P802.1CQ PDU Format v2

Antonio de la Oliva (Interdigital)

(aoliva@it.uc3m.es)

Roger Marks (EthAirNet Associates)

(roger@ethair.net)

2020-11-05

Copyright information

Portions of this document (brief quotes from IEEE Std 1722) are Copyright © IEEE. As a result, per IEEE copyright policy:

- this document is "Previously Published"

- the contributors fulfil the responsibility to immediately inform the WG Chair that the contribution requires permission from copyright owner(s) and cannot be presented or included in the draft until that permission is granted, and offering to assist the WG Chair in requesting the permission, if possible

- the WG Chair is responsible to use the IEEE-SA Permission Request and Response Form Templates to request permission (http://standards.ieee.org/develop/stdsreview.html)

Background

- MAAP is currently specified in IEEE Std 1722
- Intention to move MAAP to P802.1CQ and enhance it
- See "MAAP Integration into P802.1CQ"
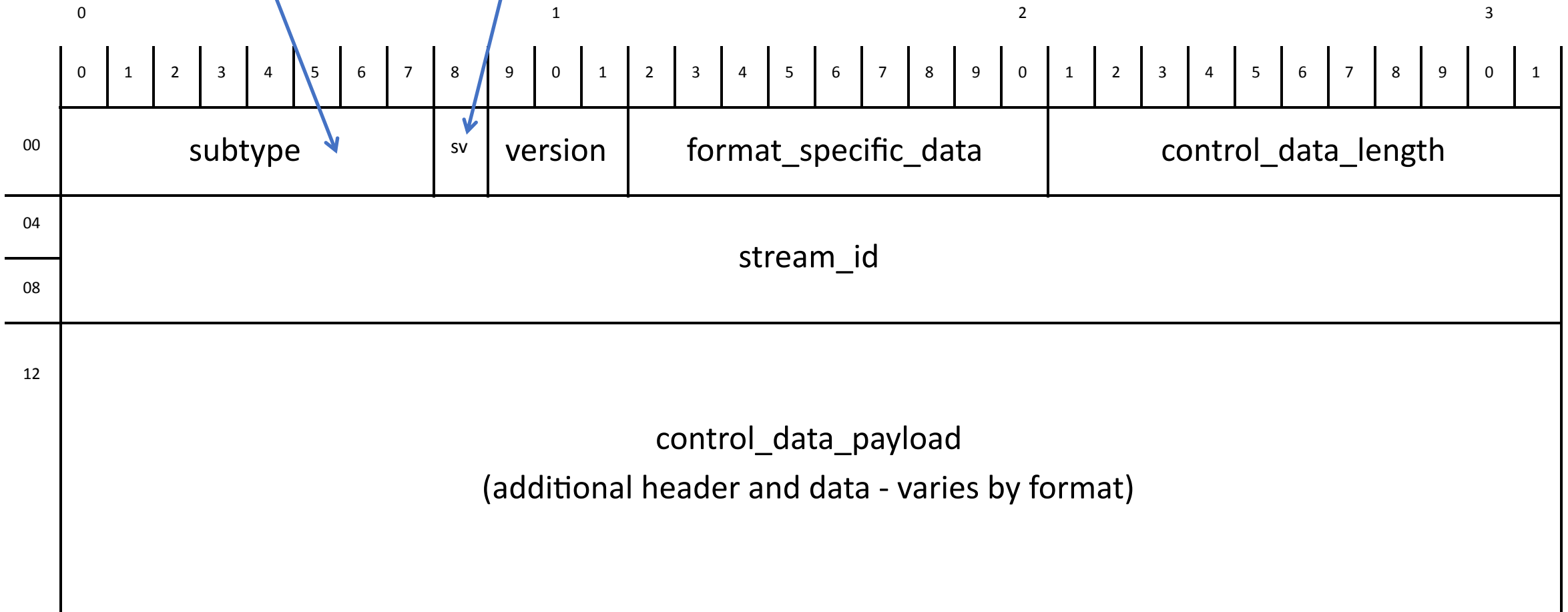 - cq-marks-oliva-MAAP
  - 2020-10-26

