A-MPDU Aggregation

Problems in the creation of the A-MPDU frames

Our objective and context

To create a “fake” AP using a wireless device in monitor mode

- make it able to manage the association with a normal 802.11n device
- Send a number of A-MPDUs to it, and study the savings in terms of air time and efficiency

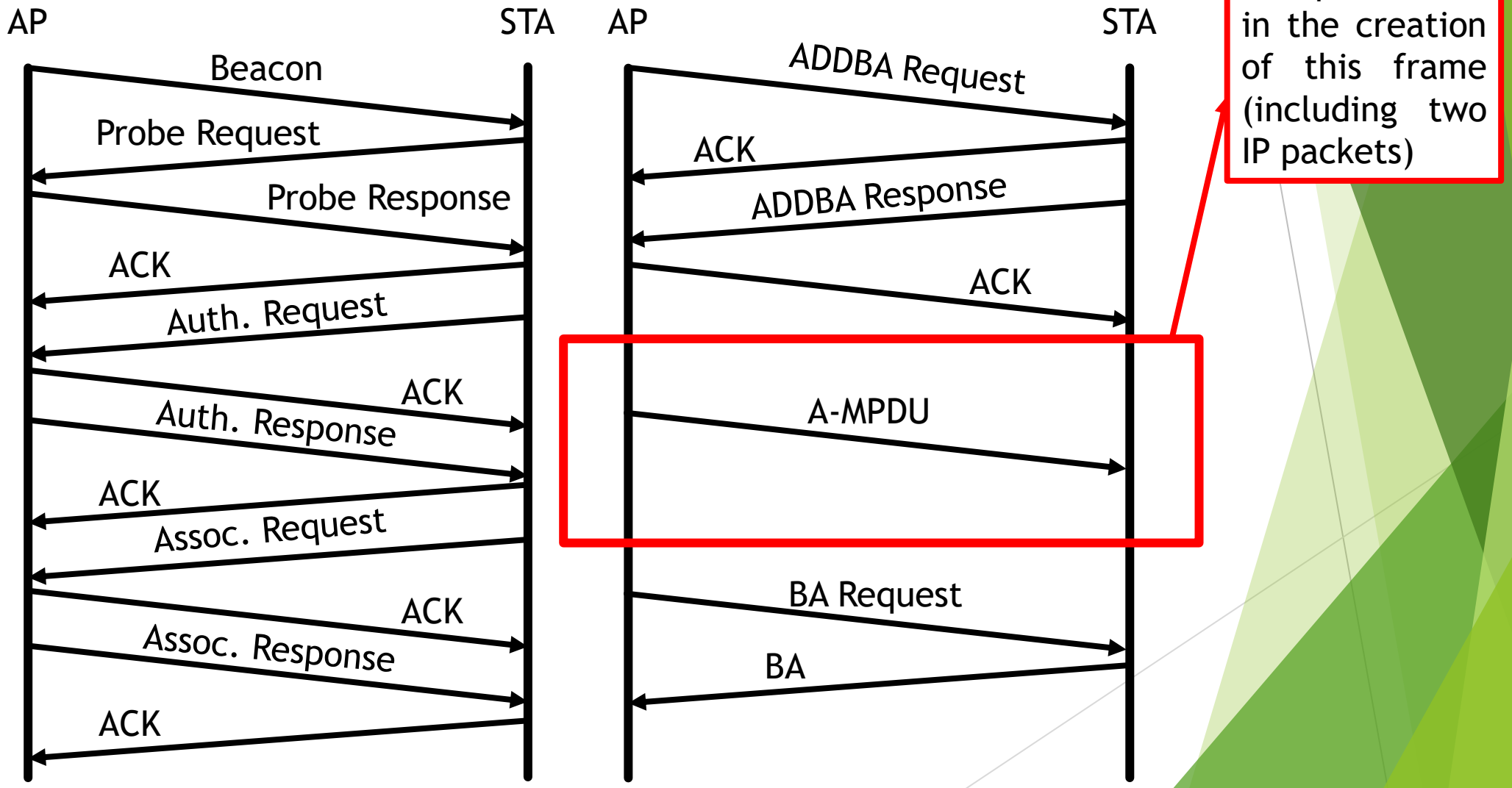
- The source code is being shared here
  - https://github.com/Wi5/wi5-aggregation

- This is part of a Final Degree Project in Unizar.es. It is also a collaboration with Wi-5 project.

