**PRÁCTICA 4: TCP Y DNS**

**1. Comunicación de aplicaciones usando el protocolo TCP .**

**1.1. Análisis inicial de la captura de TCP**

Utilizando Wireshark, abre la captura tcp.cap. En dicha captura se muestra una comunicación TCP entre dos aplicaciones. Contesta a las siguientes preguntas:

1. ¿Cuál es la dirección IP y el puerto del cliente TCP y la dirección IP y el puerto del servidor TCP?

Dirección IP cliente 10.0.0.2 puerto del cliente 60709.

Direccion IP servidor 11.0.0.2 puerto del servidor 34000.

2. ¿Cuántos segmentos TCP se han enviado desde el cliente al servidor?

Se han enviado seis con origen 10.0.0.2 que es el cliente.

3. ¿Cuántos segmentos TCP se han enviado desde el servidor al cliente?

Se han enviado cuatro con origen 11.0.0.2 que es el servidor.

4. Indica qué extremo cierra antes la conexión.

El 10.0.0.2, cliente, que es el primero que manda FIN.

**1.2. Números de secuencia**

Sigue analizando la captura tcp.cap. En el menú de Wireshark, seleccionando en el menú Edit→Preferences→Protocols→TCP, puedes desactivar la opción Relative Sequence Numbers & Window Scaling. De esta forma podrás observar los números de secuencia reales, en lugar de los números relativos que muestra por omisión Wireshark.

1. ¿Cuántos bytes de datos envía el servidor al cliente? Razona la respuesta. Indica cuáles son los números de secuencia del SYN y del FIN que envía el servidor, y qué relación tienen con la cantidad de datos enviada por el servidor al cliente.

En este tipo de ejercicios el servidor nunca envía nada al cliente, ya que solo le

está enviando ACK (que nunca transmiten información). Para saber qué cantidad

de datos lleva cada segmento, tenemos que mirar el apartado donde pone “Len =

...”

Luego el número de secuencia del SYN que envía es servidor es el 1363273423 y

el del FIN es 1363273424. En este caso el número de secuencia es el mismo del

inicio +1, ya que en ningún momento envía datos. Y si observamos len vemos que es 0 es entendible porque al ser el servidor está esperando y solo asiente.

2. ¿Cuántos bytes de datos envía el cliente al servidor? Razona la respuesta. Indica cuáles son los números de secuencia del SYN y del FIN que envía el cliente, y qué relación tienen con la cantidad de datos enviada por el cliente al servidor.

El cliente al servidor le envía 11 datos en total. Primero en el segmento 6 le envía 5

y luego en el segmento 8, termina enviándole otros 6 bytes más.

Los números de SYN que envía el cliente es 1367801698 y el FIN es 1367801710.

Con estos números podemos saber que entre ellos nos encontramos con una

diferencia de 12 bytes de datos y como el flag FIN ocupa 1, si hacemos la resta 12-

1 = 11 bytes que es la misma cantidad que vimos anteriormente.

**1.3. RTT**

Continúa analizando la captura tcp.cap.

1. Para cada uno de los segmentos de datos que envía el cliente al servidor, indica cuál es el RTT. Observa para ello los tiempos de envío de los segmentos y los de recepción de sus correspondientes asentimientos.

El segmento número 3 se envía en 0.000134 y se asiente 0.009918 al hacer la resta observamos que su RTT es 0.009784.

El segmento número 6 se envía en 1.962313 y se asiente en 1963330 que es el segmento 7 al realizar la resta observamos que su RTT es 0.0001017.

El segmento número 8 se envía en 3.251756 y se recibe su asentimiento en 3.252305 que es segmento 9, al realizar la resta observamos que su RTT es 0.0000549.

Por último, el segmento número 10 se envía en 4.170910 y se recibe su asentimiento en 4.172336 que es el segmento 11 y su RTT es 0.0001426.

**1.4. MSS.**

Abre ahora el archivo de captura tcp-mss-pmtu.cap. En dicha captura se muestra el principio de una comunicación TCP. Ordena en Wireshark los paquetes por la columna Time. Contesta a las siguientes preguntas:

1. Indica cuál es el valor anunciado de MSS en las cabeceras opcionales de los paquetes SYN de los dos sentidos de la conexión. Dados estos dos valores, indica qué tamaño de datos crees usarán ambos sentidos de la conexión si tienen que enviar datos.

El MSS del cliente es 660 como bien observamos en TCP-option-maximun segment size.

El MSS que hace el servidor al cliente (SYN + ACK) es Maximun segment size de

560 bytes.

Si ambos lados de la conexión usan esta opción, la consecuencia es que ambos

lados usarán segmentos con la parte de datos de tamaño igual al menor de los dos

MSS, es decir, el del servidor (560 bytes).

