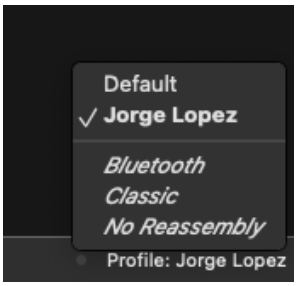
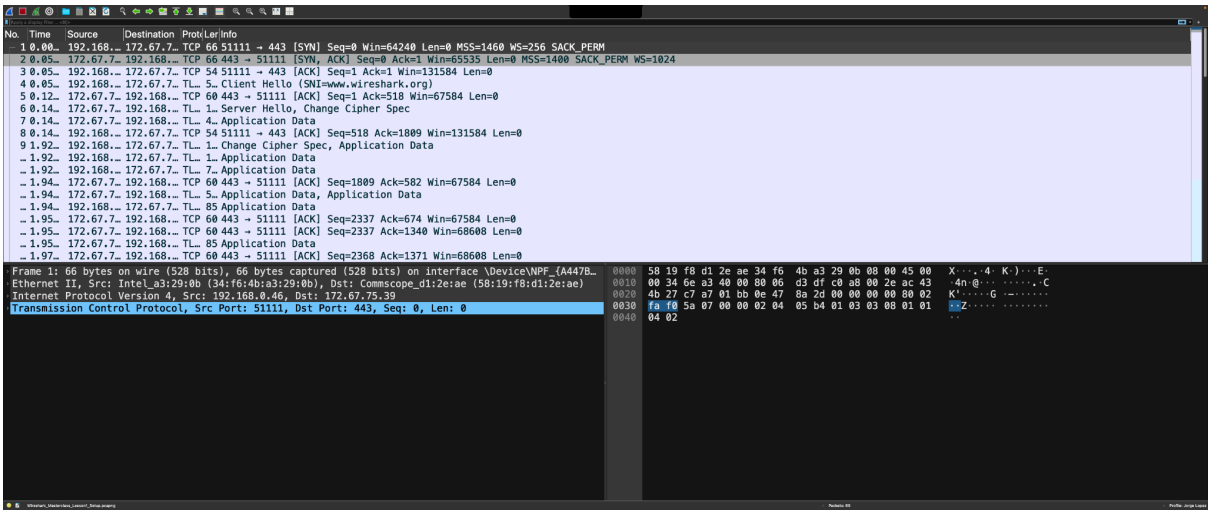
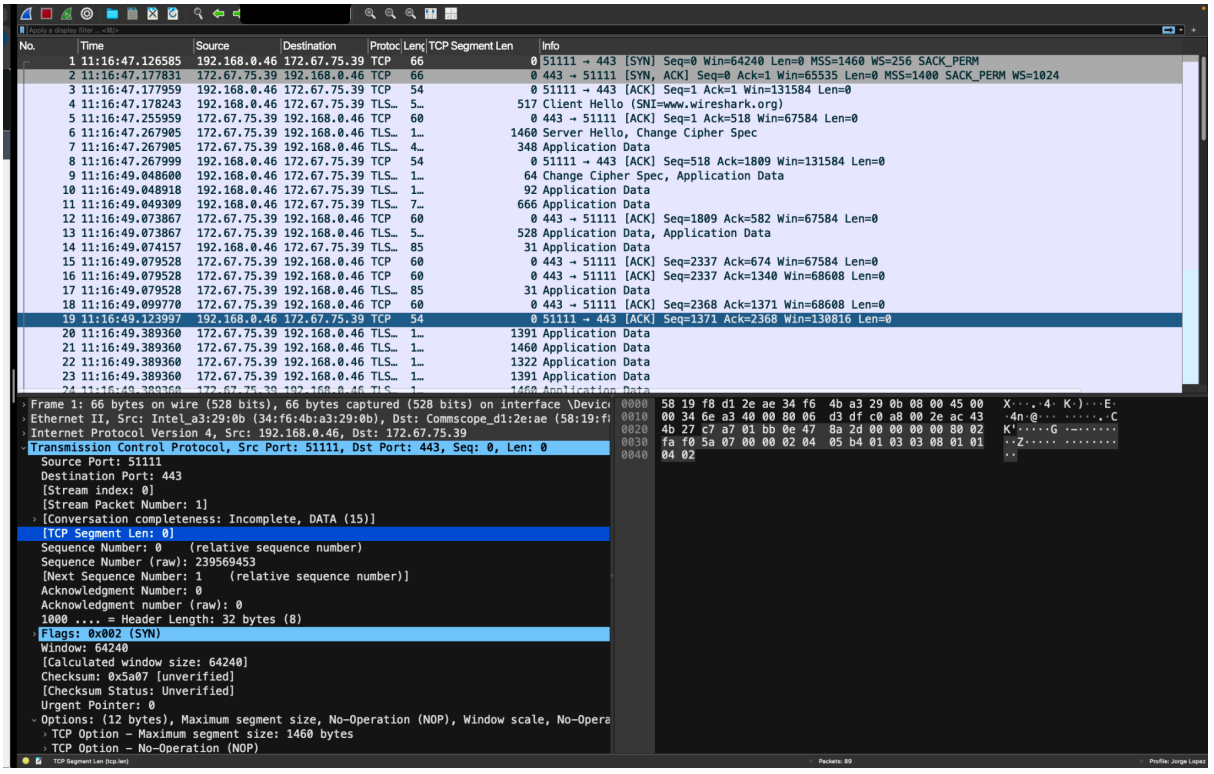
Introducción a Wireshark