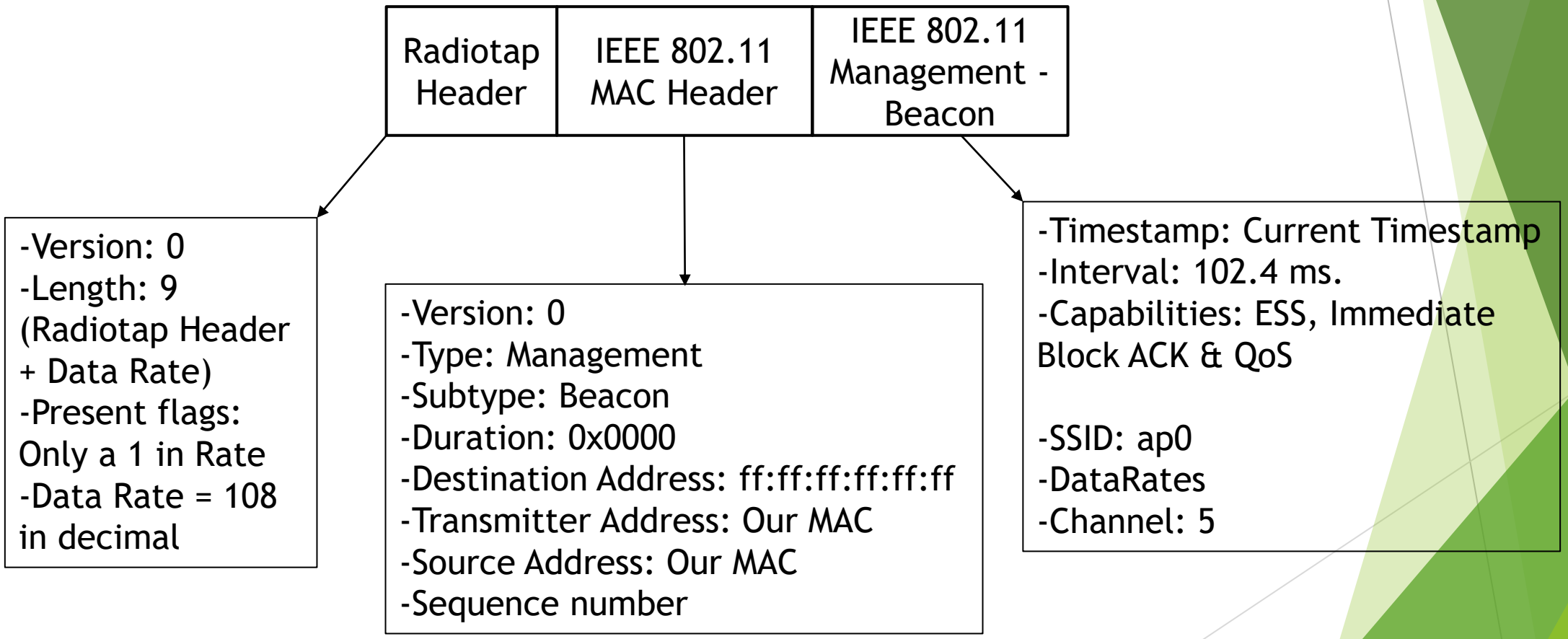
Hardware

- As an AP:
  - TP-Link TL-WN722N with the driver ath9k_htc working in monitor mode
- As the STA:
  - AR9287 Wireless Network Adapter (PCI-Express) rev 01 with the driver ath9k working in managed mode

