ELEC-C7420 - Basic principles in networking

Tutorial on wireless measurement

Introduction

In real world networks it is sometimes desirable to observe the traffic passing through a link or host in order to detect anomalies, observe the behavior of the network and obtain important statistics. One way to perform this kind of troubleshooting is capturing packets on the NIC of a given network element using a sniffer or network protocol analyzer. In the industry there are several options when it comes to capturing traffic on the network, e.g probes Accedian, Fluke Networks, which are devices designed for this purpose. And, on the other hand, there are specialized software that are installed on the computer connected to the network segment to analyze. In this exercise we will use the software *Wireshark*, which is a common tool that will let us capture packets on the interface and analyze them. Once we have captured the network frames it is possible to observe their content, see the packet header, protocol type, filter the packets according to our needs and so on.

Background

TCP/IP encapsulation

On the internet the packets routing in the networks are encapsulated according to the OSI layer model or the TCP/IP stack protocols in order to follow the standards for each protocol, which are defined by different organizations like IEEE or IETF. Similarly, in the receiving host a process of de-encapsulation is conducted in order to retrieve the payload or data.

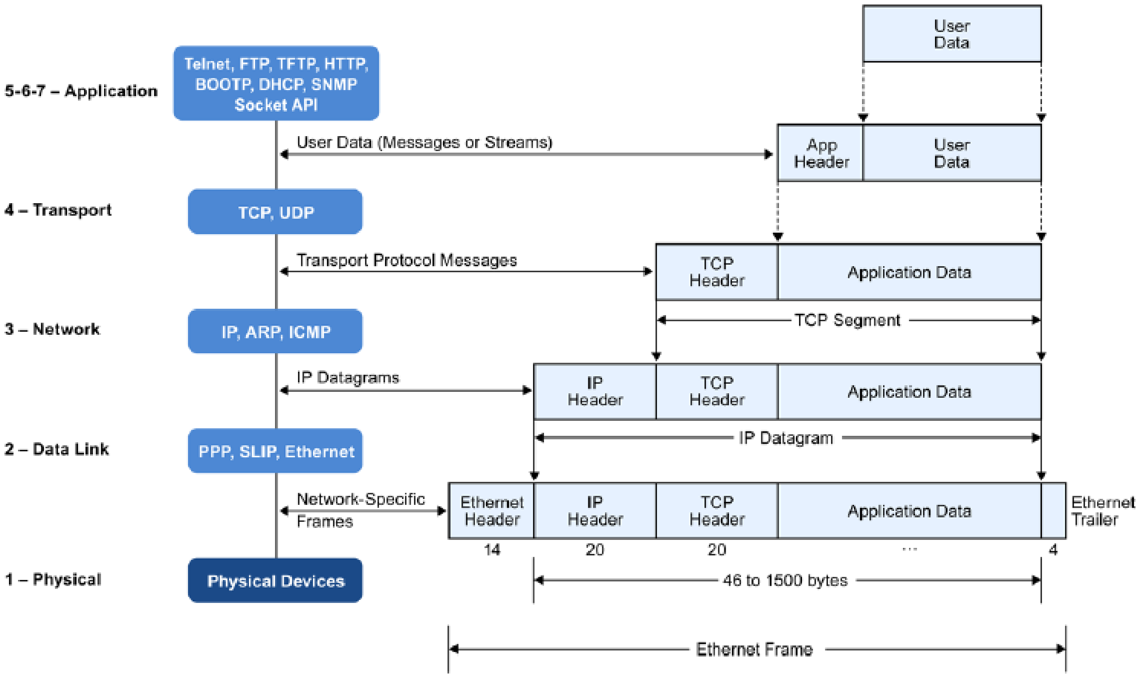


Figure 1. TCP/IP Network stack encapsulation

Wi-Fi:

Wi-Fi stands from Wireless Fidelity which is a general term that refers to the standard defined by the IEEE to define Wireless Local Area Networks (WLANs). This technology framework is defined in the IEEE 802.11 set of standards that comply with the regulations for this technology. Besides, there is the Wi-Fi Alliance that is an industry consortium that establishes the guidelines for the implementation and inter-operability, from different manufacturers, to the Wi-Fi. There are several Wi-Fi standards that have been developing along time, between the most popular implementations we have:

- IEEE 802.11b:
  - Appeared in 1999
  - Frequency band: 2.4 Ghz
  - Speed: 11Mbps (theoretical) working in 30m coverage area
  - May suffer interference from devices working in 2.4GHz

- IEEE 802.11a
  - Introduced in 2001
  - Frequency band: 5.0 Ghz
  - Speed: 54Mbps (theoretical)
  - Not compatible with IEEE 802.11b

- IEEE 802.11g
  - Introduced in 2003
  - Combination of features of the standards b and a
  - Speed: 54Mbps
  - Frequency band: 2.4 Ghz
  - Compatible with b

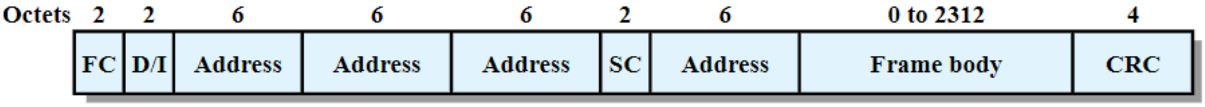
Elements of a Wi-Fi Network:

- Access Point (AP): when there is a infrastructure mode network, the AP is the element in charge of interconnect one or more hosts attached to it with each other or with other networks, e.g the Internet.

- Wi-Fi NIC: is the interface between the host and the AP, this device enables the wireless communication and the traffic.

- Hotspots: is a geographical area enabled with Wi-Fi access through one or more access points to allow users connect to the Internet. For example a public area with Wi-Fi access or a Café.

Following the 802.11 MAC frame format:



Octets	2	2	6	6	6	2	6	0 to 2312	4
	FC	D/I	Address	Address	Address	SC	Address	Frame body	CRC

FC = Frame control
D/I = Duration/connection ID
SC = Sequence control

Figure 2. 802.11 MAC format

- Frame Control: indicates the type of frame: data, management or control.
- Duration/Connection ID: when used as duration field, it indicates the time a channel is successful in the transmission of MAC frames.
- Address: depending on the context, it could be the MAC address of the Transmitter/Receiver or the MAC address of the Access Point.
- Sequence Control: it contains 4 bits to define fragmentation and a 12 bit sequence number for the frames exchanged between the transmitter and receiver.
- Frame body: contains the protocol data unit.
- Frame Check Sequence (FCS): contains the 32-bit Cyclic Redundancy Check.

Network sniffer

A network sniffer or a traffic sniffer is a tool used for observing the data traffic of the network passing through a given network interface in a host or device. The main function of this tool is to capture, or also *sniff* , the packets sent and received by the host where the sniffer is running with the objective of display or store the content of the protocols involved in the network interaction. This method of network analysis is considered to be passive because it captures all the traffic for further observation, however, the sniffer does not inject or modify any frame capture in/out the NIC to be analyzed.

In the figure 3 we can observe the basic structure of the network sniffer, which consists of the operating system running on the device, the applications that could be HTTP, FTP or SSH for instance and the packet sniffer. The packet sniffer contains a module that implements the capture of the packets (raw bits and bytes data) that are the bits encapsulated in the various protocols we studied in the lectures and a packet analyzer module which is a software implementation able to understand the protocols frame formats and display it accordingly. So, for example, the packet analyzer knows exactly how many bytes have each header or field in all the protocols that it is able to analyze, showing us the content for every packet in the right order.

During our practice experience we will use a packet sniffer called Wireshark [http://www.wireshark.org/] which is an open source network protocol analyzer that runs in several operating systems like Windows, Linux and macOS. It is able to read and write in various capture formats like tcpdump (libcap) and pcap. Also it is possible to capture and read live data from several data link layer protocols like Ethernet, Ethernet II or IEEE 802.11.

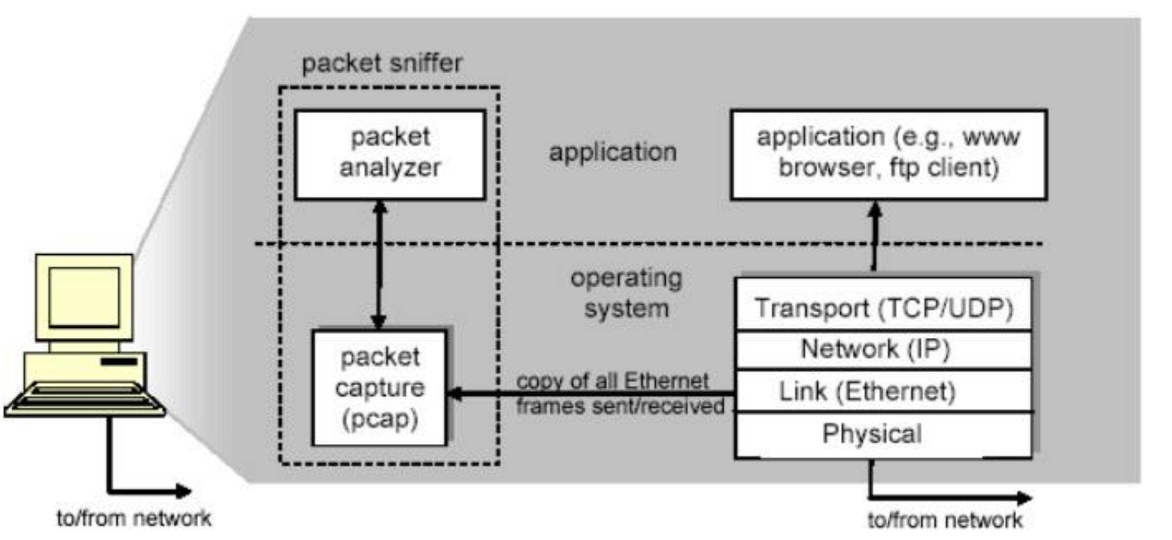


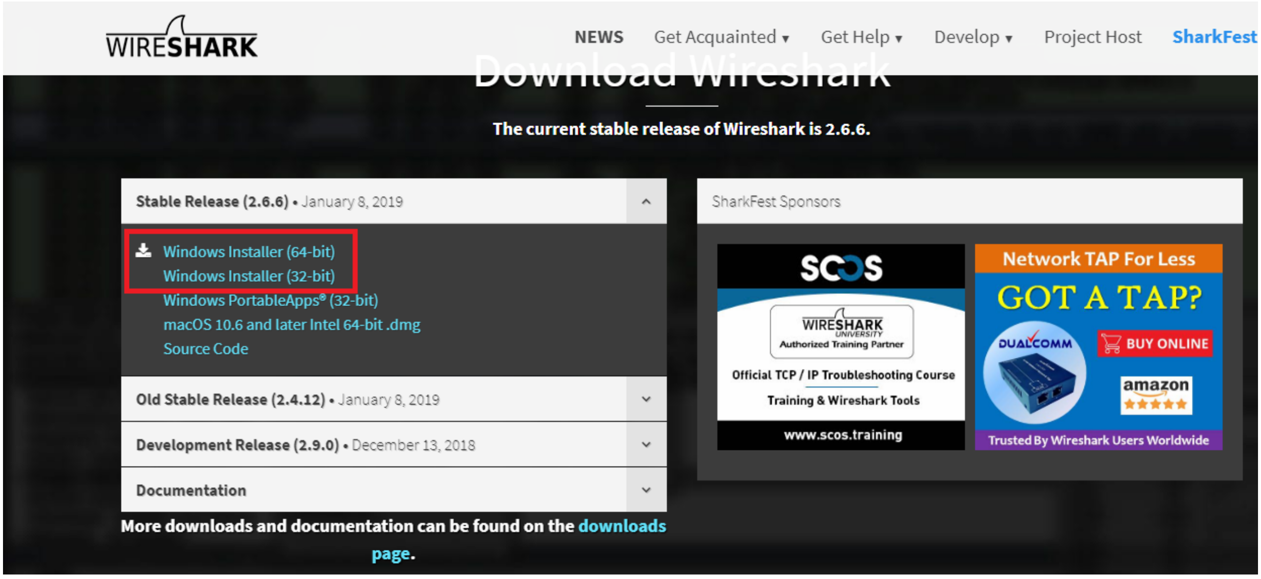
Figure 3. Network sniffer or packet sniffer structure

Getting Wireshark
Some Linux distributions have the Wireshark already installed, you can launch the application directly in that case. Otherwise, it can be downloaded from the following link:

https://www.wireshark.org/#download

Windows installation:

For windows installation select the following options according to your case and follow the step-by-step installation wizard:



Download Wireshark

The current stable release of Wireshark is 2.6.6.