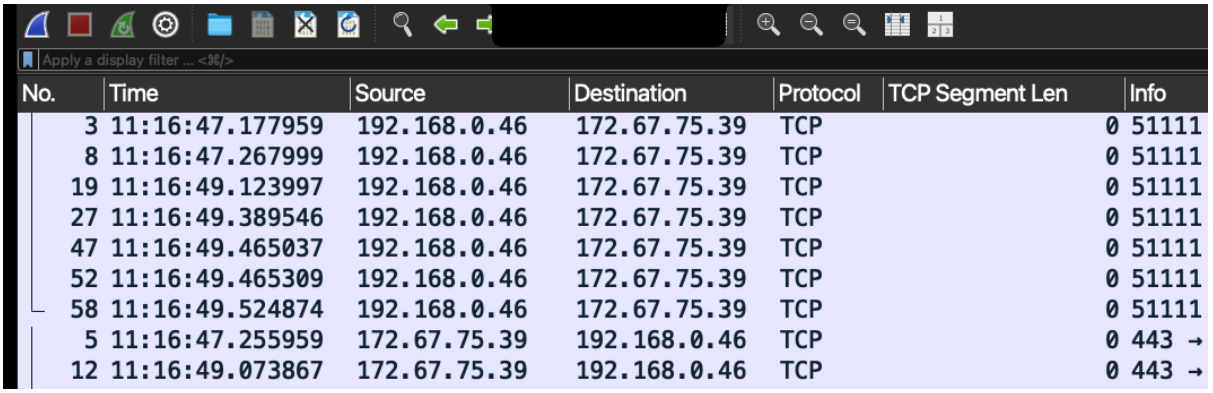
Jorge Luis Lopez 221038

- **Primera parte: personalización del entorno**



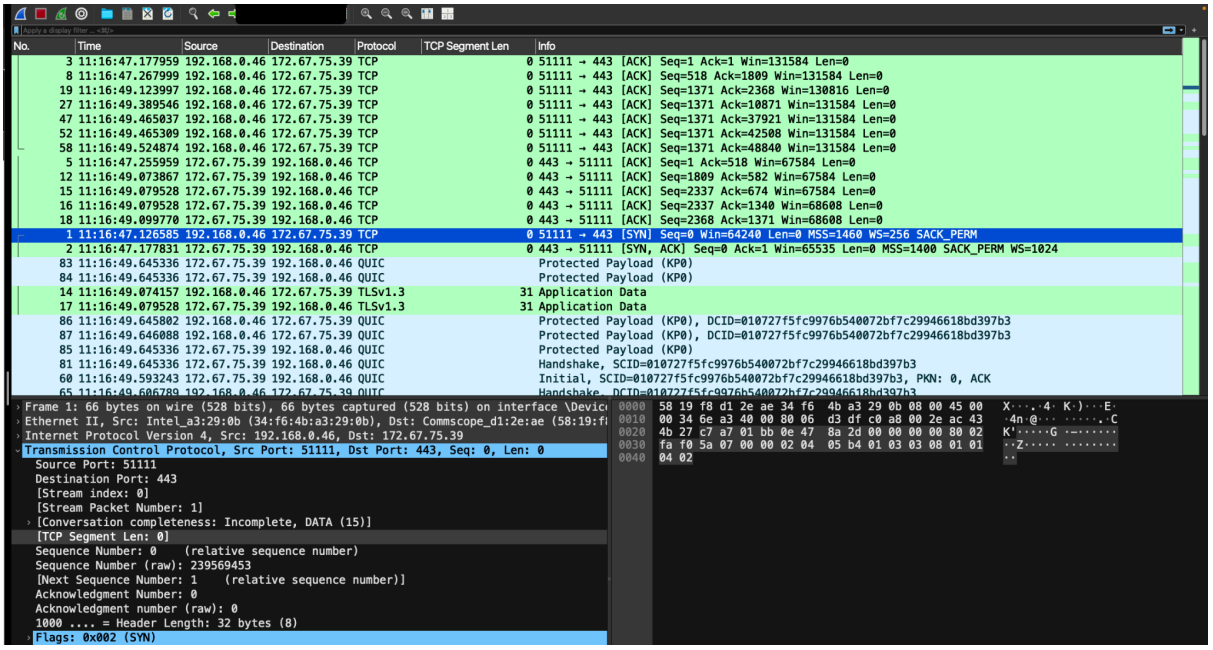






Apply a display filter ... <⌘/>

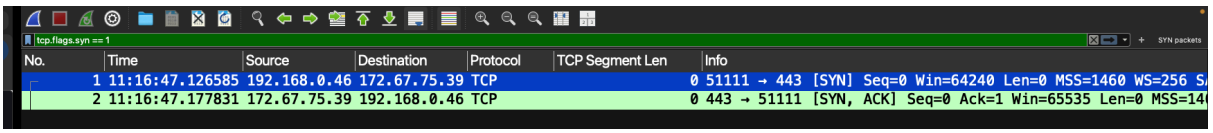
No.	Time	Source	Destination	Protocol	TCP Segment Len	Info
3	11:16:47.177959	192.168.0.46	172.67.75.39	TCP	0	51111
8	11:16:47.267999	192.168.0.46	172.67.75.39	TCP	0	51111
19	11:16:49.123997	192.168.0.46	172.67.75.39	TCP	0	51111
27	11:16:49.389546	192.168.0.46	172.67.75.39	TCP	0	51111
47	11:16:49.465037	192.168.0.46	172.67.75.39	TCP	0	51111
52	11:16:49.465309	192.168.0.46	172.67.75.39	TCP	0	51111
58	11:16:49.524874	192.168.0.46	172.67.75.39	TCP	0	51111
5	11:16:47.255959	172.67.75.39	192.168.0.46	TCP	0	443 →
12	11:16:49.073867	172.67.75.39	192.168.0.46	TCP	0	443 →



