IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LB193 CID 1003 topic | | | | |
| Date: 2013-04-29 | | | | |
| 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 1003, 277, 278, 280, 283, 285 of TGmc LB 193.

**Revision Notes**

**R0:**

Initial

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Commenter Name** | **P.L** | **SC** | **Comment** | **Proposed Change** | **Resolution** |
| 1003 | Matthew Fischer | 483.26 | 8.3.3.2 | Why should it be "optional" to include the element? "The Extended Capabilities element is optionally present if any of the fields in this element are nonzero." - Does a STA keep checking beacons to see if anything is different? And if it does, and the ExCap IE is present one time, and then is not present another time, then does the capability disappear with the IE? Or is the capability still there? - see also 8.3.3.6 assoc response, for example. | Change "is optionally present" to "is present" in this subclause and in subclauses describing other mgmt frame formats that include the extended cap IE. | Reject – the word “optionally” provides enough vaguery in the meaning of the absence of the IE to warrant caution on the part of a receiver in the absence of the IE and the resulting ambiguity, while initially disturbing ends up providing the AP with an opportunity to reduce Beaconing overhead. With the word in place, the AP can reduce Beacon overhead by not always including the ExCap IE in the Beacon. If the word “optionally” is removed, then the AP must always include the IE in every Beacon as long as at least one bit is set in the ExCap IE. |

**Discussion**

Suppose that the commenter’s suggested change is made.

In that case, the language would be:

***The Extended Capabilities element is present if any of the fields in this element are nonzero.***

Now we can definitively say that if the element is absent, all bits are zero.

Whereas, with the original language, if the element is NOT present, one cannot safely say that all of the bits are zero – that is, with the word “optionally” in the sentence, it is ok to optionally NOT include the IE even if some bits are “1”. Therefore, the existing language provides the option for an AP to NOT include the IE in every Beacon, potentially saving some airtime by allowing the AP to only transmit the IE in some, but not all Beacons.

**Proposed Resolution:**

Reject.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 277 | Matthew Fischer | 837.01 | 9.3.3 | I believe that CW should be reset for every successful exchange, regardless of frame type.  The intent of CW is to control the offered load to the medium at each STA and the offered load to the medium at each STA should depend on the current total offered load on the medium, that is the sum of all offered loads of all participating STAs because the intent of controlling the offered load to the medium is to reduce the collision rate to allow some traffic to be successfully transmitted onto the medium.  The occurrence of a collision is a rough guage of the offered load on the medium, and the rate of transmission failures is a rough guage of the rate of collisions, and therefore, in the protocol, a transmission failure is the mechanism that determines when the offered load should be reduced by increasing CW.  A transmission success causes a CW reset in the standard - but if CW was intended to set the offered load of a STA to a value that is appropriate vis-a-vis the current total offered load of the medium, then the CW achieved after a transmission failure or two should remain unaltered when a success finally occurs - i.e. CW should not reset after a success, but instead, perhaps CW should remain at whatever value it has achieved to that point. However, because the offered load on the medium is dynamic, a STA should not keep the CW at the increased value, so a better algorithm might be to reduce CW by one step after each successful transmission, instead of resetting it to CWmin.  But, the algorithm that is in the 802.11 protocol is based on the 802.3 backoff where the reset to CWmin after success was appropriate due to a difference in exactly how the backoff is counted in 802.3 vs 802.11 - namely, the allowance in 802.3 to count backoff when the medium is BUSY. I believe that no one working on 802.11 at the time that the backoff mechanism was created really thought too hard about this difference in backoff with respect to this particular problem - the difference was considered and did affect other aspects of the backoff mechanism in 802.11, but not this one. A reset to CWmin is appropriate when backoff is not suspended during BUSY. It is inappropriate when backoff is suspended.  But this fact was missed when 802.11 was written, so instead of maintaining CW, we have CW falling to its low CWmin value after a successful transmission, and the indication of a successful transmission is when the MAC acknowledgement is received to an MSDU or MMPDU or part thereof.  In any case, I can determine no reason to discriminate any particular type of frame from another when determining if a transmission was successful and therefore, that the offered load at this STA should be adjusted. Any completed exchange will verify that a collision has NOT occurred, and if a collision has not occurred, a STA should be allowed to presume that the offered load on the medium is reduced (not my choice, but 802.11's choice) so that the STA may use a reduced CW (which again, is not really the best thing to do, but that is what the protocol says to do).  So - my comment is that the successful return of any expected response to any transmitted should be considered to have satisfied the condition that allows the CW to be reset to CWmin because the rationale for allowing the reset in the existing cases has been satisfied in the additional, currently unspecified cases.  I.e. the existing language here indicates that the CW can only be reset in response to a successful MSDU or MMPDU or part thereof, instead of including any frame, such as, for example, an RTS, PS-Poll, BAR or BA. | Allow the reset of CW to CWmin after any successful frame exchange that included a response to an initial transmission, even when no MSDU or MMPDU is included in the transaction. | Revise – TGm editor to make changes shown in 11-13/xxxxry under the section for CID 277. In general these changes are in agreement with and adopt the suggested change and also change the resetting of the SSRC and SLRC. |

