

**UNIVERSIDAD DEL VALLE DE GUATEMALA**

**Facultad de Ingeniería**

**Redes – Jorge Yaz**



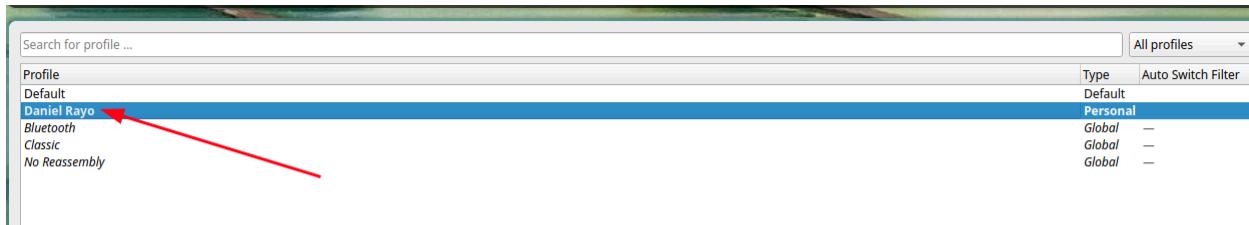
**Lab 1**

Daniel Alfredo Rayo Roldán, 22933

# Wireshark

## 1.1 Personalización del entorno

Cree un perfil con su primer nombre y primer apellido (edit -> configuration profile)



Descargue el archivo <https://www.cloudshark.org/captures/e6fb36096dbb> (Export -> Download)

The screenshot shows the CloudShark interface with a message: 'Important Announcement: CS Personal is taking a break'. It displays a file named 'Wireshark\_Masterclass\_Lesson1\_Setup.pcapng' (73.2 kb · 89 packets) with a 'more info' link. Below the file list are standard Wireshark navigation and analysis tools. A red arrow points to the 'Download File' button in the top right corner of the main content area. The table below lists the first 17 captured network frames.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.46	172.67.75.39	TCP	66	51111 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.051246	172.67.75.39	192.168.0.46	TCP	66	443 → 51111 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM=1 WS=1024
3	0.051374	192.168.0.46	172.67.75.39	TCP	54	51111 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
4	0.051658	192.168.0.46	172.67.75.39	TLSv1.3	571	Client Hello
5	0.129374	172.67.75.39	192.168.0.46	TCP	60	443 → 51111 [ACK] Seq=1 Ack=518 Win=67584 Len=0
6	0.141320	172.67.75.39	192.168.0.46	TLSv1.3	1514	Server Hello, Change Cipher Spec
7	0.141320	172.67.75.39	192.168.0.46	TLSv1.3	402	Continuation Data
8	0.141414	192.168.0.46	172.67.75.39	TCP	54	51111 → 443 [ACK] Seq=518 Ack=1809 Win=131584 Len=0
9	1.922015	192.168.0.46	172.67.75.39	TLSv1.3	118	Change Cipher Spec, Application Data
10	1.922333	192.168.0.46	172.67.75.39	TLSv1.3	146	Application Data
11	1.922724	192.168.0.46	172.67.75.39	TLSv1.3	720	Application Data
12	1.947282	172.67.75.39	192.168.0.46	TCP	60	443 → 51111 [ACK] Seq=1809 Ack=582 Win=67584 Len=0
13	1.947282	172.67.75.39	192.168.0.46	TLSv1.3	582	Application Data, Application Data
14	1.947572	192.168.0.46	172.67.75.39	TLSv1.3	85	Application Data
15	1.952943	172.67.75.39	192.168.0.46	TCP	60	443 → 51111 [ACK] Seq=2337 Ack=674 Win=67584 Len=0
16	1.952943	172.67.75.39	192.168.0.46	TCP	60	443 → 51111 [ACK] Seq=2337 Ack=1340 Win=68608 Len=0
17	1.952943	172.67.75.39	192.168.0.46	TLSv1.3	85	Application Data

Aplique el formato de tiempo Time of Day (view -> Time Display Format)

Wireshark interface showing a list of network captures. A red arrow points to the 'Time' column header.