MAAP Compatibility

- Per 1722-2016: *The maap_version field identifies the version of MAAP being used. The current version of MAAP is one (1).*
- Forward Compatibility, per 1722-2016:
  - All MAAP AVTPDUs received that contain a higher version number and a *message type that is defined in the implemented version of MAAP* shall be interpreted using the implemented version of MAAP, *ignoring all unknown fields*. This requires that future versions of MAAP maintain compatibility with the message types implemented in all previous versions of MAAP.
  - All MAAP AVTPDUs received that contain a higher version number and a *message type that is not defined in the implemented version of MAAP* shall be ignored.
- Backward Compatibility, per 1722-2016:
  - MAAP AVTPDUs that carry a protocol version lower than the protocol version implemented by the receiver shall be interpreted according to the protocol definition corresponding to the protocol version received in the MAAP AVTPDU.
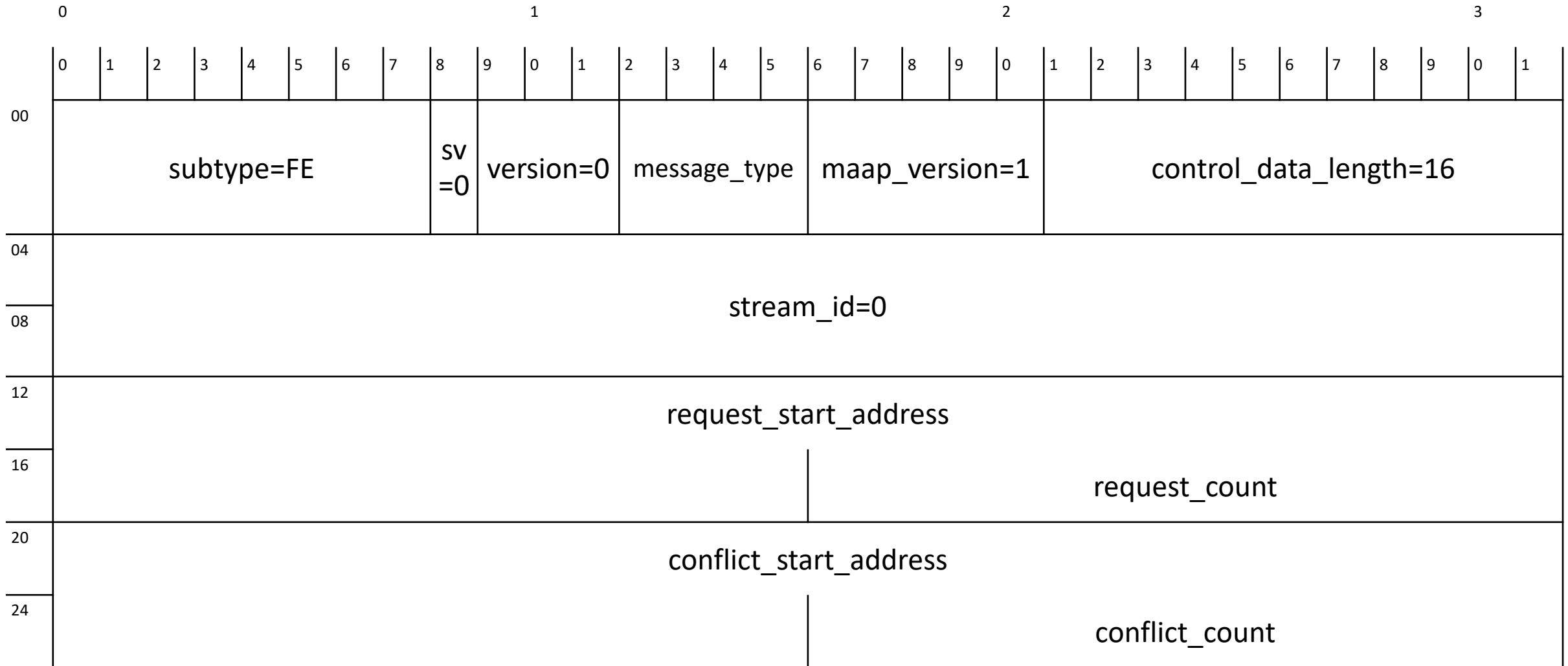
IEEE 1722 AVTPDU common control header

