Departamento de Informática
Universidad Técnica Federico Santa María
Campus San Joaquín

Wireshark

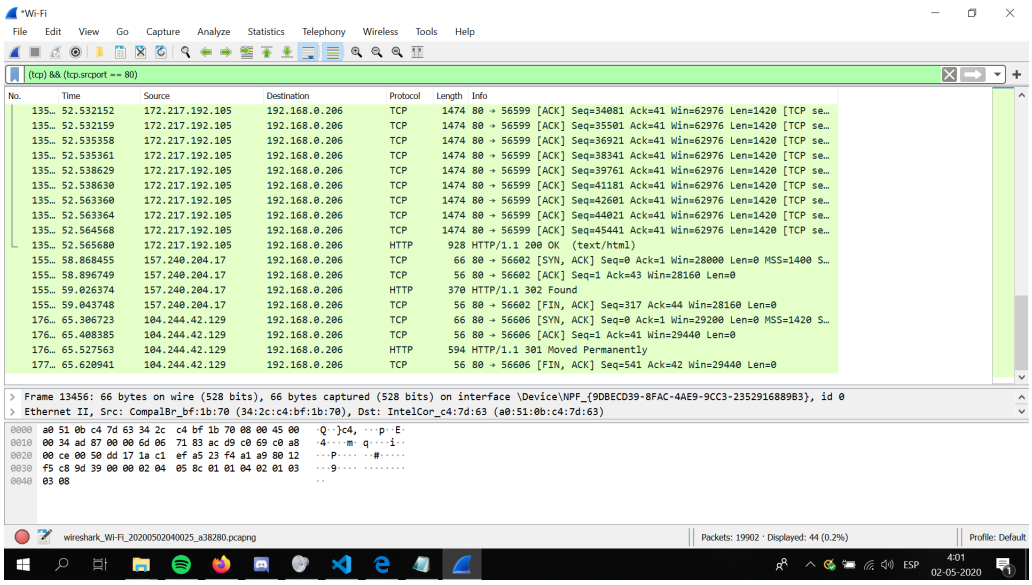
Sebastián Campos
201773517-1

Axel Reyes
201773502-3

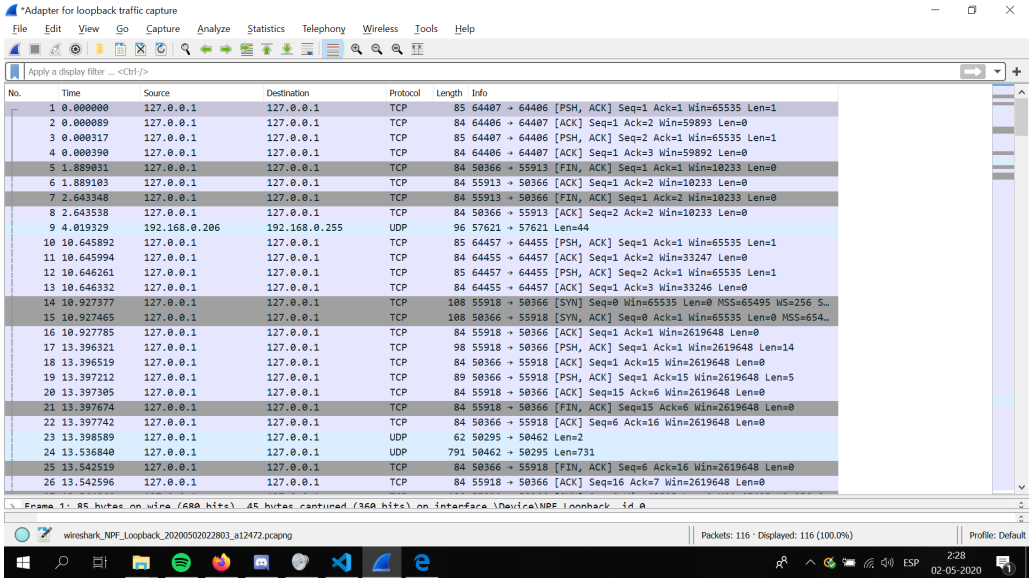
Mayo 2020

1. Referente a los mensajes realizados por las aplicaciones: ¿Qué tipos de protocolo espera ver? ¿Cuáles encontró? Justifique sus expectativas y las diferencias que encuentre.