**CID 277**

**Discussion**

But wait! There’s more!

It turns out that there is another issue – the STA retry counters should be reset to zero whenever they reach their respective limits. I.e. the SSRC and SLRC cause a CW reset when they reach limits dot11ShortRetryLimit and dot11LongRetryLimit, but the counters themselves are not currently allowed to be reset until a frame is successfully transmitted. This effectively means that these counters can continue counting PAST the limits!

**Proposed changes**

***TGm editor, change the 3rd paragraph of 9.3.3 Random backoff time as follows:***

**9.3.3 Random backoff time**

The CW shall be reset to aCWmin after every successful attempt to transmit a frame, when SLRC reaches dot11LongRetryLimit, or when SSRC reaches dot11ShortRetryLimit. The SSRC shall be reset to 0 when a CTS frame is received in response to an RTS frame, when a BlockAck frame is received in response to a BlockAckReq frame, when an (#1198)Ack frame is received in response to the transmission of a frame in a PSDU(#358) of length greater than dot11RTSThreshold containing all or part of an MSDU or MMPDU, when a frame with a group address in the Address1 field is transmitted or when the SSRC reaches dot11ShortRetryLimit. The SLRC shall be reset to 0 when an (#1198)Ack frame is received in response to transmission of a frame containing all or part of an MSDU or MMPDU of , when a frame with a group address in the Address1 field is transmitted or when the SSLC reaches dot11LongRetryLimit.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 278 | Matthew Fischer | 410.01 | 8.3.1.9 | The highest indicated modulation and stream combinations for some PHYs result in phy rates that will reduce throughput efficiency to exceedingly low levels if the maximum block ack window size is not allowed to increase beyond the existing 64. | Increase the maximum allowed MPDUs in the Block Ack frame to 256 by creating a new form of Block Ack that supports a longer BA window and a longer BA bitmap. | Revise – generally accept the proposed changes tp allow a larger BA window size – see 11-13-xxxxry. |

**CID 278**

**Discussion**

Sounds great!

See 11-13-xxxxry.

**Proposed changes**

See 11-13-xxxxry.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 280 | Matthew Fischer | 1211.01 | 11.4.3.4.4 | Part e) contains an incorrect reference - it says "priority" - now, from the MSDU perspective, it is a priority that is passed across the MAC interface, but at the receiving side of an exchange, the priority has been translated into a TID value and that is the only thing that the recipient has on hand to determine the correct PN space for comparison purposes - so i believe that TID should be the item of reference here and not priority. Throughout clause 11, only priority is used, and in many cases, priority is the correct term, because the reference is to the parameter of the MAC SAP. For example, in 11.4.2.1.2, there are several instances of "priority field", but correctly, this should be "priority parameter". Note that in 11.4.3.3.1, there is a line that includes fields of an MPDU, of which one named field is "priority" - again, no such field exists. 11.4.3.3.4 does it correctly. There are not too many instances to fix. | Examine all instances of "priority" within clause 11 and fix those that incorrectly refer to a field that does not exist, and fix those that refer to a parameter to say parameter instead of something else. Change priority to TID as appropriate. | Revise – tgm editor to make changes shown in 11-13/xxxxry under the heading for CID 280. |

**CID 280**

**Discussion**

Note that within 11.4.2.3.2, we find:

**11.4.2.3.2 Motivation for the TKIP MIC**

*The Priority field refers to the priority parameter of the MA-UNITDATA.request primitive.*

And this occurs when the construction of the MIC is being described and therefore, an unambiguous, correct reference does exist where the actual generation of the MIC is being discussed.

