Memoria Bloque 2 de Prácticas

Redes y Sistemas Distribuidos

Resumen de contenidos:

- **Cliente-Servidor básico sobre UDP**
- **Cliente-Servidor básico sobre TCP**

Nombre y Apellidos: Marcos Hidalgo Baños

Titulación: Ingeniería Informática D -- Grupo de Prácticas 3

Práctica Bloque II

Apellidos, Nombre: Hidalgo Baños, Marcos

Titulación: Grado de Ingeniería Informática (Grupo D)

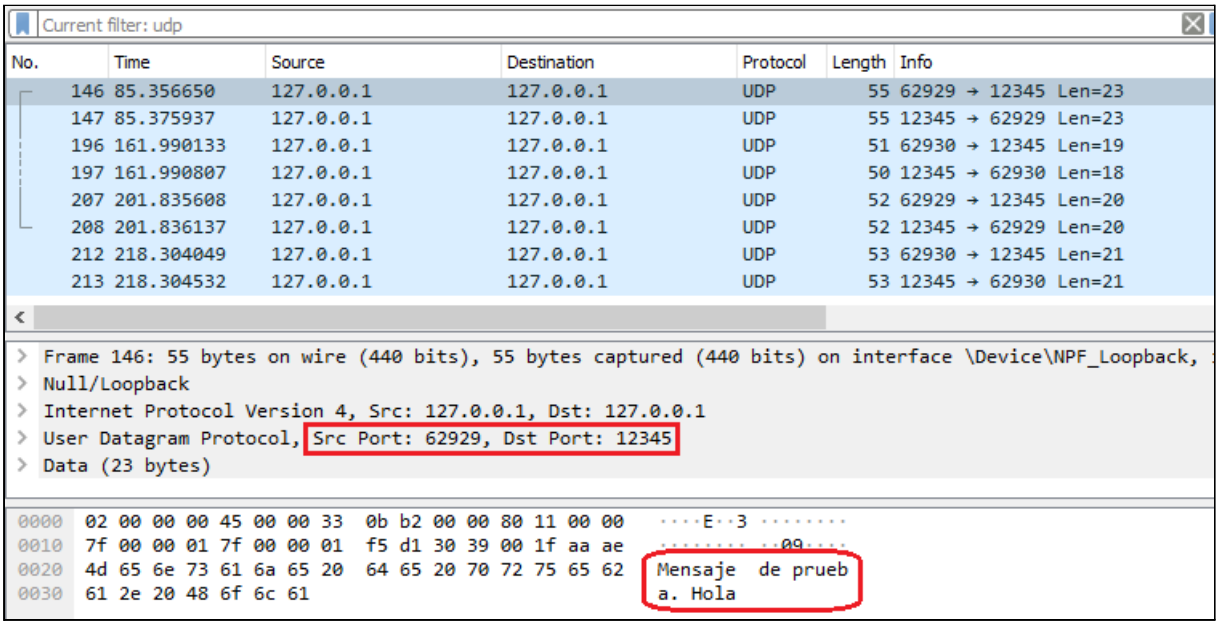
PC de la práctica: Ordenador propio

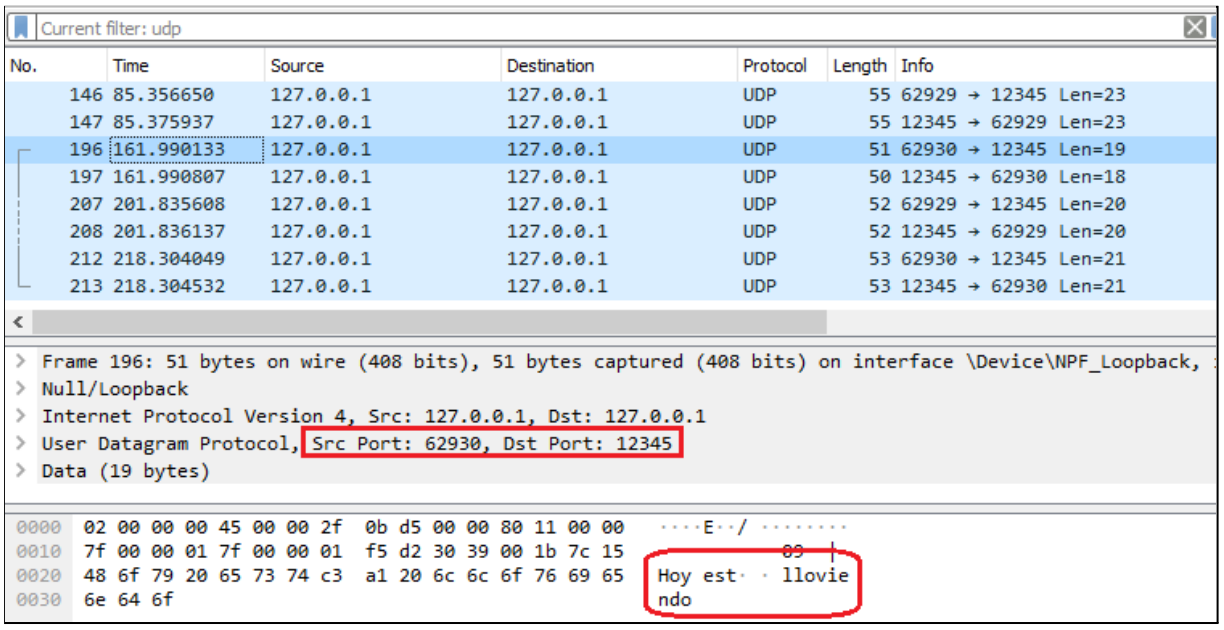
Las debidas explicaciones del código utilizado para la realización de esta práctica se encuentran en forma de comentarios en el propio código adjuntado junto con el consiguiente informe

Parte I: Protocolo UDP

Usando la traza UDP1 (b2e1-3.pcapng).