No.	Time	Source	Destination	Protocol	TCP Segment Len	Info
3	11:16:47.177959	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
8	11:16:47.267999	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=518 Ack=1809 Win=131584 Len=0
19	11:16:49.123997	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1371 Ack=2368 Win=130816 Len=0
27	11:16:49.389546	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1371 Ack=10871 Win=131584 Len=0
47	11:16:49.465037	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1371 Ack=37921 Win=131584 Len=0
52	11:16:49.465309	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1371 Ack=42508 Win=131584 Len=0
58	11:16:49.524874	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [ACK] Seq=1371 Ack=48840 Win=131584 Len=0
5	11:16:47.255959	172.67.75.39	192.168.0.46	TCP	0	443 → 51111 [ACK] Seq=1 Ack=518 Win=67584 Len=0
12	11:16:49.073867	172.67.75.39	192.168.0.46	TCP	0	443 → 51111 [ACK] Seq=1809 Ack=582 Win=67584 Len=0
15	11:16:49.079528	172.67.75.39	192.168.0.46	TCP	0	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	0	443 → 51111 [ACK] Seq=2337 Ack=1340 Win=68608 Len=0
18	11:16:49.099770	172.67.75.39	192.168.0.46	TCP	0	443 → 51111 [ACK] Seq=2368 Ack=1371 Win=68608 Len=0
1	11:16:47.126585	192.168.0.46	172.67.75.39	TCP	0	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	0	443 → 51111 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=1024
83	11:16:49.645336	172.67.75.39	192.168.0.46	QUIC		Protected Payload (KP0)
84	11:16:49.645336	172.67.75.39	192.168.0.46	QUIC		Protected Payload (KP0)
14	11:16:49.074157	192.168.0.46	172.67.75.39	TLSv1.3	31	Application Data
17	11:16:49.079528	172.67.75.39	192.168.0.46	TLSv1.3	31	Application Data
86	11:16:49.645802	192.168.0.46	172.67.75.39	QUIC		Protected Payload (KP0), DCID=010727f5fc9976b540072bf7c29946618bd397b3
87	11:16:49.646088	192.168.0.46	172.67.75.39	QUIC		Protected Payload (KP0), DCID=010727f5fc9976b540072bf7c29946618bd397b3
85	11:16:49.645336	172.67.75.39	192.168.0.46	QUIC		Protected Payload (KP0)
81	11:16:49.645336	172.67.75.39	192.168.0.46	QUIC		Handshake, SCID=010727f5fc9976b540072bf7c29946618bd397b3
60	11:16:49.593243	172.67.75.39	192.168.0.46	QUIC		Initial, SCID=010727f5fc9976b540072bf7c29946618bd397b3, PKN: 0, ACK
65	11:16:49.606789	192.168.0.46	172.67.75.39	QUIC		Handshake, DCID=010727f5fc9976b540072bf7c29946618bd397b3

```
Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Devic
Ethernet II, Src: Intel_a3:29:0b (34:f6:4b:a3:29:0b), Dst: Commscope_d1:2e:ae (58:19:f
Internet Protocol Version 4, Src: 192.168.0.46, Dst: 172.67.75.39
Transmission Control Protocol, Src Port: 51111, Dst Port: 443, Seq: 0, Len: 0
  Source Port: 51111
  Destination Port: 443
  [Stream index: 0]
  [Stream Packet Number: 1]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 0    (relative sequence number)
  Sequence Number (raw): 239569453
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1000 .... = Header Length: 32 bytes (8)
  Flags: 0x002 (SYN)
```

```
0000  58 19 f8 d1 2e ae 34 f6  4b a3 29 0b 08 00 45 00   X·····4· K·)···E·
0010  00 34 6e a3 40 00 80 06  d3 df c0 a8 00 2e ac 43   ·4n·@··· .......C
0020  4b 27 c7 a7 01 bb 0e 47  8a 2d 00 00 00 00 80 02   K'·····G ·-......
0030  fa f0 5a 07 00 00 02 04  05 b4 01 03 03 08 01 01   ··Z····· ........
0040  04 02                                             ··
```