2. Mira los tamaños de las cabeceras de los distintos segmentos, medidos en palabras de 4 bytes y razona por qué no todos los segmentos tienen el mismo tamaño. ¿Crees que el tamaño de la cabecera puede influir en el tamaño máximo de datos que posteriormente utilizará TCP para enviar segmentos?

Si escogemos los segmentos que tienen SYN activado, veremos que además de

las que suelen tener los otros segmentos, estos llevan las cabeceras de MSS y la

de Window Scale. Es decir, tendrían 5 cabeceras en opciones, por lo cual

estaríamos hablando de 20 bytes, que tendríamos que sumarle a la cabecera que

ya lleva consigo TCP.

Mientras que los que no tienen el SYN activado, cuentan con 12 bytes de opciones,

es decir, sus 20 de cabecera normal en TCP más esos 12 que se añaden.

3. Teniendo en cuenta que en el instante en el que se envía el segmento número 4, la máquina 11.0.0.11 tenía más de 2.000 bytes de datos que enviar a la máquina 12.0.0.12, ¿por qué envía menos? ¿Cuántos bytes envía en ese segmento 4?¿Cómo se justifica el número de bytes de datos que contiene el segmento 4 dados los valores de MSS anunciados en los segmentos SYN?

Le envía menos datos de los que tiene para enviar porqué si nos fijamos en el MSS,

vemos que lo máximo que se permiten enviar es de 560 bytes y si deducimos el

porque solo le manda 548, es porque esos son los bytes sin contar los 12 bytes de

las opciones.

4. Explica qué tipo de paquete son los paquetes 5 y 7 de la captura. ¿Qué significan y cuál puede ser la razón de que se hayan enviado? ¿Qué campo de la cabecera IP de los paquetes 4 y 6 ha provocado que se envíen los paquetes 5 y 7?

Son protocolos ICMP de error DESTINATION UNREACHABLE (FRAGMENTATION NEEDED) nos quiere decir que ese paquete no ha podido llegar a su máquina destino y que

necesita ser fragmentado para poder hacerlo. Como en TCP se intenta que los segmentos no sean fragmentados, seguramente en la cabecera de IP si la miramos nos encontraríamos con que nos dice que tiene activado el flag DON’T FRAGMENT y al llegar a una máquina intermedio que ha leído eso (11.0.0.1), le ha enviado este mensaje a 11.0.0.11.

Como bien hemos dicho antes, si nos vamos a la cabecera de IP del segmento 4 y 6, vemos que está activado la opción del flag DON’T FRAGMENT.

5. Mira el paquete 8 de la captura. ¿Por qué se retransmite este segmento? Comprueba el tamaño de su campo de datos. Explica la relación de este valor con la información contenida en los paquetes 5 y 7.

El ICMP que se envía incluye el tamaño máximo de segmento que puede aceptar sin fragmentación (MSS). El origen al recibir un ICMP de este tipo, disminuye el tamaño de segmento y lo retransmite.

**1.5. Funcionamiento básico de la ventana anunciada**

En la captura tcp-window.cap se muestra el principio de una comunicación TCP. Para este apartado, utiliza en Wireshark los números de secuencia relativos al principio de la conexión. Ayudándote de la gráfica tcptrace, en la segunda sección Análisis de gráficas tcptrace de conexiones TCP, contesta a las siguientes preguntas relativas al sentido de comunicación 12.0.0.100 → 13.0.0.100:

1. ¿Cuál es el número de secuencia del primer byte de datos contenido en el paquete número 6?

Su número de seq es 1.

2. ¿Cuál es el número de secuencia del último byte de datos contenido en el paquete número 6?

Creemos que el número de secuencia es 548 porque el primero del 6 es 1 y el segmento 7 empieza en 549 por eso restamos uno y obtenemos el 548.

3. El paquete número 7, ¿cuántos bytes de datos asiente?

Asiente todos los bytes es decir los 548.

4. En el momento de enviar el paquete número 8, ¿cuál es el último valor de ventana anunciada por el receptor que ha recibido el emisor? ¿En qué número de paquete venía anunciado? ¿Cuántos bytes de datos puede enviar como máximo el emisor en ese momento, en uno o más segmentos, antes de volver a recibir otro ACK del receptor?

Es 2240 que es la que esta anunciando el segmento 7. Puede enviar 2240 bytes.

5. En el momento de enviar el paquete número 13, ¿cuál es el último valor de ventana anunciada por el receptor que ha recibido el emisor? ¿En qué número de paquete venía anunciado? ¿Cuántos bytes de datos puede enviar como máximo el emisor en ese momento, en uno o más segmentos, antes de volver a recibir otro ACK del receptor?

6. ¿Podría haber enviado el emisor un segmento con datos nuevos en el instante 0.321000 segundos? ¿Por qué?

