# CS 536 Spring 2024

## Lab 3: Traffic Monitoring, Remote Command Server, Fast and Reliable File Transport

**Alireza Lotfi**

# Contents

**1 Problem 1** **3**
1.1 Environment Preparation . . . . . 3
1.2 Capturing Ethernet Frames . . . . . 3
1.3 MAC Address Verification . . . . . 4
1.4 Filter Application and Frame Inspection . . . . . 4
1.5 IP Examination . . . . . 5
1.6 Port Validation . . . . . 5
1.7 Application Layer Payload Analysis . . . . . 6

**2 Problem 3** **7**
2.1 Running the code . . . . . 7
2.2 Performance evaluation . . . . . 8

**3 Bonus** **9**
3.1 Overview . . . . . 9
3.2 m2_ch153_2024-02-19_12.53.41.307.pcap . . . . . 11
3.2.1 802.11 Network Type . . . . . 11
3.2.2 Frequency Band . . . . . 11
3.2.3 Basic Service Sets (BSS) . . . . . 11
3.2.4 Types of Frames . . . . . 12
3.2.5 Data Rates . . . . . 12
3.2.6 Signal Levels (SNR) of High-Traffic BSS . . . . . 12
3.2.7 MAC Addresses of Access Points . . . . . 13
3.3 m2_ch124_2024-02-19_12.57.45.517.pcap . . . . . 14
3.3.1 802.11 Network Type . . . . . 14
3.3.2 Frequency Band . . . . . 15
3.3.3 Basic Service Sets (BSS) . . . . . 15
3.3.4 Types of Frames . . . . . 16
3.3.5 Data Rates . . . . . 16
3.3.6 Signal Levels (SNR) of High-Traffic BSS . . . . . 16
3.3.7 MAC Addresses of Access Points . . . . . 17

3.4 Comparison . . . . . 18
3.4.1 Network Types . . . . . 19
3.4.2 Data Rate Types . . . . . 19
3.4.3 BSS . . . . . 19

# 1 Problem 1

The file used in wireshark analysis is available in the the main directory of `lab3`, named `testlogfile`.

## 1.1 Environment Preparation

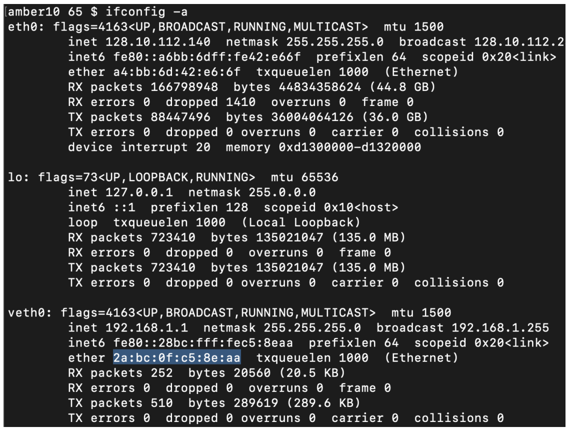
- The environment was set up by running a server and client application as shown in Figure 1.
- The server was initiated using the command:
  `./ssftpd.bin 192.168.1.1 50000`.
- The client was launched using:
  `veth './ssftp.bin testfile 192.168.1.1 50009 192.168.1.2 256000'`.


(a) Server (b) Client

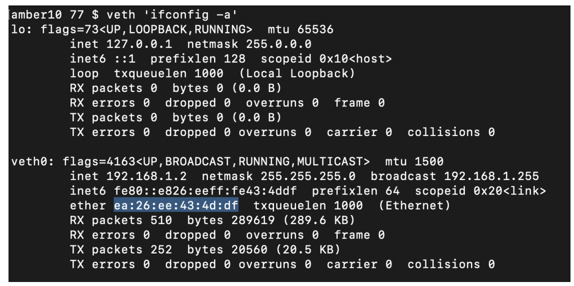
Figure 1: Environment initialization

## 1.2 Capturing Ethernet Frames

- Network traffic (24 Ethernet frames) generated between the server and client applications was captured using:
  `sudo /usr/local/etc/tcpdumpwrap-veth0 -c 24 -w - > testlogfile`




(a) Source (Server)




(b) Destination (Client)

Figure 2: Mac address verification

## 1.3 MAC Address Verification

- MAC addresses associated with `192.168.1.1` and `192.168.1.2` were confirmed using the `ifconfig -a` command on an Amber machine, revealing the addresses `ea:26:ee:43:4d:df` and `2a:bc:0f:c5:8e:aa` respectively as shown in Figure 2.

