IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SB2 Resolutions CIDs 11010 11024 11025 11026 | | | | |
| Date: 2013-08-27 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Matthew Fischer | Broadcom | 190 Mathilda Place, Sunnyvale, CA 94086 | +1 408 543 3370 | [mfischer@broadcom.com](mailto:mfischer@broadcom.com) |
|  |  |  |  |  |

Abstract

Addressing CIDs 11010 11024 11025 10026 of TGac SB2.

**Revision Notes**

**R1:**

CID 11026 Baseline does effectively prohibit multi-TID/AC A-MPDU – therefore, proposed changes modified by adding a NOTE to show the first step of the multi-indirectional path that must be followed to find the prohibition. Also deleted text that proposed rules on response transmissions following multi-TID A-MPDUs. Discussion modified to reflect this fact.

**R0:**

Initial

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Commenter Name** | **P.L** | **SC** | **Comment** | **Proposed Change** | **Resolution** |
| 11010 | Adachi, Tomoko | 86.59 | 8.4.27.2 | Related to my comment, # i-154 (10154), which was rejected in the previous Sponsor Ballot, there is no clear definition that cipher suite selector value 00-0F-AC:4 (CCMP) is CCMP with a 128-bit key and cipher suite selector value 00-0F-AC:10 (CCMP-256) is CCMP with a 256-bit key. On the other hand, GCMP-128 and the newly added GCMP-256 are explained in 11.4.5.1. Trying to add the descriptions of these relations in 11.4.3.1 by referring to 11.4.5.1, it is better to change the name CCMP in 8.4.2.27.2 to CCMP-128 to differentiate with the case just saying the name of the protocol. | Change the term CCMP which is for CCMP with a 128-bit key to CCMP-128 throughout the draft. Change the third paragraph in 11.4.3.1 to read "The AES algorithm is defined in FIPS PUB 197-2001. All AES processing used within CCMP uses AES with either a 128-bit key (CCMP-128) or a 256-bit key (CCMP-256)." | Revise – As the commenter hints, the term CCMP is used to mean both a specific implementation of CCMP (e.g. CCMP-128) and the generic function of CCMP. This is indeed, not a good idea. The same problem exists for BIP. TGac editor to make changes shown in 11-13-1007r0 under the heading for CID 11010 to introduce the terms CCMP-128 and BIP-CMAC-128 as needed. |

**CID 11010**

**Discussion:**

While the previous rejection states that “CCMP-256” is distinct from “CCMP”, there are a few lines of the draft that include different terms:

**11.4.3.1 General**

CCM is defined in IETF RFC 3610. CCM is a generic mode that can be used with any block-oriented encryption algorithm. CCM has two parameters (M and L). CCMP with a 128-bit key uses the following values for the CCM parameters:

And CCMP with a 256-bit key uses the following values for the CCM parameters:

So, in fact, the draft uses the term “CCMP” to be generic for CCMP, and not as an equivalent for CCMP-128. Many other instances of this genericity exist in the text.

A similar problem exists for BIP, where there are cipher suite selectors named BIP and BIP-CMAC-256 and BIP-GMAC-128 and BIP-GMAC-256, yet there are clauses and sentences using the term “BIP” to be inclusive of EVERY ONE OF THESE VARIANTS, which potentially confuses the singular BIP with the plural BIP.



**Proposed changes:**

***TGac editor: Please modify the following specific items within various subclauses, as shown*:**

**3.2 Definitions specific to IEEE 802.11**

**payload protected (PP) aggregate medium access control (MAC) service data unit (A-MSDU):** An AMSDU

that is protected with CTR with CBC-MAC Protocol (CCMP) or GCMP but does not include the A-MSDU

Present field (bit 7 of the QoS Control field) in the construction of the additional authentication data (AAD).

**per-frame sequence counter:** For Temporal Key Integrity Protocol (TKIP), the counter that is used as the

nonce in the derivation of the per-frame encryption key. For Counter mode with Cipher-block chaining

Message authentication code Protocol (CCMP) or GCMP, the per-frame initialization vector (IV).

