IEEE P802.11
Wireless LANs

|  |
| --- |
| Comment Resolutions on REVmd draft 4.0 |
| Date: September 8, 2020 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Menzo Wentink | Qualcomm | Utrecht, The Netherlands | +31-65-183-6231 | mwentink@qti.qualcomm.com |

Abstract

This document contains comment resolutions for REVmd draft 4.0, addressing the following CIDs:

1. 5013, 5069, 5070, 5014, 5034, 5035, 5036

5013 (4725): max retry limit

5014 (4761): dec operator

The baseline for this document is Draft P802.11REVmd D4.0.

|  |  |  |  |
| --- | --- | --- | --- |
| **CID Identifiers** | **Comment** | **Proposed Change** | **Proposed Resolution** |

|  |  |  |  |
| --- | --- | --- | --- |
| 501310.3.4.41753.45Wentink, MenzoBased on CID 4725 on draft 3.0. | At 1753.45 the lifetime is missing.The retry limit MIB variables currently have a maximum of 255, which is too small. 255 retries of an RTS may take only in the order of 10 ms, but a period of temporary interference might take much longer than that, and it is not always smart to discard a packet that soon.(This comment is related to CID 4725 on REVmd draft 3.0.) | 1753.45 add "or lifetime" after "retry limit", to read"Error recovery shall be attempted by retrying transmissions for frame exchange sequences that the initiating STA infers have failed. Retries shall continue, for each failing frame exchange sequence, until the transmission is successful, or until the relevant retry limit or lifetime is reached, whichever occurs first. A STA shall maintain an SRC and an LRC for each MSDU or MMPDU awaiting transmission. These counts are incremented and reset independently of each other."4124.62 change "255" to "65535"(in dot11ShortDEIRetryLimit).4125.30 change "255" to "65535" (in dot11UnsolicitedRetryLimit).4164.4 change "255" to "65535"(in dot11ShortRetryLimit). | Accepted |

|  |  |  |  |
| --- | --- | --- | --- |
| CID 507010.3.4.41753.44RISON, Mark | Followup to CID 4725. Packets can fail to be transmitted because they hit their retry limit or because they hit their lifetime limit | In the referenced subclause change"Retries shall continue, for each failing frame exchange sequence, until the transmission is successful, or until the relevant retry limit is reached, whichever occurs first."to"Retries shall continue, for each failing frame exchange sequence, until the transmission is successful, until the relevant retry limit is reached, or until the relevant lifetime limit is reached, whichever occurs first." |  |

|  |  |  |  |
| --- | --- | --- | --- |
| CID 5069C.3.RISON, Mark | Followup to CID 4725. We don't have unlucky packets, just unlucky connections. The next packet to a given peer is just as likely to fail as the previous given the same TXVECTOR. It makes sense to have a Tx lifetime per AC as delivering a late real-time packet is a worthless whereas best effort frames should be retried until the link times out, but for a given AC there's often no point hitting a retry limit and then just moving on to the next packet to that AC+destination. It should be possible to transmit packets up to a lifetime, without worrying about hitting a retry limit per se. | In C.3 change "255" to "65535" in (the upper limit for) dot11ShortDEIRetryLimit, in dot11UnsolicitedRetryLimit, dot11ShortRetryLimit. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| CID 503410.47.62082.1RISON, Mark | Follow-up to CID 4625.The "(inclusive)" is confusing because the A[b:c] operators inclusivity is already specified in Subclause 1.5 | Delete the "(inclusive)" |  |

|  |  |  |  |
| --- | --- | --- | --- |
| CID 50351.5151.63RISON, Mark | Follow-up to CID 4625.Calling the operator dec() is confusing because it's not generating a decimal per se, it's just generating a number, which you can choose to encode in whatever form you want (typically in binary, actually, when it goes into a field) | Change "dec(" to "int (" throughout | AcceptedNote to editor: These changes are made as part of the resolution for CID 5014. |