tcp.flags.syn == 1

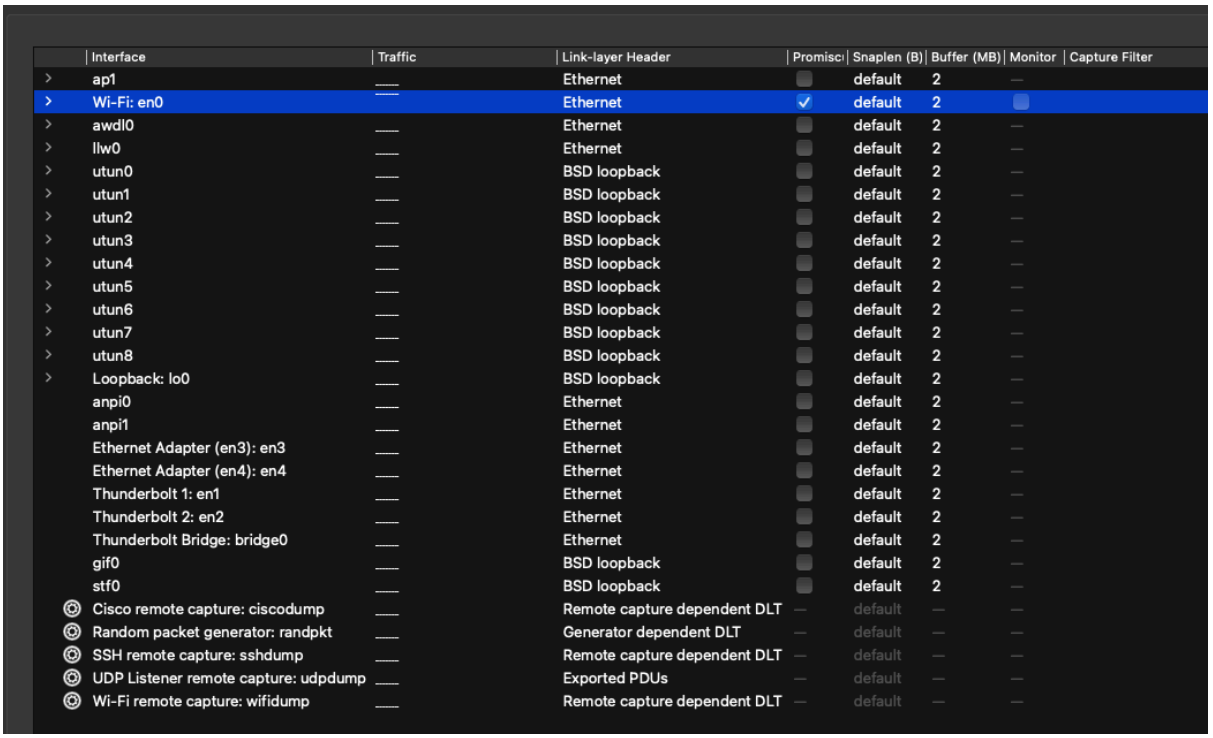
SYN packets

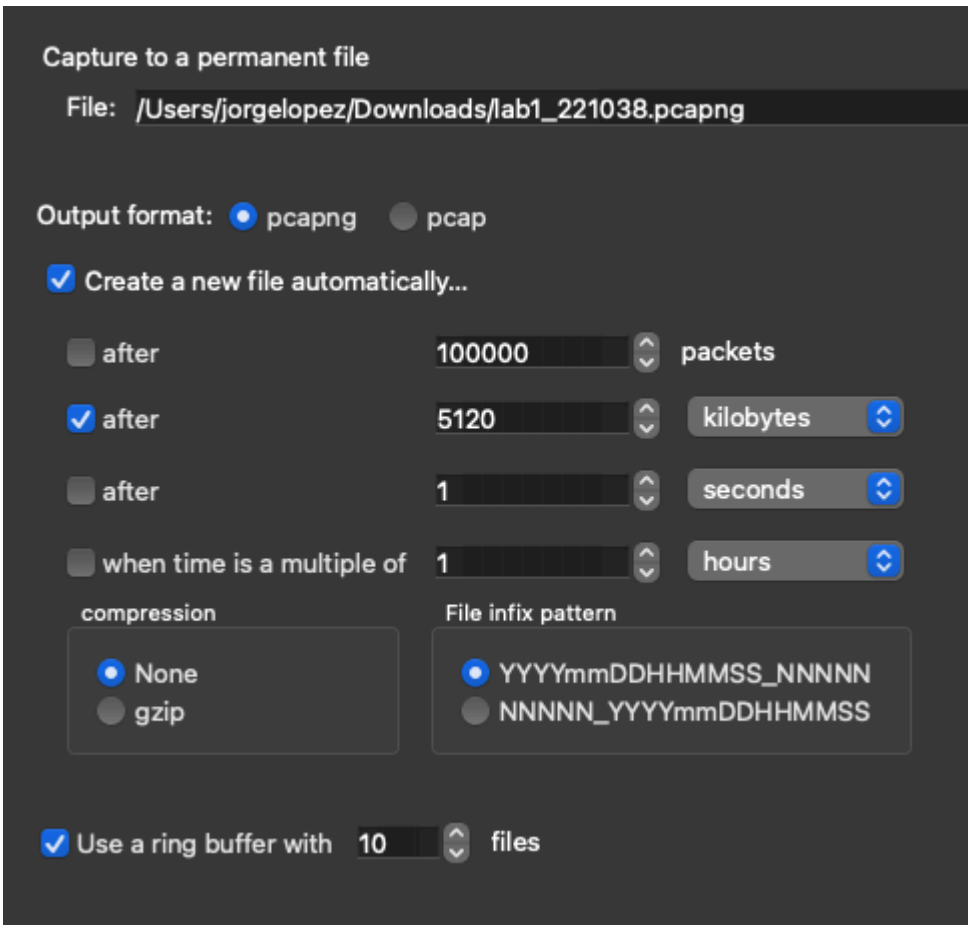
No.	Time	Source	Destination	Protocol	TCP Segment Len	Info
1	11:16:47.126585	192.168.0.46	172.67.75.39	TCP	0	51111 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 S
2	11:16:47.177831	172.67.75.39	192.168.0.46	TCP	0	443 → 51111 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=14