Ejercicio 1.





- **¿Cuál es el puerto que usa el cliente?**

  *El Cliente 1 utiliza el puerto **62929** y el Cliente 2 utiliza el puerto **62930**.*

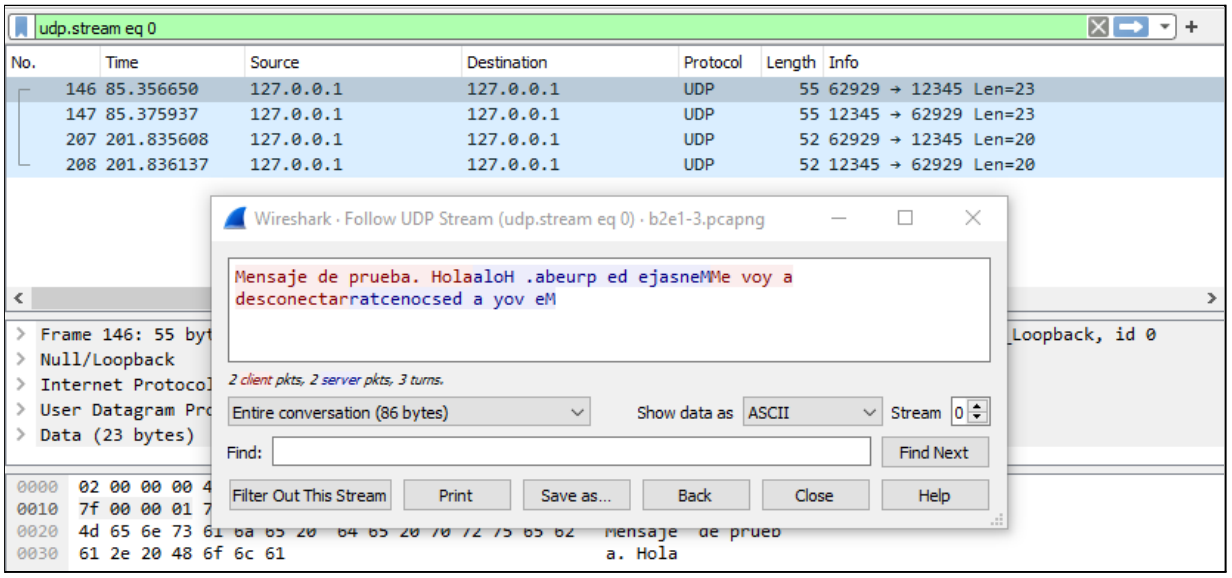
- **¿Y el servidor?**

  *El servidor emplea el puerto **12345**, tal y como le indicamos como parámetro.*

- **¿Qué tipo de puerto es cada uno de ellos?**

  *Un puerto puede ser de entrada o de salida, que visto en Wireshark reciben el nombre de **Destination Port** y **Source Port** respectivamente.*

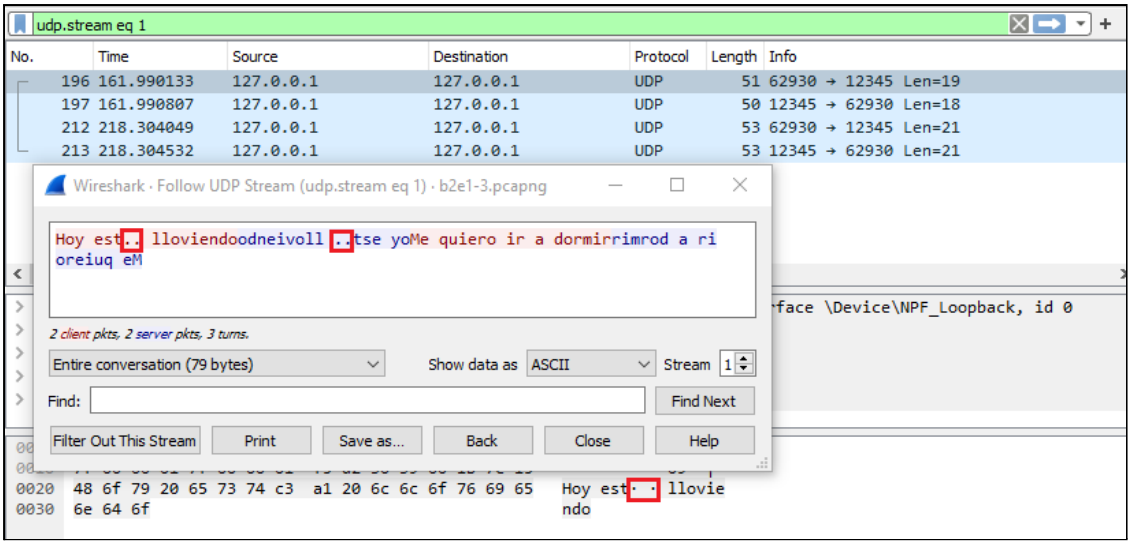
Ejercicio 2. Wireshark ofrece la opción de "Follow UDP stream", pero en UDP no existe tal concepto.