Stable Release (2.6.6) • January 8, 2019

- Windows Installer (64-bit)
- Windows Installer (32-bit)
- Windows PortableApps® (32-bit)
- macOS 10.6 and later Intel 64-bit .dmg
- Source Code

Old Stable Release (2.4.12) • January 8, 2019

Development Release (2.9.0) • December 13, 2018

Documentation

More downloads and documentation can be found on the downloads page.

SharkFest Sponsors

SCOS
WIRESHARK UNIVERSITY Authorized Training Partner
Official TCP / IP Troubleshooting Course
Training & Wireshark Tools
www.scos.training
Network TAP For Less
GOT A TAP?
DUALCOMM
BUY ONLINE
amazon
Trusted By Wireshark Users Worldwide

Figure 4. Download windows installers

Ubuntu installation:

Add the stable official Personal Package Archive (PPA):

```
sudo add-apt-repository ppa:wireshark-dev/stable
```

Update the repositories:

```
sudo apt-get update
```

Install Wireshark:

```
sudo apt-get install wireshark
```

Debian installation:

Add the repositories on the apt source list:

```
sudo nano /etc/apt/sources.list
```

Add the following lines to the sources.list file:

```
deb http://ftp.debian.org/debian/ stable main contrib non-free
deb http://ftp.de.debian.org/debian jessie main
```

Update the repositories:

```
sudo apt-get update
```

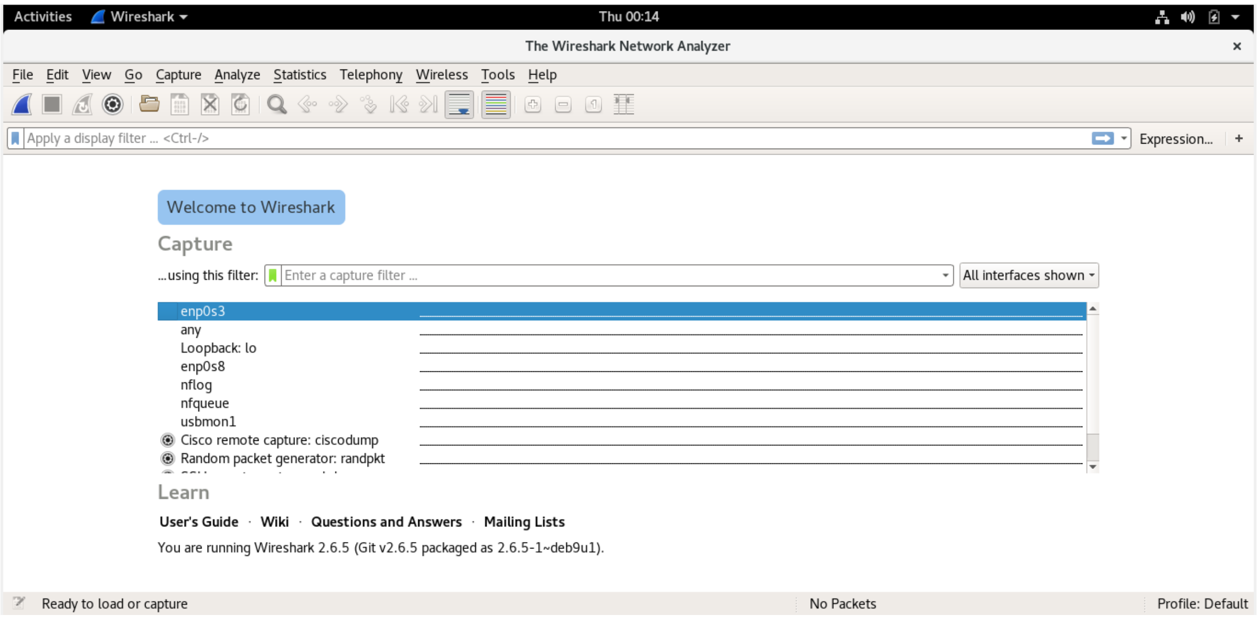
Install Wireshark:

```
sudo apt-get install wireshark
```

Note: if you have any issue or problem during the installation you can refer to the Q&A forum of wireshark for help on common issues: https://ask.wireshark.org/questions/

Getting started with Wireshark

When you launch the program you will get the Wireshark GUI like the one shown in figure 5. At this point the sniffer is not capturing any packet yet.


Activities
Wireshark
Thu 00:14
The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
+
Welcome to Wireshark
Capture
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
No Packets
Profile: Default

Apply a display filter ... <Ctrl-/>

...using this filter: Enter a capture filter ... All interfaces shown

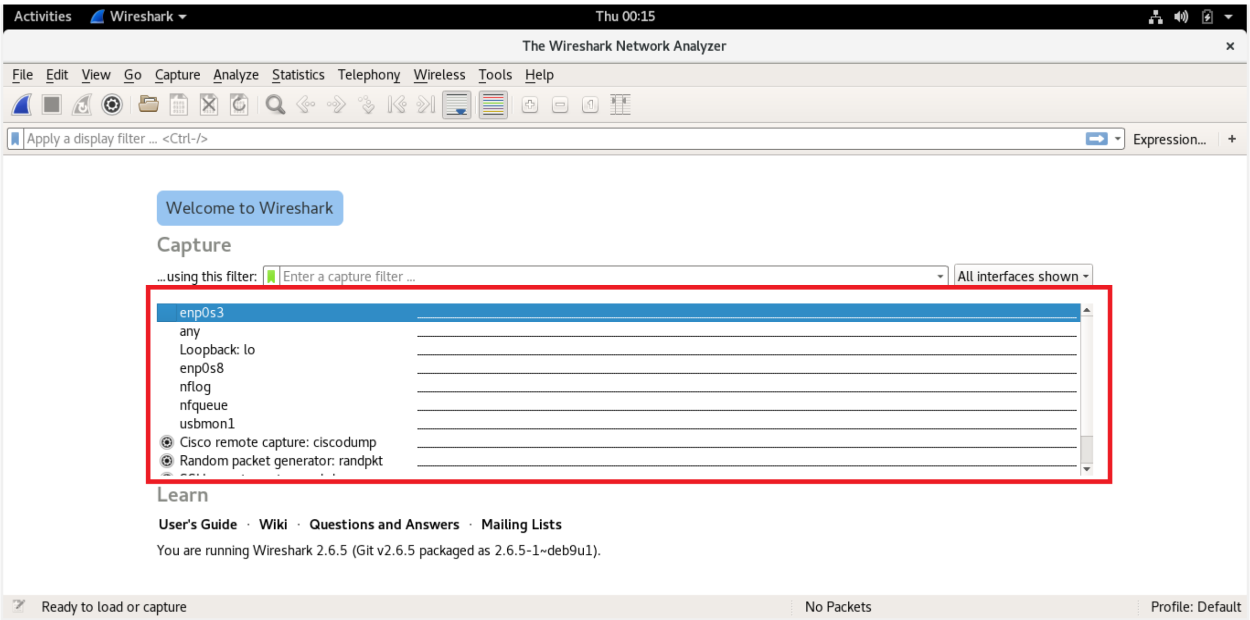
enp0s3
any
Loopback: lo
enp0s8
nflog
nfqueue
usbmon1
Cisco remote capture: ciscodump
Random packet generator: randpkt

You are running Wireshark 2.6.5 (Git v2.6.5 packaged as 2.6.5-1~deb9u1).

Ready to load or capture

Figure 5. Wireshark main window

Then, we must the select the interface where we wish to capture the traffic, e.g the wireless LAN interface or the Ethernet interface. Note that in some cases there could be multiple interfaces, we have to be sure which one is the one we are interested on.


Activities
Wireshark
Thu 00:15
The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
+
Welcome to Wireshark
Capture
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
No Packets
Profile: Default

Apply a display filter ... <Ctrl-/>

...using this filter: Enter a capture filter ... All interfaces shown

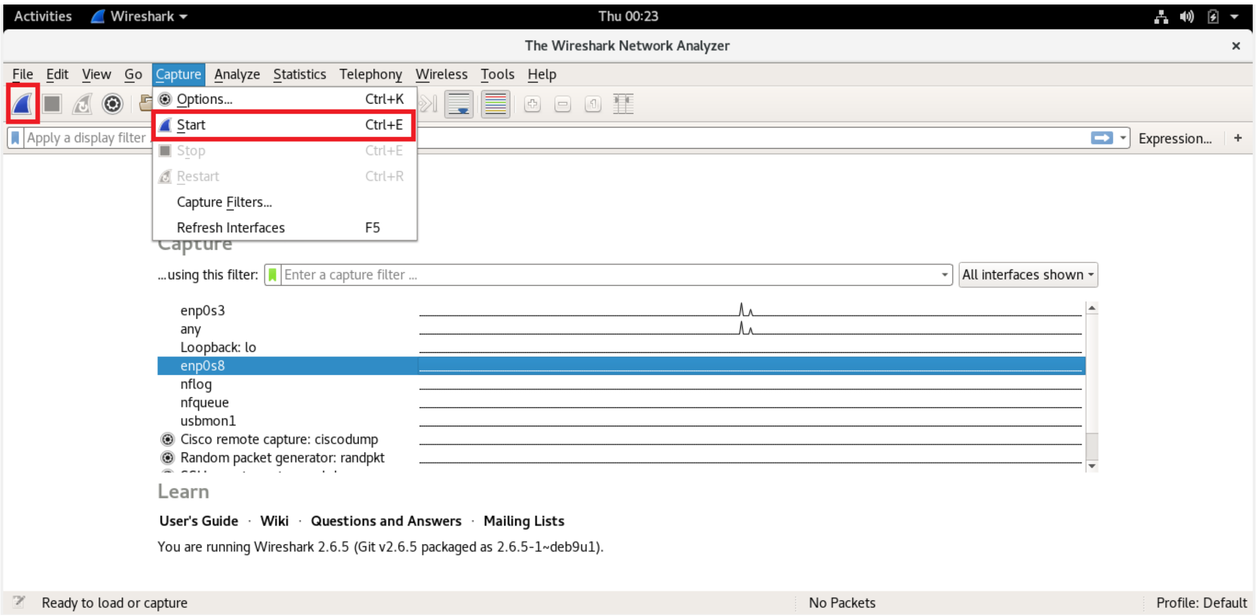
enp0s3
any
Loopback: lo
enp0s8
nflog
nfqueue
usbmon1
Cisco remote capture: ciscodump
Random packet generator: randpkt

You are running Wireshark 2.6.5 (Git v2.6.5 packaged as 2.6.5-1~deb9u1).

Ready to load or capture

Figure 6. List of interfaces

Once we select the desired interface where we will capture the traffic, click on the menu Capture → Start (Control + E) to start capturing the traffic going through this interface, or click the corresponding button according to the figure 7:

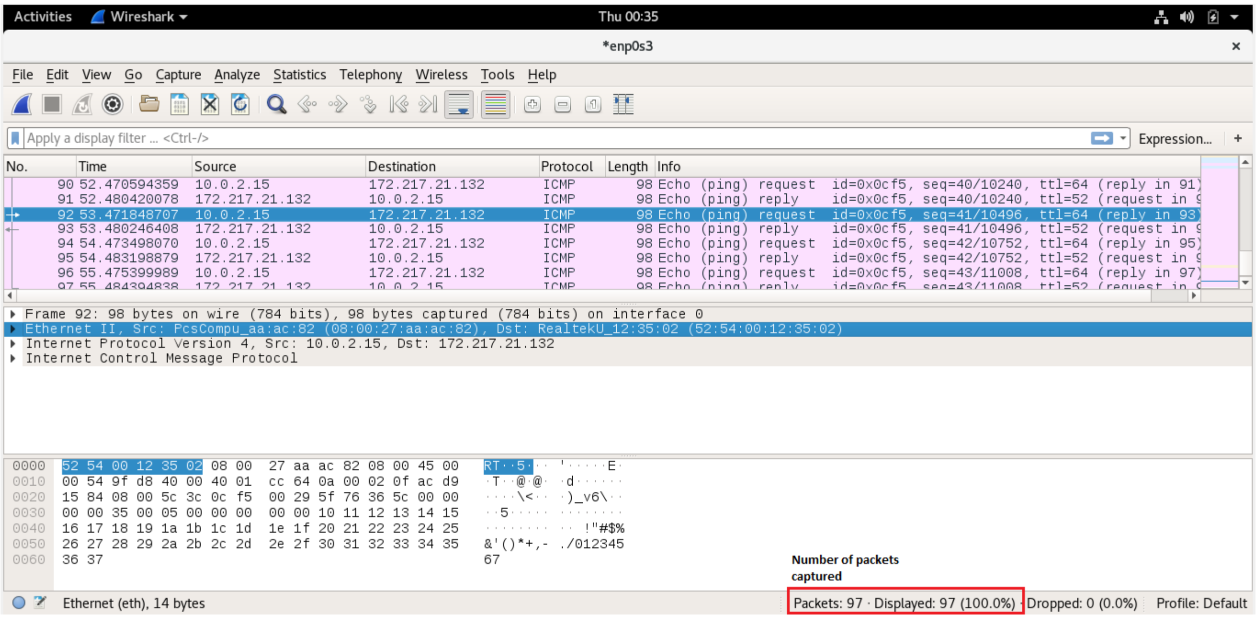


You are running Wireshark 2.6.5 (Git v2.6.5 packaged as 2.6.5-1~deb9u1).