## 1.4 Filter Application and Frame Inspection

Utilizing Wireshark with the filter, relevant Ethernet frames were identified for analysis as shown in Figure 3. This filter condition is designed to selectively capture Ethernet frames that meet specific criteria:

- `eth.dst == ea:26:ee:43:4d:df`: Filters for Ethernet frames where the destination MAC address (`eth.dst`) matches `ea:26:ee:43:4d:df`.
- `eth.src == 2a:bc:0f:c5:8e:aa`: Filters for Ethernet frames where the source MAC address (`eth.src`) matches `2a:bc:0f:c5:8e:aa`.

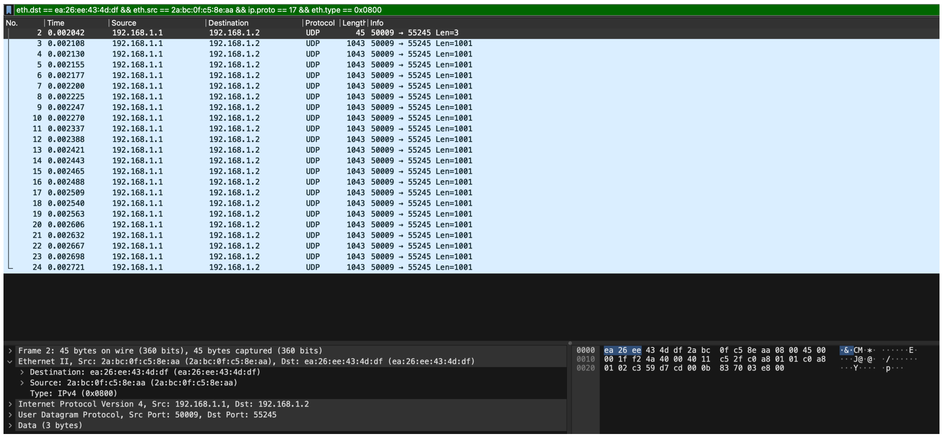



Figure 3: Packet filtering

- `ip.proto == 17`: Filters for Ethernet frames where the IP protocol (`ip.proto`) is set to 17, which corresponds to the User Datagram Protocol (UDP).
- `eth.type == 0x0800`: Filters for Ethernet frames where the Ethernet type field (`eth.type`) is set to `0x0800`, indicating an IPv4 packet.

Combining these conditions using logical AND (`&&`) allows for precise filtering of network traffic.

## 1.5 IP Examination

The last 8 bytes of the IP header within each frame were scrutinized, confirming consistency with the source and destination IP addresses utilized by the file transfer application as shown in Figure 4.

## 1.6 Port Validation

The first four bytes of the UDP header in each frame were validated, ensuring alignment with the source (server: 50009) and destination (client: 55245) ports as shown in Figure 5.

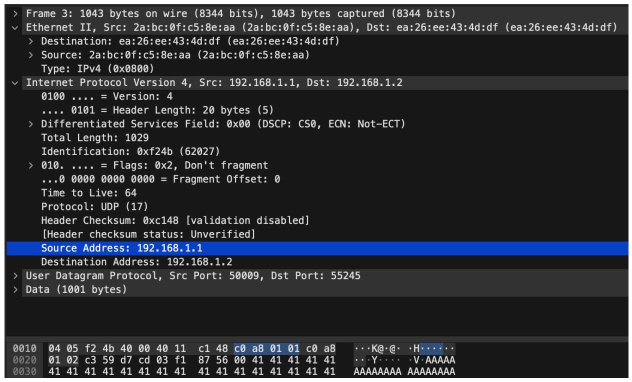



Figure 4: IP checking

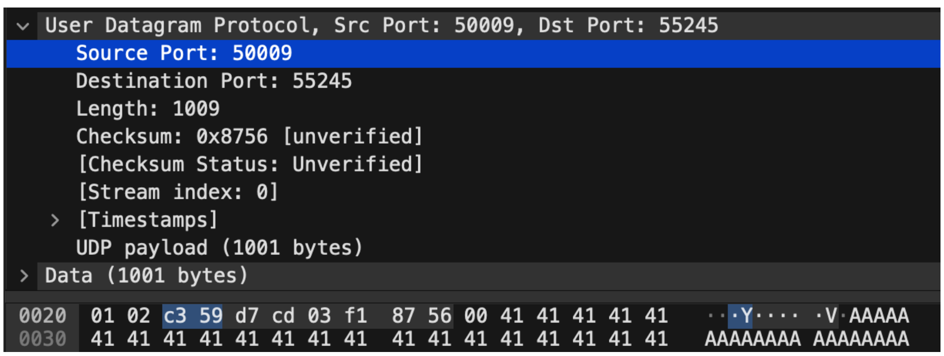



Figure 5: Port checking

## 1.7 Application Layer Payload Analysis

Hexadecimal inspection of the remaining bytes of each frame revealed application layer payload content consistent with the expected data exchanged between the client and server as shown in Figure 6.