|  |  |  |  |
| --- | --- | --- | --- |
| CID 50361.5151.63RISON, Mark | Follow-up to CID 4625.The A[b:c] operator is not defined in 1.5; instead its definition is repeated in multiple places, possibly incompletely.The definition of the dec(A[b:c]) operator does not cover the use in Table 10-15 (Settings for the TXVECTOR parameter PARTIAL\_AID for an NDP), where the argument to the dec() is not just an A[b:c].There is lots of repetition of the rules for ordering in MAC addresses | Make the changes shown under "Proposed additional changes:" under CID 4625 in 20/0435r14 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 50141.5152.38Wentink, MenzoBased on CID 4761 from draft 3.0:CID 476110.191816.41Sun, Li-Hsiang | The formula dec(BSSID[39:47]) is1) inconsistent with the definition on p152: "dec(A[b:c]) is the cast from binary to decimal operator, where c is the least significant bit in binary value [b:c]". Bit 47 should be MSB not LSB2) inconsistent with NOTE1 on p1817, where bit 47 is indeed calculated as MSB(This comment is related to CID 4761 on REVmd draft 3.0.) | Either fix all the dec() on p1816 and p1817, or revise the definition on p152 and NOTE1 on p1817 | RevisedMake changes as specified in <this document> under CID 4761, which are changes in the direction suggested by the commenter. |

**Discussion**

From 1.5 (Terminology for mathematical, logical, and bit operations):

dec(A[b:c]) is the cast from binary to decimal operator, where c is the least significant bit in binary value [b:c].

From 9.2.2 (Conventions):



From 9.4.2.45 (Multiple BSSID element):

This clause has the following example:

8c:fd:0f:7f:1e:**f5** & ZERO[45:47] = 8c:fd:0f:7f:1e:**f0** (A0:A1:A2:A3:A4:A5)

So f5 is changed to f0 by setting bits [45:47] to 0. This implies that bit 47 is considered the LSB of A5, contrary to the address convention in 9.2.2, which has the LSB of A5 numbered as bit 40, but it would be in line with the current definition of the dec operator.

Therefore, in Multiple BSSID, the bit numbering inside the address octets is assumed to be MSB to LSB (lower bit number to higher bit number inside the address octets).

In this definition, bit 39 is the LSB of A4, which is appended as an MSB to bits [40:47] of A5, implying that in Multiple BSSID, A0-A5 are assumed to be ordered MSO to LSO.

(MSO is the most significant octet (A0) and LSO is the least significant octet (A5).)

From 10.19 (Group ID and partial AID in VHT and CMMG PPDUs):

The Partial AID for a packet addressed to the AP is defined as dec(BSSID[39:47]) (see Table 10-13).

There is an example for BSSID = 00-21-6A-AC-53-52 (A0-A1-A2-A3-A4-A5), for which the Partial AID for packets to the AP is 164.

So the 9 bits [39:47] have decimal value 164:

dec 164 = 0b010100100 (MSB to LSB) = 001001010 (LSB to MSB, numbered [39:47]).

A4 = 0x53 = 0b01010011 = 1100101***0*** (LSB to MSB, numbered 32:39)

A5 = 0x52 = 0b01010010 = **01001010** (LSB to MSB, numbered 40:47)

The LSB to MSB ordering is as defined in 9.2.2, where the LSB is on the left, has the lower bit number, and is transmitted first.

164 is obtained as follows (in 9.2.2 LSB to MSB notation):

 1100101***0*01001010**

Therefore, in Partial AID, the bit numbering inside the address octets is assumed to be LSB to MSB (lower bit number to higher bit number inside the address octets).

In this definition, bit 39 is the MSB of A4, which is appended as an LSB to bits [40:47] of A5, implying that in Partial AID, A0-A5 are assumed to be ordered LSO to MSO.

(LSO is the least significant octet (A0) and MSO is the most significant octet (A5).)

Therefore, both the bit numbering and the octet significance ordering appear to be different in the definitions of Multiple BSSID and Partial AID.

The intent of the proposed changes is to remove this inconsistency, by clarifying the *dec* and [b:c] operators and by removing the dependency of the Multiple BSSID description on a bit numbering. The Multiple BSSID definition is rewritten as an operation on the hexadecimal representation of the MAC address. The Multiple BSSID definition is also limited to an operation only on A5, because the BSSID index can not be larger than 255 (see 9.4.2.73 (Multiple BSSID-Index element)). The intent is to not change existing implementations.