- **¿Cómo es capaz Wireshark de decidir que un mensaje pertenece a un "flujo" u a otro?**

  Wireshark puede filtrar por flujo o conversación porque resulta fácil determinar que dos mensajes son de la misma conversación ya que los puertos de destino y origen coinciden en ambos casos, aunque se hayan intercambiado entre sí.

Ejercicio 3. Examine un mensaje que lleve tildes.



- **¿Coincide el tamaño indicado en el campo longitud de la cabecera de UDP con la cantidad de letras enviadas?**

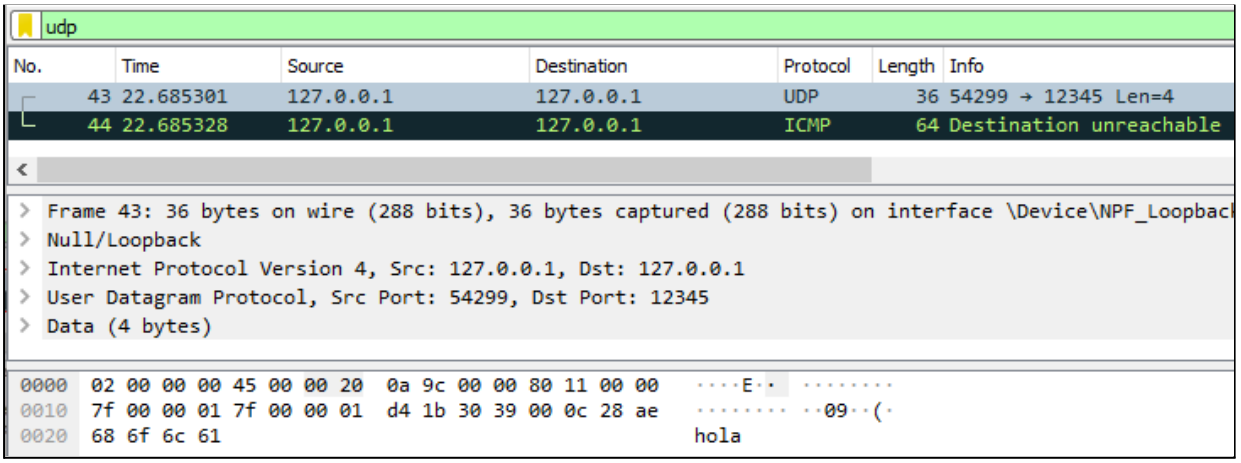
  No coincide con lo esperado.
  Resulta sospechoso que no se muestren los caracteres con tilde...

- **¿Por qué?**

  Como era de esperar, la culpa de esta incoherencia la tienen las tildes. Como los caracteres con tilde se transmiten como dos puntos, en lugar de ocupar 1 byte (lo que le corresponde) se utilizan 2 bytes (uno por cada punto).

Usando la traza UDP2 (b2e4.pcapng).

Ejercicio 4.



- **¿Por qué consigue enviar si no hay ningún servidor activo?**

  El protocolo UDP permite que un cliente mande un mensaje sin que se haya establecido una conexión previa con un servidor.

- **¿Recibe alguna respuesta?**

  Como podemos observar, el destino es inalcanzable y no se recibe ninguna respuesta.

- **En caso afirmativo, indique qué significa esa respuesta y si es tratada o no.**

  Como consecuencia, el cliente se queda esperando indefinidamente una respuesta.

Sin traza.

Ejercicio 5. Asegure que captura la excepción de la creación del socket UDP y que muestra (método `getMessage()`) el error que se produce (modifique el código si no lo hacía). Intente abrir dos veces el servidor con los mismos parámetros.

- **¿Qué error indica que se produce?**

  Address already in use: Cannot bind

- **¿Qué debería hacer para solucionar ese error y tener dos servidores del mismo tipo en su equipo?**

  Asignar un puerto diferente a cada uno de ellos

Parte II: Protocolo TCP

Usando la traza TCP1 (b2e6-9.pcapng).

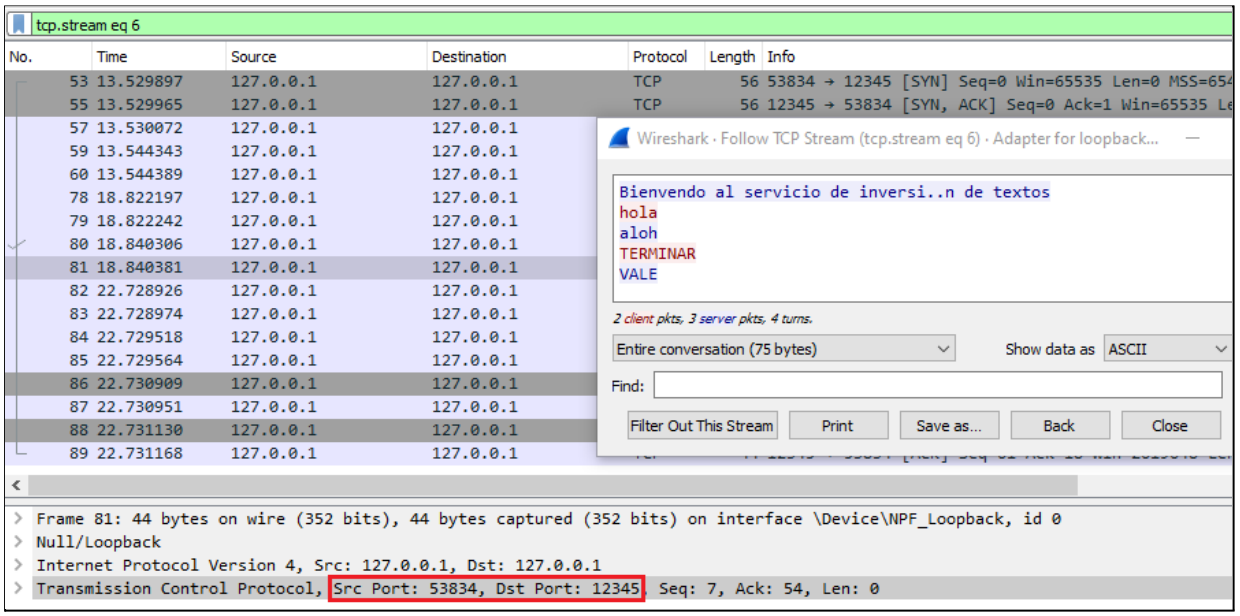
Ejercicio 6. Identifique una trama de la comunicación y use la opción "`Follow TCP stream`" para ver el intercambio de información entre cliente y servidor.

- **¿Cuál es el puerto que usa el cliente?**

  53834

- **¿Y el servidor?**

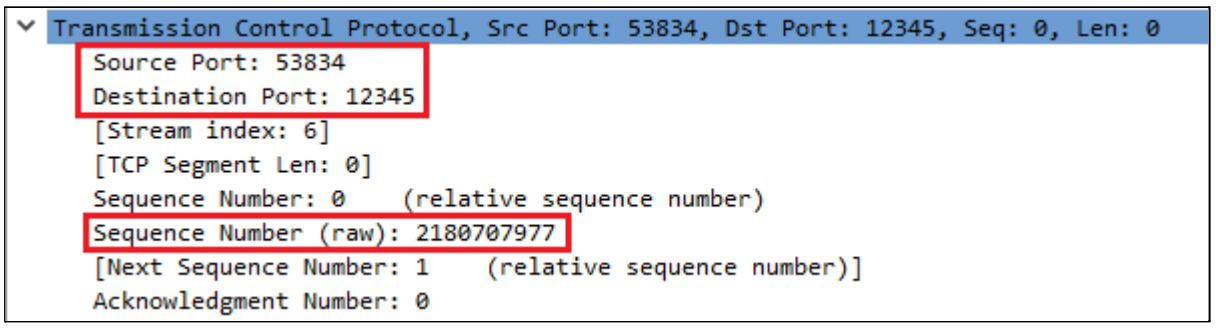
  12345



Ejercicio 7.

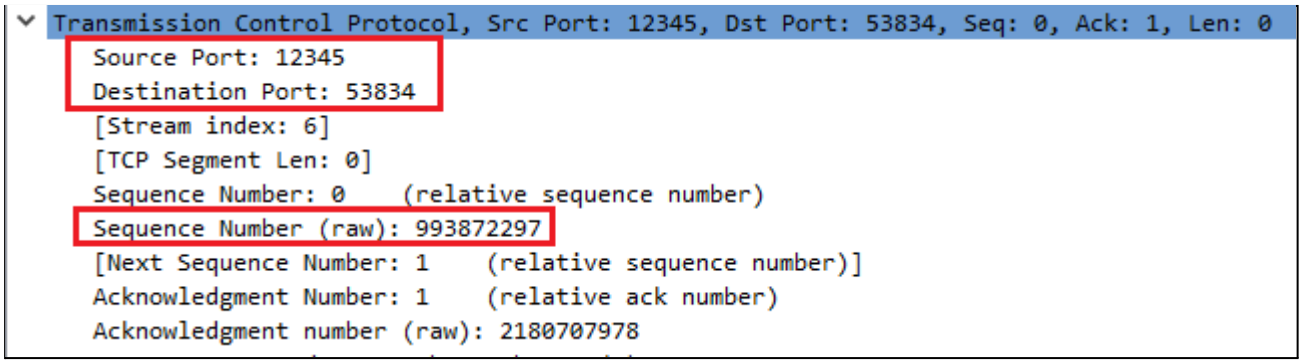
- **¿Cuál es el número de secuencia que usa el cliente TCP hacia el servidor?**

  Del cliente al servidor: 2180707977



- **¿Y las respuestas del servidor al cliente?**

  Del servidor al cliente: 993872297



Ejercicio 8. Indique los segmentos relacionados con las siguientes actividades y qué métodos de `Socket` y `ServerSocket` son responsables del intercambio de estos segmentos:

a) Inicialización de la conexión. (Color rojo)

Servidor → *ServerSocket server = new ServerSocket (port, backlog);*
→ *Socket client = server.accept();*

Cliente → *Socket serverSocket = new Socket (servername, serverPort);*

b) Envío de datos. (Color azul)

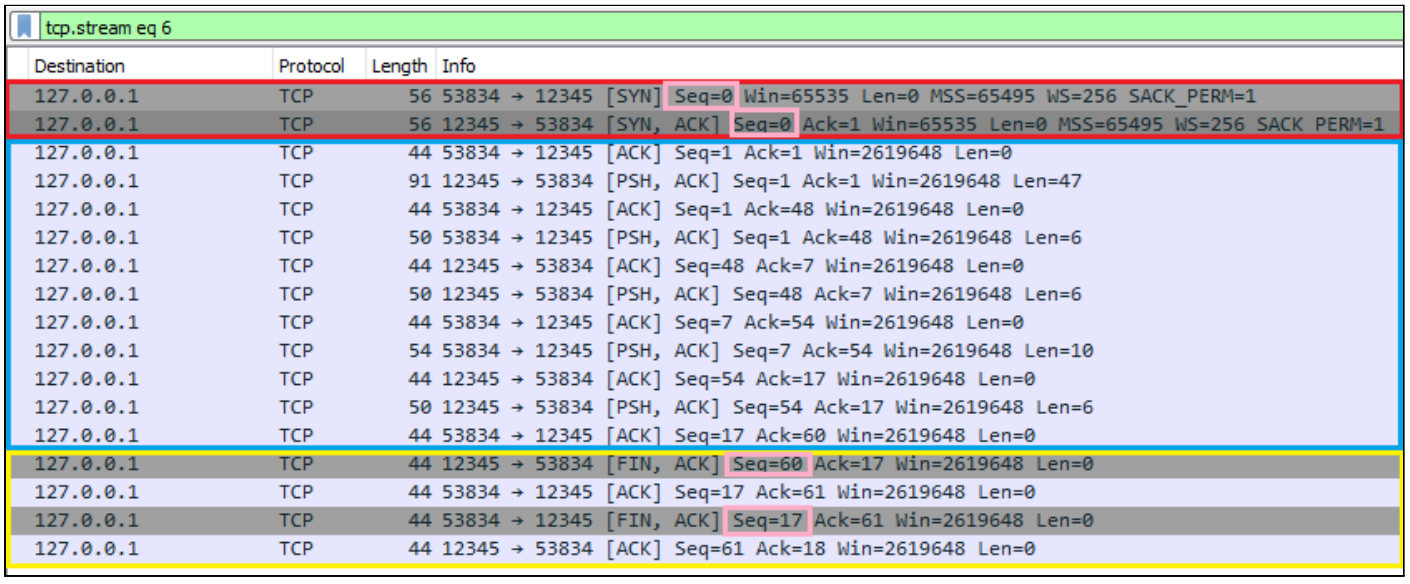
Envío de datos → *PrintWriter out = new PrintWriter (socket.getOutputStream(), true);*
→ *out.println(mensaje);*
→ *out.flush();*

Recibo de datos → *BufferedReader in = new BufferedReader (*
new InputsStreamReader (socket.getInputStream()));
→ *String line = in.readLine();*

c) Finalización de la conexión. (Color amarillo)

Cierre Buffers → *in.close();*
→ *out.close();*

Cierre de sockets → *serviceSocket.close();*
→ *client.close();*



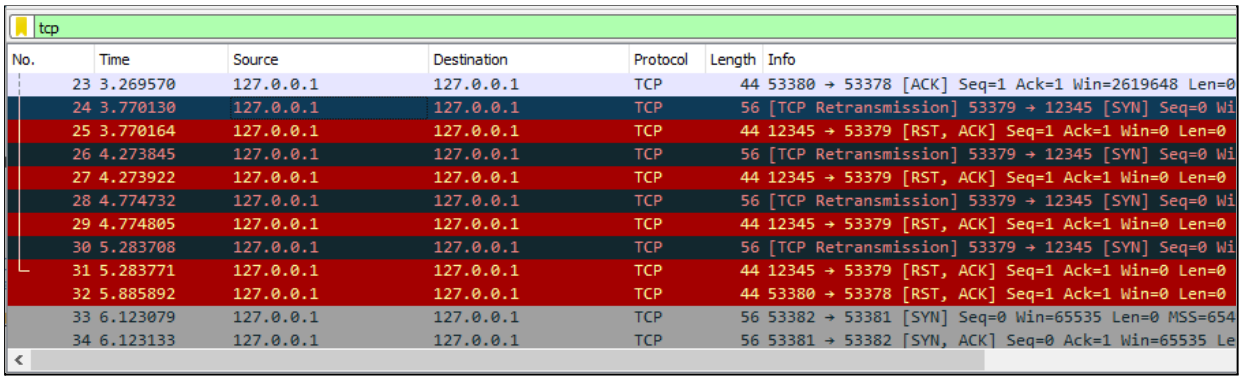
Ejercicio 9.

- **¿Cuántos números de secuencia se consumen en cada lado (cliente y servidor) durante el inicio y cierre de la conexión?**

  Teniendo en cuenta que los números de secuencia comienzan en 0, se acaba para el cliente en 17 y el del servidor en 61.

Usando la traza TCP2 (b2e10.pcapng).

Ejercicio 10.



tcp

No.	Time	Source	Destination	Protocol	Length	Info
23	3.269570	127.0.0.1	127.0.0.1	TCP	44	53380 → 53378 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
24	3.770130	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53379 → 12345 [SYN] Seq=0 Wi
25	3.770164	127.0.0.1	127.0.0.1	TCP	44	12345 → 53379 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
26	4.273845	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53379 → 12345 [SYN] Seq=0 Wi
27	4.273922	127.0.0.1	127.0.0.1	TCP	44	12345 → 53379 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
28	4.774732	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53379 → 12345 [SYN] Seq=0 Wi
29	4.774805	127.0.0.1	127.0.0.1	TCP	44	12345 → 53379 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
30	5.283708	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53379 → 12345 [SYN] Seq=0 Wi
31	5.283771	127.0.0.1	127.0.0.1	TCP	44	12345 → 53379 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
32	5.885892	127.0.0.1	127.0.0.1	TCP	44	53380 → 53378 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
33	6.123079	127.0.0.1	127.0.0.1	TCP	56	53382 → 53381 [SYN] Seq=0 Win=65535 Len=0 MSS=654
34	6.123133	127.0.0.1	127.0.0.1	TCP	56	53381 → 53382 [SYN, ACK] Seq=0 Ack=1 Win=65535 Le

- **¿Recibe algún tipo de respuesta el intento de conexión del cliente?**

  La respuesta recibida por parte del servidor para el cliente es un rechazo de conexión.

- **En caso afirmativo ¿tiene alguna característica especial?**

  Tal y como se ve en la captura de pantalla, se producen múltiples rechazos de conexión ya que el cliente trata de establecer una comunicación varias veces.

Usando la traza TCP3 (b2e11-12.pcapng).

Ejercicio 11.

- **¿Se logran conectar los 3 clientes?**

  El primer cliente puede hacerlo sin problemas, pero el segundo se mantiene en la lista de espera hasta que termine su predecesor.

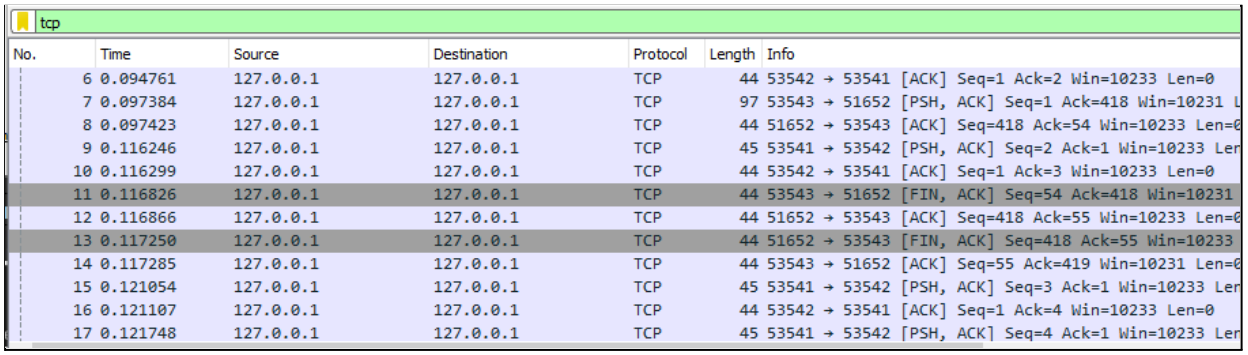
  El tercer cliente ni siquiera puede acceder al sistema porque se definió el tamaño de dicha lista para que solamente pudiera albergar a un único cliente en espera.

- **En caso de que alguno no se haya podido conectar, ¿se le indica de alguna forma que la cola está llena?**

  Sí, se le muestra que el servidor no está operativo y disponible para su uso.

Cliente 1.

```
Conectado al servidor con IP '127.0.0.1' y con puerto '12345'
Bienvendo al servicio de inversión de textos
Introduzca un texto a enviar (END para acabar):
```

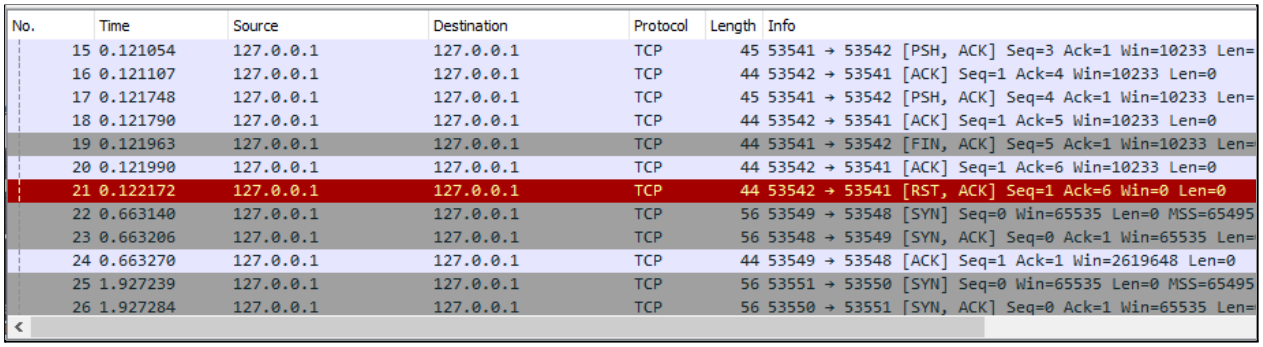


tcp

No.	Time	Source	Destination	Protocol	Length	Info
6	0.094761	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=2 Win=10233 Len=0
7	0.097384	127.0.0.1	127.0.0.1	TCP	97	53543 → 51652 [PSH, ACK] Seq=1 Ack=418 Win=10231 L
8	0.097423	127.0.0.1	127.0.0.1	TCP	44	51652 → 53543 [ACK] Seq=418 Ack=54 Win=10233 Len=0
9	0.116246	127.0.0.1	127.0.0.1	TCP	45	53541 → 53542 [PSH, ACK] Seq=2 Ack=1 Win=10233 Len
10	0.116299	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=3 Win=10233 Len=0
11	0.116826	127.0.0.1	127.0.0.1	TCP	44	53543 → 51652 [FIN, ACK] Seq=54 Ack=418 Win=10231
12	0.116866	127.0.0.1	127.0.0.1	TCP	44	51652 → 53543 [ACK] Seq=418 Ack=55 Win=10233 Len=0
13	0.117250	127.0.0.1	127.0.0.1	TCP	44	51652 → 53543 [FIN, ACK] Seq=418 Ack=55 Win=10233
14	0.117285	127.0.0.1	127.0.0.1	TCP	44	53543 → 51652 [ACK] Seq=55 Ack=419 Win=10231 Len=0
15	0.121054	127.0.0.1	127.0.0.1	TCP	45	53541 → 53542 [PSH, ACK] Seq=3 Ack=1 Win=10233 Len
16	0.121107	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=4 Win=10233 Len=0
17	0.121748	127.0.0.1	127.0.0.1	TCP	45	53541 → 53542 [PSH, ACK] Seq=4 Ack=1 Win=10233 Len

Cliente 2.

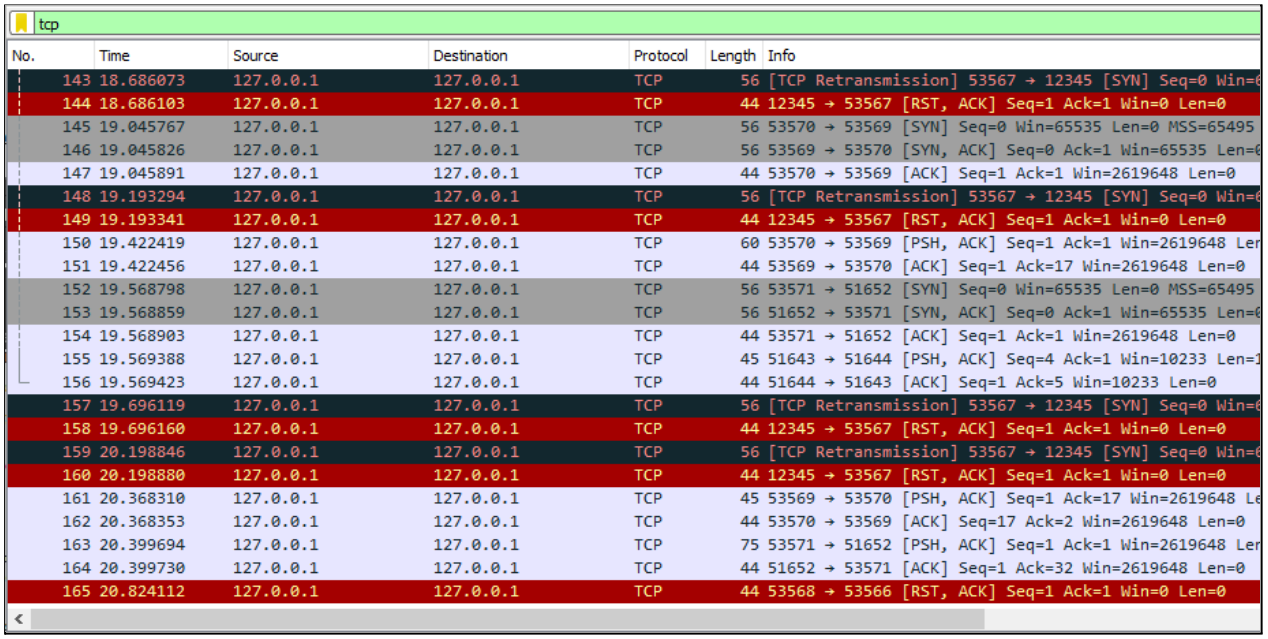
```
Conectado al servidor con IP '127.0.0.1' y con puerto '12345'
```