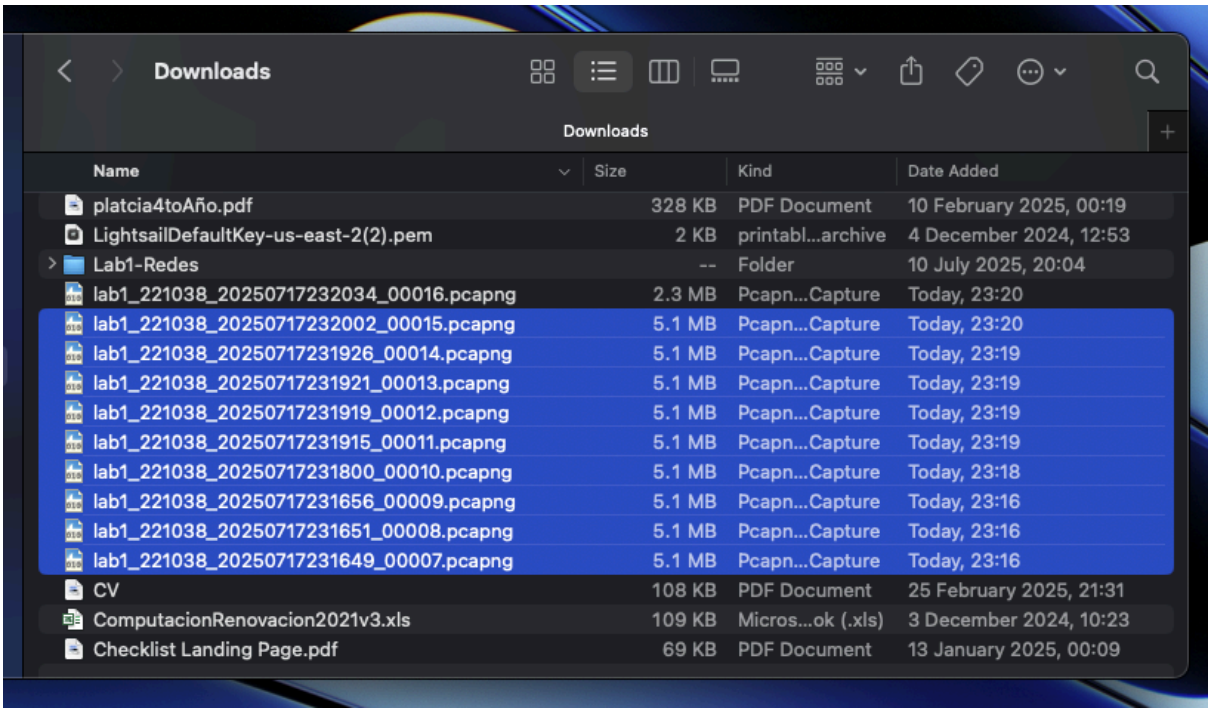
- **Segunda parte: configuración de la captura de paquetes**

```
Last login: Sun Jul 13 08:02:20 on console
jorgelopez@Jorges-MacBook-Air ~ % ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        nd6 options=201<PERFORMNUD,DAD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
anpi0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
```

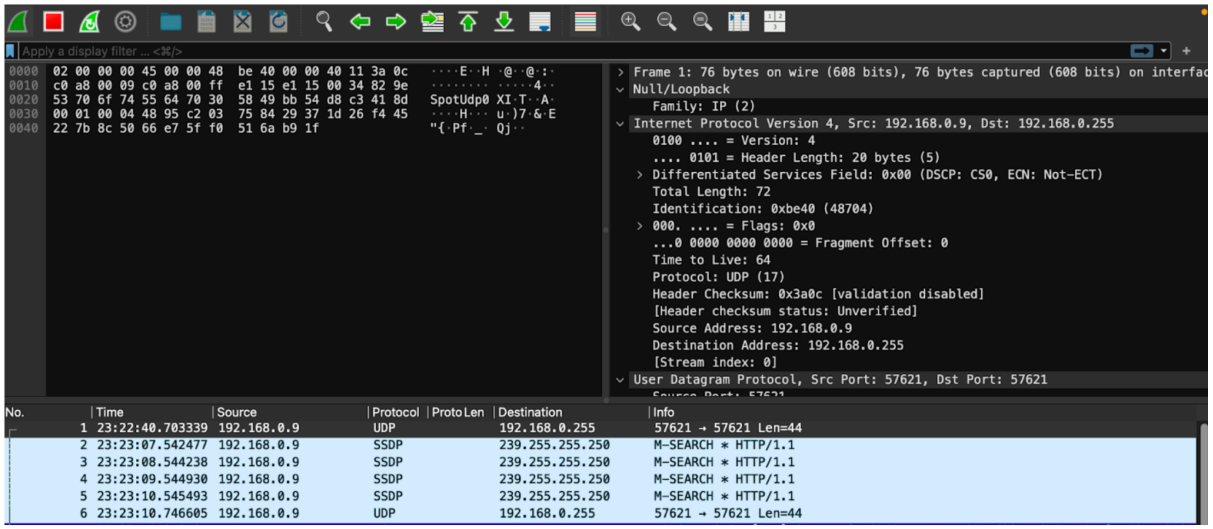
Mi interfaz de red activa es: **en0**
Dirección IPv4: `192.168.0.5`
Las demás interfaces (`lo0`, `utun`, `awdl0`, etc.) no se utilizan para la conexión a internet en este momento.