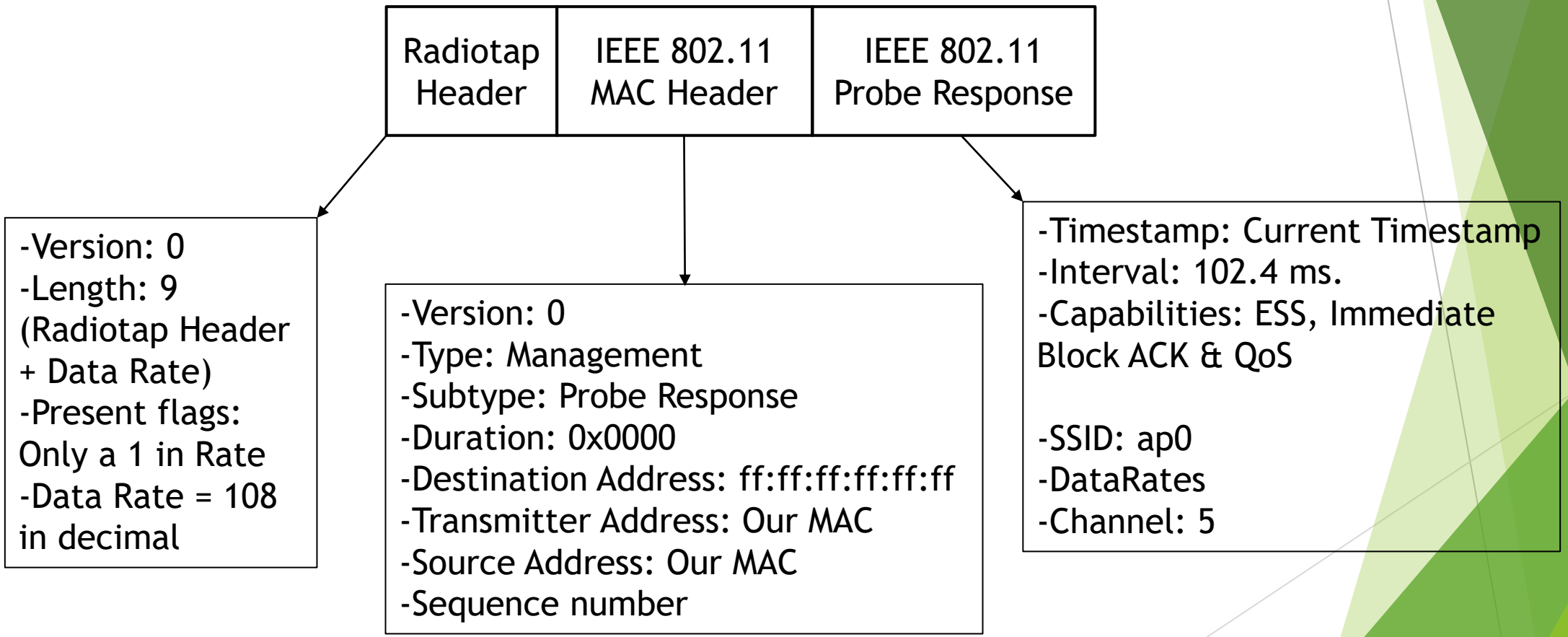
Traffic Scheme (Connection & Data)



