IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Assorted CRs on REVmd draft 3.0 | | | | |
| Date: March 19, 2020 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Nehru Bhandaru | Broadcom | 250 Innovation Drive, San Jose CA | +1 408 391 2159 | [nehru.bhandaru@broadcom.com](mailto:nehru.bhandaru@broadcom.com) |
| Mike Montemurro | Blackberry |  |  | [montemurro.michael@gmail.com](mailto:montemurro.michael@gmail.com) |

Abstract

This document contains proposed resolutions for following CIDs against REVmd draft 3.0:

Resolutions in earlier versions of this document were discussed and adopted for the following CIDs

4031, 4032, 4033, 4086, ~~4088, 4089, 4090, 4091~~, 4092, 4093, 4188, 4230, 4308, ~~4326~~, 4388,

This document proposes resolutions for 4087, 4204, 4417, 4465, 4522, 4602, 4612, 4672, 4728 for discussion and possible adoption into the standard.

The baseline for this document is Draft P802.11REVmd D3.0.

**Revision History**

00: Initial version

01: Update based on Mark Rison’s comments – 02/13/20

02: Update based on Mark Rison’s comments – 02/15/20

03: Update based on 11md ad hoc

04: Update from teleconf on March 6, 2020

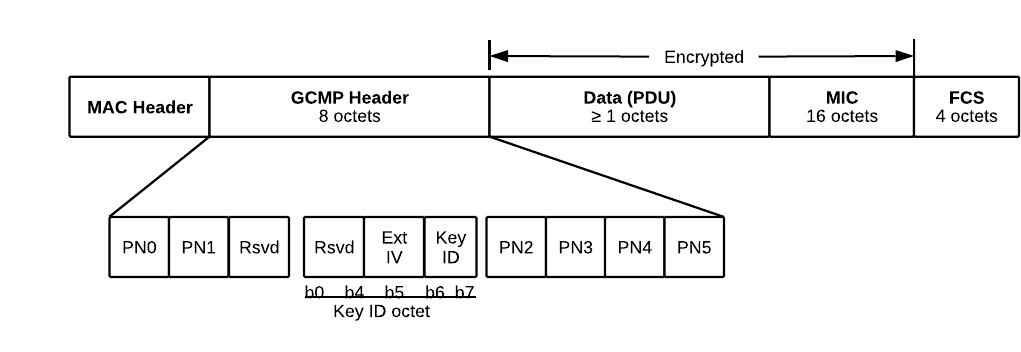
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Clause Number** | **Page** | **Line** | **Comment** | **Proposed Change** | **Resolution** |
| 4031 | 12.5.3.3.2 | 2603 | 49 | dot11PNExhaustionThreshold has been changed to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Fix dot11PNExhaustionThreshold to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Revise.  TGm Editor: Change the sentence as follows  (#2500)If the PN ~~is~~ ~~larger than dot11PNExhaustionThreshold~~ exceeds the threshold that is defined in dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh , an MLME-PN-EXHAUSTION.indication  primitive shall be generated. |
| 4032 | 12.5.4.4 | 2612 | 27 | dot11PNExhaustionThreshold has been changed to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Fix dot11PNExhaustionThreshold to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Revise.  TGm Editor: Change the sentence as follows  (#2500)If the PN ~~is~~ ~~larger than dot11PNExhaustionThreshold~~ exceeds the threshold that is defined in dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh , an MLME-PN-EXHAUSTION.indication  primitive shall be generated. |
| 4033 | 12.5.5.3.2 | 2616 | 31 | dot11PNExhaustionThreshold has been changed to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Fix dot11PNExhaustionThreshold to dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh. | Revise.  TGm Editor: Change the sentence as follows  (#2500)If the PN ~~is~~ ~~larger than dot11PNExhaustionThreshold~~ exceeds the threshold defined in dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh , an MLME-PN-EXHAUSTION.indication  primitive shall be generated. |
| 4086 | 12.5.3.3.2 | 2603 | 46 | "For PV1 MPDUs, the PN shall never repeat for a series of encrypted MPDUs using the same temporal key and TID/ACI." The above implies that the PN are allowed to be repeated for the same Key if the TID/ACI is different. This appears to violates the rule that the same PN shall never be reused for the same key. Refer to 12.5.3.1 General (P2601L14) which states that reuse of a PN with the same temporal key voids all security guarantees. | Review whether encrpytion of PV1 MPDUs violates the rule that the same PN shall never be reused for the same key. If it does, ensure that the same PN is never resued for the same key for PV1 MPDUs as well regardless of the TID/ACI. | Revise.  PN is allowed to repeat for a given temporal key for PV1, but not PV0. See discussion later in this document. Change the restriction on PN to restriction on nonce.  TGm Editor: Change 12.5.3.1 and12.5.5.1 as specified later in this document - 11-20/0246r4 |
| 4087 | 12.5.3.4.1 | 2608 | 25 | As per 12.5.3.3 (P2607L59), the MIC is also encrypted along with the plaintext MPDU, so it is not possible to obtain the original MIC at this stage. The original MIC can only be obtained after CCM decryption stage. The figure 12-23 is misleading, either it should be clarified that the MIC that is fed into the CCM decryption block is encrypted MIC, or the entire encrypted MPDU (instead of MIC and data) should be passed to the CCM decryption block. | Rectify the Figure 12-23 as per comment. Specifically, the MIC that is fed into the CCM decryption module should be "encrypted MIC" | Revise. It coukd be made clearer that MIC is the cncrypted MIC.  This also appies to GCM – figure 12-29  TGm Editor: Revise as indicated in this document 11-20/0246r6 |