**robust-security-network-association- (RSNA-) capable equipment:** A device that contains a station

(STA) that is able to create RSNAs. Such a device might use pre-RSNAs because of configuration. Notice

that RSNA-capable does not imply full compliance with the RSNA Protocol Implementation Conformance

Statement (PICS). A legacy device that has been upgraded to support Temporal Key Integrity Protocol

(TKIP) might be RSNA-capable, but is not compliant with the PICS if it does not also support Counter mode

with Cipher-block chaining Message authentication code Protocol (CCMP) using CCMP-128.

**signaling and payload protected (SPP) aggregate medium access control (MAC) service data unit (AMSDU):**

An A-MSDU that is protected with CTR with CBC-MAC Protocol (CCMP) or GCMP and that includes the

A-MSDU Present field (bit 7 of the QoS Control field) in the construction of the additional authentication

data (AAD).

**3.3 Abbreviations and acronyms**

***TGac editor: Add a new acronym in the appropriate location in the subclause:***

**GCMP Galois counter mode protocol**

**4.3.4.3 Robust security network association (RSNA)**

An RSNA defines a number of security features in addition to wired equivalent privacy (WEP) and IEEE

802.11 authentication. These features include the following:

— Enhanced authentication mechanisms for STAs

— Key management algorithms

— Cryptographic key establishment

— Enhanced data cryptographic encapsulation mechanisms, such as Counter mode with Cipher-block

chaining Message authentication code Protocol (CCMP), Galois Counter Mode Protocol (GCMP), and optionally, Temporal Key Integrity

Protocol (TKIP).

— Fast basic service set (BSS) transition (FT) mechanism

— Enhanced cryptographic encapsulation mechanisms for robust management frames

**4.5.4.1 General**

An RSNA uses the IEEE 802.1X authentication service along with enhanced data cryptographic

encapsulation mechanisms, such as TKIP, CCMP and GCMP to provide access control. The IEEE 802.11 station

management entity (SME) provides key management via an exchange of IEEE 802.1X EAPOL-Key frames.

Data confidentiality and data integrity are provided by RSN key management together with the enhanced

data cryptographic encapsulation mechanisms.

**4.5.4.4 Data confidentiality**

***TGac editor: Nothing needs to be done here, TGad already took care of it.***

IEEE Std 802.11 provides several cryptographic algorithms to protect data traffic, including: WEP, TKIP,

and CCMP. WEP and TKIP are based on the ARC419 algorithm, and CCMP is based on the advanced

encryption standard (AES). A means is provided for STAs to select the algorithm(s) to be used for a given

association.

IEEE Std 802.11 provides one security protocol, CCMP, for protection of individually addressed robust

management frames. This standard does not provide data confidentiality for group addressed robust

management frames.

IEEE Std 802.11 provides one security protocol, CCMP, for protection of individually addressed and group

addressed data frames between mesh STAs.

**4.5.4.7 Replay detection**

The replay detection mechanism defines a means by which a STA that receives a data or protected Robust

Management frame from another STA can detect whether the received frame is an unauthorized

retransmission. This replay protection mechanism is provided for data frames for STAs that use enhanced

data cryptographic encapsulation mechanisms. The replay protection mechanism is also provided for robust

management frames for STAs that use CCMP, GCMP and Broadcast/Multicast Integrity Protocol (BIP).

**4.5.4.9 Robust management frame protection**

Management frame protection protocols in an infrastructure BSS or IBSS apply to robust management

frames after RSNA PTK establishment for protection of individually addressed frames is completed and

after delivery of the IGTK to protect group addressed frames. Robust management frame protection is

implemented by CCMP, GCMP, BIP, and the SA Query procedure.

Management frame protection protocols in an MBSS apply to individually addressed frames after

establishment of the RSNA MTK, and to group addressed frames indicated as "Group Addressed Privacy" in

Table 8-38. Robust management frame protection is implemented with CCMP and GCMP.