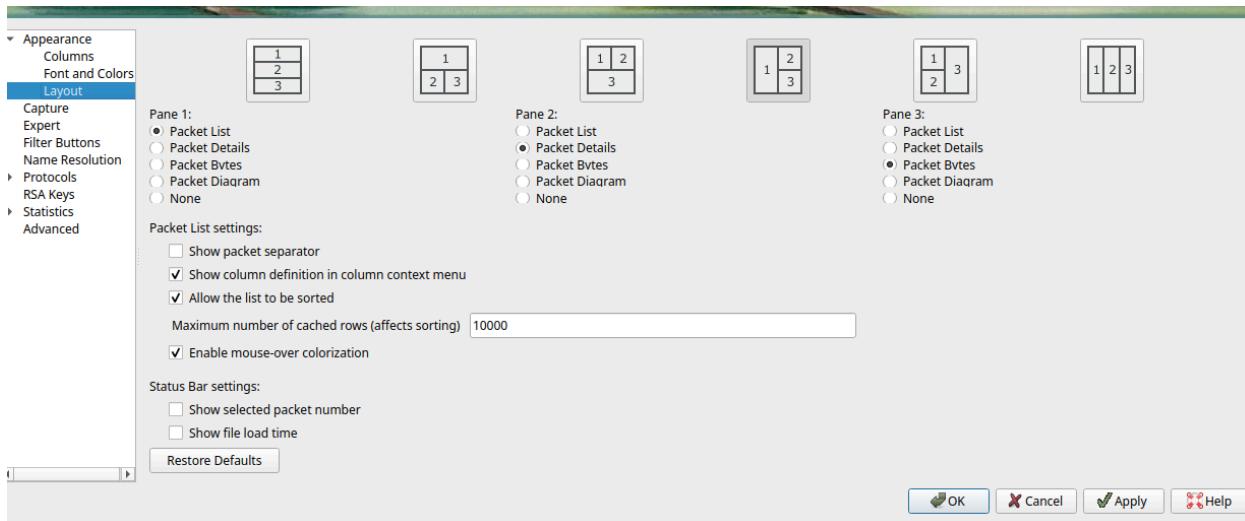
No.	Time	Source	Destination	Protocol	Info
1	11:16:47.126585	192.168.0.46	172.67.75.39	TCP	51111 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2	11:16:47.177831	172.67.75.39	192.168.0.46	TCP	443 → 51111 [SYN, ACK] Seq=1 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=1024
3	11:16:47.177959	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
4	11:16:47.178243	192.168.0.46	172.67.75.39	TLSv1.3	Client Hello (SNI=www.wireshark.org)
5	11:16:47.255959	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=1 Ack=518 Win=67584 Len=0
6	11:16:47.267995	172.67.75.39	192.168.0.46	TLSv1.3	Server Hello, Change Cipher Spec
7	11:16:47.267995	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
8	11:16:47.267999	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=518 Ack=1809 Win=131584 Len=0
9	11:16:49.048600	192.168.0.46	172.67.75.39	TLSv1.3	Change Cipher Spec, Application Data
10	11:16:49.048918	192.168.0.46	172.67.75.39	TLSv1.3	Application Data
11	11:16:49.049399	192.168.0.46	172.67.75.39	TLSv1.3	Application Data
12	11:16:49.073867	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=1809 Ack=582 Win=67584 Len=0
13	11:16:49.073867	172.67.75.39	192.168.0.46	TLSv1.3	Application Data, Application Data
14	11:16:49.074157	192.168.0.46	172.67.75.39	TLSv1.3	Application Data
15	11:16:49.079528	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2337 Ack=674 Win=67584 Len=0
16	11:16:49.079528	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2337 Ack=1340 Win=68608 Len=0
17	11:16:49.079528	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
18	11:16:49.099770	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2368 Ack=1371 Win=68608 Len=0
19	11:16:49.123997	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1371 Ack=2368 Win=130816 Len=0
20	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
21	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
22	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
23	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
24	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
25	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
26	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
27	11:16:49.389546	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1371 Ack=10871 Win=131584 Len=0
28	11:16:49.461370	172.67.75.39	192.168.0.46	TLSv1.3	Application Data
29	11:16:49.464847	172.67.75.39	192.168.0.46	TLSv1.3	Application Data

Agregue una columna con la longitud del protocolo (preferences -> column -> +)

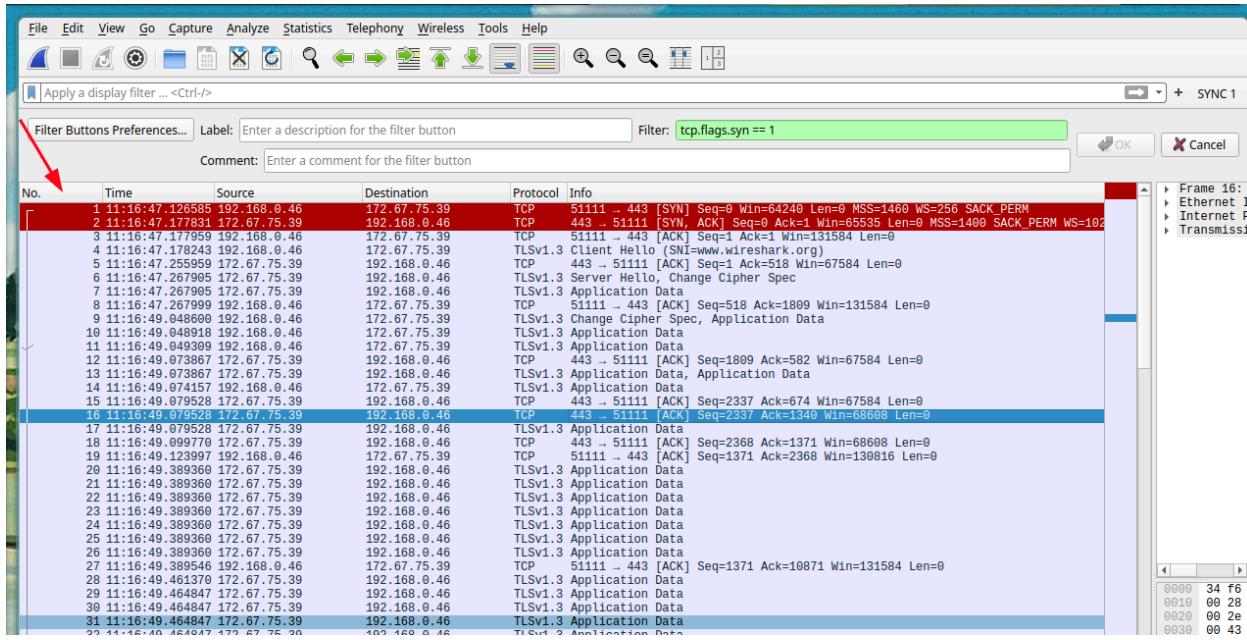
Wireshark interface showing the same list of captures with a new 'Longitud' (Length) column added. A red box highlights the new column.