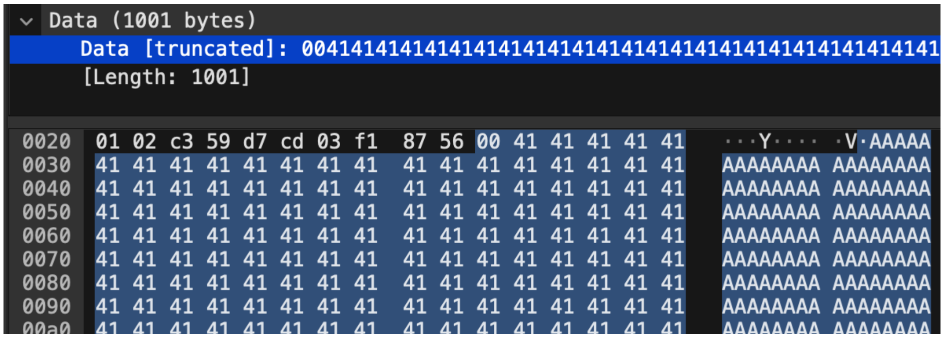
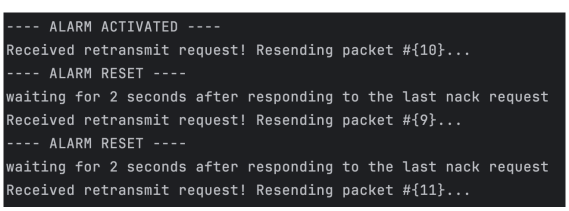



Figure 6: Payload checking

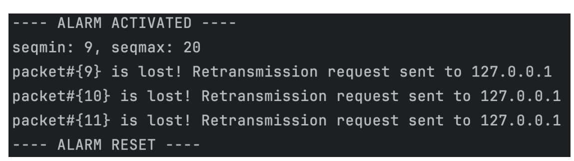
# 2 Problem 3

## 2.1 Running the code

I tested the code on my **macOS** device which worked perfectly. I have used 100 microseconds SIGALRM in the client as 150 millisecond wasn't being triggered due to short communication time. The code was tested by manually dropping specific packets (packets 9, 10, 11) to observe the reaction on the server side. The file size used for the experiment was 25600 bytes. The output for both client and the server are located in `/lab3/v2/output` directory, named `client_output.txt` and `server_output.txt` respectively. A snippet of this output is presented in Figure 7.




(a) Server




(b) Client

Figure 7: Signal checking

## 2.2 Performance evaluation

Upon analyzing the results, we noticed that the packet loss rates did not significantly differ between the two versions. However, we observed a substantial increase in completion time for larger file sizes in comparison to the previous version from lab2. To illustrate this, we have provided the difference in completion time for a file size of 256000 bytes in Table 1 (the last two rows belong to the new experiment).

Table 1: Comparison of Performance Metrics for File Size 256000

| Payload Size (bytes) | File Size (bytes) | Completion Time (ms) | Transfer Speed (bps) | Payload Packets | Percentage of bytes not received |
|---|---|---|---|---|---|
| 256000 | 1000 | 6.00 | 341.33M | 256 | 0.00% |
| 256000 | 1400 | 5.00 | 409.6M | 183 | 0.00% |
| 256000 | 1000 | 11.00 | 186.18 M | 256 | 0.00% |
| 256000 | 1400 | 4.00 | 512.00 M | 183 | 0.00% |

In the first two rows, the completion times are 6.00 ms and 5.00 ms, while in the last two rows, the completion times are longer at 11.00 ms and 4.00 ms, respectively. This suggests that the method used in the last two rows may be less efficient in terms of completion time.

In terms of transfer speed, the first two rows have speeds of 341.33 Mbps and 409.6 Mbps, while the last two rows have slower speeds of 186.18 Mbps and 512.00 Mbps, respectively. The slower transfer speeds in the last two rows may be due to the different method used, or it could be a result of other factors such as network congestion or interference.

# 3 Bonus

## 3.1 Overview

In Wireshark, we can filter for specific criteria using display filters.

- **802.11 Network Type**

  `wlan.fc.type_subtype == 8` for 802.11g or `wlan.fc.type_subtype == 12` for 802.11a frames.

- **Frequency Band**

  We can directly filter by frequency band in Wireshark using `radiotap.channel.freq`. For example, we can use `radiotap.channel.freq == 5680` to find packets with the channel frequency of 5680.

- **Basic Service Sets (BSS)**

  `wlan.bssid == [BSSID]` to filter packets belonging to a specific BSS.

- **Types of Frames**

  In 802.11 networks, frames are categorized based on their types and subtypes, each identified by a specific numeric value. These values are represented in the `wlan.fc.type_subtype` field in Wireshark.