**5.1.2 Security services**

Security services in IEEE Std 802.11 are provided by the authentication service and the CCMP, GCMP and BIP

mechanisms. The scope of the security services provided is limited to station-to-station data and robust

management frame transmissions. When CCMP or GCMP is used, the data confidentiality service is provided for data

frames and individually addressed robust management frames. For the purposes of this standard, CCMP and GCMP are

viewed as logical services located within the MAC sublayer as shown in the reference model, Figure 4-14

(in 4.9). Actual implementations of CCMP and GCMP are transparent to the LLC and other layers above the MAC

sublayer.

The security services provided by CCMP and GCMP in IEEE Std 802.11 are as follows:

a) Data Confidentiality;

b) Authentication; and

c) Access control in conjunction with layer management.

BIP provides message integrity and access control for group addressed robust management frames.

During the authentication exchange, both parties exchange authentication information as described in

Clause 11 and Clause 12.

The MAC sublayer security services provided by CCMP, GCMP and BIP rely on information from nonlayer-2

management or system entities. Management entities communicate information to CCMP, GCMP and BIP through a

set of MAC sublayer management entity (MLME) interfaces and MIB attributes; in particular, the decision

tree for CCMP, GCMP and BIP defined in 11.8 is driven by MIB attributes.

**6.3.19.1.2 Semantics of the service primitive**

***TGac editor: In the table, in the column for valid range for the row of KeyID, change “and CCMP” to “, CCMP and GCMP”***

**8.4.2.27 RSN element**

**8.4.2.27.1 General**

***TGac editor: Within the NOTE in this subclause, change all instances of CCMP to CCMP-128 and change BIP to BIP-CMAC-128.***

**8.4.2.27.2 Cipher suites**

***TGac editor: In tables 8-99 and 8-100, please change CCMP to “CCMP-128” and BIP to “BIP-CMAC-128”.***

**11.4 RSNA confidentiality and integrity protocols**

**11.4.3 CTR with CBC-MAC Protocol (CCMP)**

**11.4.3.1 General**

***Change as follows:***

Subclause CTR with CBC-MAC Protocol (CCMP) specifies all variants of CCMP, which provide data confidentiality, authentication, integrity, and replay protection. CCMP-128 is mandatory for RSN compliance.

CCMP is based on the CCM of the AES encryption algorithm. CCM combines CTR for data confidentiality and CBC-MAC for authentication and integrity. CCM protects the integrity of both the MPDU Data field and selected portions of the IEEE 802.11 MPDU header.

The AES algorithm is defined in FIPS PUB 197-2001. All AES processing used within CCMP uses AES with either a 128-bit key (CCMP-128) or a 256-bit key (CCMP-256)~~and a 128-bit block size~~.

CCM is defined in IETF RFC 3610. CCM is a generic mode that can be used with any block-oriented encryption algorithm. CCM has two parameters (M and L).~~, and~~ CCMP-128 uses the following values for the CCM parameters:

— M = 8; indicating that the MIC is 8 octets.

— L = 2; indicating that the Length field is 2 octets, which is sufficient to hold the length of the largest possible IEEE 802.11 MPDU, expressed in octets.

And CCMP-256 uses the following values for the CCM parameters:

— M = 16; indicating that the MIC is 16 octets.

— L = 2; indicating that the Length field is 2 octets, which is sufficient to hold the length of the largest possible IEEE 802.11 MPDU, expressed in octets.

CCM requires a fresh temporal key for every session. CCM also requires a unique nonce value for each frame protected by a given temporal key, and CCMP uses a 48-bit packet number (PN) for this purpose. Reuse of a PN with the same temporal key voids all security guarantees.

**11.4.3.2 CCMP MPDU format**

***Change the length of the MIC field in Figure 11-16 from "8 octets" to "variable".***

***Change the 2nd paragraph as follows:***

CCMP-128 processing expands the original MPDU size by 16 octets, 8 octets for the CCMP Header field and 8 octets for the MIC field. CCMP-256 processing expands the original MPDU size by 24 octets, 8 octets for the CCMP Header field and 16 octets for the MIC field. The CCMP Header field is constructed from the PN, ExtIV, and Key ID subfields. PN is a 48-bit PN represented as an array of 6 octets. PN5 is the most significant octet of the PN, and PN0 is the least significant. Note that CCMP does not use the WEP ICV.