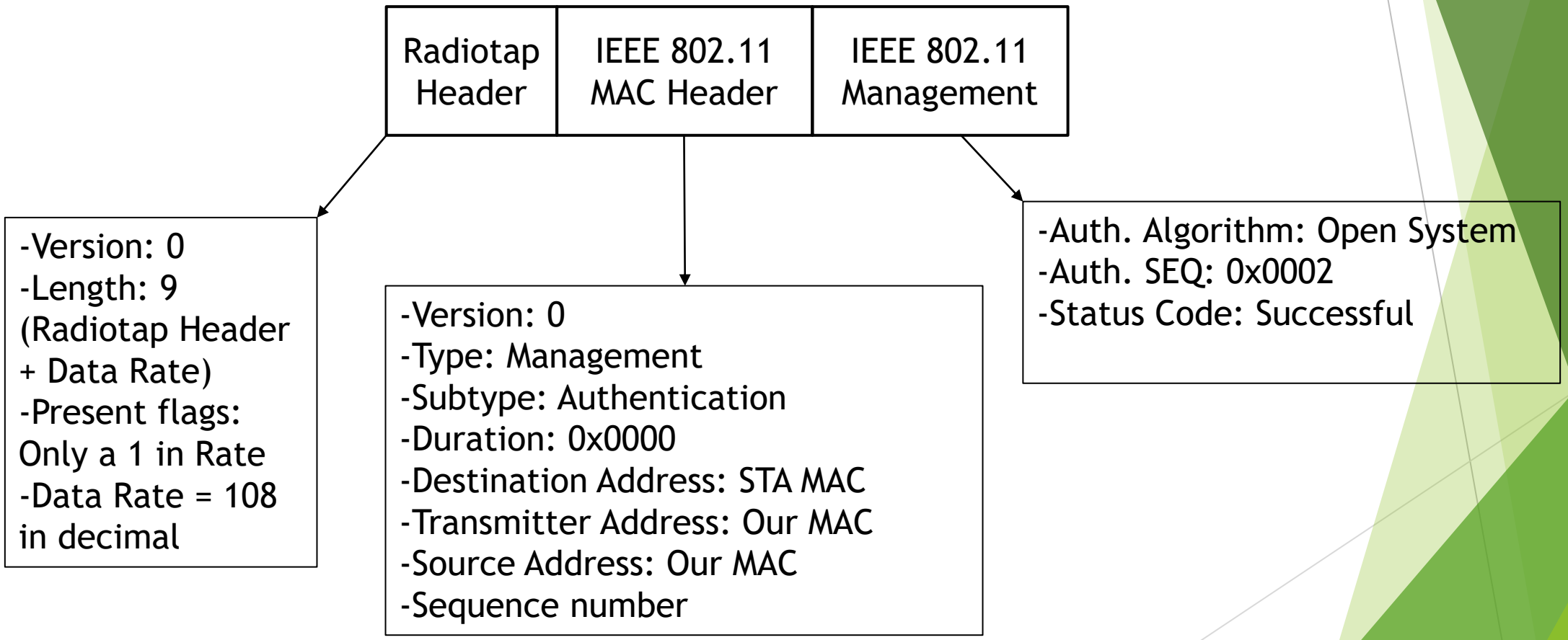
Beacon Packet (Fields and values we fill)



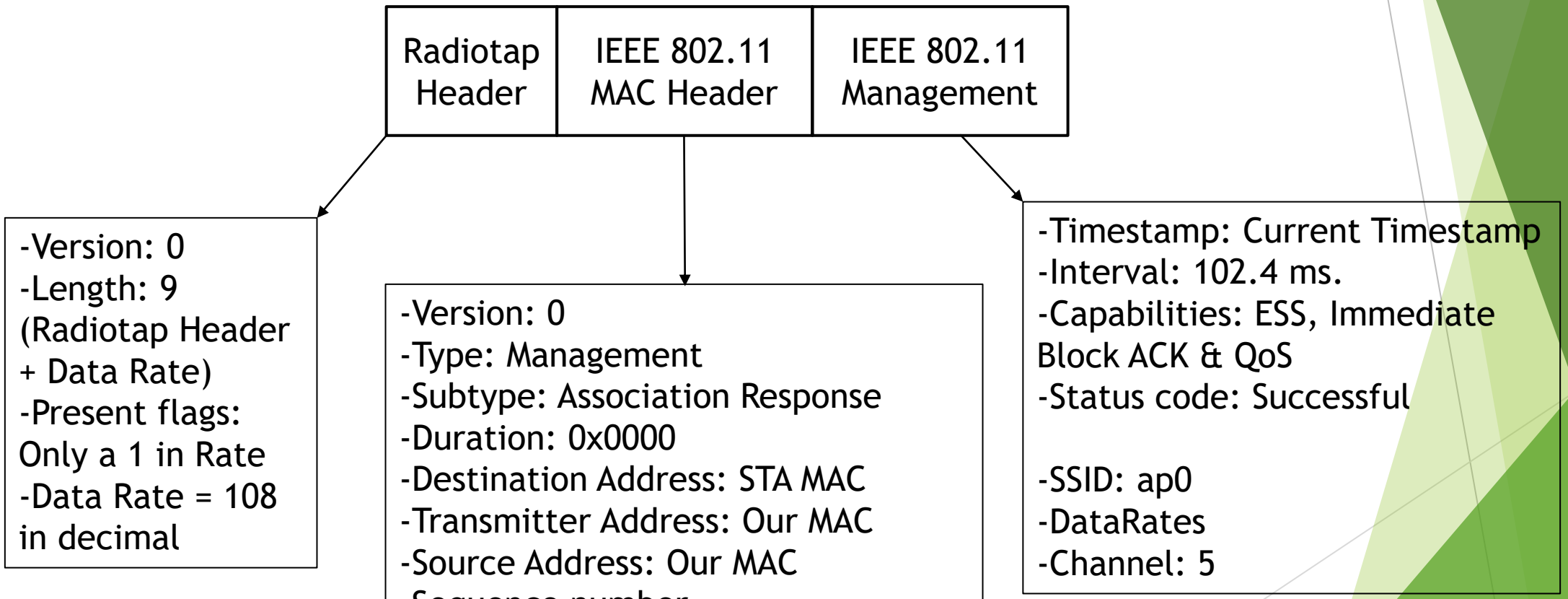
Probe Response (Fields and values we fill)