Ready to load or capture | No Packets | Profile: Default

Figure 7. Starting the capture

When the capture is started we can observe the packets being captured on the screen in real-time and the packet counting increasing as shown in the figure 8:


Activities
Wireshark
Thu 00:35
*enp0s3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No. Time           Source          Destination     Protocol Length Info
 90 52.470594359   10.0.2.15       172.217.21.132  ICMP     98 Echo (ping) request  id=0x0cf5, seq=40/10240, ttl=64 (reply in 91)
 91 52.480420078   172.217.21.132  10.0.2.15       ICMP     98 Echo (ping) reply    id=0x0cf5, seq=40/10240, ttl=52 (request in 9
 92 53.471848707   10.0.2.15       172.217.21.132  ICMP     98 Echo (ping) request  id=0x0cf5, seq=41/10496, ttl=64 (reply in 93)
 93 53.480246408   172.217.21.132  10.0.2.15       ICMP     98 Echo (ping) reply    id=0x0cf5, seq=41/10496, ttl=52 (request in 9
 94 54.473498070   10.0.2.15       172.217.21.132  ICMP     98 Echo (ping) request  id=0x0cf5, seq=42/10752, ttl=64 (reply in 95)
 95 54.483198879   172.217.21.132  10.0.2.15       ICMP     98 Echo (ping) reply    id=0x0cf5, seq=42/10752, ttl=52 (request in 9
 96 55.475399989   10.0.2.15       172.217.21.132  ICMP     98 Echo (ping) request  id=0x0cf5, seq=43/11008, ttl=64 (reply in 97)
 97 55.484394838   172.217.21.132  10.0.2.15       ICMP     98 Echo (ping) reply    id=0x0cf5, seq=43/11008, ttl=52 (request in 9
```

```
▸ Frame 92: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 172.217.21.132
▸ Internet Control Message Protocol
```

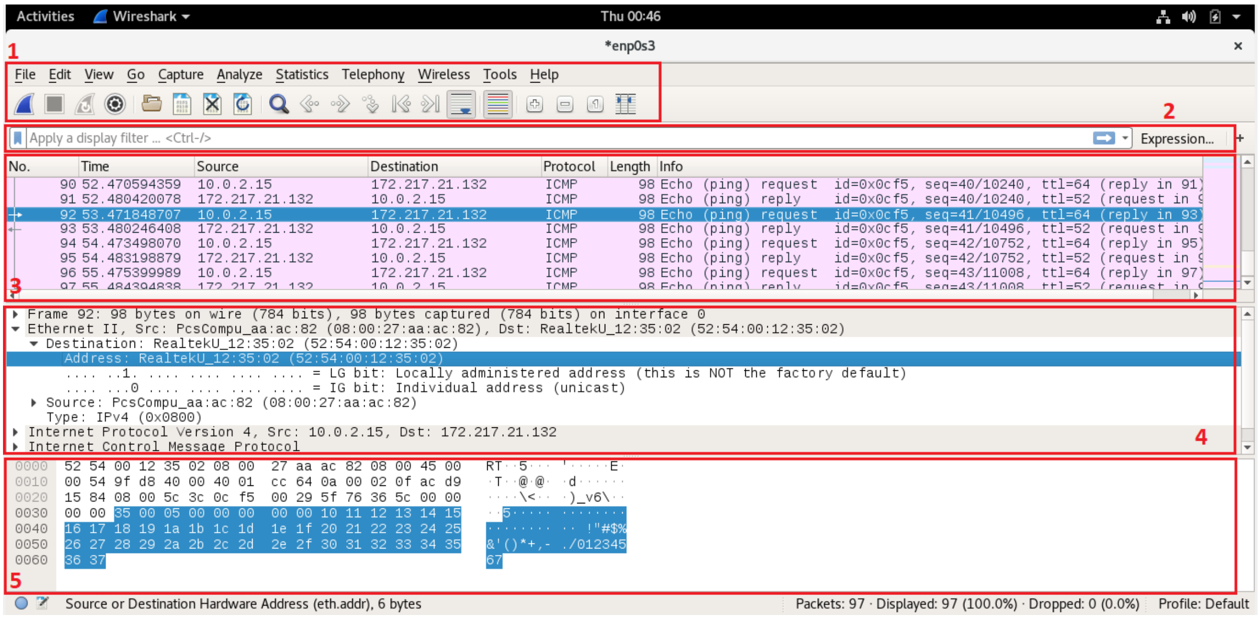
```
0000  52 54 00 12 35 02 08 00  27 aa ac 82 08 00 45 00   RT··5···'·····E·
0010  00 54 9f d8 40 00 40 01  cc 64 0a 00 02 0f ac d9   ·T··@·@··d······
0020  15 84 08 00 5c 3c 0c f5  00 29 5f 76 36 5c 00 00   ····\<···)_v6\··
0030  00 00 35 00 05 00 00 00  00 00 10 11 12 13 14 15   ··5·············
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ·········· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,-./012345
0060  36 37                                              67
```

Number of packets captured

Ethernet (eth), 14 bytes | Packets: 97 · Displayed: 97 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

Figure 8. Process of packet capture

The resulting capture screen has five important elements where we can obtain the information of the packets, see the content and values of each protocol header and filter the trace according to our needs. These components are shown in the figure 9:


Activities
Wireshark
Thu 00:46
*enp0s3
1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
2
Apply a display filter ... <Ctrl-/>
Expression...
3
4
5

```
No.   Time            Source           Destination      Protocol Length Info
   90 52.470594359    10.0.2.15        172.217.21.132   ICMP     98 Echo (ping) request  id=0x0cf5, seq=40/10240, ttl=64 (reply in 91)
   91 52.480420078    172.217.21.132   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x0cf5, seq=40/10240, ttl=52 (request in 9
   92 53.471848707    10.0.2.15        172.217.21.132   ICMP     98 Echo (ping) request  id=0x0cf5, seq=41/10496, ttl=64 (reply in 93)
   93 53.480246408    172.217.21.132   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x0cf5, seq=41/10496, ttl=52 (request in 9
   94 54.473498070    10.0.2.15        172.217.21.132   ICMP     98 Echo (ping) request  id=0x0cf5, seq=42/10752, ttl=64 (reply in 95)
   95 54.483198879    172.217.21.132   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x0cf5, seq=42/10752, ttl=52 (request in 9
   96 55.475399989    10.0.2.15        172.217.21.132   ICMP     98 Echo (ping) request  id=0x0cf5, seq=43/11008, ttl=64 (reply in 97)
   97 55.484394838    172.217.21.132   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x0cf5, seq=43/11008, ttl=52 (request in 9
```

```
▸ Frame 92: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▾ Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
      Address: RealtekU_12:35:02 (52:54:00:12:35:02)
      .... ..1. .... .... .... .... = LG bit: Locally administered address (this is NOT the factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▸ Source: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82)
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 172.217.21.132
▸ Internet Control Message Protocol
```

```
0000  52 54 00 12 35 02 08 00  27 aa ac 82 08 00 45 00   RT··5···'·····E·
0010  00 54 9f d8 40 00 40 01  cc 64 0a 00 02 0f ac d9   ·T··@·@··d······
0020  15 84 08 00 5c 3c 0c f5  00 29 5f 76 36 5c 00 00   ····\<···)_v6\··
0030  00 00 35 00 05 00 00 00  00 00 10 11 12 13 14 15   ··5·············
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ·········· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,-./012345
0060  36 37                                              67
```

Source or Destination Hardware Address (eth.addr), 6 bytes | Packets: 97 · Displayed: 97 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

Figure 9. Wireshark GUI

1) **Command menus:** they are the dropdown menus where we can start or stop captures, open or save previous captures, select interfaces, exit the application and obtain useful statistics from the trace. Also there are shortcut buttons below the menus that help us to perform several tasks quickly.

2) **Filter box:** in this dialog box we can apply filters to the captured packets to match our desired criteria with the objective to find specific packets, port numbers, protocols, MAC addresses, IP addresses and so on. A detail review on how to use filters in wireshark will be discussed in the following section.

3) **Packet list:** the packet list shows all the packets captured during the sniffing process or in a previous capture opened in Wireshark. It has several columns that help us with the packet reading:

   - No.: contains the packet number in order ascending order, i.e if a packet has a position number higher than other packet it means that this packet has been captured after the packet with lower number.

   - Time: shows the time when a given packet has been captured. Note that this time is not the UTC or epoc time, i.e the first packet was captured in the second 0 and so on.

   - Source: contains the source IP address of the packet

   - Destination: contains the destination IP address of the packet

   - Protocol: shows the protocol encapsulating the corresponding packet

   - Length: contains the length in bytes of the packet.

   - Info: contains a short description of the packet.

4) **Details:** in this pane we can observe the details of the header of the selected packet, all the encapsulation protocols and the fields of each one, as well as the values of every field.

5) **Content:** in this section we observe the content of the captured frame in both ASCII (right column) and hexadecimal (left column)

[Note: for a more detailed information of every menu and feature, please refer to the Wireshark user guide: https://www.wireshark.org/download/docs/user-guide.pdf]

Applying filters in Wireshark

The Wireshark GUI and also the command line implementation (Tshark) possess a powerful filter section which allows us to eliminate the packets or information that is not our interest and focus only in the packets that matters for our analysis. It means, that if a packet or packets meet the filter criteria that we implement, they will be the only packets listed on the packet trace facilitating the analysis and troubleshooting. The filters, for example, let us compare the fields of a specific protocol against a desired value, match fields with other fields and validate the capture of a certain protocol.

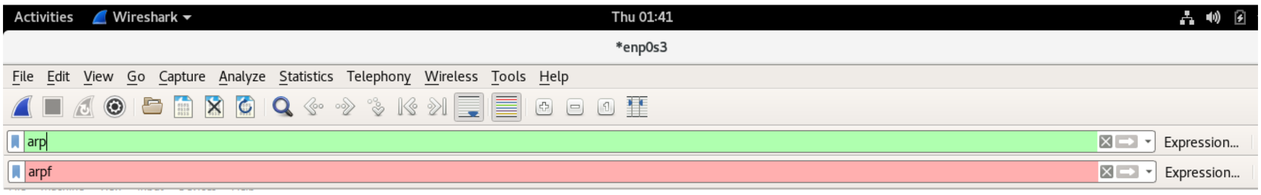
The fields values on the packet headers can be compared with other values as well. This comparison operators can be expressed using English-like abbreviations or using C-like symbols:

`eq, ==`	Equal
`ne, !=`	Not Equal
`gt, >`	Greater Than
`lt, <`	Less Than
`ge, >=`	Greater than or Equal to
`le, <=`	Less than or Equal to

Also, the filters supports the multiple test with joint Boolean connectors:

&&	- logical conjunction (i.e. AND)
||	- logical disjunction (i.e. OR)
!	- logical negation (i.e. NOT)

Note that the filter box turns red when the filter expression is invalid, it turns green when the filter expression is valid and turns yellow when it may produce unexpected results:

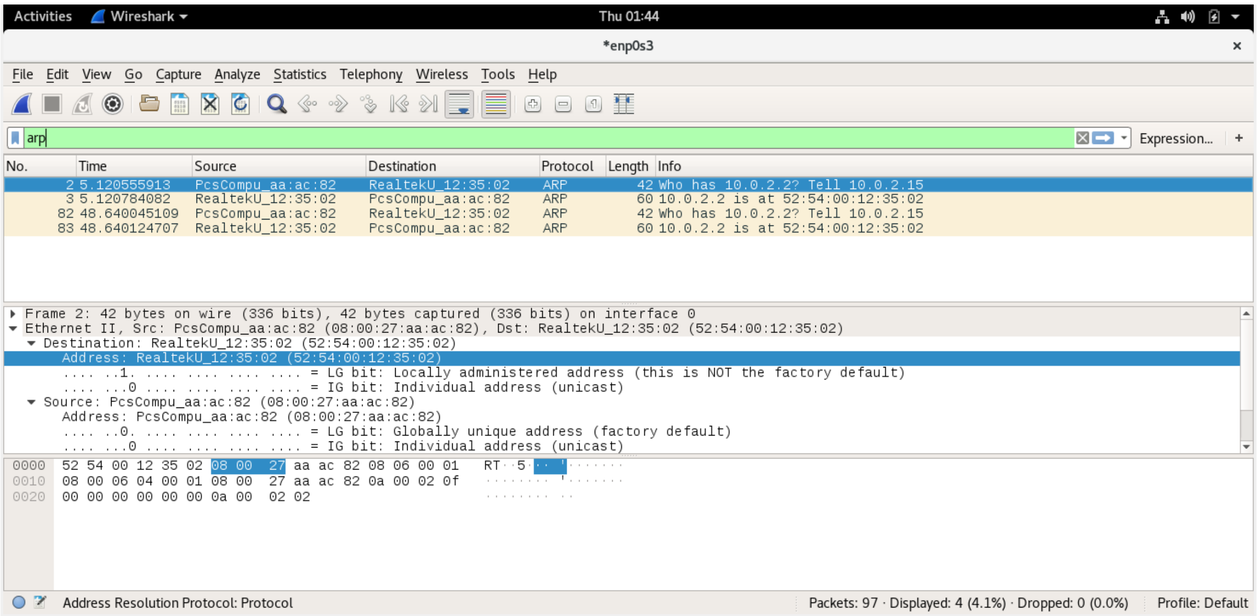

Activities
Wireshark
Thu 01:41
*enp0s3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
Expression...

arp

arpf

Figure 10. Filters with valid and invalid expressions

A simple filter expression could be the packet protocol name. If we input the protocol “ARP” as shown in figure 11, Wireshark will only display the ARP protocol packets:


Activities
Wireshark
Thu 01:44
*enp0s3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
+

arp