No, porque el paquete 13 avisa que la ventana esta agotada.

7. Identifica en la gráfica tcptrace de este sentido de la comunicación en qué otro periodo de tiempo se produce una situación similar a la que ocurre al enviarse el paquete número 13.

En el paquete numero 18.

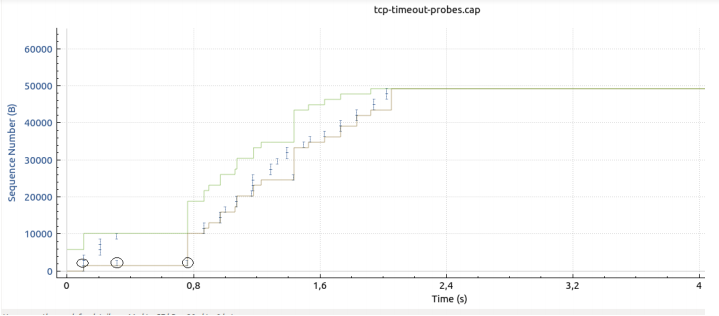
**1.6. Retransmisiones y asentimientos**

En la captura tcp-timeout-probes.cap se muestra el principio de una comunicación TCP. Para este apartado, utiliza en Wireshark los números de secuencia relativos al principio de la conexión. Ayudándote de la gráfica tcptrace, contesta a las siguientes preguntas relativas al sentido de comunicación 12.0.0.100 → 13.0.0.100:

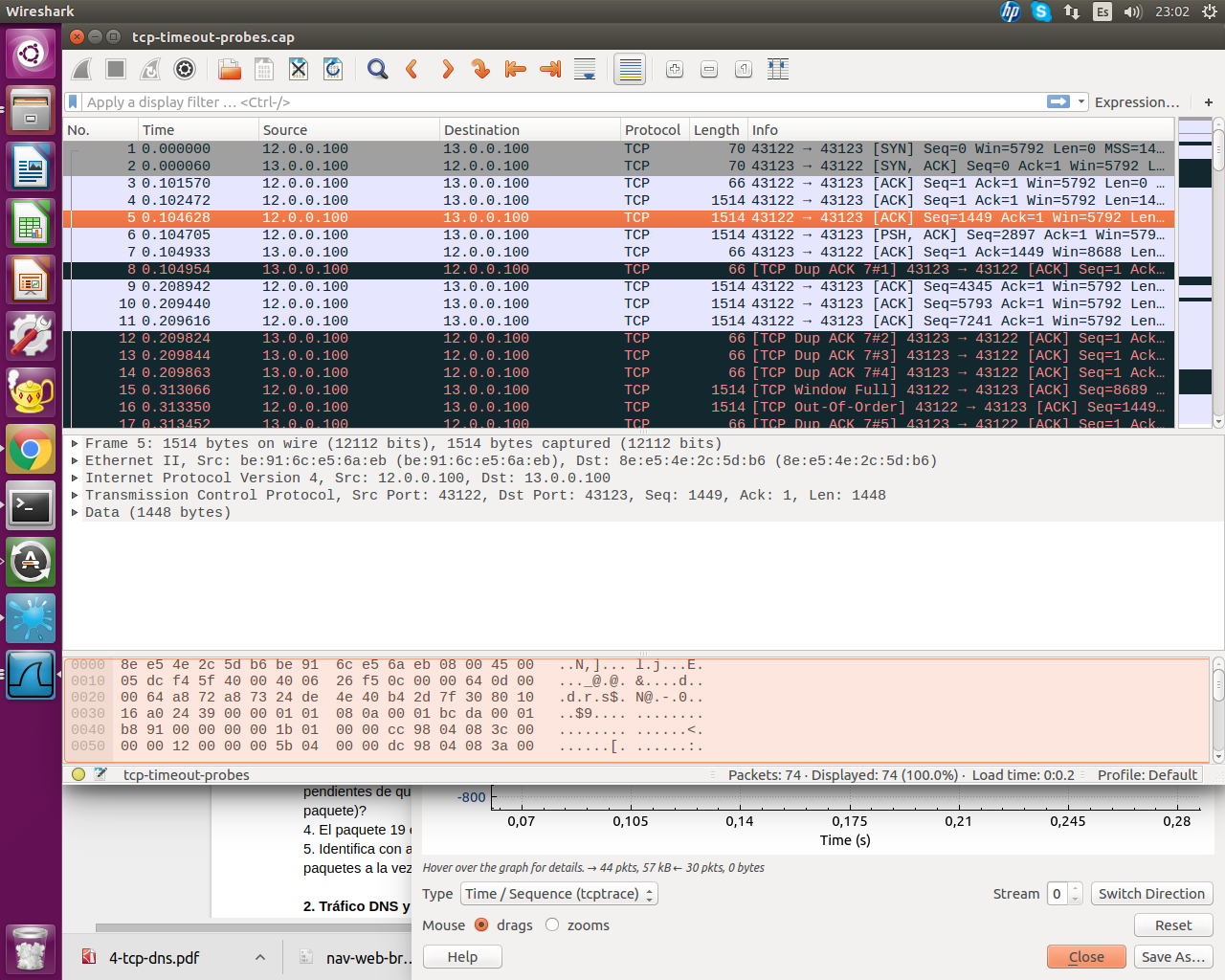
1. El paquete número 16 es una retransmisión, aunque Wireshark no lo etiqueta como tal: a) Mirando la gráfica tcptrace, ¿cómo puede identificarse que dicho paquete 16 es una retransmisión?