**11.4.3.3 CCMP cryptographic encapsulation**

**11.4.3.3.6 CCM originator processing**

***Change the last paragraph as follows:***

The CCM originator processing provides authentication and integrity of the frame body and the AAD as well as data confidentiality of the frame body. The output from the CCM originator processing consists of the encrypted data and an ~~8 additional octets of~~ encrypted MIC (see Figure 11-16).

**11.4.3.4 CCMP decapsulation**

**11.4.3.4.2 CCM recipient processing**

***Change the 2nd paragraph as follows:***

There are four inputs to CCM recipient processing:

— *Key*: the temporal key (16 octets).

— *Nonce*: the nonce (13 octets) constructed as described in 11.4.3.3.4.

— *Encrypted frame body*: the encrypted frame body from the received MPDU. The encrypted frame body includes the ~~an 8-octet~~ MIC.

— *AAD*: the AAD (22-30 octets) that is the canonical MPDU header as described in 11.4.3.3.3.

**11.4.4.1 BIP overview**

***Change as follows:***

BIP provides data integrity and replay protection for group addressed robust management frames after successful establishment of an IGTKSA (see 11.5.1.1.9 (IGTKSA).).

BIP-CMAC-128 provides data integrity and replay protection, using AES-128 in CMAC Mode with a 128-bit integrity key and a CMAC TLen value of 128 (16 octets). BIP-CMAC-256 provides data integrity and replay protection, using AES-256 in CMAC Mode with a 256-bit integrity key and a CMAC TLen value of 128 (16 octets). NIST SP 800-38B defines the CMAC algorithm and NIST SP 800-38D defines the GMAC algorithm. ~~All~~ BIP processing uses AES with a 128-bit or 256-bit integrity key ~~and a 128-bit block size,~~ and a CMAC TLen value of 128 (16 octets). The CMAC output for both BIP-CMAC-128 and BIP-CMAC-256 is truncated to 64 bits:

MIC = L(CMAC Output, 0, 64)

Where L is defined in 11.6.1 (Key hierarchy).

BIP-GCMP-128 uses AES with a 128-bit integrity key and BIP-GCMP-256 uses AES with a 256-bit integrity key. The authentication tag for both BIP-GCMP-128 and BIP-GCMP-256 is not truncated and shall be 128 bits (16 octets).

BIP uses the IGTK to compute the MMPDU MIC. The authenticator shall distribute one new IGTK and IGTK PN (IPN) whenever it distributes a new GTK. The IGTK is identified by the MAC address of the transmitting STA plus an IGTK identifier that is encoded in the MME Key ID field.

**11.4.4.5 BIP transmission**

***Change as follows:***

When a STA transmits a protected group addressed robust management frame, it shall

a) Select the IGTK currently active for transmission of frames to the intended group of recipients and construct the MME (see 8.4.2.57 (Management MIC element)) with the MIC field masked to 0 and the KeyID field set to the corresponding IGTK KeyID value. The transmitter shall insert a monotonically increasing non-negative integer into the MME IPN field. For BIP-GMAC-128 and BIP-GMAC-256, the initialization vector passed to GMAC shall be a concatentation of address 2 from the MAC header of the MPDU and the non-negative integer inserted into the MMP IPN field.

b) Compute AAD as specified in 11.4.4.3 (BIP AAD construction).

c) Compute an integrity value ~~AES-128-CMAC~~ over the concatenation of (AAD || Management Frame Body including MME), and insert the 64-bit output into the MME MIC field. For BIP-CMAC-128, the integrity value is 64-bits and is computed using AES-128-CMAC; for BIP-CMAC-256, the integrity value is 128-bits and is computed using AES-256-CMAC; for BIP-GMAC-128, the integrity value is 128-bits and is computed using AES-128-GMAC; and, for BIP-GMAC-256, the integrity value is 128-bits and is computed using AES-256-GMAC.

d) Compose the frame as the IEEE 802.11 header, management frame body, including MME, and FCS. The MME shall appear last in the frame body.

e) Transmit the frame.