| ~~4088~~ | 12.5.3.4.1 | 2608 | 50 | As per 12.5.3.3 (P2607L59), the MIC is also encrypted along with the plaintext MPDU, so it is not possible to obtain the original MIC at this stage. The original MIC can only be obtained after CCM decryption stage. The text "The MIC is extracted..." is misleading, at this stage this is encrypted MIC, the original MIC can only be obtained after passing through the CCM decryption block. | Reword to convey that the MIC that is used in the CCM integrity checking is only obtained after decryption of the encrypted MIC. | Addressed by Rojan in 20/0407 |
| ~~4089~~ | 12.5.3.4.1 | 2609 | 8 | As per 12.5.3.3 (P2607L59), the MIC is also encrypted along with the plaintext MPDU, so it is not possible to obtain the original MIC at this stage. The original MIC can only be obtained after CCM decryption stage. The text "The MIC is extracted..." is misleading, at this stage this is encrypted MIC, the original MIC can only be obtained after passing through the CCM decryption block. | Reword to convey that the MIC that is used in the CCM integrity checking is only obtained after decryption of the encrypted MIC. | Addressed by Rojan in 20/0407 |
| ~~4090~~ | 12.5.3.4.2 | 2609 | 50 | "CCM recipient processing checks the authentication and integrity of the frame body and the AAD as well as decrypting the frame body. The plaintext is returned only if the MIC check is successful." The above sentence is not clear at best, or is not accurate. The authentication and integrity check can only be performed once the original MIC has been decrypted. It should be explained that the decryption should happen first to obtain the plaintext MPDU and the original MIC. The MIC needs to be re-calculated over the plaintext MPDU following the procedure in 12.5.3.3 and compared with the decrypted MIC to verify that the MIC is correct. | Clarify that decryption should happen first to obtain the plaintext MPDU and the original MIC. The MIC needs to be re-calculated over the plaintext MPDU following the procedure in 12.5.3.3 and compared with the decrypted MIC to verify that the MIC is correct. | Addressed by Rojan in 20/0407 |
| ~~4091~~ | 12.5.3.4.3 | 2609 | 61 | "The decapsulation process succeeds when the calculated MIC matches the MIC value obtained from decrypting the received encrypted MPDU." It should be elaborated clearly how the MIC is calculated for the MIC check. | Clarify how the MIC is calculated for the MIC check. | Addressed by Rojan in 20/0407 |
| 4092 | 12.5.4.4 | 2612 | 1 | What field is this (MME Sequence Number)? this seems to be the only occurrence. Is it supposed to be the IPN/BIPN field? | Use the correct field name. | Revise.  This sentence seems redundant since later in the section it is specified to insert the IPN/BIPN etc. into the IPN/BIPN field of the MME (Management MIC Element). Remove this redundant sentence,  TGm Edtior: Change as follows  ~~The MME Sequence Number field represents a sequence number whose length is 6 octets.~~ |
| 4093 | 12.5.5.2 | 2615 | 7 | Figure 12-26: In GCMP isn't MIC also encrypted? P2617L25 mentions that it is. The figure should be amended showing MIC as encrypted. | Amend Figure 12-26 to show MIC as encrypted. | Revise.  TGm Editor change figure as per 11-20-0246r4  To be resolved along with 4087-4091 – see note |
| 4188 | 12.4.5.4 | 2574 | 14 | "a salt is passed to the KDF consisting of " is not using the normative form used in surrounding sentences | Change the cited text to "the salt passed to the KDF shall consist of " | Revise.  TGm Editor: Change as suggested.  Replace  “  a salt is passed to the KDF consisting of a concatenation of  the rejected groups from each peer's Rejected Groups element  “ with  “  …a salt consisting of the concatenation of the rejected groups from each peer's Rejected Groups element shall be passed to the KDF” |