  - **Subtype 8 (Beacon Frame)**
  - **Subtype 0 (Management Frame)**
  - **Subtype 11 (RTS Frame)**
  - **Subtype 12 (CTS Frame)**

  - **Subtype 4 (Probe Request Frame)**
  - **Subtype 5 (Probe Response Frame)**

- **Data Rates**

  `wlan_radio.data_rate` to filter by data rates. For example:

  `wlan_radio.data_rate == 54` for 54 Mbps.

- **Signal Levels (SNR) of High-Traffic BSS**

  We can use `wlan_radio.snr` to find the SNR value.

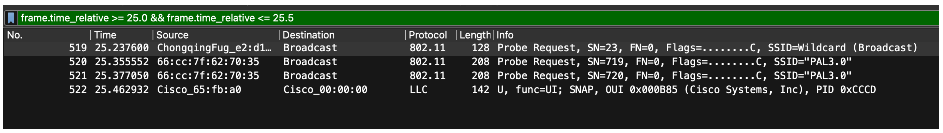
- **MAC Addresses of Access Points**

  We can use the filter `wlan.sa == [MAC address]` to filter based on the source MAC address and `wlan.da == [MAC address]` to filter based on the destination address.

Now that we have described the required commands for analyzing the packets, we will perform the filters for the provided data and compare the results at the end.

## 3.2 m2_ch153_2024-02-19_12.53.41.307.pcap

We have selected a 0.5 sec interval to analyze the packets which is shown if Figure 8.

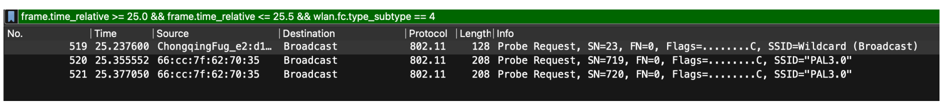

frame.time_relative >= 25.0 && frame.time_relative <= 25.5

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 519 | 25.237600 | ChongqingFug_e2:d1… | Broadcast | 802.11 | 128 | Probe Request, SN=23, FN=0, Flags=........C, SSID=Wildcard (Broadcast) |
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |
| 522 | 25.462932 | Cisco_65:fb:a0 | Cisco_00:00:00 | LLC | 142 | U, func=UI; SNAP, OUI 0x000B85 (Cisco Systems, Inc), PID 0xCCCD |

Figure 8: Interval selection

### 3.2.1 802.11 Network Type

For the network type we were able to identify only type 4. However, we had a different protocol LLC which was also available in this time frame. Results are depicted in Figure 9.

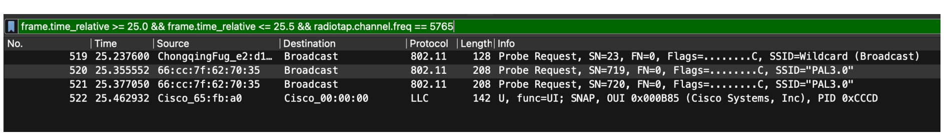

frame.time_relative >= 25.0 && frame.time_relative <= 25.5 && wlan.fc.type_subtype == 4

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 519 | 25.237600 | ChongqingFug_e2:d1… | Broadcast | 802.11 | 128 | Probe Request, SN=23, FN=0, Flags=........C, SSID=Wildcard (Broadcast) |
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |

Figure 9: Type 4

### 3.2.2 Frequency Band

All packets in this time frame were using the same channel frequency of 5765. Results are depicted in Figure 10.

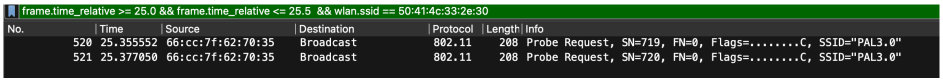

frame.time_relative >= 25.0 && frame.time_relative <= 25.5 && radiotap.channel.freq == 5765

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 519 | 25.237600 | ChongqingFug_e2:d1… | Broadcast | 802.11 | 128 | Probe Request, SN=23, FN=0, Flags=........C, SSID=Wildcard (Broadcast) |
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |
| 522 | 25.462932 | Cisco_65:fb:a0 | Cisco_00:00:00 | LLC | 142 | U, func=UI; SNAP, OUI 0x000B85 (Cisco Systems, Inc), PID 0xCCCD |

Figure 10: Channel frequency

### 3.2.3 Basic Service Sets (BSS)

Out of these packets (4 total), only 2 belongs to PAL3.0. Results are depicted in Figure 11.



frame.time_relative >= 25.0 && frame.time_relative <= 25.5  && wlan.ssid == 50:41:4c:33:2e:30

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |

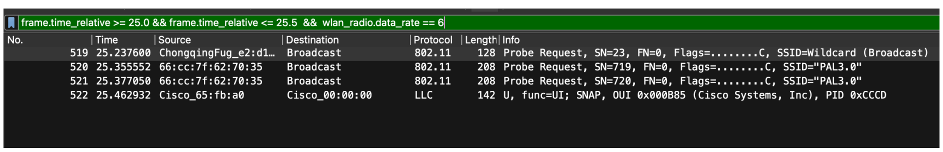
Figure 11: BSS checking

#### 3.2.4 Types of Frames

The only frame type available in this time frame is Probe. It can be checked in Figure 9 of the first section of this analysis.