**11.4.4.6 BIP reception**

***Change as follows:***

When a STA with management frame protection negotiated receives a group addressed robust management frame protected by BIP-CMAC-128, BIP-CMAC-256, BIP-GMAC-128 or BIP-GMAC-256, it shall

a) Identify the appropriate IGTK key and associated state based on the MME KeyID field. If no such IGTK exists, silently drop the frame.

b) Perform replay protection on the received frame. The receiver shall interpret the MME IPN field as a 48-bit unsigned integer. It shall compare this MME IPN integer value to the value of the receive replay counter for the IGTK identified by the MME Key ID field. If the integer value from the received MME IPN field is less than or equal to the replay counter value for this IGTK, the receiver shall discard the frame and increment the dot11RSNAStatsCMACReplays counter by 1. ~~The receiver shall extract and save the received MIC value, and compute the AES-128-CMAC over the concatenation of (AAD || Management Frame Body including MME) with the MIC field masked to 0 in the MME. If the result does not match the received MIC value, then the receiver shall discard the frame and increment the dot11RSNAStatsCMACICVErrors~~

c) If the replay protection succeeds, compute AAD for this management frame, as specified in 11.4.4.3 (BIP AAD construction). For BIP-GMAC-128 and BIP-GMAC-256, an initialization vector for GMAC is constructed as the concatenation of address 2 from the MAC header of the MPDU and the 48-bit unsigned integer from the MME IPN field.

d) Extract and save the received MIC value, and compute a verifier ~~the AES-128-CMAC~~ over the concatenation of (AAD || Management Frame Body || MME) with the MIC field masked to 0 in the MME. For BIP-CMAC-128, the verifier is AES-128-CMAC; for BIP-CMAC-256, the integrity value is 128-bits and is computed using AES-256-CMAC; for BIP-GMAC-128, the verifier is AES-128-GMAC; and, for BIP-GMAC-256, the verifier is AES-256-GMAC. If the result does not match the received MIC value, then the receiver shall discard the frame and increment the dot11RSNAStatsCMACICVErrors counter by 1.

e) Update the replay counter for the IGTK identified by the MME Key ID field with the integer value of the MME IPN field.

If management frame protection is negotiated, group addressed robust management frames that are received without BIP protection shall be discarded.

**B.4.4.1 MAC protocol capabilities**

***TGac editor: Within the section for PC34 RSNA, change the protocol description column entry as shown:***

Robust security network association (RSNA)

RSN element

Group cipher suite

Pairwise cipher suite list

Counter mode with Cipher-block chaining Message authentication code Protocol (CCMP) data confidentiality Protocol, using CCMP-128

CCMP cryptographic encapsulation Procedure using CCMP-128

CCMP decapsulation procedure using CCMP-128

**C.3 MIB Detail**

dot11RSNAConfigPairwiseCipherSizeImplemented OBJECT-TYPE

SYNTAX Unsigned32 (0..4294967295)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is a capability variable.

Its value is determined by device capabilities.

This object indicates the length in bits of the pairwise cipher key. This

should be 256 for TKIP and 128 or 256 for CCMP and 128 or 256 for GCMP."