Capture to a permanent file
File: /Users/jorgelopez/Downloads/lab1_221038.pcapng
Output format: pcapng pcap
Create a new file automatically...
after 100000 packets
after 5120 kilobytes
after 1 seconds
when time is a multiple of 1 hours
compression
None
gzip
File infix pattern
YYYYmmDDHHMMSS_NNNNN
NNNNN_YYYYmmDDHHMMSS
Use a ring buffer with 10 files


Downloads
Downloads
Name
Size
Kind
Date Added
platcia4toAño.pdf 328 KB PDF Document 10 February 2025, 00:19
LightsailDefaultKey-us-east-2(2).pem 2 KB printabl...archive 4 December 2024, 12:53
Lab1-Redes -- Folder 10 July 2025, 20:04
lab1_221038_20250717232034_00016.pcapng 2.3 MB Pcapn...Capture Today, 23:20
lab1_221038_20250717232002_00015.pcapng 5.1 MB Pcapn...Capture Today, 23:20
lab1_221038_20250717231926_00014.pcapng 5.1 MB Pcapn...Capture Today, 23:19
lab1_221038_20250717231921_00013.pcapng 5.1 MB Pcapn...Capture Today, 23:19
lab1_221038_20250717231919_00012.pcapng 5.1 MB Pcapn...Capture Today, 23:19
lab1_221038_20250717231915_00011.pcapng 5.1 MB Pcapn...Capture Today, 23:19
lab1_221038_20250717231800_00010.pcapng 5.1 MB Pcapn...Capture Today, 23:18
lab1_221038_20250717231656_00009.pcapng 5.1 MB Pcapn...Capture Today, 23:16
lab1_221038_20250717231651_00008.pcapng 5.1 MB Pcapn...Capture Today, 23:16
lab1_221038_20250717231649_00007.pcapng 5.1 MB Pcapn...Capture Today, 23:16
CV 108 KB PDF Document 25 February 2025, 21:31
ComputacionRenovacion2021v3.xls 109 KB Micros...ok (.xls) 3 December 2024, 10:23
Checklist Landing Page.pdf 69 KB PDF Document 13 January 2025, 00:09

- **Tercera parte: análisis de paquetes HTTP**



Análisis:

En la captura realizada sobre la interfaz de loopback (lo0) se observa principalmente tráfico de descubrimiento de servicios locales: por un lado, paquetes UDP broadcast dirigidos a la dirección 192.168.0.255 (puerto 57621 → 57621, longitud de 44 bytes), que corresponden a mensajes de anuncio o consulta de un servicio propio de la máquina; y por otro, varias solicitudes SSDP “M-SEARCH * HTTP/1.1” enviadas a la dirección multicast 239.255.255.250, empleadas para detectar dispositivos y servicios UPnP en la red local. Al tratarse de comunicaciones de red interna, no aparecen aquí peticiones HTTP estándar hacia servidores remotos—estas solo se capturan al monitorizar la interfaz física (en0). La presencia de estos protocolos de descubrimiento indica que la máquina está intentando localizar otros nodos y servicios en su subred antes de establecer conexiones basadas en TCP o HTTP.

a. Versión HTTP del navegador
Al inspeccionar la primera línea de la petición registrada, encontramos un encabezado que inicia con

GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1

Esto confirma que el navegador está utilizando la versión HTTP/1.1 para comunicarse con el servidor.

b. Versión HTTP del servidor
En la respuesta correspondiente al GET anterior, el servidor devuelve una línea de estado que comienza por HTTP/1.1 200 OK
De modo que, al igual que el cliente, el servidor también está ejecutando HTTP/1.1.