#### 3.2.5 Data Rates

All packets in the selected interval use the rate of 6 Mb/s. Results are depicted in Figure 20.



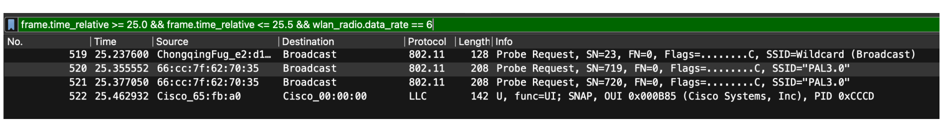
frame.time_relative >= 25.0 && frame.time_relative <= 25.5  &&  wlan_radio.data_rate == 6

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 519 | 25.237600 | ChongqingFug_e2:d1… | Broadcast | 802.11 | 128 | Probe Request, SN=23, FN=0, Flags=........C, SSID=Wildcard (Broadcast) |
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |
| 522 | 25.462932 | Cisco_65:fb:a0 | Cisco_00:00:00 | LLC | 142 | U, func=UI; SNAP, OUI 0x000B85 (Cisco Systems, Inc), PID 0xCCCD |

Figure 12: Data rate = 6 Mb/s

#### 3.2.6 Signal Levels (SNR) of High-Traffic BSS

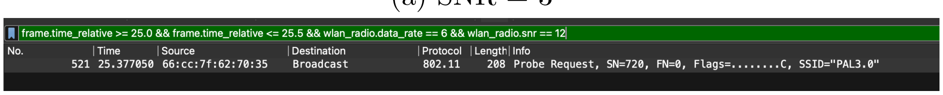
SNR values differ so much as we the factors such as noise level play a crucial role for computing the SNR. Results are depicted in Figure 13.



frame.time_relative >= 25.0 && frame.time_relative <= 25.5 && wlan_radio.data_rate == 6

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 519 | 25.237600 | ChongqingFug_e2:d1… | Broadcast | 802.11 | 128 | Probe Request, SN=23, FN=0, Flags=........C, SSID=Wildcard (Broadcast) |
| 520 | 25.355552 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=719, FN=0, Flags=........C, SSID="PAL3.0" |
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |
| 522 | 25.462932 | Cisco_65:fb:a0 | Cisco_00:00:00 | LLC | 142 | U, func=UI; SNAP, OUI 0x000B85 (Cisco Systems, Inc), PID 0xCCCD |

(a) SNR = 3



frame.time_relative >= 25.0 && frame.time_relative <= 25.5 && wlan_radio.data_rate == 6 && wlan_radio.snr == 12

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 521 | 25.377050 | 66:cc:7f:62:70:35 | Broadcast | 802.11 | 208 | Probe Request, SN=720, FN=0, Flags=........C, SSID="PAL3.0" |

(b) SNR = 12

Figure 13: Different SNR values

### 3.2.7 MAC Addresses of Access Points

In this example, we have checked a packed which belongs to CISCO Systems. Knowing the mac addresses we can easily find them. Otherwise, we can use `wlan.ssid` to filter the packets that include a mac address and then use the addresses for our filter. Results are depicted in Figure 14.

