiliary security header of the frame to be unsecured, according to the representation indicated by the FrameCounterMode of that field, as follows:

* If the FrameCounterMode is set to 0x00, the frame counter shall be set to the smallest integer N such that N = FrameCounter (modulo 232) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.
* If the FrameCounterMode is set to 0x01, the frame counter shall be set to the smallest integer N such that N = FrameCounter (modulo 256) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.
* If the FrameCounterMode is set to 0x02, the frame counter shall be set to the smallest integer N such that N = DSN (modulo 256) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.

Note – add language on detection of stale frames! (and interval boundaries checks)

j2) shall set the scaled frame counter to the difference between the resulting frame counter and the KeyOffset element of the KeyDescriptor.

j3) If the scaled frame counter value is negative, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of KEY\_ERROR.

j4) If the scaled frame counter is greater than or equal to 0xfffffffff, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of COUNTER\_ERROR.

***Change the first sentence of Step k) as follows:***

k) The procedure shall then use the ExtAddress element of the DeviceDescriptor, the scaled frame counter, the security level, and the Key element of the KeyDescriptor to produce the unsecured frame according to the CCM\* inverse transformation process described in the security operations (see 7.6.3.5).

***Change Step o) as follows:***

o) If the FrameCounter element is equal to 0xffffffffffff, the procedure shall set the Blacklisted element of

the KeyDeviceDescriptor.

**7.6 Security suite specifications**

**7.6.1 PIB security material**

Change the macFrameCounter element of Table 85 as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute | Identifier | Type | Range | Description | Default |
| *macFrameCounter* | 0x77 | Integer | 0x000000000000-0xffffffffffff | The outgoing frame counter for this device | 0x000000000000 |

Insert the following elements to Table 85 as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute | Identifier | Type | Range | Description | Default |
| *macAutoRequestFrameCounterMode* | TBD | Integer | 0x00-0x07 | The frame counter mode used for automatic data requests. This attribute is invalid if the *macAutoRequestSecurityLevel* attribute is set to 0x00. | 0x00 |
| *macCurrentTime* | TBD | Integer | 0x000000000000-0xffffffffffff | The most recent absolute device time, as measured just prior to performing an incoming or outgoing frame security procedure, in 16 kHz accuracy. | 0x000000000000 |

*Editorial note –* The *macAutoRequestFrameCounterMode* attribute should be placed right below the *macAutoRequestSecurityLevel* attribute; the other attribute should be inserted at the end of the table.

Insert the KeyOffset attribute in Table 89 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| KeyOffSet | Integer | 0x000000000000-0xffffffffffff | The start time counter value for this key, prior to which this key shall not be used. A value of 0 indicates that there are no time restrictions as to when to start using this key. |

*Editorial note –* The KeyOffset attribute should be placed right before the KeyDeviceList attribute.

Insert the AcknowledgementFrameIdentifier attribute in Table 90 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| AcknowledgementFrameIdentifier | Integer | 0x00-TBD | See Table {TBD}. |

*Editorial note –* The AcknowledgementFrameIdentifier attribute should be placed right after the CommandFrameIdentifier attribute.

Insert the AcknowledgementFrameIdentifier attribute in Table 92 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| AcknowledgementFrameIdentifier | Integer | 0x00-TBD | See Table {TBD}. |

*Editorial note –* The AcknowledgementFrameIdentifier attribute should be placed right after the CommandFrameIdentifier attribute.

Change the FrameCounter element of Table 93 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| *macFrameCounter* | Integer | 0x000000000000-0xffffffffffff | The incoming frame counter of the device in this DeviceDescriptor. This value is used to ensure sequential freshness of frames. |

**7.6.2 Auxiliary security header**

Change Figure 74 as follows:

|  |  |  |
| --- | --- | --- |
| **Octets: 1** | **0/1/4/6** | **0/1/5/9** |
| Security Control | Frame Counter | Key Identifier |

**7.6.2.2 Security control field**

Change Figure 76 as follows:

|  |  |  |
| --- | --- | --- |
| **Bits: 0–2** | **3–4** | **5-7** |
| Security Level | Key Identifier Mode | Frame Counter Mode |

Insert after 7.6.2.2.2 (Key Identifier Mode subfield) the following subclause:

**7.6.2.2.3 Frame Counter Mode subfield**

The Frame Counter Mode subfield is 3 bits in length and is used to indicate the particular representation of the frame counter field (see 7.6.2.3). The frame counter mode subfield shall be set to one of the nonreserved values listed in Table {98a}.

Table{98a}– Values of the frame counter mode

|  |  |  |  |
| --- | --- | --- | --- |
| Frame counter mode | Frame Counter Mode subfield b2 b1 b0 | Description | Frame Counter subfield length |
| 0x00 | ‘000’ | Frame counter is determined from the 4-octet Frame Counter subfield of the auxiliary security header in conjunction with *macCurrentTime*. This mode corresponds to the frame counter as used in 802.15.4-2006, where the *macCurrentTime* attribute is not present (“stuck at zero”). | 4 |
| 0x01 | ‘001’ | Frame counter is determined from the 1-octet Frame Counter subfield of the auxiliary security header in conjunction with *macCurrentTime*. | 1 |
| 0x02 | ‘010’ | Frame Counter is determined from the DSN subfield of the frame header in conjunction with *macCurrentTime*. | 0 |
| 0x03-0x07 | − | Reserved | − |

**7.6.2.3 Frame counter field**

***Change 7.6.2.3 as follows:***

**7.6.2.3** The frame counter field is 0, 1, 4, or 6 octets in length, according to the representation specified by the frame counter mode subfield of the auxiliary security header (see 7.6.2), and ~~represents~~ indicates the *macFrameCounter* attribute of the originator of a protected frame. It is used to provide semantic security of the cryptographic mechanism used to protect a frame and to offer replay protection.

**7.6.3.2 CCM\* Nonce**

***Change Figure 77 as follows:***

|  |  |  |
| --- | --- | --- |
| **Octets: 8** | **4** | **1** |
| Source address | Frame counter | Security field |

***Insert the following figure right after Figure 77:***

|  |  |  |
| --- | --- | --- |
| **Bits: 0-3** | **4** | **5-7** |
| Ext. frame counter | Set to 0 | Security level |

Figure {77a} – Security field

***Change the paragraph right after Figure 77 as follows:***

The source address field shall be set to the extended address *aExtendedAddress* of the device originating the frame. ~~, the frame counter to the value of the respective field in the auxiliary security header (see 7.6.2),~~The frame counter field and the ext. frame counter field shall be set so that the right-concatenation of the extended frame counter field and the frame counter field, as bit-strings, has the same integer value as the scaled frame counter, as reconstructed from the frame counter subfield of the auxiliary security header (see 7.6.2), taking into account the frame counter mode subfield of the security control field hereof (see 7.6.2.2.3)~~, and the~~ The security level field shall be set to the security level.