```
No.  Time          Source             Destination        Protocol Length Info
   2 5.120555913   PcsCompu_aa:ac:82  RealtekU_12:35:02  ARP          42 Who has 10.0.2.2? Tell 10.0.2.15
   3 5.120784082   RealtekU_12:35:02  PcsCompu_aa:ac:82  ARP          60 10.0.2.2 is at 52:54:00:12:35:02
  82 48.640045109  PcsCompu_aa:ac:82  RealtekU_12:35:02  ARP          42 Who has 10.0.2.2? Tell 10.0.2.15
  83 48.640124707  RealtekU_12:35:02  PcsCompu_aa:ac:82  ARP          60 10.0.2.2 is at 52:54:00:12:35:02
```

```
▸ Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▾ Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
      Address: RealtekU_12:35:02 (52:54:00:12:35:02)
      .... ..1. .... .... .... .... = LG bit: Locally administered address (this is NOT the factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▾ Source: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82)
      Address: PcsCompu_aa:ac:82 (08:00:27:aa:ac:82)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
```

```
0000  52 54 00 12 35 02 08 00  27 aa ac 82 08 06 00 01   RT··5···'·······
0010  08 00 06 04 00 01 08 00  27 aa ac 82 0a 00 02 0f   ········'·······
0020  00 00 00 00 00 00 0a 00  02 02                     ··········
```

Address Resolution Protocol: Protocol   Packets: 97 · Displayed: 4 (4.1%) · Dropped: 0 (0.0%)   Profile: Default

Figure 11. Filters showing only ARP packets

Following another filter examples:

- **`http.request`** – Display all HTTP requests.
- **`http.request || http.response`** – Display all HTTP request and responses.
- **`ip.addr == 127.0.0.1`** – Display all IP packets whose source or destination is localhost.
- **`tcp.len < 100`** – Display all TCP packets whose data length is less than 100 bytes.
- **`http.request.uri matches “(gif)$”`** - Display all HTTP requests in which the uri ends with “gif”.
- **`dns.query.name == “www.google.com”`** - Display all DNS queries for “www.google.com”.

[Note: the complete detailed list of all the filters available to apply in Wireshark can be found in the following link: https://www.wireshark.org/docs/dfref/ . Additionally, for further information regarding filters syntax and manuals, may be found in the following link: https://www.wireshark.org/docs/man-pages/wireshark-filter.html]

Wi-Fi monitoring
In this experiment we will capture traffic on an 802.11 network interface and we will observe how the 802.11 MAC frame is, the detail of its fields, the beacon frames and the retransmissions. Through this experiment we will understand how the 802.11 Wireless LAC MAC format and its fields.

Preparing the testbed
Capturing wireless frames on Windows may be problematic since it depends mainly on the drivers installed in the system and the version of Windows, most of the wlan cards do not support the monitor mode in windows, for this reason there are special tools like aircrack-ng and kismet. However, we will use Ubuntu to capture the Wi-Fi traffic on Wireshark:

Check the name of the wireless interfaces with the following command:

```
iwconfig
```

We should obtain a result similar to the following:

```
lo        no wireless extensions.

eth0      no wireless extensions.

sit0      no wireless extensions.

wlan0     unassociated  ESSID:off/any
          Mode:Managed  Channel=0  Access Point: 00:00:00:00:00:00
          Bit Rate=0 kb/s   Tx-Power:off
          Retry:on   RTS thr:off   Fragment thr:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

In this case the wireless interface is `wlan0.` Now we should set it to monitor mode:

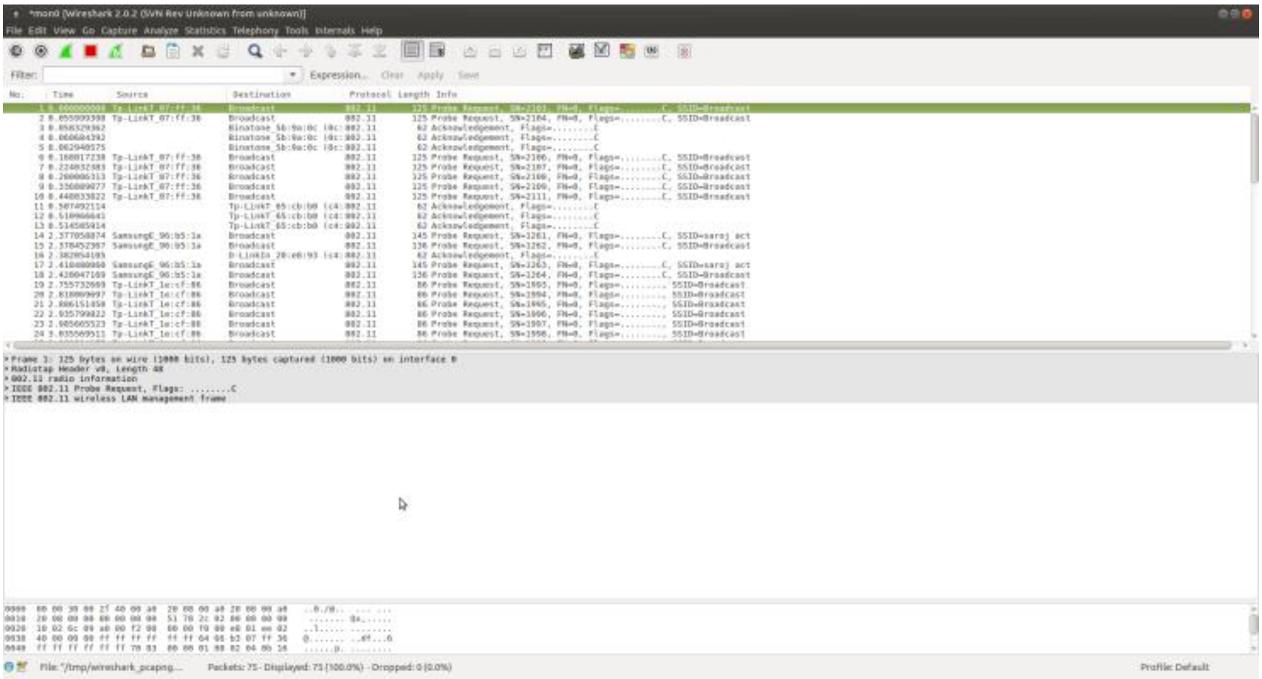
```
iwconfig wlan0 mode monitor
```

Sometimes, it is not possible to change the mode if the interface is up, for this reason we can try to disable first, change to monitor mode and then enable the interface again:

```
ifconfig wlan0 down
iwconfig wlan0 mode monitor
ifconfig wlan0 up
```

Once we have configured our wireless interface in monitor mode we are able to capture traffic on the 802.11 protocol using wireshark. The way our devices know the name of the wireless networks configured in the access point is through the so-called beacon frames, which is a frame sent by the AP in a constant interval, it contains all the information about the Wi-Fi network. The wireless NIC will send probe request frames to all the access points in the coverage area (broadcast) and the AP will response with its beacon, this kind of frames are called management frames in the

802.11 protocol, the other frames are control frames and data frames. Now we capture the traffic on the wireless interface (take in account that the device must not be connected to any wireless network in order to have a better observation of the frames). After starting the capture we should obtain a trace similar to the following:


*mon0 [Wireshark 2.0.2 (SVN Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.   Time         Source               Destination          Protocol Length Info
  1 0.000000000 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2103, FN=0, Flags=........C, SSID=Broadcast
  2 0.055999398 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2104, FN=0, Flags=........C, SSID=Broadcast
  3 0.058329362                      Binatone_5b:0a:0c (0c:802.11    62 Acknowledgement, Flags=........C
  4 0.060684392                      Binatone_5b:0a:0c (0c:802.11    62 Acknowledgement, Flags=........C
  5 0.062940575                      Binatone_5b:0a:0c (0c:802.11    62 Acknowledgement, Flags=........C
  6 0.168017238 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2106, FN=0, Flags=........C, SSID=Broadcast
  7 0.224032383 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2107, FN=0, Flags=........C, SSID=Broadcast
  8 0.280006313 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2108, FN=0, Flags=........C, SSID=Broadcast
  9 0.336089077 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2109, FN=0, Flags=........C, SSID=Broadcast
 10 0.448033822 Tp-LinkT_07:ff:38    Broadcast            802.11   125 Probe Request, SN=2111, FN=0, Flags=........C, SSID=Broadcast
 11 0.507492114                      Tp-LinkT_65:cb:b0 (c4:802.11    62 Acknowledgement, Flags=........C
 12 0.510966641                      Tp-LinkT_65:cb:b0 (c4:802.11    62 Acknowledgement, Flags=........C
 13 0.514585914                      Tp-LinkT_65:cb:b0 (c4:802.11    62 Acknowledgement, Flags=........C
 14 2.377058874 SamsungE_96:b5:1a    Broadcast            802.11   145 Probe Request, SN=1261, FN=0, Flags=........C, SSID=saroj act
 15 2.378452987 SamsungE_96:b5:1a    Broadcast            802.11   136 Probe Request, SN=1262, FN=0, Flags=........C, SSID=Broadcast
 16 2.382054105                      D-LinkIn_20:e0:93 (c4:802.11    62 Acknowledgement, Flags=........C
 17 2.418080960 SamsungE_96:b5:1a    Broadcast            802.11   145 Probe Request, SN=1263, FN=0, Flags=........C, SSID=saroj act
 18 2.420047169 SamsungE_96:b5:1a    Broadcast            802.11   136 Probe Request, SN=1264, FN=0, Flags=........C, SSID=Broadcast
 19 2.755732609 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1993, FN=0, Flags=........, SSID=Broadcast
 20 2.818009097 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1994, FN=0, Flags=........, SSID=Broadcast
 21 2.886151458 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1995, FN=0, Flags=........, SSID=Broadcast
 22 2.935799822 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1996, FN=0, Flags=........, SSID=Broadcast
 23 2.985665523 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1997, FN=0, Flags=........, SSID=Broadcast
 24 3.035569511 Tp-LinkT_1e:cf:86    Broadcast            802.11    86 Probe Request, SN=1998, FN=0, Flags=........, SSID=Broadcast
```

```
▸ Frame 1: 125 bytes on wire (1000 bits), 125 bytes captured (1000 bits) on interface 0
▸ Radiotap Header v0, Length 48
▸ 802.11 radio information
▸ IEEE 802.11 Probe Request, Flags: ........C
▸ IEEE 802.11 wireless LAN management frame
```

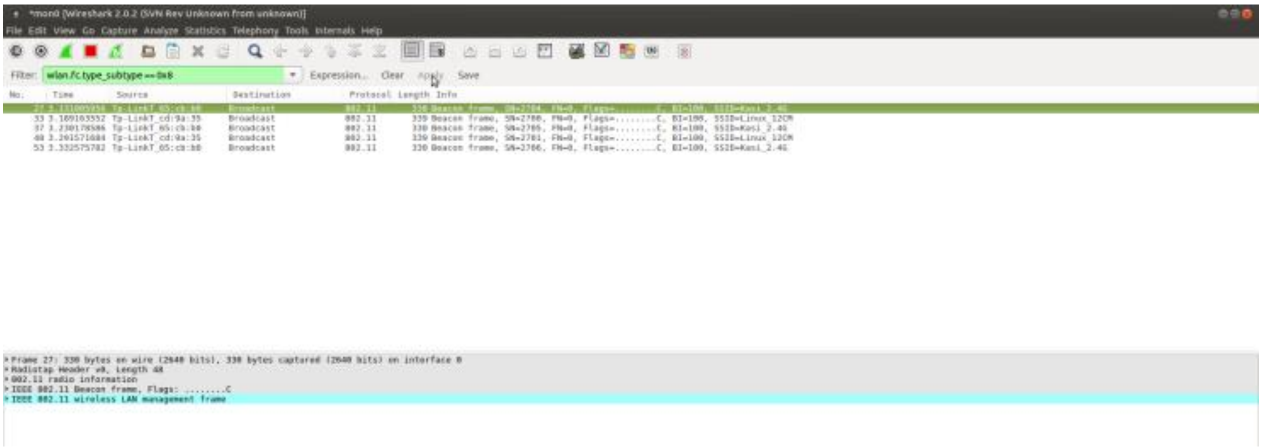
```
0000  00 00 30 00 2f 40 00 a0  20 08 00 a0 20 08 00 a0   ..0./@.. .... ...
0010  20 08 00 00 00 00 00 00  51 78 2c 02 00 00 00 00   ........ Qx,.....
0020  10 02 6c 09 a0 00 f2 00  00 00 f0 00 e8 01 ee 02   ..l..... ........
0030  40 00 00 00 ff ff ff ff  ff ff 64 66 b3 07 ff 36   @....... ..df...6
0040  ff ff ff ff ff ff 70 83  00 00 01 08 02 04 0b 16   ......p. ........
```

File: "/tmp/wireshark_pcapng... Packets: 75 · Displayed: 75 (100.0%) · Dropped: 0 (0.0%) Profile: Default

Figure 12. Packet capture on the wireless interface

Now we apply a filter to the wireshark to obtain only the beacon frames. In this case we have to consider that the beacon frames are part of the management frames with sub-type value of 8, for this reason we apply the following filter in the filter box:

```
wlan.fc.type_subtype == 0x8
```


*mon0 [Wireshark 2.0.2 (SVN Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

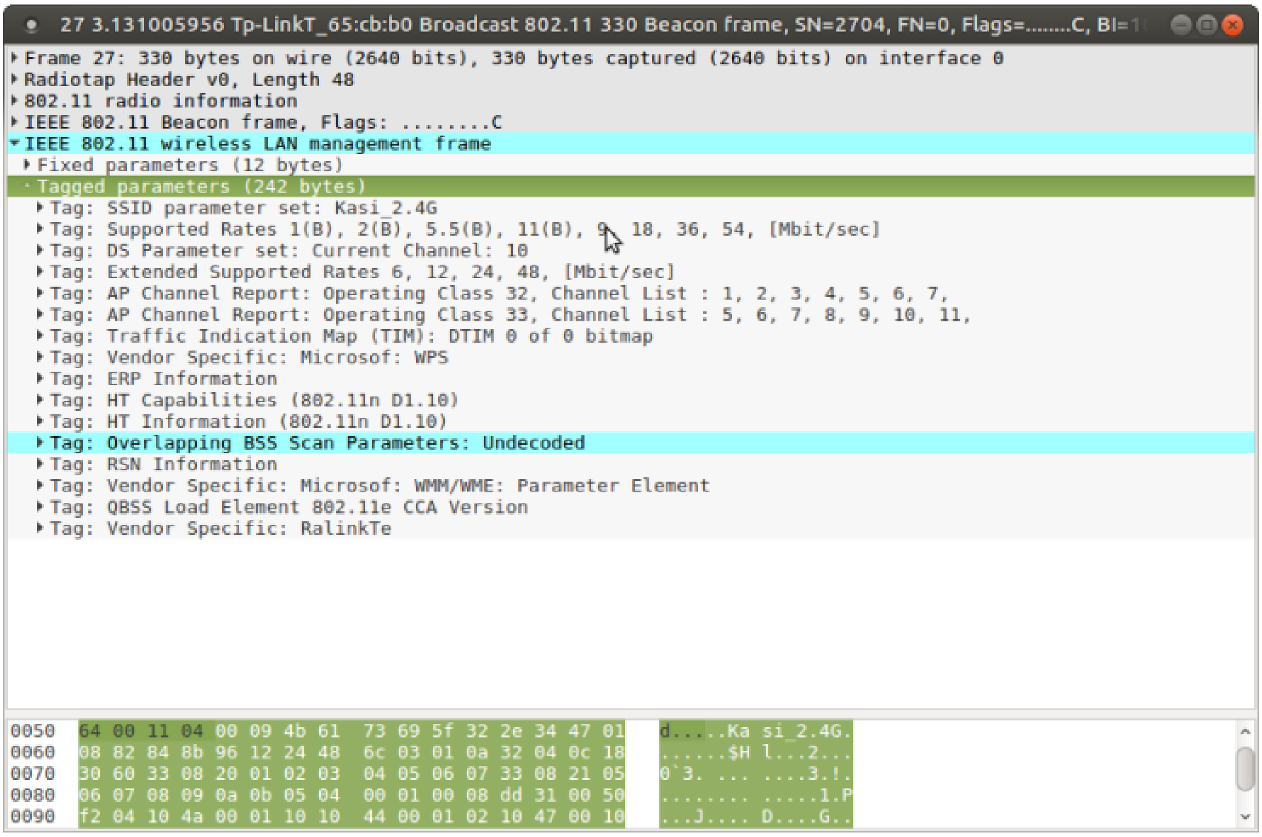
Filter: wlan.fc.type_subtype == 0x8

```
No.   Time         Source               Destination          Protocol Length Info
 27 3.131005958 Tp-LinkT_65:cb:b0    Broadcast            802.11   330 Beacon frame, SN=2704, FN=0, Flags=........C, BI=100, SSID=Kasi_2.4G
 33 3.169163552 Tp-LinkT_cd:9a:35    Broadcast            802.11   339 Beacon frame, SN=2780, FN=0, Flags=........C, BI=100, SSID=Linux_12CM
 37 3.230178586 Tp-LinkT_65:cb:b0    Broadcast            802.11   330 Beacon frame, SN=2705, FN=0, Flags=........C, BI=100, SSID=Kasi_2.4G
 48 3.291571684 Tp-LinkT_cd:9a:35    Broadcast            802.11   339 Beacon frame, SN=2781, FN=0, Flags=........C, BI=100, SSID=Linux_12CM
 53 3.332575782 Tp-LinkT_65:cb:b0    Broadcast            802.11   330 Beacon frame, SN=2706, FN=0, Flags=........C, BI=100, SSID=Kasi_2.4G
```

```
▸ Frame 27: 330 bytes on wire (2640 bits), 330 bytes captured (2640 bits) on interface 0
▸ Radiotap Header v0, Length 48
▸ 802.11 radio information
▸ IEEE 802.11 Beacon frame, Flags: ........C
▸ IEEE 802.11 wireless LAN management frame
```

Figure 12. Packet capture on the wireless interface

In this case there were only 5 beacon frames, however, it depends on the environment where the capture is being conducted, if there are more access points nearby, then the capture should have lot more beacon frames. Now we look closer to the beacon frame, in your capture you can explore every field and every value of the frame:



27 3.131005956 Tp-LinkT_65:cb:b0 Broadcast 802.11 330 Beacon frame, SN=2704, FN=0, Flags=........C, BI=1

```
▸Frame 27: 330 bytes on wire (2640 bits), 330 bytes captured (2640 bits) on interface 0
▸Radiotap Header v0, Length 48
▸802.11 radio information
▸IEEE 802.11 Beacon frame, Flags: ........C
▾IEEE 802.11 wireless LAN management frame
  ▸Fixed parameters (12 bytes)
  ▾Tagged parameters (242 bytes)
    ▸Tag: SSID parameter set: Kasi_2.4G
    ▸Tag: Supported Rates 1(B), 2(B), 5.5(B), 11(B), 9, 18, 36, 54, [Mbit/sec]
    ▸Tag: DS Parameter set: Current Channel: 10
    ▸Tag: Extended Supported Rates 6, 12, 24, 48, [Mbit/sec]
    ▸Tag: AP Channel Report: Operating Class 32, Channel List : 1, 2, 3, 4, 5, 6, 7,
    ▸Tag: AP Channel Report: Operating Class 33, Channel List : 5, 6, 7, 8, 9, 10, 11,
    ▸Tag: Traffic Indication Map (TIM): DTIM 0 of 0 bitmap
    ▸Tag: Vendor Specific: Microsof: WPS
    ▸Tag: ERP Information
    ▸Tag: HT Capabilities (802.11n D1.10)
    ▸Tag: HT Information (802.11n D1.10)
    ▸Tag: Overlapping BSS Scan Parameters: Undecoded
    ▸Tag: RSN Information
    ▸Tag: Vendor Specific: Microsof: WMM/WME: Parameter Element
    ▸Tag: QBSS Load Element 802.11e CCA Version
    ▸Tag: Vendor Specific: RalinkTe
```

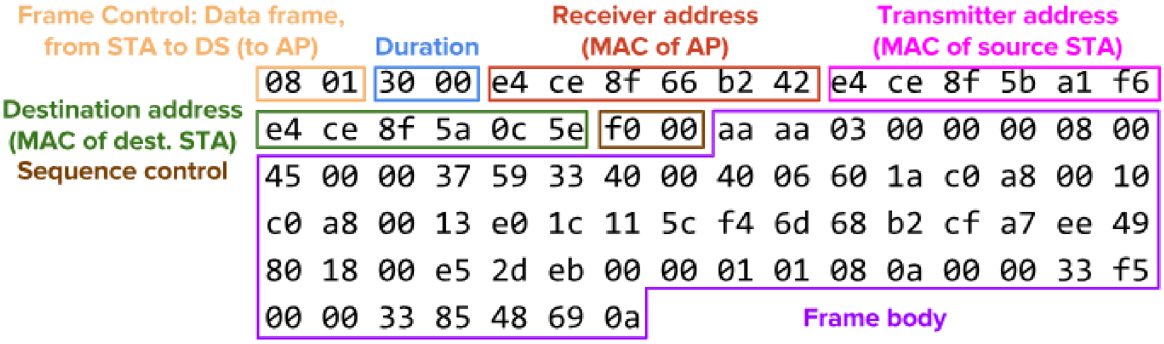
```
0050  64 00 11 04 00 09 4b 61  73 69 5f 32 2e 34 47 01   d.....Ka si_2.4G.
0060  08 82 84 8b 96 12 24 48  6c 03 01 0a 32 04 0c 18   .......$H l...2...
0070  30 60 33 08 20 01 02 03  04 05 06 07 33 08 21 05   0`3. ... ....3.!.
0080  06 07 08 09 0a 0b 05 04  00 01 00 08 dd 31 00 50   ........ .....1.P
0090  f2 04 10 4a 00 01 10 10  44 00 01 02 10 47 00 10   ...J.... D....G..
```

Figure 13. Detail of the beacon frame

One important field to observe is the SSID, which give us the name of the wireless network configured on the Access Point. Also, the beacon frame format changes depending on the version of 802.11 protocol that is being implemented in the Access Point.

Monitoring the 802.11 frame

From the 802.11 frames captured we can identify the fields from the captured packets:



```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

Frame Control: Data frame, from STA to DS (to AP)
Duration
Receiver address (MAC of AP)
Transmitter address (MAC of source STA)
Destination address (MAC of dest. STA)
Sequence control
Frame body

Figure 14. Detail of the 802.11 frame

Also we can observe the format of the acknowledge frame:

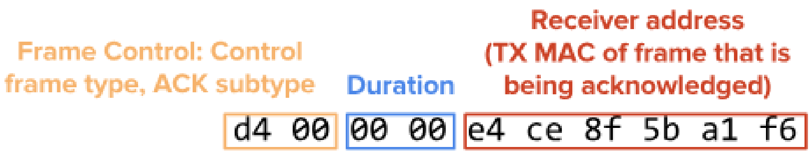
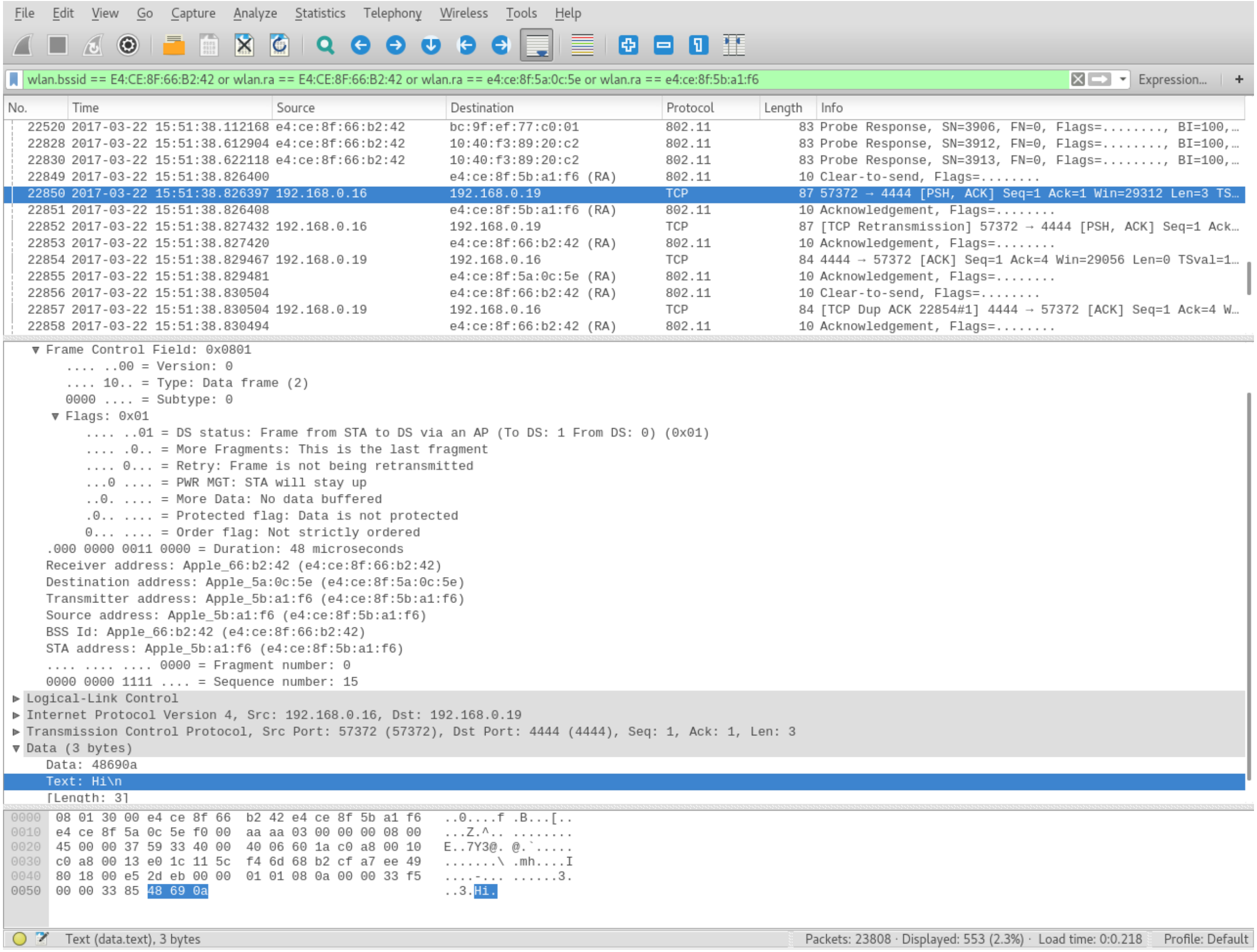

Frame Control: Control frame type, ACK subtype
Duration
Receiver address (TX MAC of frame that is being acknowledged)
d4 00
00 00
e4 ce 8f 5b a1 f6

Figure 15. Acknowledgment of the 802.11 frame

Now we will take the following packet as example, in order to analyze all the MAC layer fields and retransmissions:



```
wlan.bssid == E4:CE:8F:66:B2:42 or wlan.ra == E4:CE:8F:66:B2:42 or wlan.ra == e4:ce:8f:5a:0c:5e or wlan.ra == e4:ce:8f:5b:a1:f6
```

```
No.   Time                       Source            Destination            Protocol  Length Info
22520 2017-03-22 15:51:38.112168 e4:ce:8f:66:b2:42 bc:9f:ef:77:c0:01      802.11    83 Probe Response, SN=3906, FN=0, Flags=........, BI=100,…
22828 2017-03-22 15:51:38.612904 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2      802.11    83 Probe Response, SN=3912, FN=0, Flags=........, BI=100,…
22830 2017-03-22 15:51:38.622118 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2      802.11    83 Probe Response, SN=3913, FN=0, Flags=........, BI=100,…
22849 2017-03-22 15:51:38.826400                   e4:ce:8f:5b:a1:f6 (RA) 802.11    10 Clear-to-send, Flags=........
22850 2017-03-22 15:51:38.826397 192.168.0.16      192.168.0.19           TCP       87 57372 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=3 TS…
22851 2017-03-22 15:51:38.826408                   e4:ce:8f:5b:a1:f6 (RA) 802.11    10 Acknowledgement, Flags=........
22852 2017-03-22 15:51:38.827432 192.168.0.16      192.168.0.19           TCP       87 [TCP Retransmission] 57372 → 4444 [PSH, ACK] Seq=1 Ack…
22853 2017-03-22 15:51:38.827420                   e4:ce:8f:66:b2:42 (RA) 802.11    10 Acknowledgement, Flags=........
22854 2017-03-22 15:51:38.829467 192.168.0.19      192.168.0.16           TCP       84 4444 → 57372 [ACK] Seq=1 Ack=4 Win=29056 Len=0 TSval=1…
22855 2017-03-22 15:51:38.829481                   e4:ce:8f:5a:0c:5e (RA) 802.11    10 Acknowledgement, Flags=........
22856 2017-03-22 15:51:38.830504                   e4:ce:8f:66:b2:42 (RA) 802.11    10 Clear-to-send, Flags=........
22857 2017-03-22 15:51:38.830504 192.168.0.19      192.168.0.16           TCP       84 [TCP Dup ACK 22854#1] 4444 → 57372 [ACK] Seq=1 Ack=4 W…
22858 2017-03-22 15:51:38.830494                   e4:ce:8f:66:b2:42 (RA) 802.11    10 Acknowledgement, Flags=........
```

```
    ▼ Frame Control Field: 0x0801
        .... ..00 = Version: 0
        .... 10.. = Type: Data frame (2)
        0000 .... = Subtype: 0
      ▼ Flags: 0x01
          .... ..01 = DS status: Frame from STA to DS via an AP (To DS: 1 From DS: 0) (0x01)
          .... .0.. = More Fragments: This is the last fragment
          .... 0... = Retry: Frame is not being retransmitted
          ...0 .... = PWR MGT: STA will stay up
          ..0. .... = More Data: No data buffered
          .0.. .... = Protected flag: Data is not protected
          0... .... = Order flag: Not strictly ordered
    .000 0000 0011 0000 = Duration: 48 microseconds
    Receiver address: Apple_66:b2:42 (e4:ce:8f:66:b2:42)
    Destination address: Apple_5a:0c:5e (e4:ce:8f:5a:0c:5e)
    Transmitter address: Apple_5b:a1:f6 (e4:ce:8f:5b:a1:f6)
    Source address: Apple_5b:a1:f6 (e4:ce:8f:5b:a1:f6)
    BSS Id: Apple_66:b2:42 (e4:ce:8f:66:b2:42)
    STA address: Apple_5b:a1:f6 (e4:ce:8f:5b:a1:f6)
    .... .... .... 0000 = Fragment number: 0
    0000 0000 1111 .... = Sequence number: 15
▶ Logical-Link Control
▶ Internet Protocol Version 4, Src: 192.168.0.16, Dst: 192.168.0.19
▶ Transmission Control Protocol, Src Port: 57372 (57372), Dst Port: 4444 (4444), Seq: 1, Ack: 1, Len: 3
▼ Data (3 bytes)
    Data: 48690a
    Text: Hi\n
    [Length: 3]
```

```
0000  08 01 30 00 e4 ce 8f 66  b2 42 e4 ce 8f 5b a1 f6   ..0....f .B...[..
0010  e4 ce 8f 5a 0c 5e f0 00  aa aa 03 00 00 00 08 00   ...Z.^.. ........
0020  45 00 00 37 59 33 40 00  40 06 60 1a c0 a8 00 10   E..7Y3@. @.`.....
0030  c0 a8 00 13 e0 1c 11 5c  f4 6d 68 b2 cf a7 ee 49   .......\ .mh....I
0040  80 18 00 e5 2d eb 00 00  01 01 08 0a 00 00 33 f5   ....-... ......3.
0050  00 00 33 85 48 69 0a                               ..3.Hi.
```

```
Text (data.text), 3 bytes    Packets: 23808 · Displayed: 553 (2.3%) · Load time: 0:0.218    Profile: Default
```

Figure 16. TCP packet over 802.11 encapsulation

In the figure 16 we observe that the type inf the Frame Control header indicates the frame as “Data Frame” (recall there are other two frames: control and management):

```
Frame Control Field: 0x0801
    .... ..00 = Version: 0
    .... 10.. = Type: Data frame (2)
    0000 .... = Subtype: 0
```

Observing the flags in the frame control field we determine:

- The frame is being transmitted from a wireless host (laptop, cellphone, etc.) to the distribution system by the AP.
- The frame is not the first of a fragment that has been divided into several frames
- The frame is not a re-transmission of an already sent frame, however it is not acknowledged yet.

```
Flags: 0x01
            .... ..01 = DS status: Frame from STA to DS via an AP (To DS: 1
From DS: 0) (0x01)
    .... .0.. = More Fragments: This is the last fragment
   .... 0... = Retry: Frame is not being retransmitted
   ...0 .... = PWR MGT: STA will stay up
   ..0. .... = More Data: No data buffered
   .0.. .... = Protected flag: Data is not protected
   0... .... = Order flag: Not strictly ordered
```

Now, we see that the transmission lasted 48 microseconds:

```
.000 0000 0011 0000 = Duration: 48 microseconds
```

We can also see this value from the dumped hexadecimal values of the packet: 3000x16 which is 48 in decimal:

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

Now we can see the MAC addresses of the AP, the source and the destination. Following the MAC address of the AP:

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

MAC address of the source:

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
```

```
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

MAC address of the destination

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

Wireshark also summarize the address information in the form:

```
Receiver address: e4:ce:8f:66:b2:42
Destination address: e4:ce:8f:5a:0c:5e
Transmitter address: e4:ce:8f:5b:a1:f6
Source address: e4:ce:8f:5b:a1:f6
BSS Id: e4:ce:8f:66:b2:42
STA address: e4:ce:8f:5b:a1:f6
```