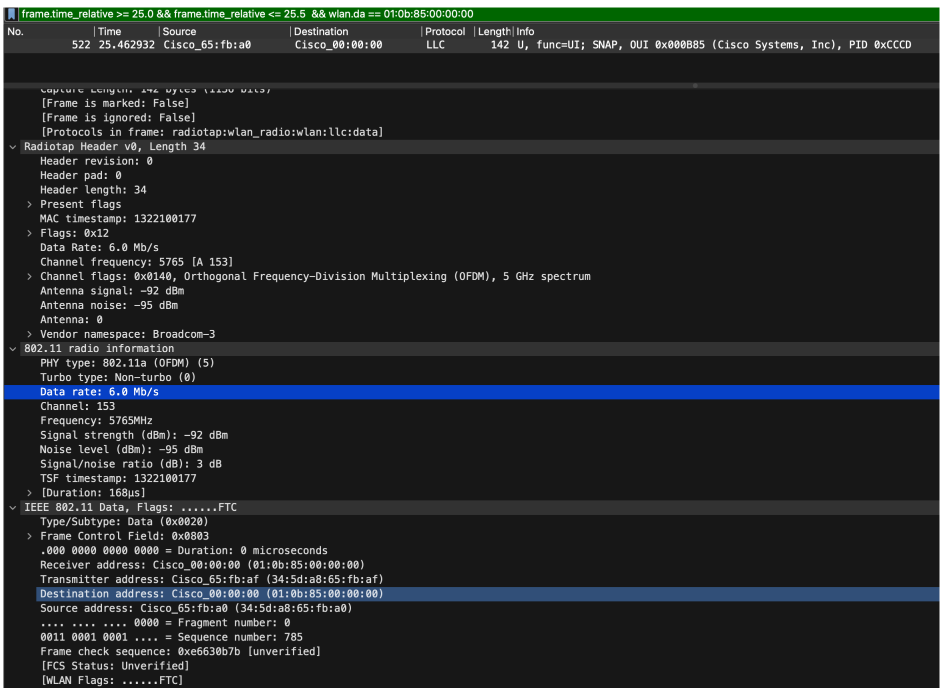



Figure 14: Type 4

## 3.3 m2_ch124_2024-02-19_12.57.45.517.pcap

We have selected a 0.5 sec interval to analyze the packets which is shown if Figure 15.

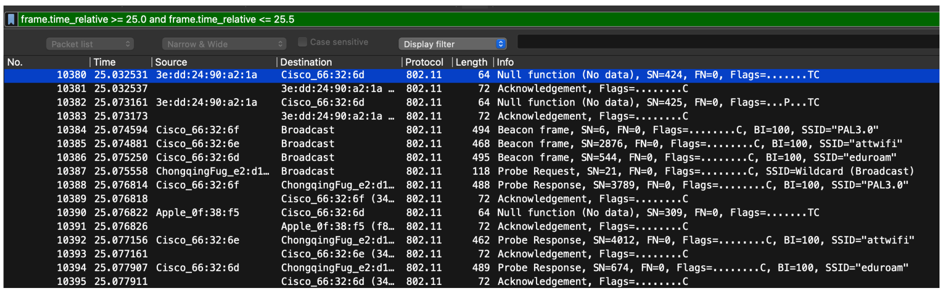
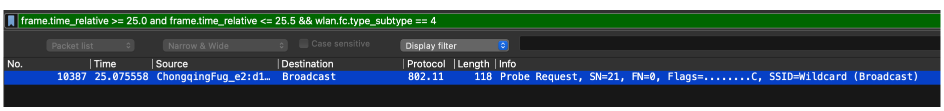


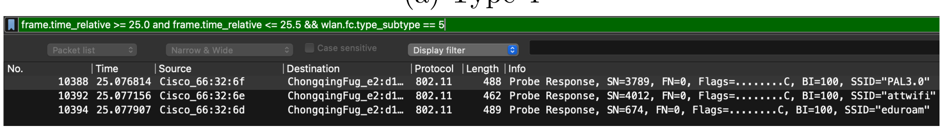

Figure 15: Interval selection

### 3.3.1 802.11 Network Type

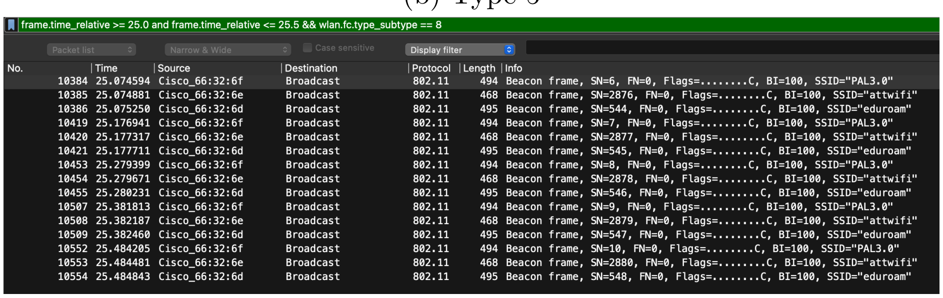
For the network type we were able to identify types 4, 5, and 8 but we couldn't find wlan.fc.type_subtype == 12. The results are depicted in Figure 16




(a) Type 4




(b) Type 5




(c) Type 8

Figure 16: Different network types

### 3.3.2 Frequency Band

Most packets were operating in the same channel as indicated in Figure 17 (freq = 5680).