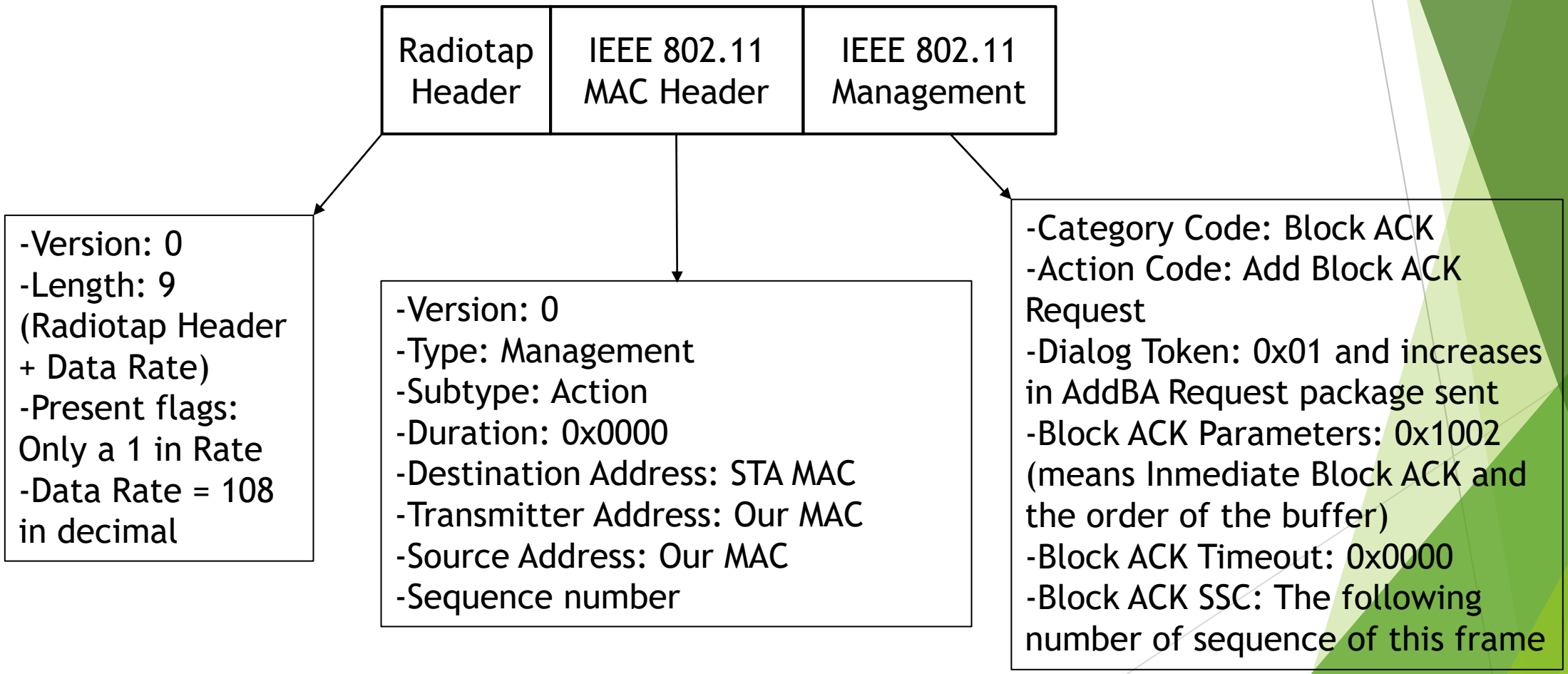
Auth. Response (Fields and values we fill)