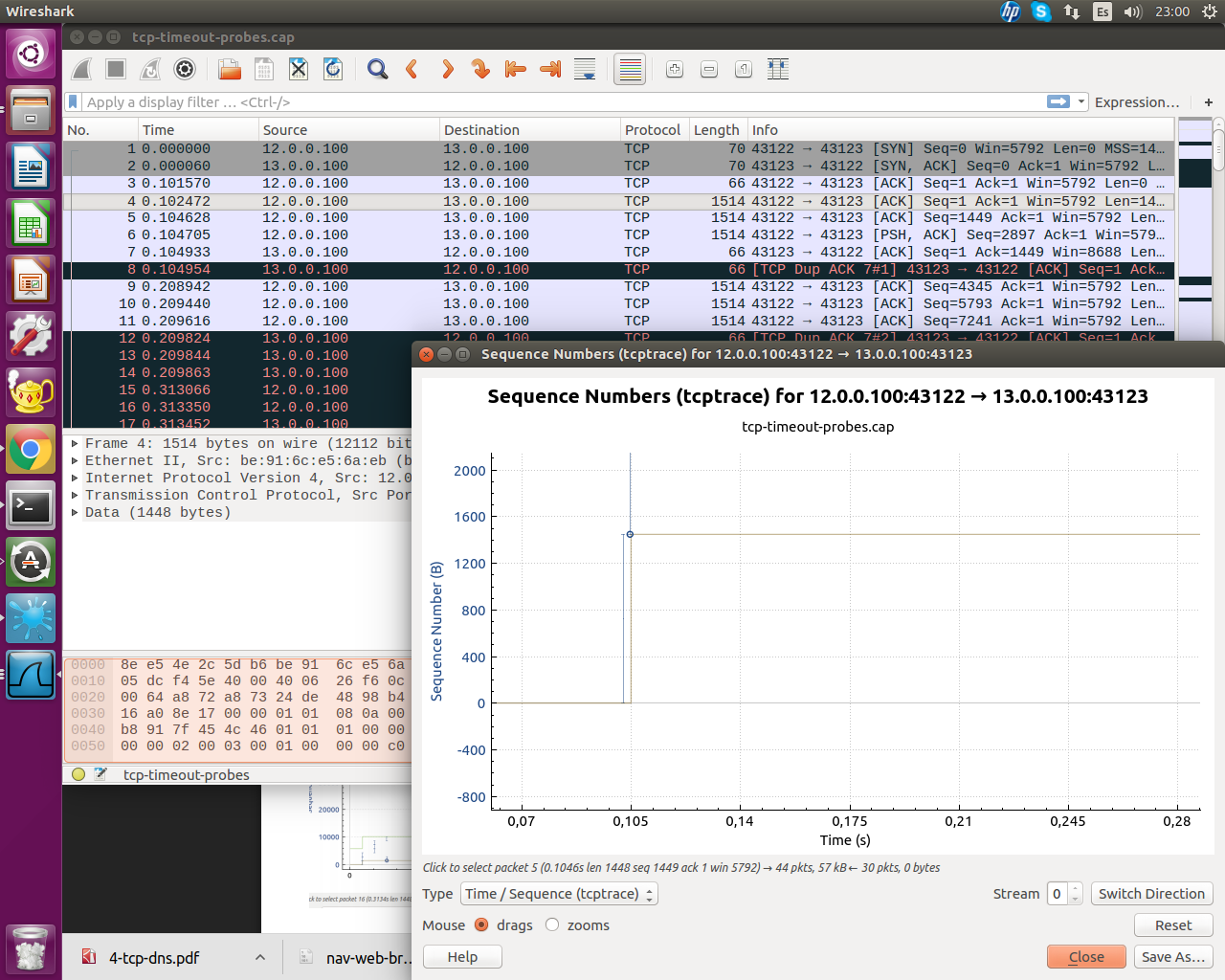
Wireshark interpreta que hay paquetes duplicados y los muestra como “TCP Out-

Of-Order“por ello sabemos que es una retransmisión. A parte analizando la gráfica nos damos cuenta de que el 5, 16 y 18 (los he redondeado en la tercera foto) tienen el mismo número de secuencia, el mismo ACK pero son retransmitidos en diferentes instantes de tiempo.