   Dentro de los protocoles que se esperó observar estaban TCP, UDP y HTTP. Luego, al utilizar Wireshark encontramos que dentro de la interfaz de Wi-Fi se encontró TCP, HTTP y protocolo de IP como se puede ver en la imagen:



   Mientras que para la interfaz de Adapter for loopback traffic capture se encontró TCP, UDP y protocolo de IP:

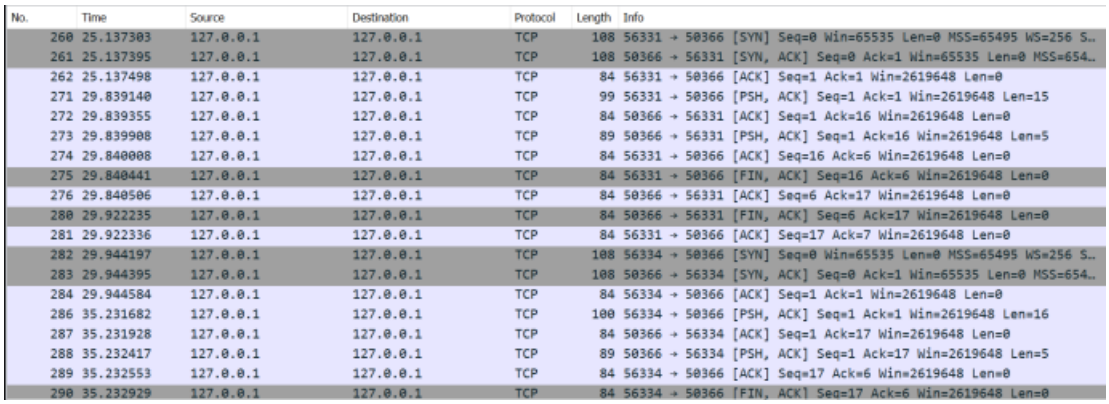


2. Las interacciones vía TCP entre el cliente y el servidor, ¿deben ocupan los mismos puertos a lo largo del tiempo? ¿Coincide con lo visto en Wireshark? Fundamente.

   No necesariamente. Si bien el servidor 'bindea' un puerto (el cuál se mantendrá constante para el mismo cliente y entre clientes), el puerto utilizado por el cliente variará, ya que el socket utilizado por el cliente no tiene ningún puerto específico asociado.

   Al recurrir a WireShark, obtuvimos las siguientes capturas:

No.	Time	Source	Destination	Protocol	Length	Info
232	6.383812	127.0.0.1	127.0.0.1	TCP	108	56326 → 50366 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 S…
233	6.383937	127.0.0.1	127.0.0.1	TCP	108	50366 → 56326 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=654…
234	6.384298	127.0.0.1	127.0.0.1	TCP	84	56326 → 50366 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
235	9.683125	127.0.0.1	127.0.0.1	TCP	84	50366 → 56326 [FIN, ACK] Seq=1 Ack=1 Win=2619648 Len=0
236	9.683242	127.0.0.1	127.0.0.1	TCP	84	56326 → 50366 [ACK] Seq=1 Ack=2 Win=2619648 Len=0
237	10.468165	127.0.0.1	127.0.0.1	TCP	84	56326 → 50366 [FIN, ACK] Seq=1 Ack=2 Win=2619648 Len=0
238	10.468286	127.0.0.1	127.0.0.1	TCP	84	50366 → 56326 [ACK] Seq=2 Ack=2 Win=2619648 Len=0
247	21.257435	127.0.0.1	127.0.0.1	TCP	108	50366 → 56329 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 S…
248	21.257516	127.0.0.1	127.0.0.1	TCP	108	56329 → 50366 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=654…
249	21.257834	127.0.0.1	127.0.0.1	TCP	84	50366 → 56329 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
250	24.997609	127.0.0.1	127.0.0.1	TCP	98	56329 → 50366 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=14
251	24.997774	127.0.0.1	127.0.0.1	TCP	84	50366 → 56329 [ACK] Seq=1 Ack=15 Win=2619648 Len=0
252	24.998395	127.0.0.1	127.0.0.1	TCP	89	50366 → 56329 [PSH, ACK] Seq=1 Ack=15 Win=2619648 Len=5
253	24.998505	127.0.0.1	127.0.0.1	TCP	84	56329 → 50366 [ACK] Seq=15 Ack=6 Win=2619648 Len=0
254	24.998993	127.0.0.1	127.0.0.1	TCP	84	56329 → 50366 [FIN, ACK] Seq=15 Ack=6 Win=2619648 Len=0
255	24.999070	127.0.0.1	127.0.0.1	TCP	84	50366 → 56329 [ACK] Seq=6 Ack=16 Win=2619648 Len=0
258	25.121941	127.0.0.1	127.0.0.1	TCP	84	50366 → 56329 [FIN, ACK] Seq=6 Ack=16 Win=2619648 Len=0
259	25.122022	127.0.0.1	127.0.0.1	TCP	84	56329 → 50366 [ACK] Seq=16 Ack=7 Win=2619648 Len=0
260	25.137303	127.0.0.1	127.0.0.1	TCP	108	56331 → 50366 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 S…