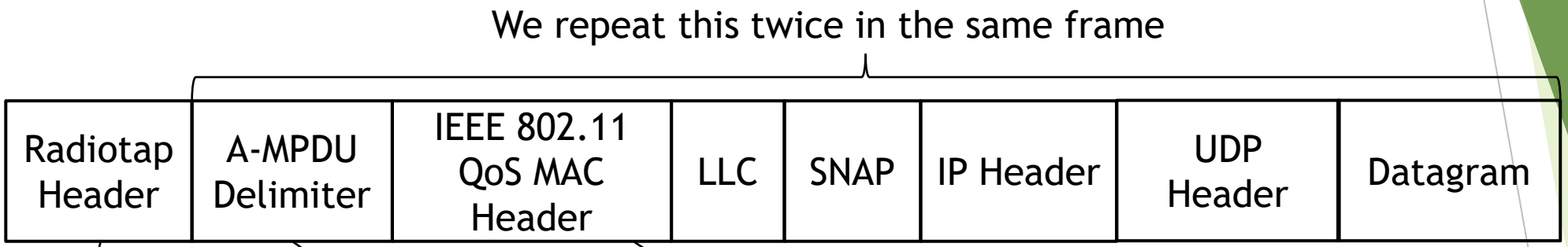
Assoc. Response (Field and values we fill)



AddBA Request (Fields and values we fill)



A-MPDU (I): structure with two IP packets



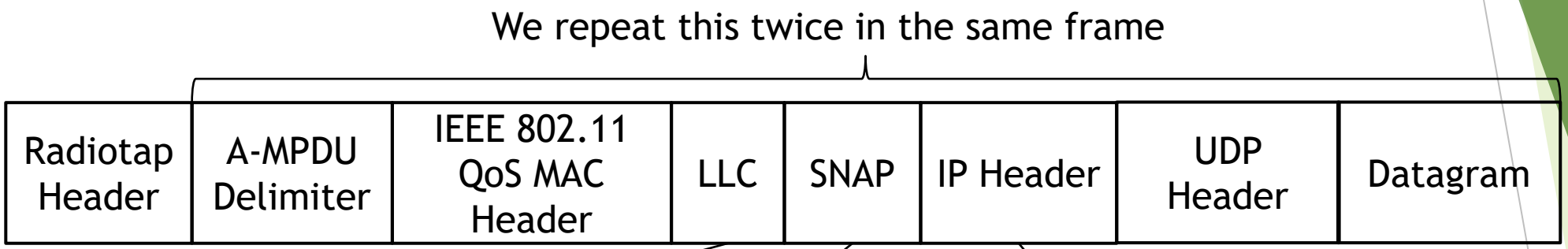
-Version: 0
-Length: 9 (Radiotap Header + Data Rate)
-Present flags: Only a 1 in Rate
-Data Rate = 108 in decimal

-Reserved&Lengh: 0x4004
-CRC: 0x47
-Delimiter Signature: 0x4E ('N')

All these values were taken from an example

-Version: 0
-Type: Data
-Subtype: QoS
-Duration: 0x0000
-Destination Address: STA MAC
-Transmitter Address: Our MAC
-Source Address: Our MAC
-Sequence number
-QoS: 0x0000