No.	Time	Source	Destination	Protocol	Info	Longitud
1	11:16:47.126585	192.168.0.46	172.67.75.39	TCP	51111 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	66
2	11:16:47.177831	172.67.75.39	192.168.0.46	TCP	443 → 51111 [SYN, ACK] Seq=1 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=1024	66
3	11:16:47.177959	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0	54
4	11:16:47.178243	192.168.0.46	172.67.75.39	TLSv1.3	Client Hello (SNI=www.wireshark.org)	571
5	11:16:47.255959	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=1 Ack=518 Win=67584 Len=0	60
6	11:16:47.267995	172.67.75.39	192.168.0.46	TLSv1.3	Server Hello, Change Cipher Spec	1514
7	11:16:47.267995	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	402
8	11:16:47.267999	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=518 Ack=1809 Win=131584 Len=0	54
9	11:16:49.048600	192.168.0.46	172.67.75.39	TLSv1.3	Change Cipher Spec, Application Data	118
10	11:16:49.048918	192.168.0.46	172.67.75.39	TLSv1.3	Application Data	146
11	11:16:49.049399	192.168.0.46	172.67.75.39	TLSv1.3	Application Data	729
12	11:16:49.073867	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=1809 Ack=582 Win=67584 Len=0	60
13	11:16:49.073867	172.67.75.39	192.168.0.46	TLSv1.3	Application Data, Application Data	582
14	11:16:49.074157	192.168.0.46	172.67.75.39	TLSv1.3	Application Data	85
15	11:16:49.079528	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2337 Ack=674 Win=67584 Len=0	60
16	11:16:49.079528	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2337 Ack=1340 Win=68608 Len=0	60
17	11:16:49.079528	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	85
18	11:16:49.099770	172.67.75.39	192.168.0.46	TCP	443 → 51111 [ACK] Seq=2368 Ack=1371 Win=68608 Len=0	60
19	11:16:49.123997	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1371 Ack=2368 Win=130816 Len=0	54
20	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1445
21	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1514
22	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1376
23	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1445
24	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1514
25	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1376
26	11:16:49.389360	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	211
27	11:16:49.389546	192.168.0.46	172.67.75.39	TCP	51111 → 443 [ACK] Seq=1371 Ack=10871 Win=131584 Len=0	54
28	11:16:49.461370	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1445
29	11:16:49.464847	172.67.75.39	192.168.0.46	TLSv1.3	Application Data	1445

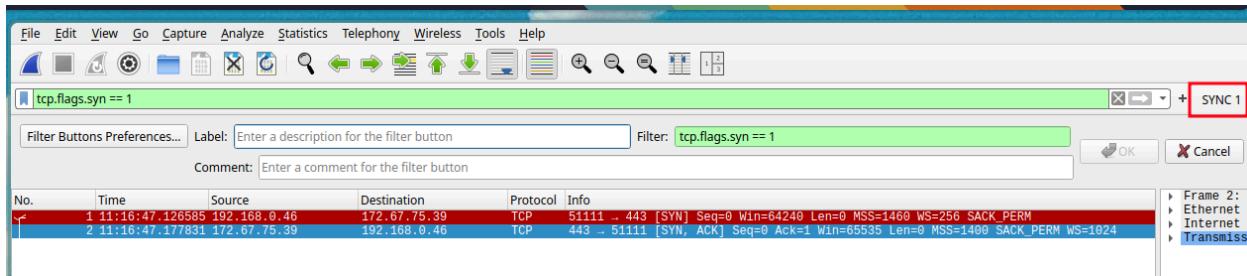
Aplique un esquema de paneles que sea de su preferencia (que no sea el esquema por defecto) (preferences -> Layout)



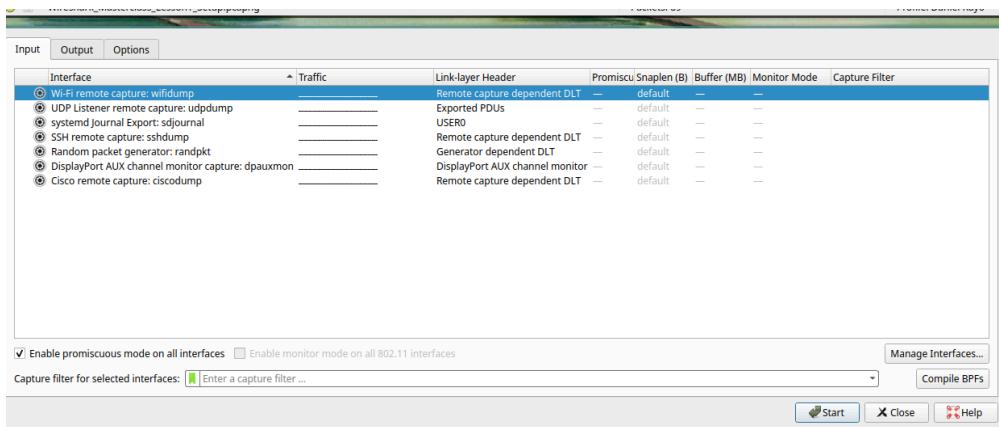
Aplique una regla de color para el protocolo TCP cuyas banderas SYN sean iguales a 1, y coloque el color de su preferencia. (View -> coloring rules -> +)