****

b) ¿Qué número de paquete es la primera transmisión de los bytes de datos que viajan en este paquete 16?





Es el 5 ya que su origen y destino son las mismas maquinas y tambien su numero de secuencia. Podemos observarlo en las imagenes que se adjuntan, tanto en la gráfica como en wireshark.

c) ¿Cuál ha sido el plazo de retransmisión aplicado a esa primera transmisión del paquete?

200 y pico milisegundos. (entre la 5 y la 16)

d) Mirando la gráfica tcptrace, ¿vuelve a ser retransmitido este paquete en la conexión? ¿Con qué plazo de retransmisión? ¿Por qué?

Vuelve a ser retransmitido en el paquete 18 esperando ahora 400 y pico milisegundos. Porque el algoritmo exponential backoff entra en acción doblando el tiempo de espera para esta segunda retransmisión

2. Identifica en la gráfica tcptrace otros segmentos que son retransmisión.

El paquete 35 también se retransmite con el paquete 45

3. En el momento de transmitir el paquete 18, ¿cuántos bytes están transmitidos y pendientes de que llegue su asentimiento? ¿Y cuántos paquetes (indica su número de paquete)?

Bytes in flight 8688(seq ack)

4. El paquete 19 es un asentimiento. ¿Cuántos bytes asiente? ¿Y cuántos segmentos?

Asiente todos los bytes desde el 1449(paquete 5)

5. Identifica con ayuda de la gráfica tcptrace otros asentimientos que asienten varios paquetes a la vez.

**2. Tráfico DNS y HTTP**

Tal y como hemos visto en teoría, el servicio DNS ofrece mecanismos para poder descubrir la dirección IP que corresponde a una máquina con la que queremos establecer un diálogo a nivel de transporte (con TCP o UDP). Uno de los ejemplos más típicos de uso de este servicio es al utilizar un navegador web. El archivo de captura nav-web-browser.pcap contiene un extracto de una captura original de tráfico de red con Wireshark. Abre ese archivo con Wireshark y contesta a las siguientes preguntas:

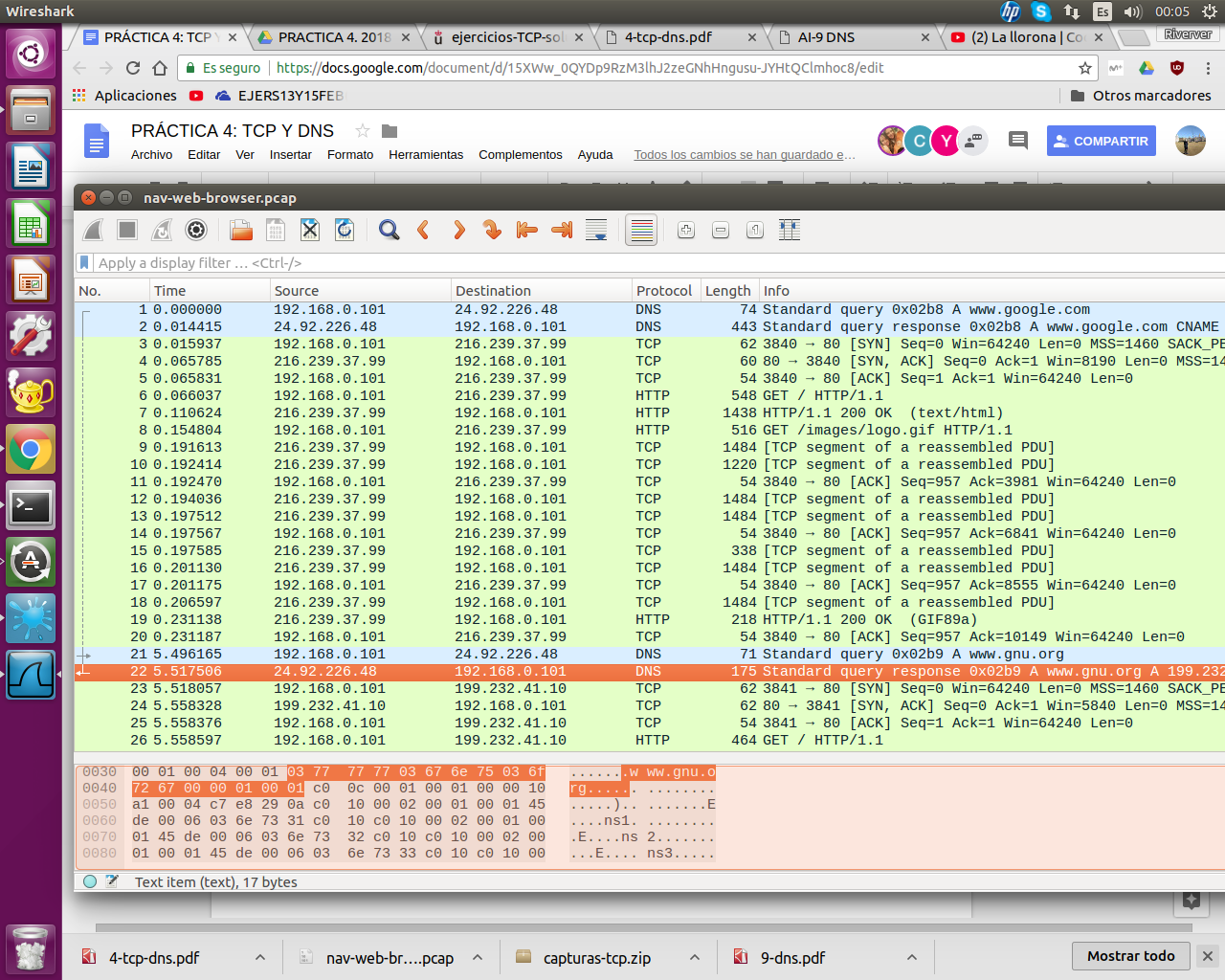
1. ¿Cuántas solicitudes de resolución de la IP correspondiente a un FQDN se realizan en esta captura? Identifica, concretamente, cuáles son los FQDN cuya dirección IP se solicita, así como la IP que el servicio DNS informa que corresponde a esos nombres de dominio.