No.	Time	Source	Destination	Protocol	Length	Info
15	0.121054	127.0.0.1	127.0.0.1	TCP	45	53541 → 53542 [PSH, ACK] Seq=3 Ack=1 Win=10233 Len=
16	0.121107	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=4 Win=10233 Len=0
17	0.121748	127.0.0.1	127.0.0.1	TCP	45	53541 → 53542 [PSH, ACK] Seq=4 Ack=1 Win=10233 Len=
18	0.121790	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=5 Win=10233 Len=0
19	0.121963	127.0.0.1	127.0.0.1	TCP	44	53541 → 53542 [FIN, ACK] Seq=5 Ack=1 Win=10233 Len=
20	0.121990	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [ACK] Seq=1 Ack=6 Win=10233 Len=0
21	0.122172	127.0.0.1	127.0.0.1	TCP	44	53542 → 53541 [RST, ACK] Seq=1 Ack=6 Win=0 Len=0
22	0.663140	127.0.0.1	127.0.0.1	TCP	56	53549 → 53548 [SYN] Seq=0 Win=65535 Len=0 MSS=65495
23	0.663206	127.0.0.1	127.0.0.1	TCP	56	53548 → 53549 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=
24	0.663270	127.0.0.1	127.0.0.1	TCP	44	53549 → 53548 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
25	1.927239	127.0.0.1	127.0.0.1	TCP	56	53551 → 53550 [SYN] Seq=0 Win=65535 Len=0 MSS=65495
26	1.927284	127.0.0.1	127.0.0.1	TCP	56	53550 → 53551 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=

Cliente 3.

```
Servidor actualmente inoperativo. Disculpe las molestias.
```



tcp

No.	Time	Source	Destination	Protocol	Length	Info
143	18.686073	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53567 → 12345 [SYN] Seq=0 Win=
144	18.686103	127.0.0.1	127.0.0.1	TCP	44	12345 → 53567 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
145	19.045767	127.0.0.1	127.0.0.1	TCP	56	53570 → 53569 [SYN] Seq=0 Win=65535 Len=0 MSS=65495
146	19.045826	127.0.0.1	127.0.0.1	TCP	56	53569 → 53570 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=
147	19.045891	127.0.0.1	127.0.0.1	TCP	44	53570 → 53569 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
148	19.193294	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53567 → 12345 [SYN] Seq=0 Win=
149	19.193341	127.0.0.1	127.0.0.1	TCP	44	12345 → 53567 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
150	19.422419	127.0.0.1	127.0.0.1	TCP	60	53570 → 53569 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len
151	19.422456	127.0.0.1	127.0.0.1	TCP	44	53569 → 53570 [ACK] Seq=1 Ack=17 Win=2619648 Len=0
152	19.568798	127.0.0.1	127.0.0.1	TCP	56	53571 → 51652 [SYN] Seq=0 Win=65535 Len=0 MSS=65495
153	19.568859	127.0.0.1	127.0.0.1	TCP	56	51652 → 53571 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=
154	19.568903	127.0.0.1	127.0.0.1	TCP	44	53571 → 51652 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
155	19.569388	127.0.0.1	127.0.0.1	TCP	45	51643 → 51644 [PSH, ACK] Seq=4 Ack=1 Win=10233 Len=
156	19.569423	127.0.0.1	127.0.0.1	TCP	44	51644 → 51643 [ACK] Seq=1 Ack=5 Win=10233 Len=0
157	19.696119	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53567 → 12345 [SYN] Seq=0 Win=
158	19.696160	127.0.0.1	127.0.0.1	TCP	44	12345 → 53567 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
159	20.198846	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 53567 → 12345 [SYN] Seq=0 Win=
160	20.198880	127.0.0.1	127.0.0.1	TCP	44	12345 → 53567 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
161	20.368310	127.0.0.1	127.0.0.1	TCP	45	53569 → 53570 [PSH, ACK] Seq=1 Ack=17 Win=2619648 Le
162	20.368353	127.0.0.1	127.0.0.1	TCP	44	53570 → 53569 [ACK] Seq=17 Ack=2 Win=2619648 Len=0
163	20.399694	127.0.0.1	127.0.0.1	TCP	75	53571 → 51652 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len
164	20.399730	127.0.0.1	127.0.0.1	TCP	44	51652 → 53571 [ACK] Seq=1 Ack=32 Win=2619648 Len=0
165	20.824112	127.0.0.1	127.0.0.1	TCP	44	53568 → 53566 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

<u>Ejercicio 12</u>.

- **¿Los clientes en espera (es decir los que están en la cola) tienen inicializada la conexión o esa inicialización se hace cuando se sacan de la cola (con el método accept)?**

  Estos clientes tienen la conexión inicializada pero esperan a que el servidor les acepte. Como podemos comprobar con el segundo cliente, muestra un mensaje de conexión establecida, pero no muestra el mensaje de bienvenida del servidor.