::= { dot11RSNAConfigPairwiseCiphersEntry 4 }

**M.6.4 CCMP test vector**

***TGac editor: Change CCMP to “CCMP-128” in this subclause.***

**M.7.1 General**

***TGac editor: Change CCMP to “CCMP-128” in this subclause.***

**M.7.2 CCMP pairwise key derivation**

***TGac editor: Change CCMP to “CCMP-128” in this subclause and in the heading of this subclause.***

**M.9.1 BIP with broadcast Deauthentication frame**

***TGac editor: Change BIP to “BIP-CMAC-128” in this subclause and in the heading of this subclause.***

**M.9.2 CCMP with unicast Deauthentication frame**

***TGac editor: Change CCMP to “CCMP-128” in this subclause and in the heading of this subclause.***

**V.2.4 Sales meeting**

***TGac editor: Change CCMP to “CCMP-128” in this subclause.***

**W.1 Clarification of Mesh Data frame format**

***TGac editor: Change CCMP to “CCMP-128” in this subclause.***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11026 | Schelstraete, Sigurd | 156.27 | 9.19.2.4 | Figure 9-19b shows TXOP sharing for MU transmission. Section 9.12.6 talks about A-MPDU padding for MU. If I'm understanding correctly, section 9.12.6 says that MPDUs with different TIDs (corresponding to different ACs) can be combined in the same A-MDPU - see paragraph starting L42 on page 147. For some reason, the padding shown in Figure 9-19b does not reflect this. | Confirm that combining of MPDUs with different ACs in the same A-MPDU is allowed. If so, should this be reflected in Figure 9-19b? | Revise – TGac editor to make changes shown in 11-13-1007r0 under the heading for CID 11026 to include some clarifications to the construction of multi-AC PPDUs. Commenter to note that the general restrictions on A-MPDU contents found in the baseline prohibit multi-AC A-MPDUs. Also, the figure here and in many other instances in the standard is only an illustrative example of one possible case of the protocol described and was never intended to convey a complete depiction of every detail and aspect of the protocol. Because there are multiple cases to choose from both at the aggregation side and the acknowledgement side, it is difficult to determine which one particular example would be included, if one were to be chosen. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11025 | Schelstraete, Sigurd | 147.38 | 9.12.6 | "The same initial value of A-MPDU\_Length[n] shall also be used as APEP\_LENGTH[n] in the PHY-TXSTART.request primitive." Does this mean that APEP\_LENGTH[n] does not include possible secondary AC MPDUs that may be added to the PSDU (see subsequent paragraph)? If so, couldn't putting this APEP\_LENGTH[n] value in VHT-SIG-B cause the receiving STA to prematurely stop processing data? | Clarify | Revise – Tgac editor to make changes shown in 11-13-1007r0 under the heading for CID 11025 which generally agrees with the very gently implied suggestion of the commenter that the text is incorrect. *Have a bit more self-confidence man*! |
| 11024 | Schelstraete, Sigurd | 147.31 | 9.12.6 | It sounds like the APEP\_LENGTHs for the different users are chosen independently. Can this really be the case when MU TXOP sharing is used? Traffic to secondary ACs should not increase the length of the MU PPDU beyond what is needed to transmit the primary AC data. As such, shouldn't NSYM and TXTIME be determined by the primary AC alone? | Clarify | Reject – the original language did not include the secondary AC contents in the determination of the PPDU length or TXTIME value as can be seen in the return and use of the PSDU\_Length value based on the initial, primary-AC-only A-MPDU\_Length values – note that the PSDU\_Length generated based on the initial A-MPDU\_Length value is used as an upper bound to the addition of secondary AC MPDUs and padding. The proposed changes to resolve other CIDs for this subclause further clarify this to be the case by placing the language relating the final APEP\_Length value to the A-MPDU\_Length value that results after performing secondary AC A-MPDU additions, if any. |

**CID 11024, 11025, 11026**

**Discussion:**

CID 11026:

The baseline text regarding the allowed contets of A-MPDUs expressly forbids the inclusion of more than one AC per A-MPDU.

CID 11025:

Commenter is correct – secondary AC additions need to be accounted for in A-MPDU\_Length[*n*]

CID 11024:

Commenter is incorrect – the PPDU\_Length is calculated once based on the length of the initial primary AC MPDU contents and is then used as the upper limit on secondary AC addition and padding addition.