From 11ax draft 7.0, which modified 11.10.14 (Multiple BSSID set) as follows:

— The set has a maximum range of 2n for at least one n, where ~~1 ≤ n ≤ 46~~

• 1 ≤ n ≤ 8 if dot11MultiBSSIDImplemented is true

• 1 ≤ n ≤ 46 if dot11MultiBSSIDImplemented (if present) is false and dot11RMMeasurementPilotActivated

is nonzero

Endianness:

little endian: 6 occurrences

big endian: 11 occurrences

significant octet: 20 occurrences

From 3.1 (Definitions)

**big endian:** The concept that, for a given multi-octet numeric representation, the most significant octet has the lowest address.

**little endian:** The concept that, for a given multi-octet numeric representation, the least significant octet has the lowest address.

***--- Start of changes for CID 4761 ---***

**1.5 Terminology for mathematical, logical, and bit operations**

***151.63 change as shown***

*A*[*b*:*c*] is the bit string consisting of bits *b* to *c* of *A*, where bit 0 of the output is the value of bit *b*. This operator is not used in this standard with *b* larger than *c*.

*int*(*S*) is the decimal value of bit string *S*, where bit 0 of *S* is the least significant bit, using the IEEE 802.11 bit conventions from 9.2.2 (Conventions).

NOTE—The *int* operator applied to a (portion of a) MAC address implies that octet 0 of the MAC address is the least significant octet under this operator.

**9.2.2 Conventions**

***782.62 change as shown***

MAC addresses are assigned as ordered sequences of bits. The Individual/Group bit is always transferred first and is bit 0 of the MAC address. Bit 47 of the MAC address is always transferred last. This is illustrated in Figure 9-1 (Representation of a 48-bit MAC address). Also see clause 8 of IEEE Std 802-2014.

A MAC address can be represented using hexadecimal values separated by hyphens, as described in IEEE Std 802.

***783.26 delete***

MAC\_ADDR[*b*:*c*] represent bits *b* to *c* inclusive of MAC address MAC\_ADDR.

**9.4.2.45 Multiple BSSID element**

***1163.32 change as shown***

The MaxBSSID Indicator field contains a value assigned to *n*, where 2*n* is the maximum number of BSSIDs in the multiple BSSID set, including the reference BSSID (see 11.10.14 (Multiple BSSID set)). The maximum value of *n* is 8. The actual number of BSSIDs in the multiple BSSID set is not explicitly signaled. BSSID(i) corresponding to the ith BSSID in the multiple BSSID set is derived as follows:

A0-A1-A2-A3-A4-A5 = Reference BSSID

B = A5 mod 2*n*

A5(i) = A5 – B + ( (B + i) mod 2*n* )

BSSID(i) = A0-A1-A2-A3-A4-A5(i)

NOTE 1—For example, for *n* = 3 and Reference BSSID = 8c-fd-0f-7f-1e-f5:

A5 = f5

B = 5

A5(5) = f2 and BSSID(5) = 8c-fd-0f-7f-1e-f2

A5(2) = f7 and BSSID(2) = 8c-fd-0f-7f-1e-f7

NOTE 2—This definition uses the hexadecimal address representation defined in IEEE std 802.

NOTE 3—This definition implicitly assumes that A0 is the most significant octet (although the definition is only on A5).

NOTE 4—The BSSID index as defined in 9.4.2.73 (Multiple BSSID-Index element) cannot be larger than 255, which effectively limits *n* to 8.

NOTE 5—Members of a Multiple BSSID set have the same value for A0, A1, A2, A3, A4, and A5(i) – ( (B + i) mod 2*n*.

**10.19 Group ID and partial AID in VHT and CMMG PPDUs**

***1808.3 change as shown***

The partial AID is a nonunique STA identifier defined in Table 10-13 (Settings for the TXVECTOR parameters GROUP\_ID and PARTIAL\_AID for VHT STAs) and Table 10-14 (Settings for the TXVECTOR parameter PARTIAL\_AID for CMMG STAs). The partial AID is carried in the TXVECTOR parameter PARTIAL\_AID of a VHT SU PPDU or a CMMG PPDU and is limited to 9 bits.