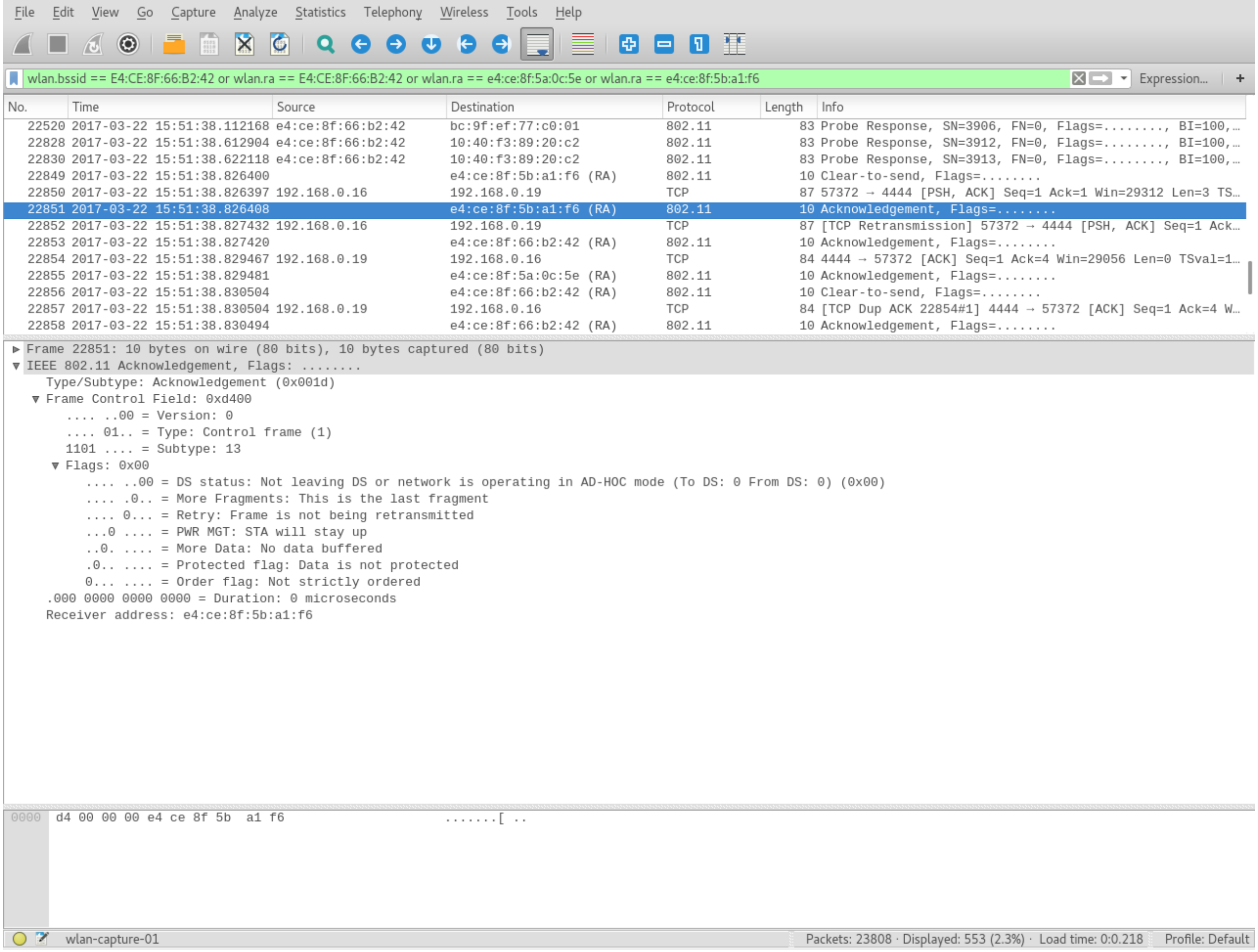
The next fields we observe are the sequence number, 15; and the fragment number, 0:

```
0000 0000 1111 .... = Sequence number: 15
.... .... .... 0000 = Fragment number: 0
```

These two values can also be obtained observing the raw hex dumped information:

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

Now, to observe the 802.11 acknowledgement we need to select the next packet on the example depicted on the figure 16:



```
wlan.bssid == E4:CE:8F:66:B2:42 or wlan.ra == E4:CE:8F:66:B2:42 or wlan.ra == e4:ce:8f:5a:0c:5e or wlan.ra == e4:ce:8f:5b:a1:f6
```

```
No.   Time                       Source            Destination              Protocol  Length Info
22520 2017-03-22 15:51:38.112168 e4:ce:8f:66:b2:42 bc:9f:ef:77:c0:01        802.11    83 Probe Response, SN=3906, FN=0, Flags=........, BI=100,…
22828 2017-03-22 15:51:38.612904 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2        802.11    83 Probe Response, SN=3912, FN=0, Flags=........, BI=100,…
22830 2017-03-22 15:51:38.622118 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2        802.11    83 Probe Response, SN=3913, FN=0, Flags=........, BI=100,…
22849 2017-03-22 15:51:38.826400                   e4:ce:8f:5b:a1:f6 (RA)   802.11    10 Clear-to-send, Flags=........
22850 2017-03-22 15:51:38.826397 192.168.0.16      192.168.0.19             TCP       87 57372 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=3 TS…
22851 2017-03-22 15:51:38.826408                   e4:ce:8f:5b:a1:f6 (RA)   802.11    10 Acknowledgement, Flags=........
22852 2017-03-22 15:51:38.827432 192.168.0.16      192.168.0.19             TCP       87 [TCP Retransmission] 57372 → 4444 [PSH, ACK] Seq=1 Ack…
22853 2017-03-22 15:51:38.827420                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Acknowledgement, Flags=........
22854 2017-03-22 15:51:38.829467 192.168.0.19      192.168.0.16             TCP       84 4444 → 57372 [ACK] Seq=1 Ack=4 Win=29056 Len=0 TSval=1…
22855 2017-03-22 15:51:38.829481                   e4:ce:8f:5a:0c:5e (RA)   802.11    10 Acknowledgement, Flags=........
22856 2017-03-22 15:51:38.830504                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Clear-to-send, Flags=........
22857 2017-03-22 15:51:38.830504 192.168.0.19      192.168.0.16             TCP       84 [TCP Dup ACK 22854#1] 4444 → 57372 [ACK] Seq=1 Ack=4 W…
22858 2017-03-22 15:51:38.830494                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Acknowledgement, Flags=........
```

```
▶ Frame 22851: 10 bytes on wire (80 bits), 10 bytes captured (80 bits)
▼ IEEE 802.11 Acknowledgement, Flags: ........
    Type/Subtype: Acknowledgement (0x001d)
  ▼ Frame Control Field: 0xd400
      .... ..00 = Version: 0
      .... 01.. = Type: Control frame (1)
      1101 .... = Subtype: 13
    ▼ Flags: 0x00
        .... ..00 = DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0 From DS: 0) (0x00)
        .... .0.. = More Fragments: This is the last fragment
        .... 0... = Retry: Frame is not being retransmitted
        ...0 .... = PWR MGT: STA will stay up
        ..0. .... = More Data: No data buffered
        .0.. .... = Protected flag: Data is not protected
        0... .... = Order flag: Not strictly ordered
    .000 0000 0000 0000 = Duration: 0 microseconds
    Receiver address: e4:ce:8f:5b:a1:f6
```

```
0000  d4 00 00 00 e4 ce 8f 5b  a1 f6                    .......[ ..
```

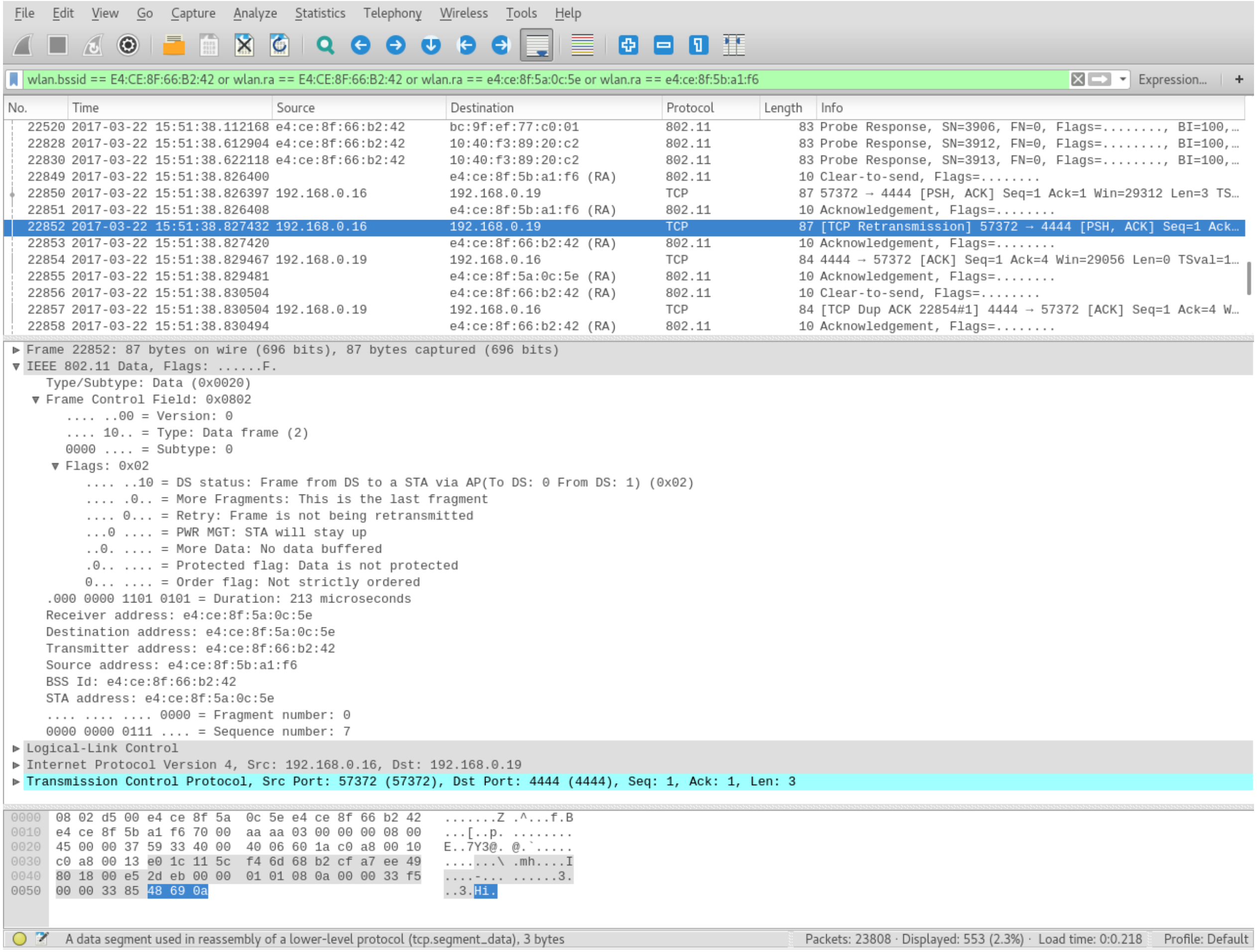
```
wlan-capture-01        Packets: 23808 · Displayed: 553 (2.3%) · Load time: 0:0.218    Profile: Default
```

Figure 17. 802.11 acknowledge frame

The ACK frames only have three fields:

- Frame Control bits are set to 1101, this means it is an ACK frame.
- Duration field is set to 0 if acknowledging a complete data frame or the final fragment in a fragment burst.
- Receiver address field is transmitter address of the frame that is being acknowledged.