However, within the CCMP subclauses, there is at least one reference to the Priority field of an MPDU, and such field does not exist.

There is a description of how to determine the value of the priority field of the NONCE, and it can be genericized for use by the rest of the subclause.

**Proposed changes**

***TGm editor, make the changes shown:***

**11.4.2.1.2 TKIP cryptographic encapsulation**

a) TKIP MIC computation protects the MSDU Data field and corresponding SA, DA, and Priority parameter values. The computation of the MIC is performed on the ordered concatenation of the SA, DA, Priority, and MSDU Data parameters. The MIC is appended to the MSDU Data field. TKIP discards any MIC padding prior to appending the MIC.

**11.4.2.1.3 TKIP decapsulation**

d) The MIC verification step recomputes the MIC over the MSDU SA, DA, Priority, and MSDU Data parameters (but not the TKIP MIC field). The calculated TKIP MIC result is then compared bit-wise to the received MIC.

**11.4.2.3.3 Definition of the TKIP MIC**

Michael operates on each MSDU including the Priority parameter, 3 reserved octets, SA parameter, and DA parameter. An MSDU consists of octets *m*0...*mn*–1 where *n* is the number of MSDU octets, including SA, DA, Priority, and Data parameters. The message is padded at the end with a single octet with value 0x5a, followed by between 4 and 7 zero octets. The number of zero octets is chosen so that the overall length of the padded MSDU is a multiple of four. The padding is not transmitted with the MSDU; it is used to simplify the computation over the final block. The MSDU is then converted to a sequence of 32-bit words *M*0 ...*MN*–1, where *N* = (*n*+5)/4.(#272) By construction, *MN*–1 = 0 and *MN*–2 ≠0.

**11.4.3.3 CCMP cryptographic encapsulation**

**11.4.3.3.1 General**

c) Construct the CCM Nonce block from the PN, A2, and the priority value of the MPDU where A2 is

MPDU Address 2. If the Type field of the Frame Control field is 10 (Data frame) and there is a QC field present in the MPDU header, the priority value of the MPDU is equal to the value of the QC TID (bits 0 to 3 of the QC field). If the Type field of the Frame Control field is 00 (Management frame), and the frame is a QMF, the priority value of the MPDU is equal to the value in the ACI subfield of the Sequence Number field. Otherwise, the priority value of the MPDU is equal to the fixed value 0

**11.4.3.3.4 Construct CCM nonce**

The Priority subfield of the Nonce Flags field shall be set to the priority value of the MPDU.(11ae)