**Proposed changes:**

***TGac editor: Within subclause 9.12.6 A-MPDU padding for VHT PPDU, change the text as shown:***

**9.12.6 A-MPDU padding for VHT PPDU**

The A-MPDU\_Length[n] for user n is initialized as the length of the resulting A-MPDU pre-EOF padding. This initial value of A-MPDU\_Length[n] for user n is used as the APEP\_LENGTH[n] parameter value for the PLME-TXTIME.request primitive (see 6.5.7 (PLME-TXTIME.request)). The PLME-TXTIME.request primitive is then invoked once for the VHT PPDU. The PLME-TXTIME.confirm primitive (see 6.5.8 (PLME-TXTIME.confirm)) provides the TXTIME parameter and PSDU\_LENGTH[] parameters for all the users for the transmission.

Subsequently, for each user *n*, as permitted by the rules for EDCA TXOP Sharing (see 9.19.2.2a Sharing an EDCA TXOP), a VHT STA may add A-MPDU subframes to the A-MPDU for that user that meets either of the following conditions:

— Have a TID that maps to an AC that is not the primary AC

— Have 0 in the MPDU Length field and 0 in the EOF field

provided that each added subframe and the resulting A-MPDU meet all of the following:

— A-MPDU content constraints (see 9.12.1 (A-MPDU contents)) for the intended recipient

— Length limit constraints (see 8.6.1 (A-MPDU format) and 9.12.2 (A-MPDU length limit rules)) for the intended recipient

— MPDU start spacing constraints (see 9.12.3 (Minimum MPDU Start Spacing field)) for the intended Recipient

and provided that, after incrementing the A-MPDU\_Length[*n*] with the length of each such added A-MPDU subframe, the relationship A-MPDU\_Length[*n*] <= PSDU\_LENGTH[*n*] is true.

NOTE – An A-MPDU is prohibited by the rules in 9.12.1 (A-MPDU contents) from carrying MPDUs of more than one TID.

Subsequently, for each user *n*, a VHT STA may add A-MPDU subframes to the A-MPDU for that user that meet the following condition:

— Have 0 in the MPDU Length field

provided that each added subframe and the resulting A-MPDU meet the following condition:

— Length limit constraints (see 8.6.1 (A-MPDU format) and 9.12.2 (A-MPDU length limit rules)) for the intended recipient

and provided that, after incrementing the A-MPDU\_Length[*n*] with the length of each such added A-MPDU subframe, the relationship A-MPDU\_Length[*n*] <= PSDU\_LENGTH[*n*] is true.

An implementation may reduce the A-MPDU\_Length[*n*] by the amount of padding for user *n* which was added subsequent to the addition of a subframe for user *n* that contains 1 in the EOF field.

finalused for the VHT PPDU

Padding is then added for each user such that the resulting A-MPDU contains exactly PSDU\_LENGTH octets for that user as follows:

**22.2.2 TXVECTOR and RXVECTOR parameters**

***TGac editor: In table* Table 22-1—TXVECTOR and RXVECTOR parameters, *in the row that contains “APEP\_LENGTH” in the column labelled “Parameter”, please modify the entry in the column labelled “value”, as shown*:**

If equal to 0, indicates a VHT NDP PPDU for both RXVECTOR and TXVECTOR.

If greater than 0 in the TXVECTOR, indicates the number of octets in the range 1 to 1 048 575 in the A-MPDU pre-EOF padding (see 9.12.2 (A-MPDU length limit rules)) carried in the PSDU. This parameter is used to determine the number of OFDM symbols in the Data field that do not appear after a subframe with 1 in the EOF subfield and, after being rounded up to a 4 octet boundary with the two LSBs removed, is placed in the VHT-SIG-B Length field.

NOTE—The rounding up of the APEP\_LENGTH parameter to a 4-octet word boundary could result in a value that is larger than the PSDU\_LENGTH calculated using the equations in 22.4.3 (TXTIME and PSDU\_LENGTH calculation).

If greater than 0 in the RXVECTOR, this parameter is the value obtained from the VHT-SIG-B Length field multiplied by 4.

**References:**