In this capture there is a re-transmission from the AP to the destination, because the same payload of the next packet in the trace is transmitted from the AP to the destination:



```
wlan.bssid == E4:CE:8F:66:B2:42 or wlan.ra == E4:CE:8F:66:B2:42 or wlan.ra == e4:ce:8f:5a:0c:5e or wlan.ra == e4:ce:8f:5b:a1:f6
```

```
No.   Time                       Source            Destination              Protocol  Length Info
22520 2017-03-22 15:51:38.112168 e4:ce:8f:66:b2:42 bc:9f:ef:77:c0:01        802.11    83 Probe Response, SN=3906, FN=0, Flags=........, BI=100,…
22828 2017-03-22 15:51:38.612904 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2        802.11    83 Probe Response, SN=3912, FN=0, Flags=........, BI=100,…
22830 2017-03-22 15:51:38.622118 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2        802.11    83 Probe Response, SN=3913, FN=0, Flags=........, BI=100,…
22849 2017-03-22 15:51:38.826400                   e4:ce:8f:5b:a1:f6 (RA)   802.11    10 Clear-to-send, Flags=........
22850 2017-03-22 15:51:38.826397 192.168.0.16      192.168.0.19             TCP       87 57372 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=3 TS…
22851 2017-03-22 15:51:38.826408                   e4:ce:8f:5b:a1:f6 (RA)   802.11    10 Acknowledgement, Flags=........
22852 2017-03-22 15:51:38.827432 192.168.0.16      192.168.0.19             TCP       87 [TCP Retransmission] 57372 → 4444 [PSH, ACK] Seq=1 Ack…
22853 2017-03-22 15:51:38.827420                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Acknowledgement, Flags=........
22854 2017-03-22 15:51:38.829467 192.168.0.19      192.168.0.16             TCP       84 4444 → 57372 [ACK] Seq=1 Ack=4 Win=29056 Len=0 TSval=1…
22855 2017-03-22 15:51:38.829481                   e4:ce:8f:5a:0c:5e (RA)   802.11    10 Acknowledgement, Flags=........
22856 2017-03-22 15:51:38.830504                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Clear-to-send, Flags=........
22857 2017-03-22 15:51:38.830504 192.168.0.19      192.168.0.16             TCP       84 [TCP Dup ACK 22854#1] 4444 → 57372 [ACK] Seq=1 Ack=4 W…
22858 2017-03-22 15:51:38.830494                   e4:ce:8f:66:b2:42 (RA)   802.11    10 Acknowledgement, Flags=........
```

```
▶ Frame 22852: 87 bytes on wire (696 bits), 87 bytes captured (696 bits)
▼ IEEE 802.11 Data, Flags: ......F.
    Type/Subtype: Data (0x0020)
  ▼ Frame Control Field: 0x0802
      .... ..00 = Version: 0
      .... 10.. = Type: Data frame (2)
      0000 .... = Subtype: 0
    ▼ Flags: 0x02
        .... ..10 = DS status: Frame from DS to a STA via AP(To DS: 0 From DS: 1) (0x02)
        .... .0.. = More Fragments: This is the last fragment
        .... 0... = Retry: Frame is not being retransmitted
        ...0 .... = PWR MGT: STA will stay up
        ..0. .... = More Data: No data buffered
        .0.. .... = Protected flag: Data is not protected
        0... .... = Order flag: Not strictly ordered
    .000 0000 1101 0101 = Duration: 213 microseconds
    Receiver address: e4:ce:8f:5a:0c:5e
    Destination address: e4:ce:8f:5a:0c:5e
    Transmitter address: e4:ce:8f:66:b2:42
    Source address: e4:ce:8f:5b:a1:f6
    BSS Id: e4:ce:8f:66:b2:42
    STA address: e4:ce:8f:5a:0c:5e
    .... .... .... 0000 = Fragment number: 0
    0000 0000 0111 .... = Sequence number: 7
▶ Logical-Link Control
▶ Internet Protocol Version 4, Src: 192.168.0.16, Dst: 192.168.0.19
▶ Transmission Control Protocol, Src Port: 57372 (57372), Dst Port: 4444 (4444), Seq: 1, Ack: 1, Len: 3
```

```
0000  08 02 d5 00 e4 ce 8f 5a  0c 5e e4 ce 8f 66 b2 42   .......Z .^...f.B
0010  e4 ce 8f 5b a1 f6 70 00  aa aa 03 00 00 00 08 00   ...[..p. ........
0020  45 00 00 37 59 33 40 00  40 06 60 1a c0 a8 00 10   E..7Y3@. @.`.....
0030  c0 a8 00 13 e0 1c 11 5c  f4 6d 68 b2 cf a7 ee 49   .......\ .mh....I
0040  80 18 00 e5 2d eb 00 00  01 01 08 0a 00 00 33 f5   ....-... ......3.
0050  00 00 33 85 48 69 0a                               ..3.Hi.
```

A data segment used in reassembly of a lower-level protocol (tcp.segment_data), 3 bytes | Packets: 23808 · Displayed: 553 (2.3%) · Load time: 0:0.218 | Profile: Default

Figure 18. 802.11 re-transmission, same data payload

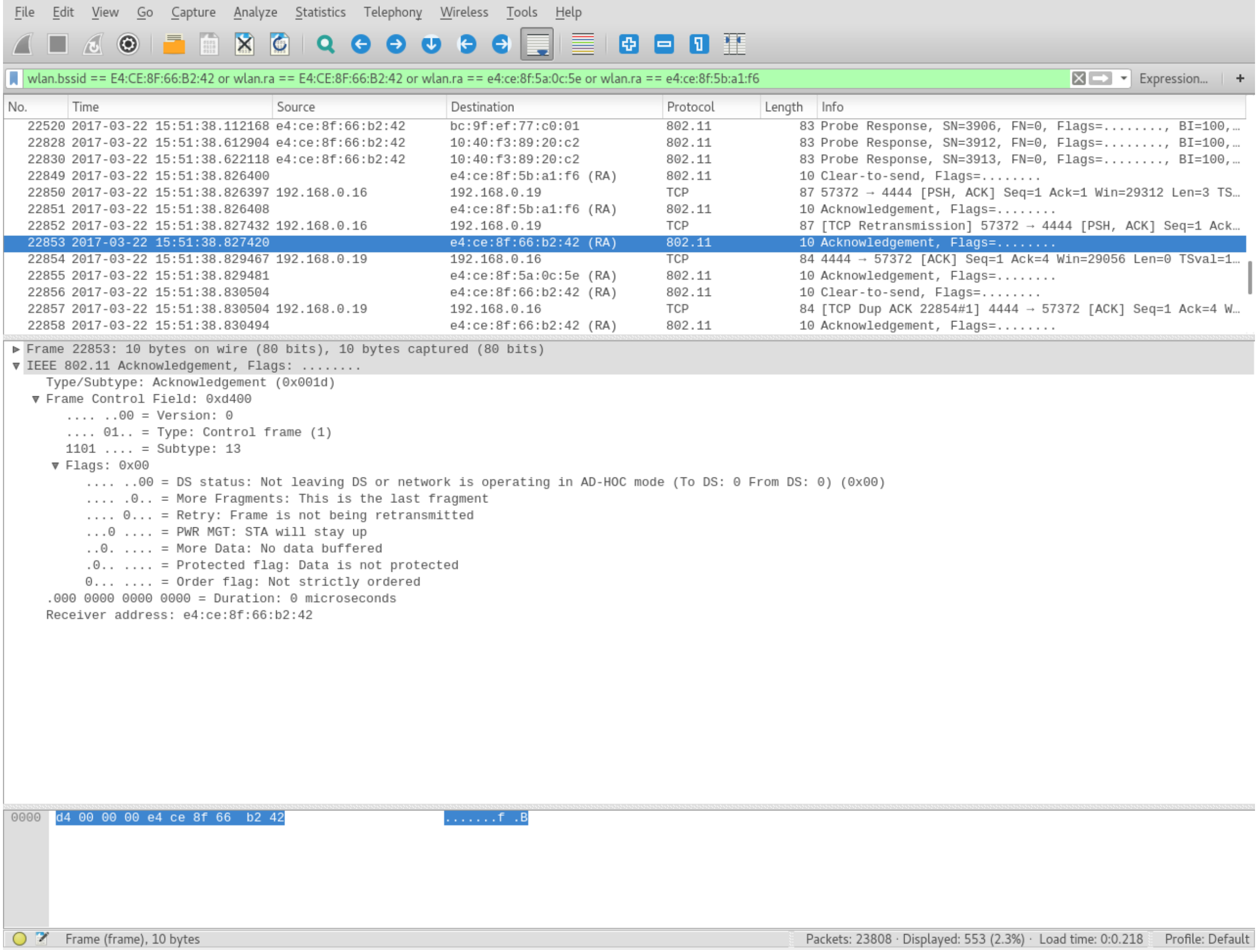
In this case we can detect some variations in the packet headers:

- The DS bits are different, which means that in this case the frame is sent from the AP to the distribution system.
- The duration has changed.
- The addresses are different, since the frame is travelling from the AP to the destination, instead of travelling from the source to the AP.
- The sequence number is different.

However, as mentioned before the data payload remains the same:

```
08 01 30 00 e4 ce 8f 66 b2 42 e4 ce 8f 5b a1 f6
e4 ce 8f 5a 0c 5e f0 00 aa aa 03 00 00 00 08 00
45 00 00 37 59 33 40 00 40 06 60 1a c0 a8 00 10
c0 a8 00 13 e0 1c 11 5c f4 6d 68 b2 cf a7 ee 49
80 18 00 e5 2d eb 00 00 01 01 08 0a 00 00 33 f5
00 00 33 85 48 69 0a
```

Finally, after the re-transmission, we confirm that the frame is acknowledged by the receiver in the 802.11 ACK frame, which tells to the AP that the frame has been received at the destination:



```
wlan.bssid == E4:CE:8F:66:B2:42 or wlan.ra == E4:CE:8F:66:B2:42 or wlan.ra == e4:ce:8f:5a:0c:5e or wlan.ra == e4:ce:8f:5b:a1:f6
```

```
No.   Time                       Source            Destination               Protocol   Length Info
22520 2017-03-22 15:51:38.112168 e4:ce:8f:66:b2:42 bc:9f:ef:77:c0:01         802.11     83 Probe Response, SN=3906, FN=0, Flags=........, BI=100,…
22828 2017-03-22 15:51:38.612904 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2         802.11     83 Probe Response, SN=3912, FN=0, Flags=........, BI=100,…
22830 2017-03-22 15:51:38.622118 e4:ce:8f:66:b2:42 10:40:f3:89:20:c2         802.11     83 Probe Response, SN=3913, FN=0, Flags=........, BI=100,…
22849 2017-03-22 15:51:38.826400                   e4:ce:8f:5b:a1:f6 (RA)    802.11     10 Clear-to-send, Flags=........
22850 2017-03-22 15:51:38.826397 192.168.0.16      192.168.0.19              TCP        87 57372 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=3 TS…
22851 2017-03-22 15:51:38.826408                   e4:ce:8f:5b:a1:f6 (RA)    802.11     10 Acknowledgement, Flags=........
22852 2017-03-22 15:51:38.827432 192.168.0.16      192.168.0.19              TCP        87 [TCP Retransmission] 57372 → 4444 [PSH, ACK] Seq=1 Ack…
22853 2017-03-22 15:51:38.827420                   e4:ce:8f:66:b2:42 (RA)    802.11     10 Acknowledgement, Flags=........
22854 2017-03-22 15:51:38.829467 192.168.0.19      192.168.0.16              TCP        84 4444 → 57372 [ACK] Seq=1 Ack=4 Win=29056 Len=0 TSval=1…
22855 2017-03-22 15:51:38.829481                   e4:ce:8f:5a:0c:5e (RA)    802.11     10 Acknowledgement, Flags=........
22856 2017-03-22 15:51:38.830504                   e4:ce:8f:66:b2:42 (RA)    802.11     10 Clear-to-send, Flags=........
22857 2017-03-22 15:51:38.830504 192.168.0.19      192.168.0.16              TCP        84 [TCP Dup ACK 22854#1] 4444 → 57372 [ACK] Seq=1 Ack=4 W…
22858 2017-03-22 15:51:38.830494                   e4:ce:8f:66:b2:42 (RA)    802.11     10 Acknowledgement, Flags=........
```

```
▶ Frame 22853: 10 bytes on wire (80 bits), 10 bytes captured (80 bits)
▼ IEEE 802.11 Acknowledgement, Flags: ........
    Type/Subtype: Acknowledgement (0x001d)
  ▼ Frame Control Field: 0xd400
      .... ..00 = Version: 0
      .... 01.. = Type: Control frame (1)
      1101 .... = Subtype: 13
    ▼ Flags: 0x00
        .... ..00 = DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0 From DS: 0) (0x00)
        .... .0.. = More Fragments: This is the last fragment
        .... 0... = Retry: Frame is not being retransmitted
        ...0 .... = PWR MGT: STA will stay up
        ..0. .... = More Data: No data buffered
        .0.. .... = Protected flag: Data is not protected
        0... .... = Order flag: Not strictly ordered
    .000 0000 0000 0000 = Duration: 0 microseconds
    Receiver address: e4:ce:8f:66:b2:42
```

```
0000  d4 00 00 00 e4 ce 8f 66  b2 42                     .......f .B
```

```
Frame (frame), 10 bytes          Packets: 23808 · Displayed: 553 (2.3%) · Load time: 0:0.218    Profile: Default
```

Figure 19. second 802.11 ACK of the same frame