c. Lenguajes que acepta el navegador

Dentro de los encabezados de la petición, aparece la línea

Accept-Language: en-US,en;q=0.5
lo cual indica que el navegador prefiere contenido en inglés de Estados Unidos y acepta también otros dialectos de inglés con un factor de calidad (“q”) de 0.5.

d. Bytes de contenido devueltos por el servidor

En la sección de encabezados de la respuesta HTTP encontramos

Content-Length: 4832
Este valor nos dice que el servidor ha enviado 4832 bytes de contenido en el cuerpo de la respuesta.

e. Dónde “escuchar” los paquetes y si conviene instalar Wireshark en el servidor
Para diagnosticar problemas de rendimiento (latencia, pérdidas o retransmisiones) es más efectivo capturar el tráfico en puntos clave de la red, como el enlace cliente→switch y el enlace switch→servidor, utilizando un puerto mirror (SPAN) o un dispositivo de sniffing dedicado. Instalar Wireshark directamente en el servidor de producción no es recomendable, ya que el propio proceso de captura consume recursos de CPU y memoria, lo que podría alterar los tiempos de respuesta que se pretenden medir y añadir sesgo a los resultados.

Discusión sobre la actividad, experiencia y hallazgos

La primera parte del laboratorio (personalización del entorno) permitió familiarizarme con las opciones avanzadas de Wireshark: creación de perfiles, ajuste del formato de tiempo, columnas personalizadas (“Proto Len”), esquemas de paneles, reglas de color y botones de filtro. Gracias a ello pude optimizar la visibilidad del tráfico que me interesaba—en especial los paquetes TCP SYN.

En la segunda parte, la configuración de un ring-buffer para captura automática destacó la potencia de Wireshark para entornos de alto volumen: definir un archivo base, límite de 5 MB y rotación de 10 archivos garantiza no perder datos cuando se monitoriza por largos períodos. Aquí encontré un reto práctico al seleccionar la interfaz correcta: inicialmente capturaba en gif0 o lo0 y no veía nada de tráfico HTTP, lo que me obligó a investigar la lista de interfaces con ifconfig y usar la interfaz física (en0) o, alternativamente, el loopback para pruebas locales.

Finalmente, el análisis HTTP de la tercera parte consolidó el entendimiento de cómo inspeccionar protocolos de aplicación: localizar la petición GET, extraer versiones HTTP, encabezados Accept-Language, Content-Length y comprender el flujo request–response. Este ejercicio resaltó la importancia de capturar en el punto de la red por donde realmente circula el tráfico de interés.

Comentarios

Interfaz virtual vs. física: macOS lista muchas interfaces túnel; distinguir cuál usar requirió consultar ifconfig.

Modo monitor vs. managed: en Wi-Fi es frecuente no ver tráfico IP en modo “managed” sin activar el modo monitor.

Usabilidad de Wireshark: la gran cantidad de opciones puede abrumar al principio, pero combinar perfiles, filtros y color-rules agiliza mucho el análisis.

Conclusiones

Este laboratorio demostró que, con una configuración adecuada, Wireshark es una herramienta extremadamente flexible tanto para tareas puntuales (ver un GET HTTP) como para monitoreos continuos sin intervención (ring-buffer). Crear un perfil dedicado y personalizar la interfaz de usuario ayuda a centrarse en los datos más relevantes. Asimismo, aprender a seleccionar correctamente la interfaz de captura es esencial para no filtrar tráfico erróneo. Finalmente, el análisis de HTTP me permitió reforzar conceptos de protocolos de capa de aplicación y buenas prácticas para diagnósticos de rendimiento.

Referencias Utilizadas

Apple Inc. (n.d.). ifconfig(8) manual page. Recuperado el 17 de julio de 2025, de https://developer.apple.com/library/archive/documentation/Darwin/Reference/ManPages/man8/ifconfig.8.html

Internet Engineering Task Force. (2014). RFC 7230: Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing. Recuperado de https://tools.ietf.org/html/rfc7230

UPnP Forum. (2017). Simple Service Discovery Protocol (SSDP). Recuperado de https://openconnectivity.org/specs/upnp/2017/SSDP.pdf

Wireshark Foundation. (n.d.-a). Wireshark User’s Guide. Recuperado el 17 de julio de 2025, de https://www.wireshark.org/docs/wsug_html_chunked/

Wireshark Foundation. (n.d.-b). Ring Buffer Options. En Wireshark User’s Guide. Recuperado el 17 de julio de 2025, de https://www.wireshark.org/docs/wsug_html_chunked/ChCaptureRingBuffer.html