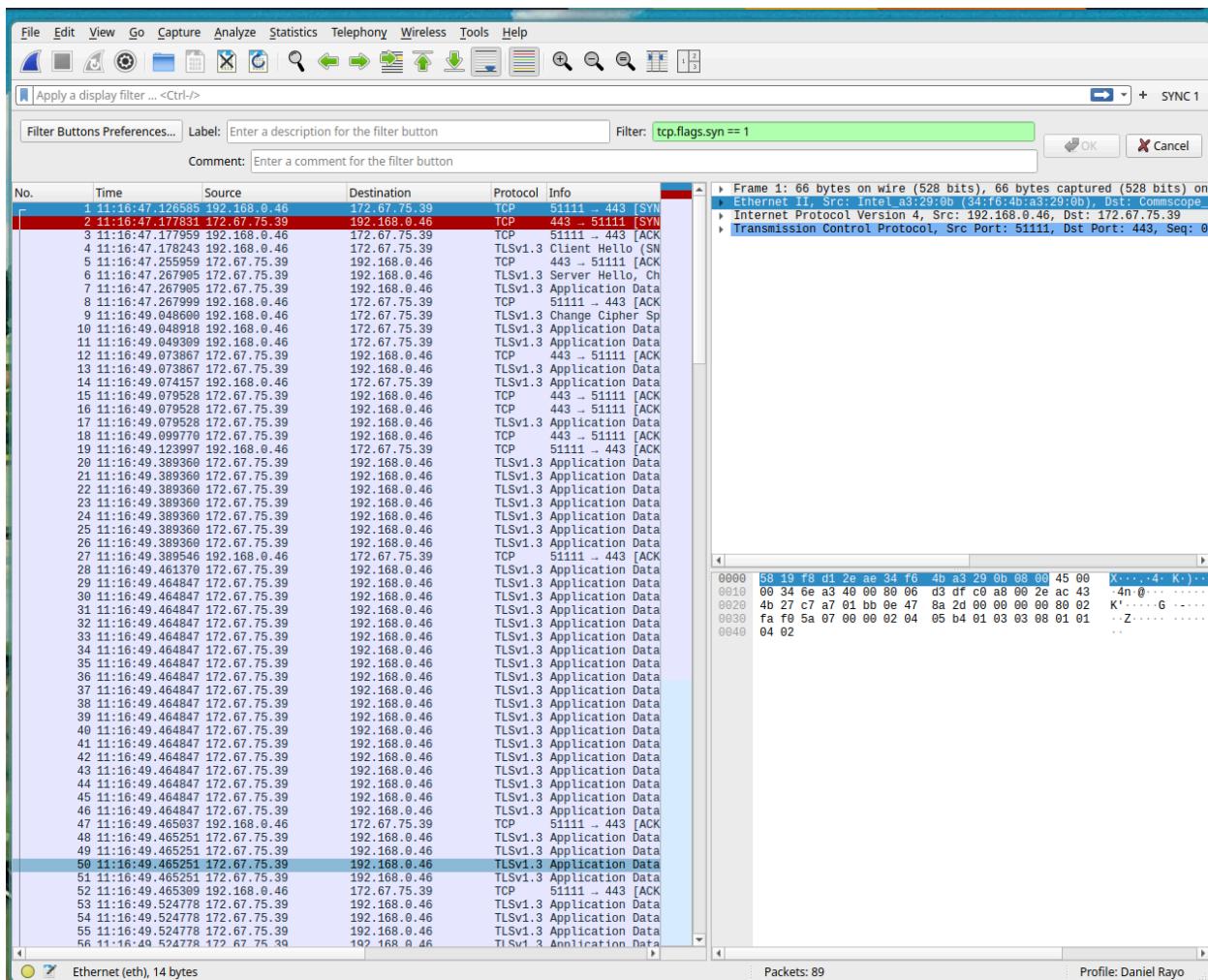
Cree un botón que aplique un filtro para paquetes TCP con la bandera SYN igual a 1 (esquina superior derecha -> +)



Oculte las interfaces virtuales (en caso aplique: capture -> options)



## Resultado final



## Lista de interfaces simplificada

Welcome to Wireshark

Open

/home/smaug/Downloads/Wireshark\_Masterclass\_Lesson1\_Setup.pcapng (73 KB)

**Capture**

...using this filter:  Enter a capture filter ...