=FE for MAAP

0 | The stream_id field is not used or is defined by the format

	0				1					2							3

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1

00	subtype								SV	version		format_specific_data					control_data_length		

04	stream_id
08	

12	control_data_payload (additional header and data - varies by format)

5

Legacy MAAP PDU format

Legacy MAAP PDU messages

message_type	Name	maap_version
0	reserved	
1	PROBE	1
2	DEFEND	1
3	ANNOUNCE	1
4-16	reserved	

Pre-existing MAAP message types, enhanced

message_type	Name	maap_version
0	reserved	
1	PROBE	1
		2
2	DEFEND	1
		2
3	ANNOUNCE	1
		2

MAAPv1 devices
- can send MAAPv1 (maap_version 1) messages

- can read MAAPv1 messages

- can read MAAPv2 message_type 1, 2, and 3
  - *All MAAP AVTPDUs received that contain a higher version number and a message type that is defined in the implemented version of MAAP shall be interpreted using the implemented version of MAAP, ignoring all unknown fields.*

MAAPv2 PDU, legacy message types (1,2,3)

	0								1								2														3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
00	subtype=FE								sv =0	version=0				message_type = 1,2,3				maap_version= 2						control_data_length=16								
04	Other control information																	Message_subtype				Other control information										
08	Other control information				ID												Other control information															
12	request_start_address																															
16																	request_count															
20	conflict_start_address																															
24																	conflict_count															

New MAAPv2 message types

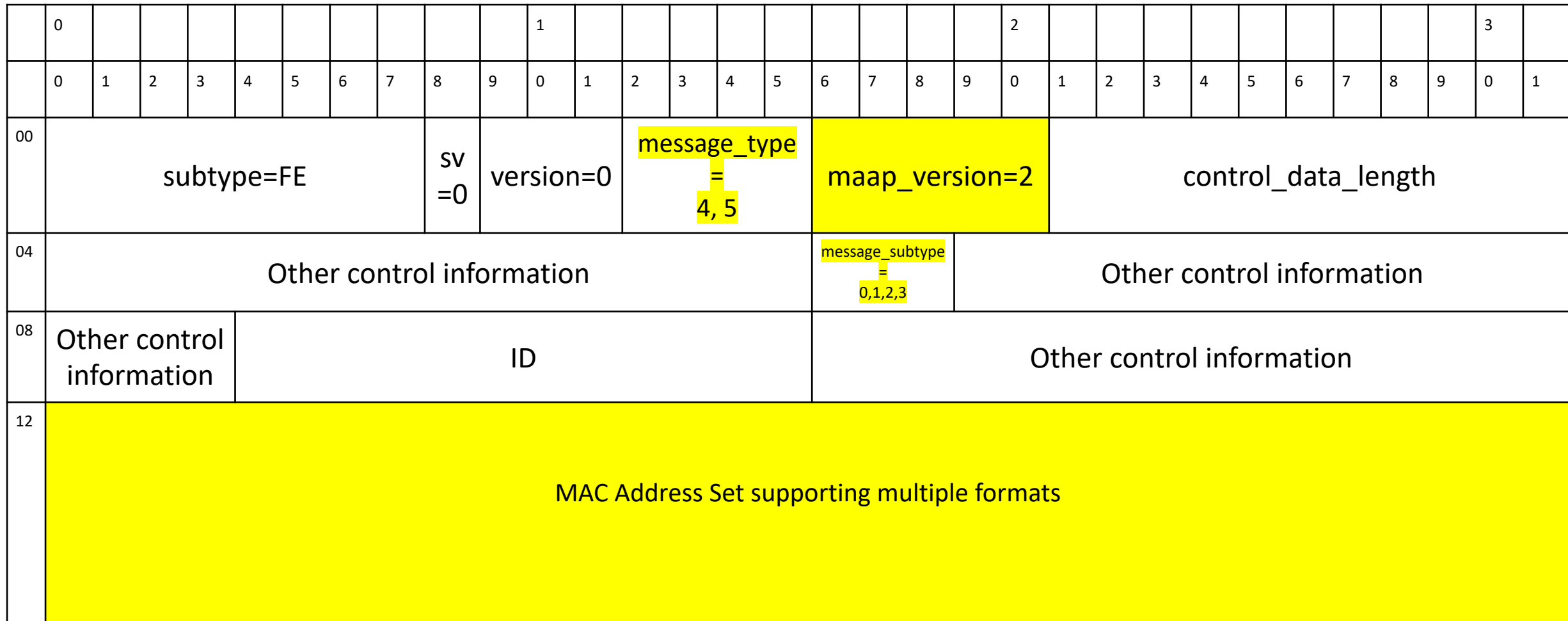| message_type	message_subtype	name	maap_version
0		reserved	
1		PROBE	1
			2
2		DEFEND	1
			2
3		ANNOUNCE	1
			2
4	0	DISCOVER	2
4	1	ADVERTISE	2
5	0	OFFER	2
5	1	REQUEST	2
5	2	ACK	2
5	3	RELEASE	2
6 – 16		reserved	