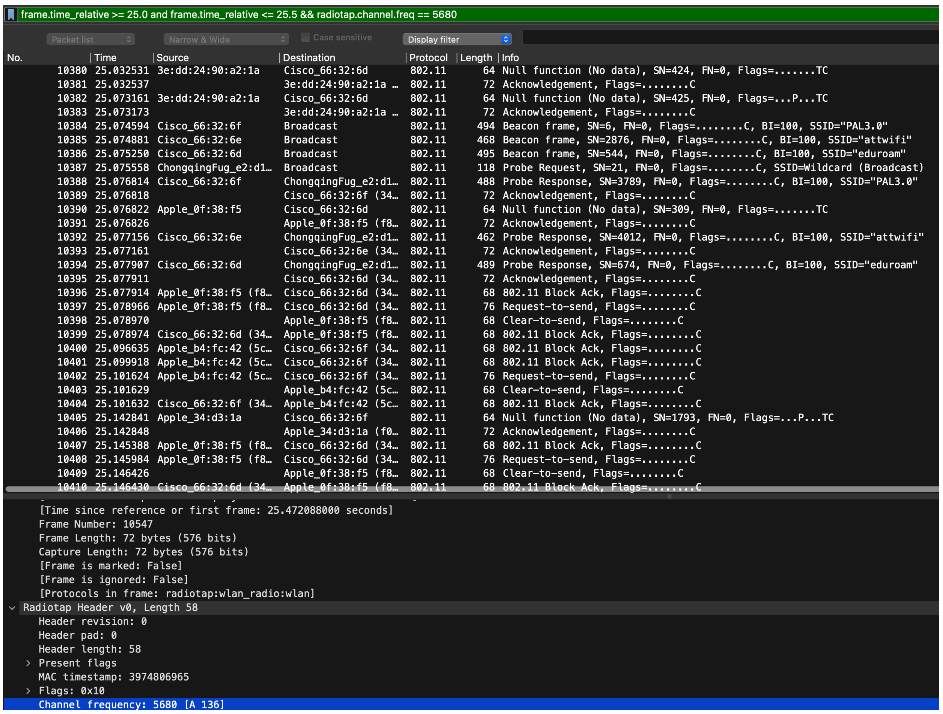



Figure 17: Frequency checking

### 3.3.3 Basic Service Sets (BSS)

As an example for this section we filtered the packets belonging to PAL3.0 as shown in Figure 18.

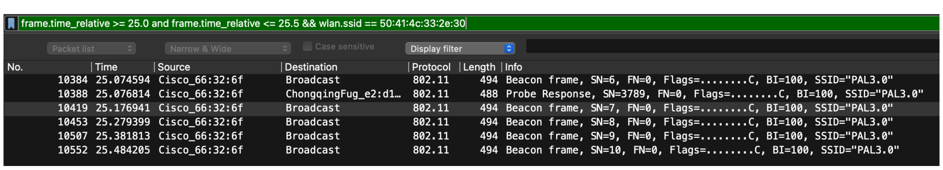
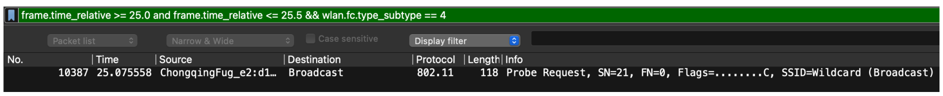


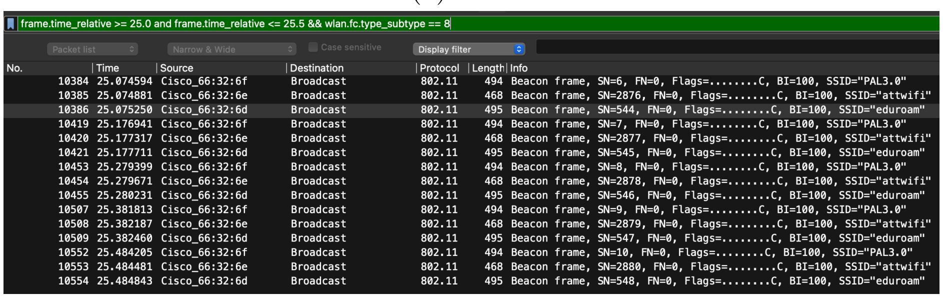

Figure 18: BSS filtering

### 3.3.4 Types of Frames

There are so many types of frames available as indicated in the overview section. For this set of packets, we could only find two type which are Probe and Beacon types. The results are demonstrated in Figure 19.