| 4230 | 12.7 |  |  | Where "KDF-Hash-Length" is used, sometimes the "Length" is not specified (cf. "Length is Q plus 256", "Length = Q + 128", "Length is the total number of bits to derive, i.e., number of bits of the PTK. The length is dependent on the negotiated cipher suites and AKM suites as defined by Table 12-7 (Cipher suite key lengths) in 12.7.2 (EAPOL-Key frames) and Table 12-10 (Integrity and key-wrap algorithms(#102)(#1188)) in ", "Length is cipher-suite dependent and is defined by the TK\_bits value in Table 12-7 (Cipher suite key lengths).") | Specify the Length in 12.7.1.6.4. In 12.7.8.2 explicitly say "Length = TK\_bits + 128" | Revise.  TGm Editor change as follows  In 12.7.1.6.4  KDF-Hash-Length is the (#246)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)) to generate a key whose length is equal to the length of the hash algorithm’s~~  ~~digest~~.  — Hash is the hash algorithm specific to the negotiated AKM (see Table 9-151 (AKM suite  selectors)).  - Length is the length of the hash algorithm’s digest.  …  In 12.7.8.2:  KDF-Hash-Length is the key derivation function defined in 12.7.1.6.2 (Key derivation function (KDF)) ~~that uses Hash to generate a key whose length is TK\_bits + 128~~  Length is TK bits + 128 |
| 4308 | 12.6.1.1.9 | 2624 | 18 | "Direction vector (whether the IGTK is used for transmit or receive)" -- how can it not be rx for a non-AP STA and tx for an AP? | Delete the cited sentence | Reject.  Direction vector is part of GTKSA and IGTKSA and is configured on the AP/STA. But direction is fixed. Needs to be consistent for IGTK and GTK (2623.54). This specification is required for IBSS and Mesh where STAs have both transmit and receive IGTKSAs/GTKSAs |
| 4326 | 12.6.18 | 2640 | 41 | "NOTE 2---Because the IEEE 802.11 Null frame does not derive from an MA-UNITDATA.request primitive, it is not protected." -- the real reason is that there is nothing to protect. Some TDLS frames, for example, are not derived from MA-UNITDATA.requests, but are protected nonetheless. It's not clear what the point of this NOTE is anyway | Delete the cited text at the referenced location, and delete the " 1" immediately above | Revise  Change cited sentence at p2640.41 as follows  NOTE 2—~~Because the~~ IEEE 802.11 Null frames are ~~does not derive from an MA-UNITDATA.request primitive, it is~~ not protected. |
| 4388 | 12 |  |  | I presume GCMP is not allowed for S1G, since there's no description of GCMP for PV1 MPDUs. Where is this restriction specified? | As it says in the comment | Revise  The comment does not propose a change to the draft. Cipher suite negotiation is not PHY specific so there's no reason to impose a requirement that GCM is not allowed for S1G STAs. However since there is no GCMP specified for PV1, perhaps a statement to that effect might help until it is specified for PV1.  TGm Editor: Add the following at 12.5.5.1 (GCMP Overview) p2614.62  An S1G STA shall use PV0 frames when using GCMP encapsulation. |
| 4204 | 12.6.10.3 | 2635 | 22 | " When the PMKSA was not created using pre-authentication, the AKM indicated in the RSNE by the STA in the (Ed)(re)association request shall be identical to the AKM used to establish the cached PMKSA in the first place. " is too far away from Table 9-151--AKM suite selectors. Furthermore, it makes the table messy with lots of insertions of "or PMKSA caching" | Add a column to the table with heading something like "Can be used with PMKSA caching" and then state that this means that the AKM can also be used for the use of a cached PMKSA for a previous AKM of that type, and cross-reference from there to 12.6.10.3 Cached PMKSAs and RSNA key management | Revise.  Rather than add a new column and make the table more complex, remove PMKSA caching from AKM Suites table and adjust the text and references to PMK caching section later in the document accordingly.  See CID 4204 discussion section later in this document.  TGm Editor: change as specified in 11-20-0246r5 |
| 4417 | 12.5.3.3.3 | 2604 | 51 | "The Fragment Number subfield is not modified." -- delete (2x), since we don't say so for any of the other not-modified fields | As it says in the comment | Revise.  Fragment number is part of the Sequence Control field. One subfield (Sequence number) of which is masked and the other (Fragment Number) is not. Current text is and would be clearer.  However, for completeness, in item 1) describing masking of the frame control field, a statement that other subfields are not modified might be included for completeness.  TGm Editor: Add the following after bullet vi) 12.5.3.3.3 2604.44, 12.5.3.3.3 bullet vi) 2605.43, and bullet 3) 12.5.4.3 2611.45  vii) Other subfields are not modified. |