A-MPDU (II): structure with two IP packets

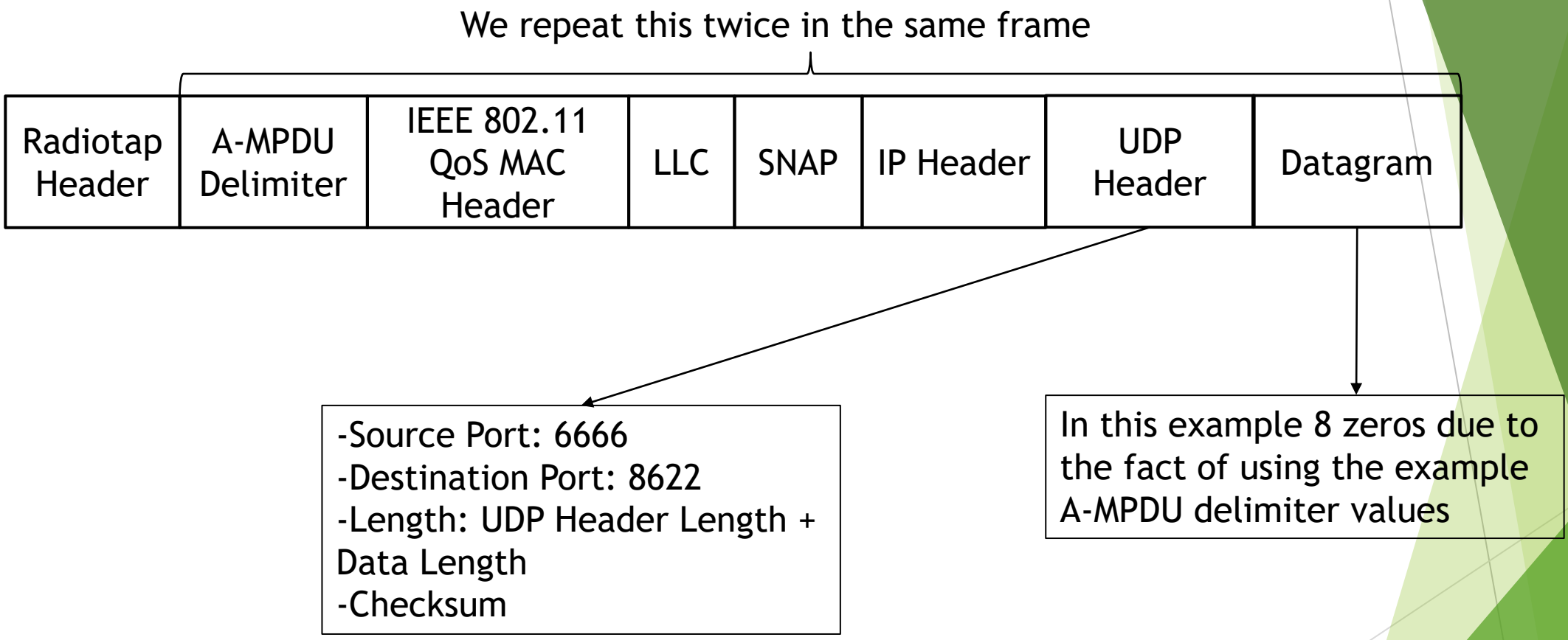


-DSAP: 0xAA
-SSAP: 0xAA
-Control: 0x03

-OID: 0x000000
-Protocol ID: 0x0800

-Internet Header Length (IHL): 5
-Version:4
-ToS:0x00
-Total Length: IP Header Length + UDP Header Length + Data Length
-ID
-Frag_off: 0
-TTL: 255
-Protocol: UDP
-Source Address: Our IP Address
-Destination Address: The STA IP Address
-Checksum