MAAPv1 devices

- can send MAAPv1 (maap_version 1) messages
- can read MAAPv1 messages
- can read MAAPv2 message_type 1, 2, and 3
  - *All MAAP AVTPDUs received that contain a higher version number and a message type that is defined in the implemented version of MAAP shall be interpreted using the implemented version of MAAP, ignoring all unknown fields.*
- ignore new MAAPv2 message_types
  - *All MAAP AVTPDUs received that contain a higher version number and a message type that is not defined in the implemented version of MAAP shall be ignored.*
  - **So MAAPv2 can use a new PDU format (but aligned with AVTPDU control header format).**

MAAPv2 devices

- can send all MAAPv2 messages
- can read all MAAPv1 and MAAPv2 messages

MAAPv2 PDU, new message types using AVTPDU common control header

	0									1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
00	subtype=FE								sv =0	version=0			message_type = 4, 5				maap_version=2				control_data_length											
04	Other control information													message_subtype = 0,1,2,3			Other control information															
08	Other control information		ID														Other control information															
12	MAC Address Set supporting multiple formats																															

Note: MAC Address Set field not drawn to scale

11

ID field and tie breaking

- Client: 12 bits flat ID (tie breaks performed based on MAC address)
- Server: 4 bits priority, 8 bits Network ID. Both set by admin.

0	1	2	3	4	5	6	7	8	9	10	11
Priority				Network ID							

- Case of multiple servers within same network-> tie break by priority
  - Please note that in the spec it is recommended servers are provisioned with disjoint address sets
- Case of merging networks -> periodic message from servers indicating their presence. With Network ID merging of networks can be detected (same network ID means servers are coordinated).
  - You do nothing
  - You claim same address to both servers (bit indicating multiple servers are found in the network with different Network ID, in the OFFER)→behavior at this stage out of scope

Conclusion

- For existing MAAPv1 message types, specify enhanced MAAPv2 versions
  - Can be read and understood by legacy MAAPv1 devices
  - New fields are ignored
- For new message types, retains the AVTPDU header format and subtype and add new fields, including message_subtype
  - ignored by legacy MAAPv1 clients.
  - enable the new functionality expected from IEEE 802.1CQ
  - e.g. server-assigned addresses and new address formats
- New PDU format presented to IEEE 1722 (20 October 2020) with encouraging positive feedback