| 4465 | 12.6.18 | 2640 | 18 | "shall delete the PTKSA, GTKSA, IGTKSA, BIGTKSA(#2116) (#1504)and any TPKSA(#59)" -- there might not be an IGTKSA or BIGTKSA either | Change to "shall delete the PTKSA, GTKSA, any IGTKSA, any BIGTKSA(#2116) (#1504)and any TPKSA(#59)". In next sentence change " and IGTKSA" to " and any IGTKSA" | Accept.  IGTKSA is option and continguent on PMF being enabled. Ditto for BIGTKSA which is present only when optional Beacon Protection applies.  TGm Editor change as suggested. |
| 4522 | 12.5.4.4 | 2612 | 30 | 11md: "NOTE--When the IPN or BIPN space is exhausted, the choices available to an implementation are to replace the corresponding key or to end communications.(#2116)" should also be stated in the other places where PN-EXHAUSTION is discussed (12.5.3.3.3 for CCMP and 12.5.5.3.2 for GCMP) | As it says in the comment | Revise.  State as note in CCMP and GCMP description as indicated by the commentor  TGm Editor change as 12.5.3.3.2 p2603.49 and p2616.32 as follows  If the PN is larger than dot11PNExhaustionThreshold, an MLME-PN-EXHAUSTION.indication  primitive shall be generated.  Note -- When PN space is exhausted, , the choices available to an implementation are to replace the corresponding key or to end communications. |
| 4602 | 12 |  |  | There is confusion (cf. CID 2137 I think) about the general concept of a temporal key, and the temporal key (TK) in PTKs (Jouni is adamant they are not the same) | As it says in the comment | TBD Reject  There is no specific change suggested by the commentor. Comment does not specify what the confusion is.  Discuss in 11md  Assign to Mark or Jouni? |
| 4612 | 12 | 2609 | 5 | "4) The nonce(#1406) value is constructed from the STA MAC Address Identified By A2, PN, and Nonce Flags fields." is just duplication of Figure 12-21--Nonce field. Ditto duplication of Figure 12-28--Nonce field for GCMP | Replace the cited text, and "3) (11ah)The nonce(#1406) value is constructed from the A2, PN, and Nonce Flags fields." in 12.5.3.4.1 and "c) The nonce(#1406) value is constructed from the A2 and PN fields." in 12.5.5.4.1, with references to the figures. Also remove the "The Nonce field has an internal structure of Nonce Flags || (11ah)STA MAC Address Identified By A2 || PN" and "The Nonce field has an internal structure of A2 || PN" duplication (of figures immediately above!) | TBD  Many a figure is followed by an explanation in the spec. The description seems correct.  In general duplication needs to be avoided in the spec. Nevertheless, explanation of fields in the text that refer to fields in the figure is useful.  Discuss in 11md |
| 4672 | 12.4.4 |  |  | It should be "set to" on tx, not "equal to" | Change at end of first para of 12.4.4.2.3, second para of 12.4.7.4 | Accept.  TGm Editor: change as suggested by replacing ‘equal’ with ‘set’ |
| 4728 |  |  |  | "To prevent key reinstallation attacks, a non-AP STA in which dot11WNMSleepModeActivated is true shall maintain a copy of the most recent GTK and most recent IGTK " -- should not quadruplicate this statement, even less so with variant wordings. We did not duplicate the statement for the original KRACK fix (this point was ignored in the resolution of CID 2551) | In 11.2.3.16.1, 12.7.7.4, 12.12.2.1, 13.5.1 delete the para starting (#1321) and replace it with "NOTE---See 6.3.19 regarding prevention of key reinstallation attacks." | TBD Reject.  There is a reference to 6.3.19 at the end of the paragraph and it seems clear enough. Also, to include any note some text is needed to provide the context for the note.  Alternatively, 6.3.119.1.4 can be modified to refer to each of these sections. Not sure if that is the right thing to do – since setkeys is a primitive that applies all the keys and the last bullet 419.55 already states that received of a …matching… request shall not change the replay counter values and thus has no effect.  Discuss in 11md |