**11.4.5.4.4 PN and replay detection(11ad)**

d) A receiver shall discard any data MPDU that is received with its PN less than or equal to the value of the replay counter that is associated with the TA and priority value of the received MPDU. If dot11RSNAProtectedManagementFramesActivated is true, a receiver shall discard any group addressed MMPDU that is received with its PN less than or equal to the value of the replay counter associated with group addressed MMPDUs. If dot11RSNAProtectedManagementFramesActivated is true, the receiver shall discard any individually addressed robust MMPDU that is received with its PN less than or equal to the value of the replay counter associated with the TA of that individually addressed MMPDU. When discarding a frame, the receiver shall increment by 1 the value of dot11RSNAStatsGCMPReplays for data frames or dot11RSNAStatsRobustMgmtGCMPReplays for robust management frames.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 283 | Matthew Fischer | 859.01 | 9.7.6.5 | ACK (and other control response frames) are supposed to be sent at rates as specified in this subclause. Generally speaking, this subclause prescribes ACK transmission at a Basic Rate, which is, for eliciting frames that are transmitted at higher rate or MCS values, typically a rate that is lower than the rate or MCS of the eliciting frame by at least two increments. When the eliciting frame is at one of the lower rates/MCS, then the ACK will often be at the same rate/MCS. Normally, this is ok - reliability of the ACK transmission is good or better than the eliciting frame because the eliciting frame usually has significantly more bits than the ACK. But if the eliciting STA has a higher TX power, then the rate for the ACK can be too high and is very likely to fail. It would be nice if the eliciting STA can signal to the responder that a lower rate/MCS will be needed. This ensures a receiveable ACK and it allows the eliciting STA to correctly compute the ACK duration for DUR field use. | Allow a control response frame to be sent at a rate/MCS lower than otherwise allowed when there is a power difference or for other reasons. Create a mechanism that allows the eliciting STA to indicate the preferred response rate/MCS. E.g. signal in the eliciting frame, a value of step down in rate/MCS from the normally computed value for the response frame. | Revise – tgm editor to make changes shown in 11-13/xxxxry under the heading for CID 283 which allow for a reduced response frame MCS selection |

**CID 283**

**Discussion**

See 11-13-xxxxry

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 285 | Matthew Fischer | 878.01 | 9.19.2.5 | In item b) The success or failure should not matter here - there should always be a backoff after finishing a TXOP. And there is the question of whether one should wait for the expiration of the TXNAV value first, but i think that part is covered. | Change "was successful" to "was completed" | Accept. |

**Discussion**

In the original sentence:

*The final transmission by the TXOP holder initiated during the TXOP for that AC was successful and the TXNAV timer has expired.*

the language attempts to differentiate the successful vs failing cases by mentioning “successful” but having a non-zero TXNAV timer is a sufficient condition and the two cases should not be separated anyway. Consider:

CASE 1:

The final frame of a TXOP was successful, included a non-zero NAV that covered MORE than the subsequently successfully received BA, and there are no more transmissions. In this case, the transmitter must wait until the end of the TXNAV timer to begin backoff in order to be fair to receivers that loaded their NAV. The sentence correctly instructs the transmitter as to the preferred behaviour.

CASE 2:

A frame of a TXOP was successful, included a non-zero NAV that covered MORE than the subsequently successfully received BA, and there are additional, failing transmissions that follow the successful exchange and the additional transmissions use all of the remaining TXNAV time. The transmitter can begin backoff after the last failing transmission, and this condition is described in text below the list of conditions, but that text uses “may”, or one might consider this to be a completely separate case:

*In addition, the backoff procedure may be invoked for an EDCAF when the transmission of a non-initial frame by the TXOP holder fails.*

CASE 3:

A frame of a TXOP was successful, included a non-zero NAV that covered MORE than the subsequently successfully received BA, and there are additional, failing transmissions that follow the successful exchange and the additional transmissions do NOT use all of the remaining TXNAV time. Ideally, the transmitter should wait for the end of TXNAV to perform backoff, but the sentence does NOT capture this condition, because the final transmission of the TXOP was NOT successful.

Nowhere, is there a clear statement that the EDCAF must perform backoff after the completion of the TXOP.

Text as it would appear after the proposed modification:

*The final transmission by the TXOP holder initiated during the TXOP for that AC was ~~successful~~completed and the TXNAV timer has expired.*

One question remains – does “transmission completed” mean that the transmitter has completed the wait for timeout for an ACK if it was not received?

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 208 | Mark RISON |  |  | The spec appears to require that if TPMFs have a separate SN counter on rx, one should remember the last SN used to ensure it is not used for non-TPMFs, but does not appear to require this in the other direction | Require it in the other direction too | ? |

**Discussion**

I assume that direction means transmitter vs receiver.

The reason to cache the last SN per RA at the transmitter is to avoid sending two distinctly different MPDUs with the same SN. This possibility arises when a single SN number space is shared by more than one RA. This caching is only required on the part of the transmitter.

The reason to cache the last SN per RA at the receiver is to detect duplicates. The text describes the duplicate detection mechanism.

It is not clear what the commenter believes is missing.

**References:**