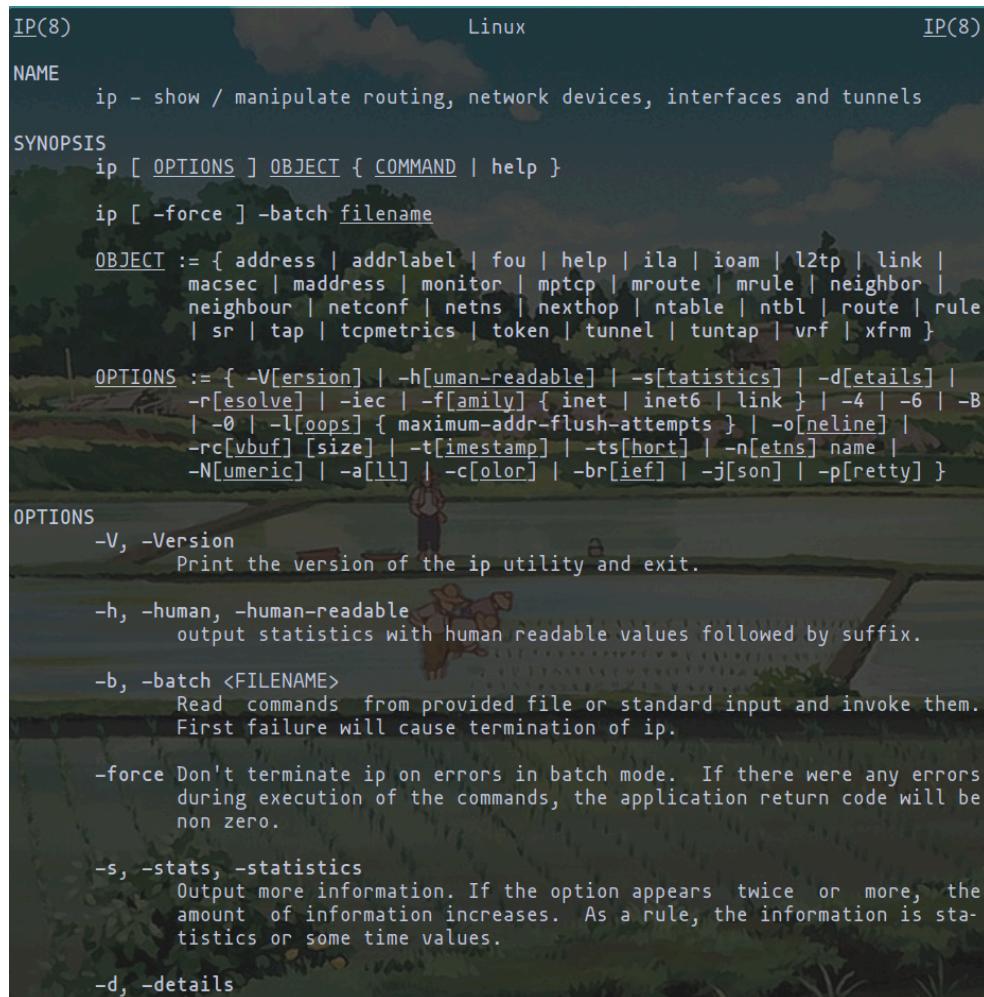
5 interfaces shown, 2 hidden ▾

- ① DisplayPort AUX channel monitor capture: dpauxmon
- ② Random packet generator: randpkt
- ③ SSH remote capture: sshdump
- ④ UDP Listener remote capture: udpdump
- ⑤ Wi-Fi remote capture: wifidump

## Learn

## 1.2 Configuración de la captura de paquetes

1. Abra una terminal y ejecute el comando ifconfig/ipconfig (dependiendo de su OS). Detalle y explique lo observado, investigue (i.e.: ‘man ifconfig’, documentación) de ser necesario.



The screenshot shows a terminal window with a dark background and a green landscape watermark. The title bar says "IP(8)". The window contains the "ip" command's man page. The "NAME" section defines "ip" as showing and manipulating routing, network devices, interfaces, and tunnels. The "SYNOPSIS" section shows the command structure: ip [OPTIONS] OBJECT {COMMAND | help}. The "OBJECT" section lists various objects like address, addrlabel, fou, help, ila, ioam, l2tp, link, macsec, maddress, monitor, mptcp, mroute, mrule, neighbor, neighbour, netconf, netns, nexthop, ntable, ntbl, route, rule, sr, tap, tcpmetrics, token, tunnel, tuntap, vrf, and xfrm. The "OPTIONS" section details options such as -V, -h, -b, -force, -s, -d, and their descriptions. The "Linux" watermark is visible in the background of the terminal window.

```
NAME
    ip - show / manipulate routing, network devices, interfaces and tunnels

SYNOPSIS
    ip [ OPTIONS ] OBJECT { COMMAND | help }

    ip [ -force ] -batch filename

    OBJECT := { address | addrlabel | fou | help | ila | ioam | l2tp | link |
                macsec | maddress | monitor | mptcp | mroute | mrule | neighbor |
                neighbour | netconf | netns | nexthop | ntable | ntbl | route | rule |
                sr | tap | tcpmetrics | token | tunnel | tuntap | vrf | xfrm }

    OPTIONS := { -V[ersion] | -h[uman-readable] | -s[tatistics] | -d[etails] |
                 -r[esolve] | -iec | -f[amily] { inet | inet6 | link } | -4 | -6 | -B |
                 -0 | -l[oops] { maximum-addr-flush-attempts } | -o[neline] |
                 -rc[vbuf] [size] | -t[imestamp] | -ts[hort] | -n[etns] name |
                 -N[umeric] | -a[ll] | -c[olor] | -br[ief] | -j[son] | -p[retty] }

OPTIONS
    -V, --Version
        Print the version of the ip utility and exit.

    -h, --human, --human-readable
        output statistics with human readable values followed by suffix.

    -b, --batch <FILENAME>
        Read commands from provided file or standard input and invoke them.
        First failure will cause termination of ip.

    -force
        Don't terminate ip on errors in batch mode. If there were any errors
        during execution of the commands, the application return code will be
        non zero.

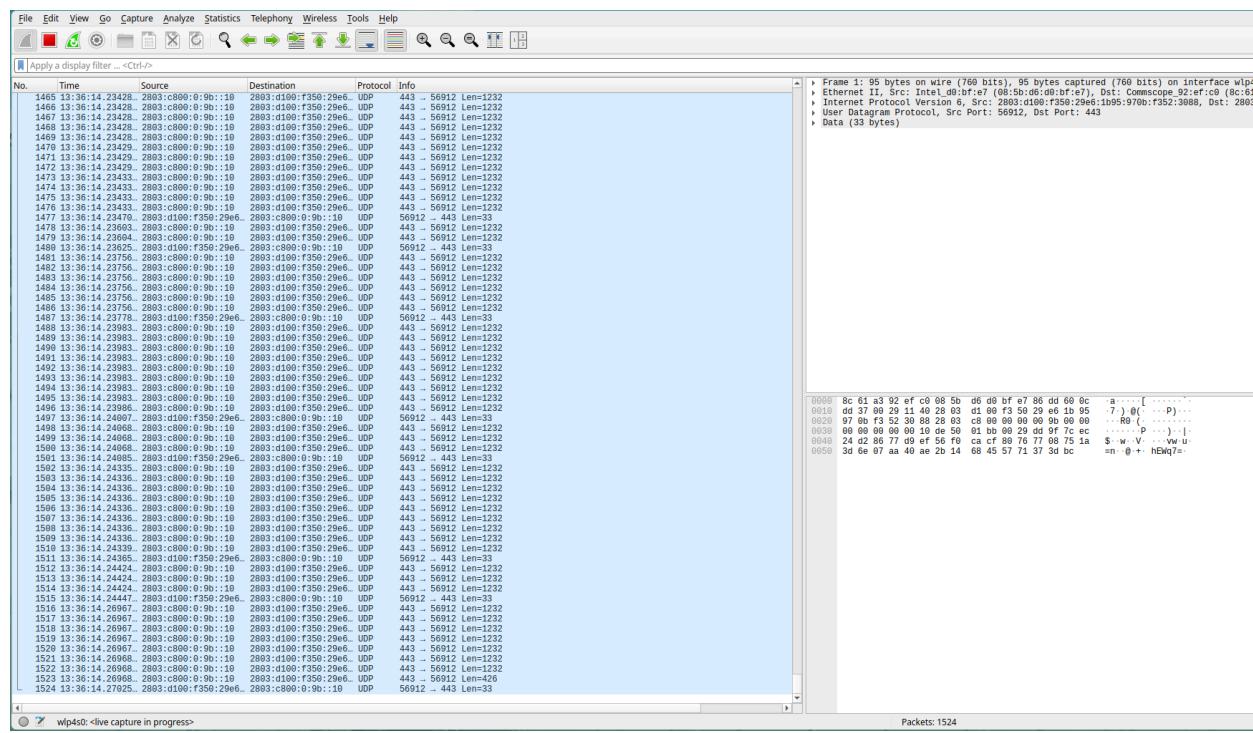
    -s, --stats, --statistics
        Output more information. If the option appears twice or more, the
        amount of information increases. As a rule, the information is sta-
        tistics or some time values.

    -d, --details
```