**CID 4093**

Discussion:

The encrypted field extends to MIC. Figure to be adjusted accordingly

TGm Editor, please replace Figure 12-26 as shown below.



**CID 4031**

Discussion

dot11PNExhaustionThresholdLow and dot11PNExhaustionThresholdHigh together represent 48-bit number and not a range.

**CID 4388**

The comment is as follows:

“I presume GCMP is not allowed for S1G, since there's no description of GCMP for PV1 MPDUs. Where is this restriction specified?”

Discussion

PV1 frames are compressed frames that are optionally used with S1G

The comment seems to ask for a restriction that S1G STAs only use CCMP - which puts a requirement on using specific ciphers with specific PHYs. We didn't even do that with TKIP. There is nothing to stop (from a specification point of view) an S1G STAs advertising and negotiating a GSM cipher suite – in that case they would use PV0 frames, since GCMP is not specified for PV1 frames.

There is no technical problem that would be solved by adding any additional text since at this time since negotiating PV1 frame exchanges with GCMP isn't a requirement. If and when required, a separate submission could address how PV1 frames may be used with GCMP.

Perhaps, to be consistent specify that GCMP shall not be used by S1G STAs

The proposed resolution doesn't actually propose a change.

**CID 4086**

Comment related to whether PV1 MPDUs reusing PN

Discussion

If PN is allowed to be repeated, statement in 12.5.3.3.1 2602.50 needs to be softened.

It is sufficient for CCM security that 13 octet nonce is unique i.e. CCM counter has to be unique for

each AES encrypted block (RFC 3610)

“A nonce N of 15-L octets. Within the scope of any encryption key

K, the nonce value MUST be unique.

“

in the construction of CCM Nonce, TID/ACI (Priority) is used in Nonce Flags (12.5.3.3.4 Construct CCM Nonce). That ensures that the CCM counter used for encryption is unique across TIDs and preserves security guarantees.

Perhaps both PV0 and PV1 construction can use the same language – but the decision seems to have been made consciously to be different – the same statement is repeated in 12.5.3.3.2 PN Processing in 11md D3.0 – and the restriction is different for PV0 and PV1

“For PV0 MPDUs, the PN shall never repeat

for a series of encrypted MPDUs using the same temporal key. (11ah)For PV1 MPDUs, the PN shall never

repeat for a series of encrypted MPDUs using the same temporal key and TID/ACI.”

Change to relax PN repetition for different priorities for PV0 may cause interoperability issues. Perhaps that is why it was left this way.

Propose to remove the sentence that reuse of PN voids all security guarantees in the General section.

**TGm Editor: Change 12.5.3.1 2601.13 as follows**

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~~ nonce value with the same temporal key voids all security guarantees.

**TGm Editor: Change 12.5.5.1 2614.50 as follows**

GCM requires a fresh temporal key for every session. GCM also requires a unique nonce value for each frame

protected by a given temporal key~~, and GCMP uses a 96-bit nonce that includes a 48-bit packet number (PN)~~

~~for this purpose~~. Reuse of a ~~PN~~ nonce value with the same temporal key voids all security guarantees. GCMP uses a 128-bit MIC.