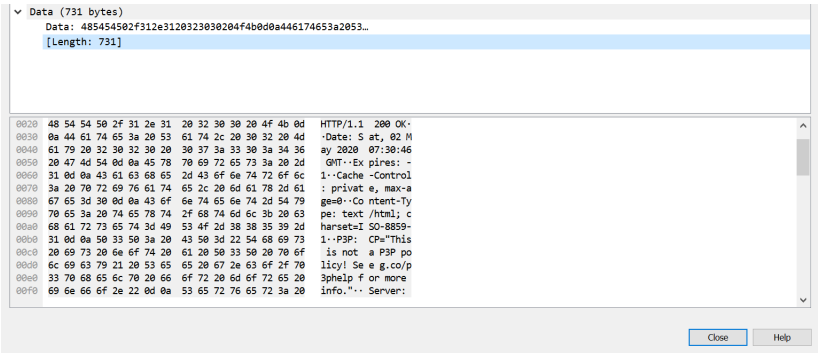


No.	Time	Source	Destination	Protocol	Length	Info
260	25.137303	127.0.0.1	127.0.0.1	TCP	108	56331 → 50366 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 S…
261	25.137395	127.0.0.1	127.0.0.1	TCP	108	50366 → 56331 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=654…
262	25.137498	127.0.0.1	127.0.0.1	TCP	84	56331 → 50366 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
271	29.839140	127.0.0.1	127.0.0.1	TCP	99	56331 → 50366 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=15
272	29.839355	127.0.0.1	127.0.0.1	TCP	84	50366 → 56331 [ACK] Seq=1 Ack=16 Win=2619648 Len=0
273	29.839908	127.0.0.1	127.0.0.1	TCP	89	50366 → 56331 [PSH, ACK] Seq=1 Ack=16 Win=2619648 Len=5
274	29.840008	127.0.0.1	127.0.0.1	TCP	84	56331 → 50366 [ACK] Seq=16 Ack=6 Win=2619648 Len=0
275	29.840441	127.0.0.1	127.0.0.1	TCP	84	56331 → 50366 [FIN, ACK] Seq=16 Ack=6 Win=2619648 Len=0
276	29.840506	127.0.0.1	127.0.0.1	TCP	84	50366 → 56331 [ACK] Seq=6 Ack=17 Win=2619648 Len=0
280	29.922235	127.0.0.1	127.0.0.1	TCP	84	50366 → 56331 [FIN, ACK] Seq=6 Ack=17 Win=2619648 Len=0
281	29.922336	127.0.0.1	127.0.0.1	TCP	84	56331 → 50366 [ACK] Seq=17 Ack=7 Win=2619648 Len=0
282	29.944197	127.0.0.1	127.0.0.1	TCP	108	56334 → 50366 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 S…
283	29.944395	127.0.0.1	127.0.0.1	TCP	108	50366 → 56334 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=654…
284	29.944584	127.0.0.1	127.0.0.1	TCP	84	56334 → 50366 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
286	35.231682	127.0.0.1	127.0.0.1	TCP	100	56334 → 50366 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=16
287	35.231928	127.0.0.1	127.0.0.1	TCP	84	50366 → 56334 [ACK] Seq=1 Ack=17 Win=2619648 Len=0
288	35.232417	127.0.0.1	127.0.0.1	TCP	89	50366 → 56334 [PSH, ACK] Seq=1 Ack=17 Win=2619648 Len=5
289	35.232553	127.0.0.1	127.0.0.1	TCP	84	56334 → 50366 [ACK] Seq=17 Ack=6 Win=2619648 Len=0
290	35.232929	127.0.0.1	127.0.0.1	TCP	84	56334 → 50366 [FIN, ACK] Seq=17 Ack=6 Win=2619648 Len=0

El puerto relacionado al servidor es el 50366, en las imágenes se puede notar que interactúa por lo menos con 4 puertos distintos: 56331, 56334, 56326 y 56329.

3. Los contenidos de los mensajes enviados entre las aplicaciones, ¿son legibles?

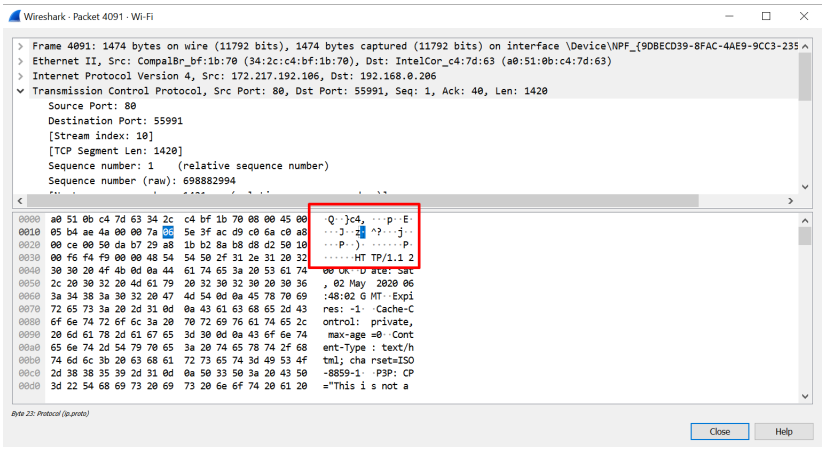
   Wireshark muestra el mensaje codificado a la izquierda, y el mensaje decodificado a la derecha, como se puede ver en la imagen:



   Por lo tanto, al necesitar que Wireshark decodifique el mensaje, este no sería legible a priori.

4. Encuentre la respuesta a la consulta HTTP recibida por el servidor, ¿el header es igual al almacenado por el cliente, o existe alguna diferencia importante? Explique.

   Al compara el mensaje decodificado mostrado por Wireshark y el que almacena el cliente, se puede observar difieren en algunos códigos al principio como se ve en la siguiente imagen:



   Luego, se puede observar el html entregado (el cual no lo guarda el cliente), y finalmente se ven unos códigos que puede que sean como los del inicio o simplemente formen parte del html: