IEEE P802.11
Wireless Access Method and Physical Layer Specifications

Title: **Making Hash, or**
A proposed change to Sections 4.1.5 and 5.2.12 in the draft IEEE p802.11-93/20b1

Author:
Barry A. Dobyns
Photonics Corporation
2940 North First Street
San Jose, CA 95134
408-955-7930
408-955-7940 fax
76527,266@compuserve.com or dobyns@acm.org

Abstract:

What's Wrong:
The editor's note in section 4.1.5 at the top of page 45 of IEEE p802.11-93/20b1 reads:
[Editor's note (GE) Hash needs to be defined]

The editor's note in section 5.2.12 near the bottom of page 65 of IEEE p802.11-93/20b1 reads:
[Hash algorithm should be defined. JES]

The editor's note in section 5.2.12 at the bottom of page 65 of IEEE p802.11-93/20b1 reads:
[Do we need to specify the depth of the MPDU_ID_CACHE to achieve interoperability??? Implication: If not, don't overspecify. JES]

How To Fix It:
Supply ~~a~~ definition, along with proposed text, like the two alternatives in this document.

Motions:
Alternative !:
Resolved, that the proposed text changes in 11-94/179, Alternative 1, "Additive MPDU-ID Hash" be incorporated into the draft standard IEEE p802.11-93/20b1 sections 4.1.5 and 5.2.12 in it's next revision by the editors.

Alternative 2:
Resolved, that the proposed text changes in 11-94/179, Alternative 2, "CCITT-CRC16 MPDU-ID Hash" be incorporated into the draft standard IEEE p802.11-93/20b1 sections 4.1.5 and 5.2.12 in it's next revision by the editors.

Proposed Change: Alternative 1 "Additive MPDU-ID Hash"

The proposed text is shown below, with change bars, in the affected section from the draft standard, IEEE p802.11-93/20b1. This alternative offers a simple MPDU-ID hashing function which exhibits good randomness by exploiting design-time knowledge about the sources of randomness of the hashed keys. It is also less complex than Alternative 2.

4.1.5 MPDU ID/ConnID

This 16 bit field shall contain the MPDU-ID for asynchronous frames and the Connection ID for time-bounded frames. The MPDU-ID value is a specific ~~the~~ hash code derived from ~~of~~ the 3 octet NID field (as defined in section 4.1.6.1), the 6 octet source address (as defined in section 4.1.6.3) and the ~~a~~ 1 octet sequence number.

These source data are divided up into octets, which are numbered as shown below in figure 4-xx.

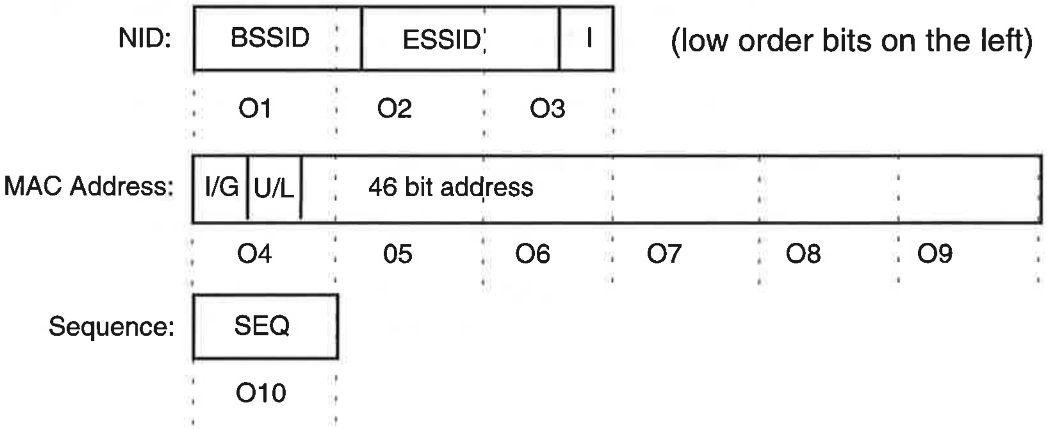

Figure 4-xx Octet Assignment for MPDU-ID Procedure

Integers are constructed from the ten octets in the source fields as shown in figure 4-xx.

Integer I1	O2	O3
Integer I2	O10	O1
Integer I3	O4	O5
Integer I4	O6	O7
Integer I5	O8	O9

(all integers shown LSB first)

Figure 4-xx, Integer Construction for MPDU-ID Procedure

The MPDU ID shall be defined as:

MPDU ID = I1 + I2 + I3 + I4 + I5

Time-Bounded ConnectionIDs station are assigned when the station requests TB bandwidth in a management Request message.

5.2.12. Duplicate Detection and Recovery

Since MAC-level acknowledgments and retransmissions are incorporated into the protocol, there is the possibility that a frame may be received more than once. Such duplicate frames shall be filtered out within the destination MAC.

Duplicate frame filtering is facilitated through the inclusion of an **MPDU ID** field within the individual frames of an MPDU, including the DATA and ACK frames. Frames which are part of the same MPDU shall have the same ID, and different MPDUs will (with a very high probability) have a different ID.

The MPDU ID is a 16 bit hash of the 2 octet Network ID field, 6 octet source address and a 1 octet sequence number maintained by the source STA, as defined in section 4.1.5. ~~. The hashing of this information into a smaller field reduces overhead, particularly within ACK frames.~~

A destination STA shall reject a frame which has the RETRY bit set in the CONTROL field as a duplicate if it receives one which matches a value of recent MPDU IDs kept in the MPDU_ID_CACHE. The MPDU_ID_CACHE shall keep the no less than 16 of the most recent ~~last X~~ MPUD IDs of directed frames addressed to that STA, multicast frames received by that STA, or broadcast frames received by that STA on a FIFO basis for the purpose of comparison with the most recent MPDU ID. [1]

There is the small possibility that a frame will be improperly rejected due to such a match; however, this occurrence would be rare and would simply result in a lost frame similar to an FCS error.

[1] In particular, a STA is not required to keep a MPDU ID cache of frames which may be received but which are not addressed to that station. A frame is addressed to a STA when the NID field and the DST field both match the NID and MAC Address of the STA. The implementation of MPDU ID cache is not constrained to any maximum depth. The implementation of MPDU ID cache is not constrained to contain only MPDU ID addressed to the STA. The implementation of MPDU ID cache is not constrained to contain only the MPDU ID.

Rationale for Alternative 1

Quality of the MPDU ID

In any given installation, it may be expected that the MAC addresses will reflect the manufacturer's ID of only a small number of vendors, therefore O8 and O9, as shown above are the most likely components to vary from one MPDU ID to the next, with O9 having the most variablilty.

In practice, an implementation of a MAC is likely to reject frames which do not have the same NID as that STA at a very low level, and will not cache the MPDU ID for those frames. Therefore, the NID components of MPDU ID, O1, O2 and O3 are highly unlikely to vary.

The sequence number is also extremely likely to vary as a component of the MPDU ID from one frame to the next.

The MPDU ID is constructed in such a way as to distribute the components which vary with the highest probability across all the bits of the MPDU ID.

Complexity of the MPDU ID

In addition, the implementation of the MPDU ID has very low complexity measure, and should be easy to implement in either hardware or software.

This MPDU ID hash has about the lowest possible complexity measure of any alternative possible, excepting those that simply truncate and ignore bits from the constituent keys altogether.

MPDU ID cache

While I do not feel that there should be a single number for all implementations which specifies the depth of the MPDU ID cache, it is unacceptable for a STA to implement a depth of 0 or 1.

In addition, a minimum depth must be chosen in order to insure that reliable duplicate rejection can be supported on all STA, and the value of 16 was chosen as one which is easily supported by a small amount of hardware, and large enough to serve as a useful tool to reject duplicate frames.

Proposed Change: Alternative 2 "CCITT-CRC16 MPDU-ID Hash"

The proposed text is shown below, with change bars, in the affected section from the draft standard, IEEE p802.11-93/20b1. This hash function for MPDU-ID relies on the existence, not yet mandated, of a reasonable CRC generator somewhere in the system which contains the MAC. This CRC generator is used as the generator of the MPDU-ID Hash function.

4.1.5 MPDU ID/ConnID

This 16 bit field shall contain the MPDU-ID for asynchronous frames and the Connection ID for time-bounded frames. The MPDU-ID value is a specific ~~the~~ hash code derived from ~~of~~ the 3 octet NID field (as defined in section 4.1.6.1), the 6 octet source address (as defined in section 4.1.6.3) and the ~~a~~ 1 octet sequence number.

The MPDU ID shall be CCITT-CRC16 of the NID, source address and sequence number:

$$\text{MPDU ID} = x^{16} + x^{12} + x^5 + 1$$

Time-Bounded ConnectionIDs station are assigned when the station requests TB bandwidth in a management Request message.

5.2.12. Duplicate Detection and Recovery

Since MAC-level acknowledgments and retransmissions are incorporated into the protocol, there is the possibility that a frame may be received more than once. Such duplicate frames shall be filtered out within the destination MAC.

Duplicate frame filtering is facilitated through the inclusion of an **MPDU ID** field within the individual frames of an MPDU, including the DATA and ACK frames. Frames which are part of the same MPDU shall have the same ID, and different MPDUs will (with a very high probability) have a different ID.

The MPDU ID is a 16 bit hash of the 2 octet Network ID field, 6 octet source address and a 1 octet sequence number maintained by the source STA, as defined in section 4.1.5. ~~. The hashing of this information into a smaller field reduces overhead, particularly within ACK frames.~~

A destination STA shall reject a frame which has the RETRY bit set in the CONTROL field as a duplicate if it receives one which matches a value of recent MPDU IDs kept in the MPDU_ID_CACHE. The MPDU_ID_CACHE shall keep the no less than 16 of the most recent ~~last X~~ MPUD IDs of directed frames addressed to that STA, multicast frames received by that STA, or broadcast frames received by that STA on a FIFO basis for the purpose of comparison with the most recent MPDU ID. [2]

There is the small possibility that a frame will be improperly rejected due to such a match; however, this occurrence would be rare and would simply result in a lost frame similar to an FCS error.

[2] In particular, a STA is not required to keep a MPDU ID cache of frames which may be received but which are not addressed to that station. A frame is addressed to a STA when the NID field and the DST field both match the NID and MAC Address of the STA. The implementation of MPDU ID cache is not constrained to any maximum depth. The implementation of MPDU ID cache is not constrained to contain only MPDU ID addressed to the STA. The implementation of MPDU ID cache is not constrained to contain only the MPDU ID.

Rationale for Alternative 2

Quality of the MPDU ID

The CCITT-CRC 16 is discussed at some length in Doc 11-94/202, and I will not repeat it's analysis here.

However, the CRC of the keys provides a good measure of insurance against duplicate MPDU-ID generation, probably better than the additive alternative.

Complexity of the MPDU ID

Implementation of the CCITT-CRC16 MPDU ID has moderate complexity measure, however, this is offset substantially by the possiblity that this CRC is used in other parts of the system. If the CRC generation portion of the system can be re-used for MPDU-ID generation, then the complexity measure is extremely low, probably as low or lower than that of the additive alternative.

MPDU ID cache

While I do not feel that there should be a single number for all implementations which specifies the depth of the MPDU ID cache, it is unacceptable for a STA to implement a depth of 0 or 1.

In addition, a minimum depth must be chosen in order to insure that reliable duplicate rejection can be supported on all STA, and the value of 16 was chosen as one which is easily supported by a small amount of hardware, and large enough to serve as a useful tool to reject duplicate frames.