**CID4728**

To prevent key reinstallation attacks, a non-AP STA in which dot11WNMSleepModeActivated is  
true shall maintain a copy of the most recent GTK and most recent IGTK " -- should not quadruplicate this statement, even less so with variant wordings. We did not duplicate the statement for the original KRACK fix (this point was ignored in the resolution of CID 2551)

**Discussion:**

Suggesgion (Mark R) make the reference 6.3.19.1.4, which is the specific requirement (“(#1321)(#2550)When the Key Type is Group, IGTK, or BIGTK(#2116), and the key matches the GTK,

IGTK, or BIGTK(#2116), if any, installed as a result of EAPOL-Key frames (see 12.7.7.4 (Group key

handshake implementation considerations)) or exiting WNM sleep mode (see 11.2.3.16.1 (WNM sleep

mode capability)) receipt of this primitive shall have no effect.”)

There is a reference to 6.3.19 at the end of the paragraph and it seems clear enough. Also, to include any note some text is needed to provide the context for the note.

Alternatively, 6.3.119.1.4 can be modified to refer to each of these sections. Not sure if that is the right thing to do – since setkeys is a primitive that applies all the keys and the last bullet 419.55 already states that received of a …matching… request shall not change the replay counter values and thus has no effect.

**CID 4204 Discussion**

**12.6.10.3 p2635.22**

Comment **“**When the PMKSA was not created using pre-authentication, the AKM indicated in the RSNE by the STA in the  
(Ed)(re)association request shall be identical to the AKM used to establish the cached PMKSA in the first  
place. " is too far away from Table 9-151--AKM suite selectors. Furthermore, it makes the table messy with lots of insertions of "or PMKSA caching"

Commentor proposes -- Add a column to the table with heading something like "Can be used with PMKSA caching" and then state that this means that the AKM can also be used for the use of a cached PMKSA for a previous AKM of that type, and cross-reference from there to 12.6.10.3 Cached PMKSAs and RSNA key management

Discussion:

The AKM Suites Table 9-151 is becoming complex

Jouni: Perhaps there are other ways to address this. Handle it clause 9… Some PMK caching does not apply to FILS and non-RSN AKMs. Delete everything about PMKSA caching from AKM Suites table

Rather than add a new column and make the table more complex, remove PMKSA caching from AKM Suites table and adjust the text and references to PMK caching section later in the document accordingly.

Remove references to PMKSA caching from AKM Suites table

Adjust the following paragraphs to reference PMKSA caching section and keep the sentence about default AKM

**TGm Editor: Change 9.4.2.24.3 AKM suites as specified below**

* AKM suites

The AKM Suite Count field indicates the number of AKM suite selectors that are contained in the AKM Suite List field. The value 0 is reserved.

The AKM Suite List field contains a series of AKM suite selectors. In an IBSS only a single AKM suite selector is specified because IBSS STAs use the same AKM suite and because there is no mechanism to negotiate the AKMP in an IBSS (see 12.6.5 (RSNA policy selection in an IBSS(#59))).