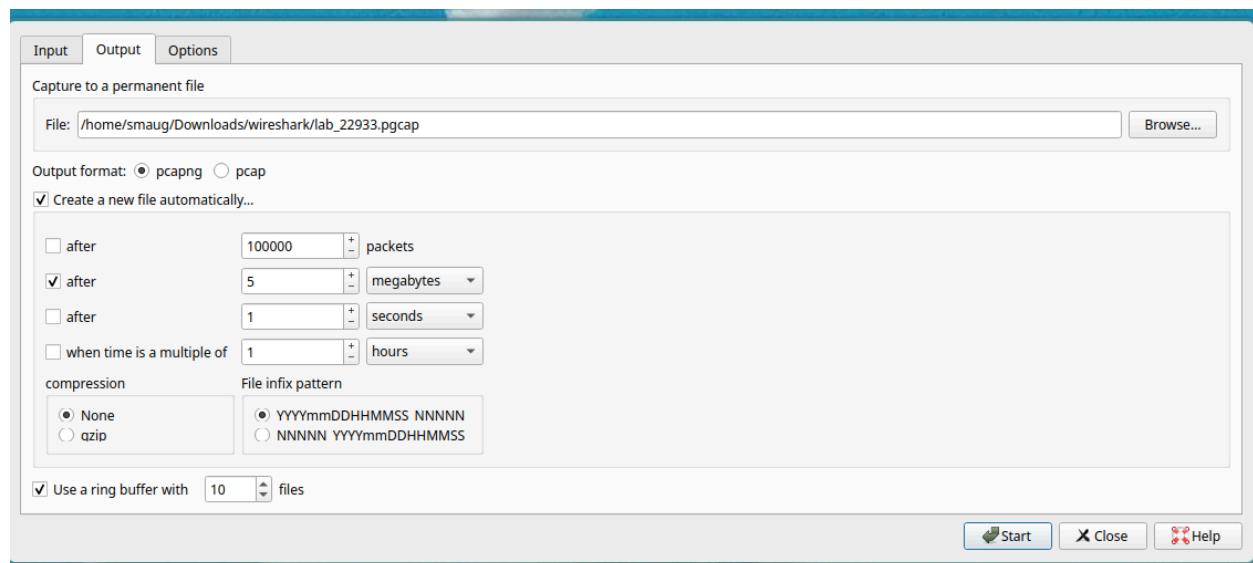
En mi caso al usar Manjaro Linux, el comando para manipular visualizar interfaces y funcionalidades de conexión se llama `ip`.

Es un comando multipropósito, que incluye muchos subcomandos como `address`, `route` para administrar las direcciones IP y la tabla de ruteo respectivamente.