***1808.12 change as shown***

BSSID[*b*:*c*] and RA[*b*:*c*] are defined as *A*[*b*:*c*] in 1.5 (Terminology for mathematical, logical, and bit operations) and 9.2.2 (Conventions).

***1809.49 change as shown***

As an example of the GROUP\_ID and PARTIAL\_AID setting, consider the case of a BSS with BSSID 00-21-6A-AC-53-5232 that has as a member a non-AP STA assigned AID 5. In VHT PPDUs sent to an AP, the GROUP\_ID is set to 0 and the PARTIAL\_AID is set to 164. In VHT PPDUs sent by the AP to the non-AP STA associated with that AP, the GROUP\_ID is set to 63 and PARTIAL\_AID is set to 229.

As an example of the PARTIAL\_AID setting, consider the case of a BSS with BSSID 00-21-6A-AC-53-52 that has as a member a non-AP CMMG STA assigned AID 5. In CMMG PPDUs sent to an AP, the PARTIAL\_AID is set to 165. In CMMG PPDUs sent by the AP to the non-AP STA associated with that AP, the PARTIAL\_AID is set to 37.

**10.21 Group ID, partial AID, Uplink Indication, and COLOR in S1G PPDUs**

***1812.30 change as shown***

The S1G partial AID is a nonunique identifier of an S1G STA as defined in Table 10-14 (Settings for the TXVECTOR parameter PARTIAL\_AID for NDP frames) and Table 10-15 (Settings for the TXVECTOR parameter PARTIAL\_AID for non-1 MHz PPDUs and non-NDP frames). The partial AID is carried in the TXVECTOR parameter PARTIAL\_AID of an S1G SU PPDU with the TXVECTOR parameter CH\_BANDWIDTH set to CBW2, CBW4, CBW8, or CBW16 and is limited to 9 bits.

***1814.14 change as shown***

As an example of the PARTIAL\_AID setting, consider the case of a BSS with BSSID 00-21-6A-AC-53-52 that has as a member a non-AP S1G STA assigned AID 5. In an NDP frame sent by the non-AP S1G STA to the S1G AP, the PARTIAL\_AID is equal to 165. In an NDP frame sent by the S1G AP to the non-AP S1G STA associated with that S1G AP, the PARTIAL\_AID is equal to 229. In a non-1 MHz S1G PPDU that is not an NDP frame and that is sent by the non-AP S1G STA to the S1G AP, the PARTIAL\_AID is set to 165. In a non-1 MHz S1G PPDU that is not an NDP frame that is sent by the S1G AP to the non-AP S1G STA associated with that S1G AP, the PARTIAL\_AID is set to 37.

**10.47.6 NDP Paging Setup**

***2081.60 change as shown***

If the Direction field of the NDP Paging frame is equal to 1, the subfields of the APDI field of the NDP Paging frame shall be set as follows:

— The PTSF subfield is set to TSF[Partial TSF Offset + 4 : Partial TSF Offset + 11], where TSF is the 8-octet value of the TSF timer and Partial TSF Offset is the value of the Partial TSF Offset field in the NDP Paging Request.

**11.10.14 Multiple BSSID set**

***2310.12 change as shown***

A multiple BSSID set is characterized as follows:

— All members of the set use a common operating class, channel, channel access functions, and antenna connector.

— The set has a maximum range of 2n for at least one n, where 1 ≤ n ≤ 8.

— Members of the set have the same value for BSSID address octets A0, A1, A2, A3, A4, and A5 – ( (B + i) mod 2*n*, where B = A5 mod 2*n*.

— All BSSIDs within the multiple BSSID set are assigned in a way that they are not available as MAC addresses for STAs using a different operating class, channel or antenna connector.

***Change all remaining occurrences of "dec(" to "int("***

(The remaining ones are in 10.21 (Group ID, partial AID, Uplink Indication, and COLOR in S1G PPDUs)).

***--- End of changes for CID 4761 ---***