Each AKM suite selector specifies an AKMP. Table 9-151 (AKM suite selectors) gives the AKM suite selectors defined by this -standard. An AKM suite selector has the format shown in Figure 9-288 (Suite selector format).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| * AKM suite selectors | | | | | |
| OUI | Suite type | Meaning | | |  |
| Authentication type | Key management type | Key derivation type | Authentication algorithm numbers (see 9.4.1.1 (Authentication Algorithm Number field))(M85) |
| 00-0F-AC | 0 | Reserved | Reserved | Reserved | Reserved |
| 00-0F-AC | 1 | Authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | RSNA key management as defined in 12.7 (Keys and key distribution~~) or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.2 (PRF) | 0 (open) |
| 00-0F-AC | 2 | PSK | RSNA key management as defined in 12.7 (Keys and key distribution), using PSK | Defined in 12.7.1.2 (PRF) | 0 (open) |
| 00-0F-AC (M117) | 3 | FT authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | FT key management as defined in 12.7.1.6 (FT key hierarchy) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over IEEE Std 802.1X ~~or PMKSA caching~~ |
| 00-0F-AC | 4 | FT authentication using PSK | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association using PSK |
| 00-0F-AC | 5 | Authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | RSNA key management as defined in 12.7 (Keys and key distribution) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 0 (open) |
| 00-0F-AC | 6 | PSK | RSNA Key Management as defined in 12.7 (Keys and key distribution) using PSK | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 0 (open) |
| 00-0F-AC | 7 | TDLS | TPK handshake | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | N/A |
| 00-0F-AC | 8 | SAE authentication(M137) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | RSNA key management as defined in 12.7 (Keys and key distribution), ~~PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ or authenticated mesh peering exchange as defined in 14.5 (Authenticated mesh peering exchange (AMPE)) | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 3 (SAE) for SAE Authentication  ~~0 (open) for PMKSA caching~~ |
| 00-0F-AC(M117) | 9 | FT authentication over SAE ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | FT key management defined in 12.7.1.6 (FT key hierarchy) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 3 (SAE) for FT Initial Mobility Domain Association.  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  ~~0 (open) for FT Initial Mobility Domain Association over PMKSA caching~~ |
| 00-0F-AC | 10 | APPeerKey Authentication with SHA-256 ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | RSNA key management as defined in 12.7 (Keys and key distribution) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | N/A |
| 00-0F-AC | 11 | Authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ using a Suite B compliant EAP method supporting SHA-256 | RSNA key management as defined in 12.7 (Keys and key distribution) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256 | 0 (open) |
| 00-0F-AC | 12 | Authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ using a Suite B compliant EAP method supporting SHA-384 | RSNA key management as defined in 12.7 (Keys and key distribution) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384 | 0 (open) |
| 00-0F-AC(M117) | 13 | FT authentication negotiated over IEEE Std 802.1X ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | FT key management as defined in 12.7.1.6 (FT key hierarchy) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over IEEE Std 802.1X ~~or PMKSA caching~~ |
| 00-0F-AC(M117)(11ai) | 14 | Key management over FILS using SHA-256 and AES-SIV-256, ~~PMKSA caching,~~ or authentication negotiated over IEEE Std 802.1X(#114) | FILS key management defined in 12.11.2.5 (Key establishment with FILS authentication) | Defined in 12.11.2.5 (Key establishment with FILS authentication) using SHA-256. | 4, 5 or 6 (FILS) for FILS Authentication  0 (open) for IEEE Std 802.1X |
| 00-0F-AC(M117)(11ai) | 15 | Key management over FILS using SHA-384 and AES-SIV-512, ~~PMKSA caching,~~ or authentication negotiated over IEEE Std 802.1X(#114) | FILS key management defined in 12.11.2.5 (Key establishment with FILS authentication) | Defined in 12.11.2.5 (Key establishment with FILS authentication) using SHA-384. | 4, 5 or 6 (FILS) for FILS Authentication  0 (open) for IEEE Std 802.1X |
| 00-0F-AC(M117)(11ai) | 16 | FT authentication over FILS with SHA-256 and AES-SIV-256, ~~PMKSA caching,~~ or authentication negotiated over IEEE Std 802.1X(#114) | FT key management as defined in 12.7.1.6 (FT key hierarchy) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-256. | 4, 5 or 6 (FILS) for FT Initial Mobility Domain Association over FILS.  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over IEEE Std 802.1X ~~or PMKSA caching~~ |
| 00-0F-AC(M117)(11ai) | 17 | FT authentication over FILS with SHA-384 and AES-SIV-512, ~~PMKSA caching,~~ or authentication negotiated over IEEE Std 802.1X(#114) | FT key management as defined in 12.7.1.6 (FT key hierarchy) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384. | 4, 5 or 6 (FILS) for FT Initial Mobility Domain Association over FILS.  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over IEEE Std 802.1X ~~or PMKSA caching~~ |
| 00-0F-AC(#4198) | 18 | Reserved | Reserved | Reserved | Reserved |
| 00-0F-AC(#170) | 19 | FT authentication using PSK | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384. | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association using PSK |
| 00-0F-AC(#171) | 20 | PSK | RSNA key management(Ed) as defined in 12.7 (Keys and key distribution) using PSK | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384. | 0 (open) |
| 00-0F-AC | (11ai)(#4198) (#171)21–255 | Reserved | Reserved | Reserved | Reserved |
| Other OUI or CID | Any | Vendor-specific | Vendor-specific | Vendor-specific | Vendor-specific |

The AKM suite selector value 00-0F-AC:1 (i.e., Authentication negotiated over IEEE Std 802.1X with RSNA key management as defined in 12.7 (Keys and key distribution) ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management))~~ is the default AKM suite when the AKM Suite List field is not included in the RSNE.