(a) Probe




(b) Beacon

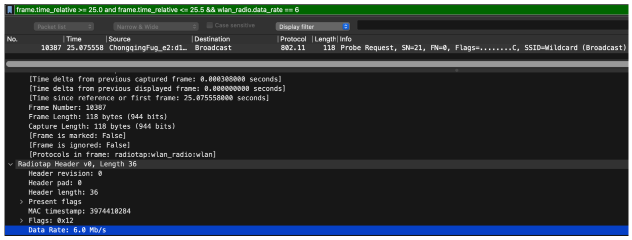
Figure 19: Different frame types

### 3.3.5 Data Rates

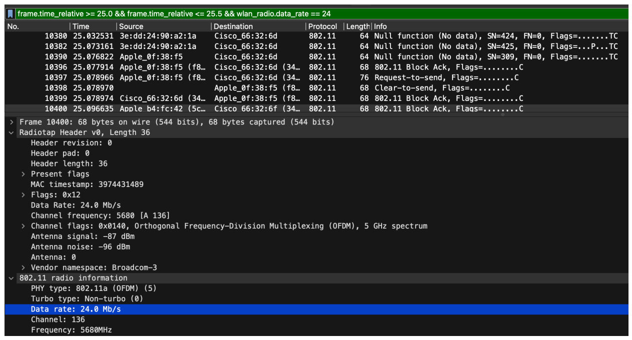
Packets in this time interval are working in various data rates. For example, so many of them are operating in 24 Mb/s but we only have one packet that is working in 6 Mb/s. The results are shown in Figure 20.

### 3.3.6 Signal Levels (SNR) of High-Traffic BSS

The SNR value can be found with other information related to it such as noise level and signal strenghth. Results available through Figure 21.




(a) 6 Mb/s




(b) 24 Mb/s

Figure 20: Data rate checking

### 3.3.7 MAC Addresses of Access Points

In order to find the access points, we should be aware of their MAC address. We can find each point as shown in Figure 22.

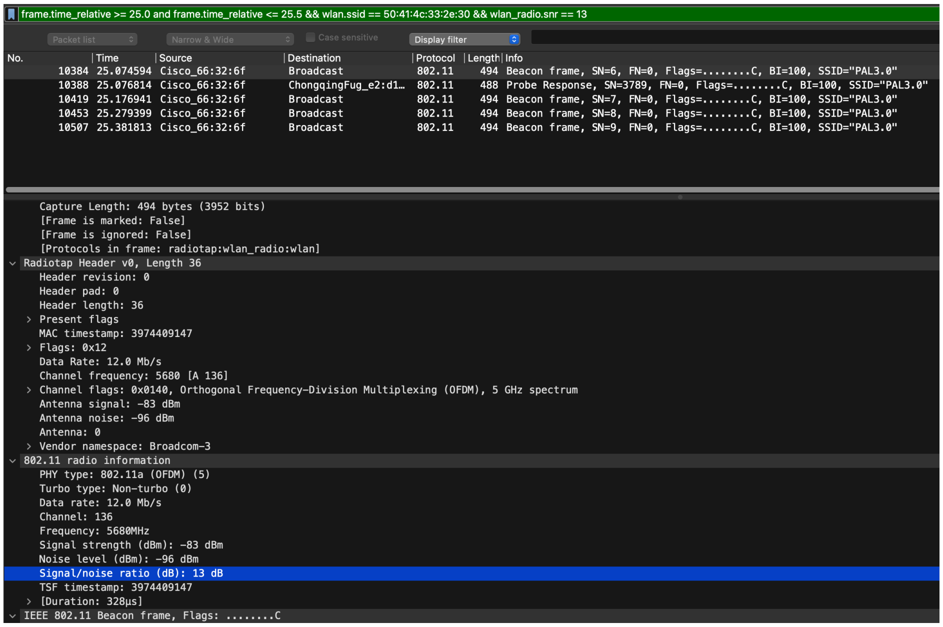


Figure 21: SNR checking

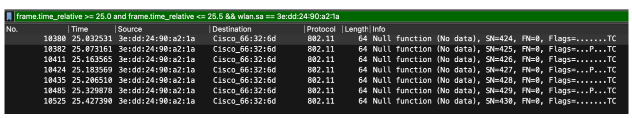


Figure 22: MAC address checking

## 3.4 Comparison

We used the same time interval for all three datasets, 25.0 - 25.5. Hence, we are going to see which parts had significant difference comparing them together. Overall, comparing all the datasets together, we will find out that datasets with more packets are having a wider variety which is a trivial information. Thus, we can see more types, data rates, etc as depicted in previous sections. Details for *657.pcap are not provided as it was just repeating the commands. However, it's data is used to check for differences between all datasets.

### 3.4.1 Network Types

In bigger datasets, we were able to see more network types. For example, in the smaller dataset we only saw Probe type but we could see both Probe and Beacon types in the bigger dataset.

### 3.4.2 Data Rate Types

As shown in previous sections, we have more rates in the given time for the two big datasets. For example, in the *307.pcap we are only using 6 Mb/s while in *517.pcap we are also using 24 Mb/s.

### 3.4.3 BSS

According to the assignment documentation, the data is gathered from university servers. So, in all datasets we can see most of the devices are connected to the internet through PAL3.0 or attwifi which is an interesting point to know.