A-MPDU (III): structure with two IP packets



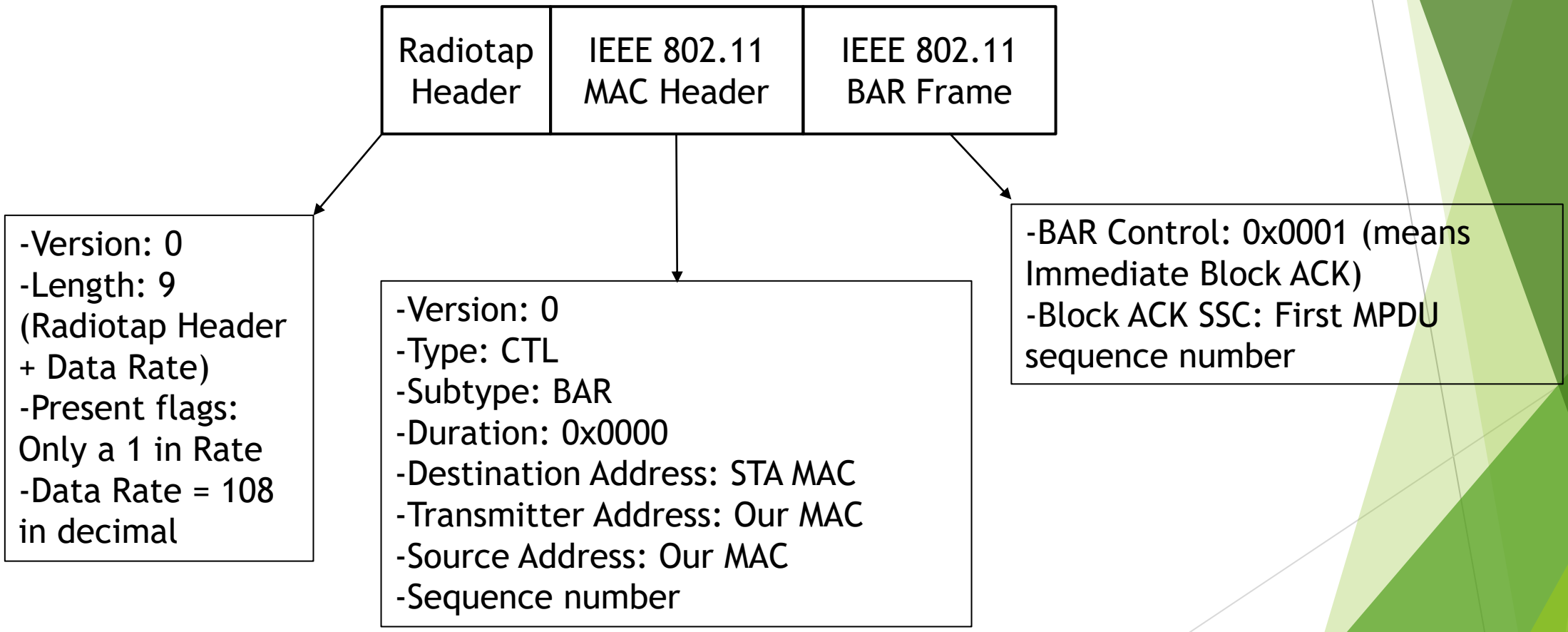
A-MPDU (IV): Questions

- This frame is giving us the problems, it could be happening because we didn’t notice that some flag of this frame or the previous ones must be enabled, we tried to enable in this frame the “A-MPDU status” that belongs to the radiotap header but we couldn't.

- Other thing that could be wrong are the values of the A-MPDU delimiter but we try to use the values used in the following program as we said previously:
  - https://github.com/rpp0/aggr-inject

- But even using those values, Wireshark thinks that the A-MPDU delimiter is the Frame Control Field, as you can **see in the attached capture**. Wireshark thinks there were Fragmented IEEE802.11 frames

BA Request


Radiotap Header
IEEE 802.11 MAC Header
IEEE 802.11 BAR Frame
-Version: 0
-Length: 9
(Radiotap Header + Data Rate)
-Present flags: Only a 1 in Rate
-Data Rate = 108 in decimal
-Version: 0
-Type: CTL
-Subtype: BAR
-Duration: 0x0000
-Destination Address: STA MAC
-Transmitter Address: Our MAC
-Source Address: Our MAC
-Sequence number
-BAR Control: 0x0001 (means Immediate Block ACK)
-Block ACK SSC: First MPDU sequence number

THANKS FOR YOUR HELP