NOTE (Ed)1—The selector value 00-0F-AC:1 specifies only that IEEE Std 802.1X-2010 is used as the authentication transport. IEEE Std 802.1X-2010 selects the authentication mechanism.

The AKM suite selector value 00-0F-AC:8 (i.e., SAE authentication with SHA-256 ~~or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)~~ ~~with SHA-256 key derivation~~) is used when either a password or PSK is used with RSNA key management.

NOTE (Ed)2—Selector values 00-0F-AC:1 and 00-0F-AC:8 can simultaneously be enabled by an Authenticator.

The AKM suite selector value 00-0F-AC:2 (PSK) is used when an alternate form of PSK is used with RSNA key management.

NOTE (Ed)3—Selector values 00-0F-AC:1 and 00-0F-AC:2 can simultaneously be enabled by an Authenticator.

The AKM suite selector value 00-0F-AC:11 is used only with cipher suite selector values 00-0F-AC:8 (GCMP-128) and 00-0F-AC:11 (BIP-GMAC-128). The AKM suite selector value 00-0F-AC:12 is used only with cipher suite selector values 00-0F-AC:9 (GCMP-256), 00-0F-AC:10 (CCMP-256), 00-0F-AC:13 (BIP-CMAC-256), and 00-0F-AC:12 (BIP-GMAC-256). The AKM suite selector value 00-0F-AC:13 is used only with cipher suite selector values 00-0F-AC:9 (GCMP-256), 00-0F-AC:10 (CCMP-256), 00-0F-AC:13 (BIP-CMAC-256), and 00-0F-AC:12 (BIP-GMAC-256).

NOTE 4—The usage of selector values with authentication algorithms(Ed) is defined in the Authentication algorithm numbers column of Table 9-151 (AKM suite selectors); see subclause 9.4.1.1 (Authentication Algorithm Number field).(M85)(11ai)

A PMKSA established using a given AKM selector value may be cached and used in a subsequent (Re)association as defined 12.6.10.3 (Cached PMKSAs and RSNA key management).

**CID 4087 Discussion**

The comment is

**“**As per 12.5.3.3 (P2607L59), the MIC is also encrypted along with the plaintext MPDU, so it is not possible to obtain the original MIC at this stage. The original MIC can only be obtained after CCM decryption stage. The figure 12-23 is misleading, either it should be clarified that the MIC that is fed into the CCM decryption block is encrypted MIC, or the entire encrypted MPDU (instead of MIC and data) should be passed to the CCM decryption block.”

We discussed this in recent 11md ad hoc/ teleconf where 11-20/0246r3 was presented.

Generally speaking these figures are only informative and there are specs from IETF (RFC 3610) and NIST GCM specification as to how the MIC is handled. With AEAD schemes such as CCM and GCM, the MIC is not exposed out of cipher text, but…

Mark Rison came up the following resolution – we can discuss

TGm Editor: Change as follows

Change "MIC" to "Encrypted MIC"

in Figure 12-23—CCMP decapsulation block diagram (also "Data" -> "Encrypted data" on the left),

Figure 12-29—GCMP decapsulation block diagram (also "Data" -> "Encrypted data" on the left),

Figure 12-18—CCMP encapsulation block diagram (also "Data" -> "Plaintext data" on the left and "Data" -> "data" on the right),

Figure 12-27—GCMP encapsulation block diagram (also "Data" -> "Plaintext data" on the left and "Data" -> "data" on the right),

and at 2608.51½, 2609.9½, 2618.29½.

Lowercase "Nonce" in Figure 12-18—CCMP encapsulation block diagram,

"Nonce" and "Header" in Figure 12-27—GCMP encapsulation block diagram,

"Nonce" in Figure 12-23—CCMP decapsulation block diagram,

"Nonce" and "Header" in Figure 12-29—GCMP decapsulation block diagram,

and ask the Editors to sharpen up Figure 12-23—CCMP decapsulation block diagram.