¿Qué dirección IP tiene el servidor DNS que contesta?

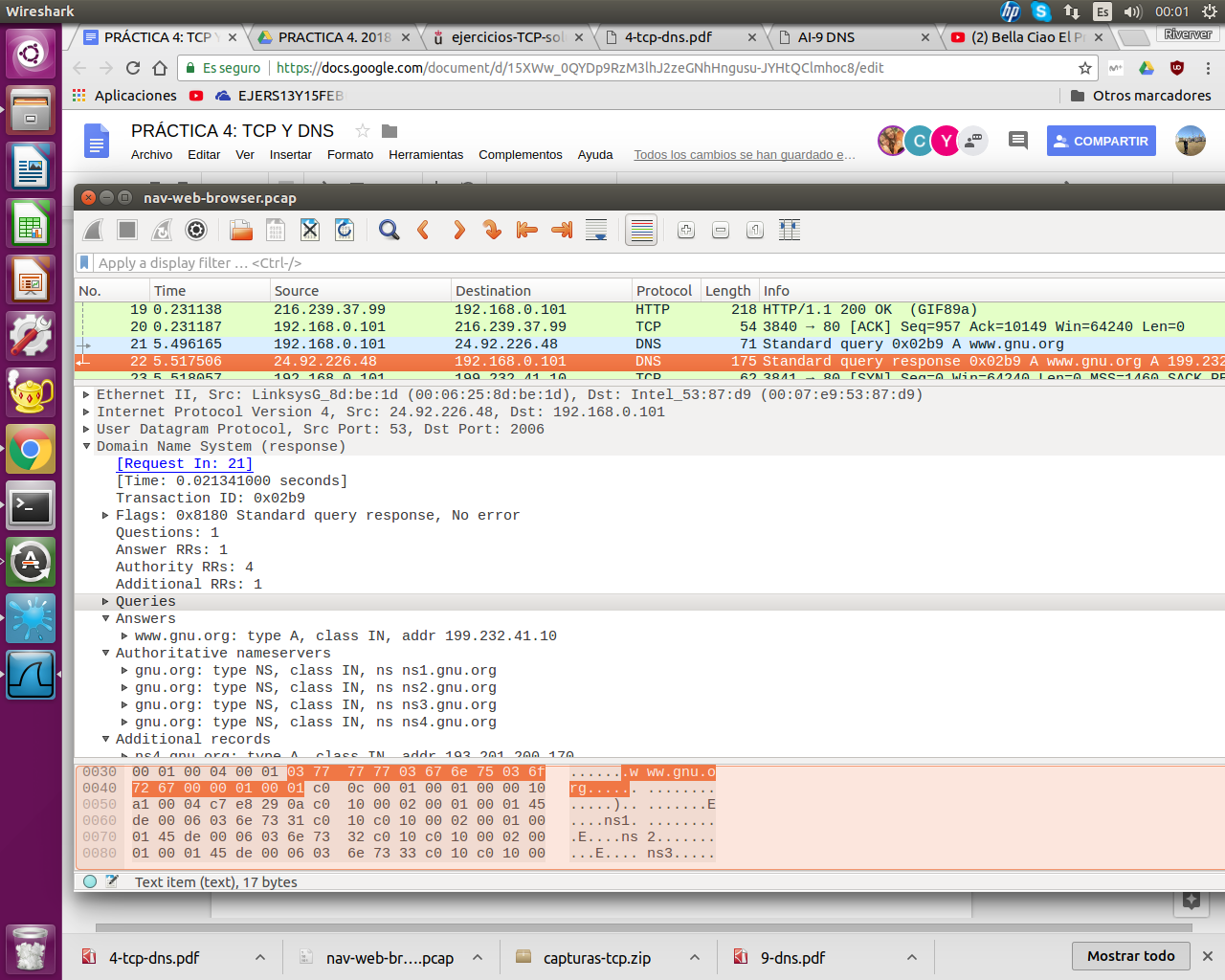
Son dos solicitudes de:

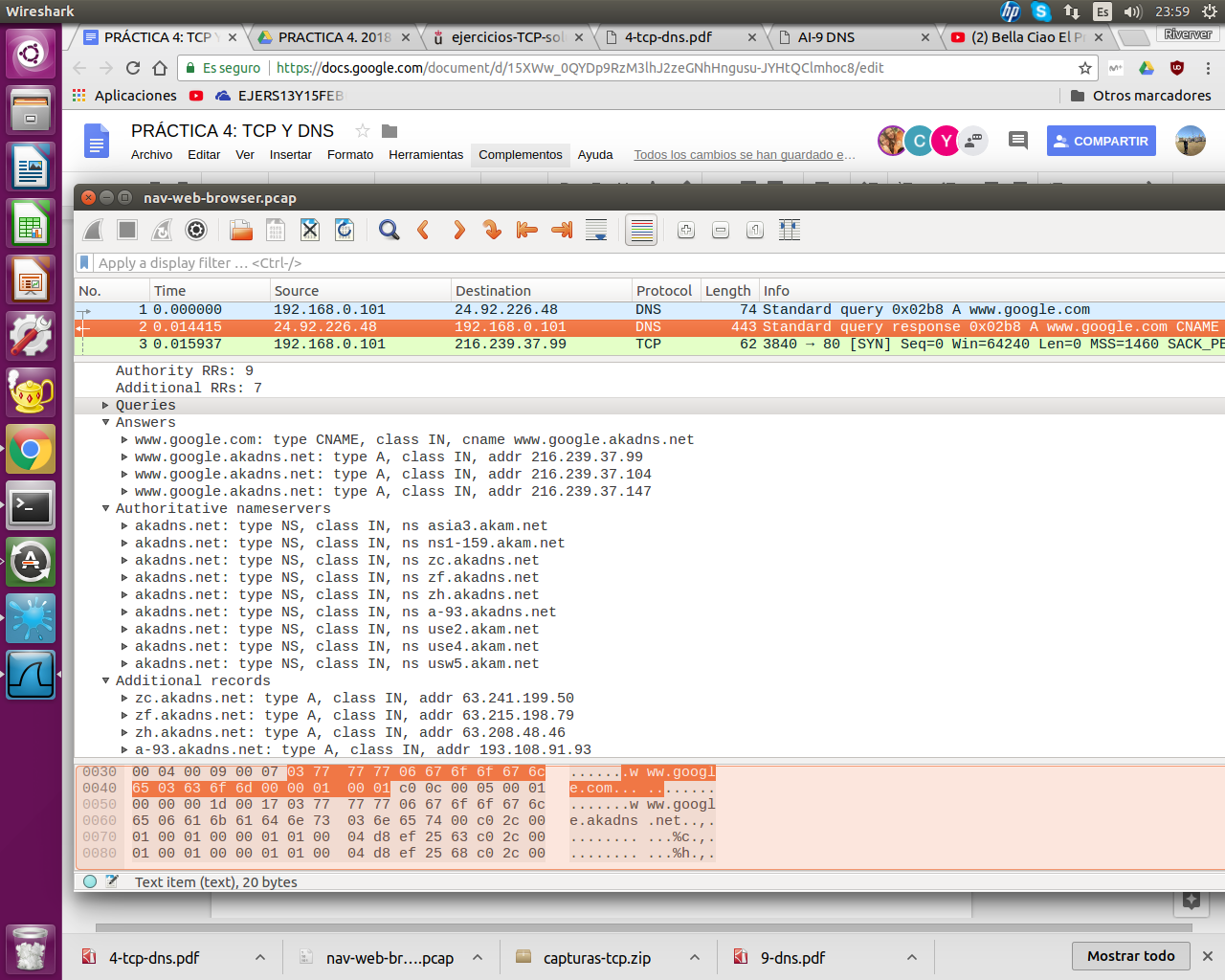
[www.google.com](http://www.google.com)

[www.gnu.org](http://www.gnu.org)

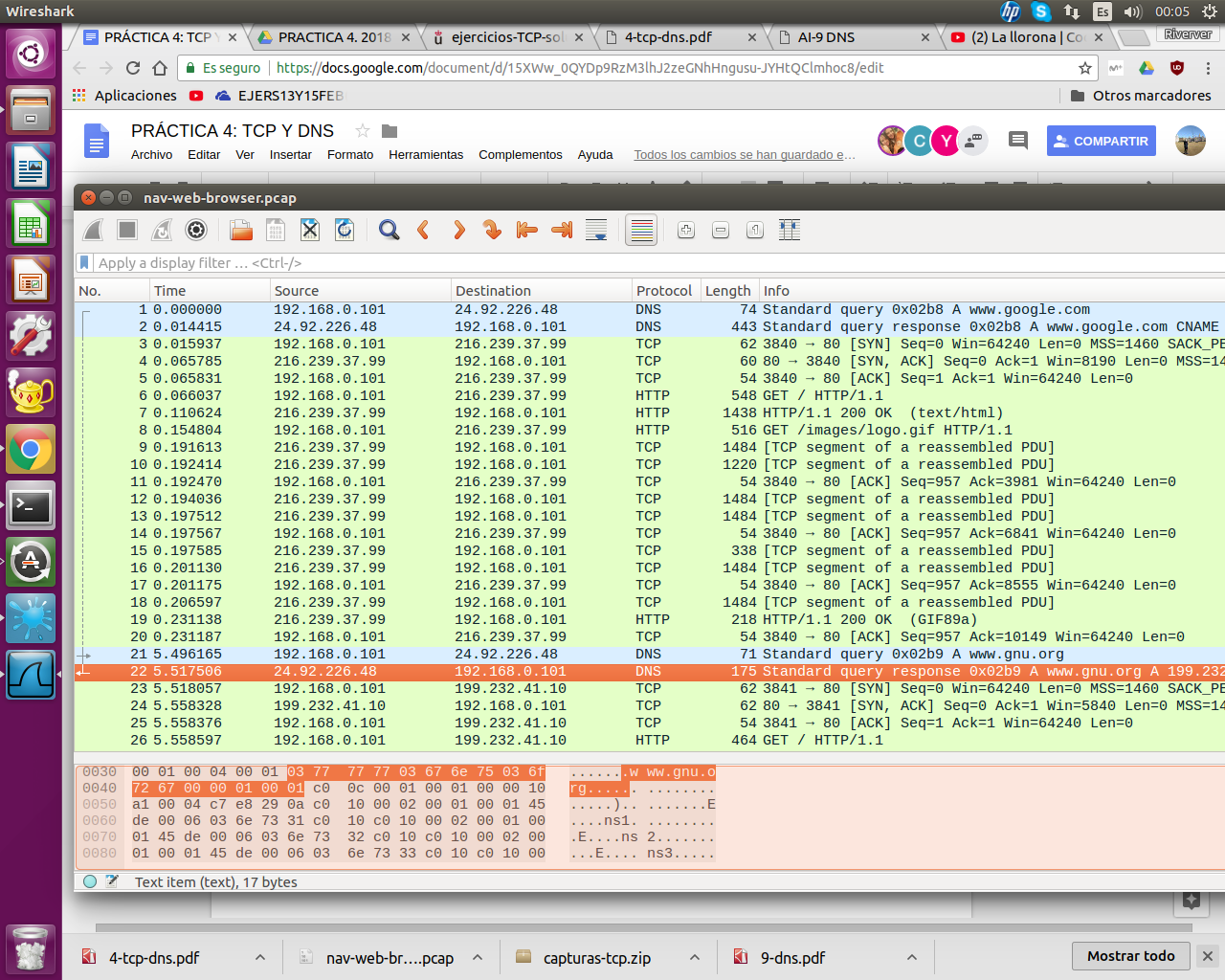


Las direcciones ip de google y gnu se muestran en las siguientes imágenes:





Y la dirección ip del servidor DNS que contesta es la que se muestra en la siguiente imagen:



2. Dibuja un pequeño diagrama en el que se indique cuantos diálogos TCP puedes identificar en esta captura. Para cada diálogo TCP debes indicar:

- La IP de la máquina cliente y la IP del servidor, así como el puerto de comunicación empleado en ambos extremos.

-Los números de secuencia anunciados en la apertura de cada conexión por ambos extremos.

-¿Cuántos de los diálogos TCP abiertos analizan correctamente? Indica, concretamente, el número de trama capturada en Wireshark que corresponde a los segmentos TCP intercambiados para cerrar cada conexión.

VER FOTO DIALOGO