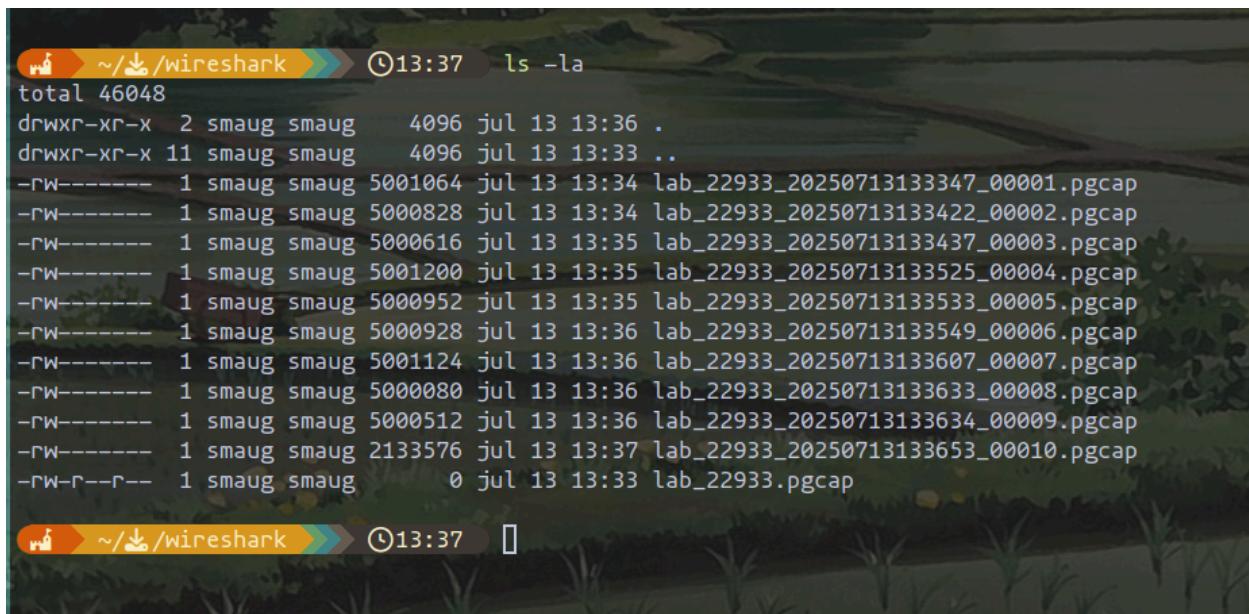
## Captura de paquetes



## Configuración utilizada para guardar los datos en paquetes



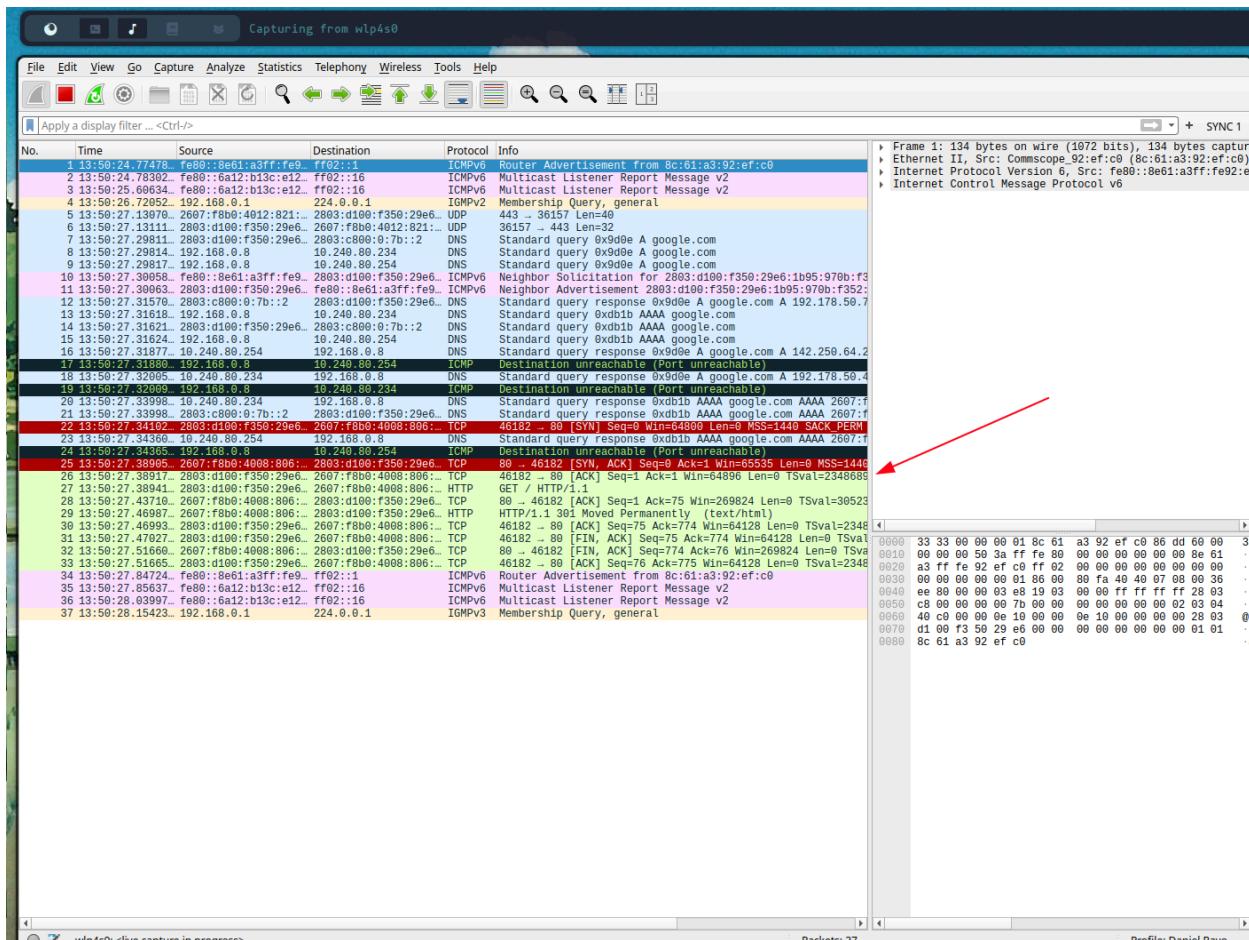
## Archivos Generados por wireshark



```
~/.wireshark 13:37 ls -la
total 46048
drwxr-xr-x 2 smaug smaug 4096 jul 13 13:36 .
drwxr-xr-x 11 smaug smaug 4096 jul 13 13:33 ..
-rw----- 1 smaug smaug 5001064 jul 13 13:34 lab_22933_20250713133347_00001.pgcap
-rw----- 1 smaug smaug 5000828 jul 13 13:34 lab_22933_20250713133422_00002.pgcap
-rw----- 1 smaug smaug 5000616 jul 13 13:35 lab_22933_20250713133437_00003.pgcap
-rw----- 1 smaug smaug 5001200 jul 13 13:35 lab_22933_20250713133525_00004.pgcap
-rw----- 1 smaug smaug 5000952 jul 13 13:35 lab_22933_20250713133533_00005.pgcap
-rw----- 1 smaug smaug 5000928 jul 13 13:36 lab_22933_20250713133549_00006.pgcap
-rw----- 1 smaug smaug 5001124 jul 13 13:36 lab_22933_20250713133607_00007.pgcap
-rw----- 1 smaug smaug 5000080 jul 13 13:36 lab_22933_20250713133633_00008.pgcap
-rw----- 1 smaug smaug 5000512 jul 13 13:36 lab_22933_20250713133634_00009.pgcap
-rw----- 1 smaug smaug 2133576 jul 13 13:37 lab_22933_20250713133653_00010.pgcap
-rw-r--r-- 1 smaug smaug 0 jul 13 13:33 lab_22933.pgcap
```

## 1.3 Análisis de paquetes

Abra su navegador, inicie una captura de paquetes en Wireshark (sin filtro) en la interfaz y acceda a la siguiente dirección: <http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html>



```

Internet Protocol Version 4, Src: 192.168.0.8, Dst: 128.119.245.12
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 442
  Identification: 0x9623 (38435)
  ▶ 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 64
  Protocol: TCP (6)
  Header Checksum: 0x6ce6 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.0.8
  Destination Address: 128.119.245.12
  [Stream index: 0]
  ▶ Transmission Control Protocol, Src Port: 53102, Dst Port: 80, Seq: 1, Ack: 1, Len: 390
  ▶ Hypertext Transfer Protocol
    ▶ GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1\r\n
      Host: gaia.cs.umass.edu\r\n
      User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
      Accept-Language: en-US,en;q=0.5\r\n
      Accept-Encoding: gzip, deflate\r\n
      Sec-GPC: 1\r\n
      Connection: keep-alive\r\n
      Upgrade-Insecure-Requests: 1\r\n
      Priority: u=0, i=\r\n
      \r\n
      [Response in frame: 35]
      Full request URI: http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html
  Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.0.8
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 490
  Identification: 0x534f (21327)
  ▶ 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 37
  Protocol: TCP (6)
  Header Checksum: 0xca8a [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 128.119.245.12
  Destination Address: 192.168.0.8
  [Stream index: 0]
  Transmission Control Protocol, Src Port: 80, Dst Port: 53102, Seq: 1, Ack: 391, Len: 438
  Hypertext Transfer Protocol
    ▶ HTTP/1.1 200 OK\r\n
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Date: Sun, 13 Jul 2025 20:08:10 GMT\r\n
      Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.33 mod_perl/2.0.11 Perl/v5...
      Last-Modified: Sun, 13 Jul 2025 05:59:01 GMT\r\n
      ETag: "51-639c93d69ba76"\r\n
      Accept-Ranges: bytes\r\n
      Content-Length: 81\r\n
      Keep-Alive: timeout=5, max=100\r\n
      Connection: Keep-Alive\r\n
      Content-Type: text/html; charset=UTF-8\r\n

```

- ¿Qué versión de HTTP está ejecutando su navegador ?  
Usa la versión HTTP/1.1
- ¿Qué versión de HTTP está ejecutando el servidor?  
Usa la versión HTTP/1.1
- ¿Qué lenguaje (si aplica) indica el navegador que acepta el servidor?  
Indica que acepta inglés estadounidense (Accept-Language: en-US, en)
- ¿Cuántos bytes de contenido fueron devueltos por el servidor?  
81 bytes fueron devueltos.

- e. En el caso que haya un problema de rendimiento mientras se descarga la página. ¿En qué elementos de la red convendría escuchar los paquetes? ¿Es conveniente instalar Wireshark en el servidor? Justifique

Dado que fallas en transmisión de paquetes pueden ser causadas por múltiples causas, es viable monitorear las partes más importantes que participan en la conexión: el cliente, el router de la red local y el servidor objetivo.

En cuanto a si es conveniente instalar wireshark en el servidor, depende del contexto. Las ventajas de hacerlo es que permite debuggear de forma remota el tráfico en el server, tener control por medio de logs, sin embargo bajo la contra de agregar *overhead* en todas las conexiones que tenga el server además que los logs pueden a tomar bastante espacio.

## Discusión

Discusión sobre la actividad, su experiencia y hallazgos. Incluir ambas partes.

Para la primera parte se pretendió crear un protocolo de comunicación utilizando solamente la comunicación verbal y oral, con la intención de poder comprender los retos a los que se enfrentan los protocolos de comunicación que implementan los sistemas de computación. En el proceso se experimentó dificultades en varios aspectos, al traducir la información a un lenguaje común (morse) para después transmitirlo por medio físicos y que la otra persona haga el proceso inverso.

Al tomar varios pasos a veces se cometieron errores en la traducción inicial, otras veces el emisor enviaba mal los caracteres, en otras ocasiones el receptor no recibía o lograba juntar todos los caracteres, y la traducción de morse era incorrecta. Este tipo de problemas son casos comunes a los que se enfrentan también los sistemas de computación.

Por otro lado, para la sección de wireshark, se descubrió el uso básico de esta herramienta para interceptar paquetes de comunicación cuando salen o entran a la computadora. El software reveló todos los detalles de diferentes capas de transporte, muchos de ellos no son intuitivos para gente no educada. Sin embargo ayudó a desvelar la gran cantidad de niveles de abstracción que se lleva para poder enviar un mensaje de un lado a otro.

## Comentarios

Para poder utilizar wireshark en su máxima capacidad el conocimiento de teoría de protocolos de comunicación es indispensable, de lo contrario gran parte de la información mostrada no se encontrará relevante.

## Conclusiones

- Los protocolos de comunicación son un acuerdo entre 2 partes, el incumplimiento por parte de uno de los miembros imposibilita la comunicación.
- Para la transmisión de mensajes, mientras más diferenciables los signos es estricto el protocolo más fácil es poder descubrir errores de comunicación y recepción de mensajes.
- El uso de wireshark como herramienta de debugeo monitoreo de aplicaciones puede ser útil dependiendo del contexto.

## Referencias

No se utilizaron referencias externas para la elaboración de este laboratorio.