Nathalie Alfaro Quesada, B90221.

Exercise One

Open “Wireshark”, then use the “File” menu and the “Open” command to open the file “WS-Ex01.pcap”. You should see 26 packets listed.

This set of packets describes a ‘conversation’ between a user’s client and a central server. This entire conversation happens automatically, after a user types something and hits enter. Look at the packets to answer the following questions in relation to this conversation.

In answering the following questions, use brief descriptions. For example, “In frame X, the client requests a web page, and in frame Y, the server delivers the content of the page.”

a) What is the IP address of the client that initiates the conversation?

La IP es 131.247.95.216 como se muestra al inicio.

b) Use the first two packets to identify the server that is going to be contacted. List the common name, and three IP addresses that can be used for the server.

El servidor que contacta es Google y las 3 IP que usan son:

- 64.233.161.99
- 64.233.161.104
- 64.233.161.147

c) What is happening in frames 3, 4, and 5?

Frame 3: El cliente 131.247.95.216 envía un paquete SYN al servidor 64.233.161.99 en el puerto 80, esto inicia la conexión TCP.

Frame 4: El servidor reconoce la solicitud del cliente y responde con un paquete SYN-ACK.

Frame 5: El cliente envía un paquete ACK y así finaliza las 3 vías.

Esto hace una conexión TCP para la comunicación HTTP entre cliente-servidor.

d) What is happening in frames 6 and 7?

Frame 6: El cliente envía un GET en HTTP al servidor.

Frame 7: El servidor responde con un TCP ACK y reconoce el GET del cliente.

e) Ignore frame eight. However, for your information, frame eight is used to manage flow control.

f) What is happening in frames nine and ten? How are these two frames related?

Frame 9: El servidor reconoce que recibió datos del cliente con un ACK TCP. Además asegura una conexión fiable.

Frame 10: El servidor envía una respuesta HTTP 200 OK al cliente, lo que significa el inicio de la entrega. Hace la transferencia del contenido de la respuesta HTTP.

g) What happens in packet 11?

Muestra al cliente solicitando otro recurso al servidor, que es parte de continuar la conversación y solicitar elementos adicionales necesarios para la página web.

h) After the initial set of packets is received, the client sends out a new request in packet 12. This occurs automatically without any action by the user. Why does this occur? See the first “hint” to the left.

Pasa automáticamente porque los navegadores web modernos solicitan múltiples recursos (como imágenes, CSS y scripts) para cargar completamente una página web.

i) What is occurring in packets 13 through 22?

Paquetes 13-19: Tienen segmentos TCP que transfieren el archivo de imagen "logo.gif" del servidor al cliente, con el cliente enviando paquetes ACK para confirmar la recepción.

Paquetes 20-21: Intercambios adicionales de ACK entre el cliente y el servidor para mantener el flujo de datos.

Paquete 22: El cliente completa la transferencia, recibiendo el archivo de imagen completamente reensamblado.

j) Explain what happens in packets 23 through 26. See the second “hint” to the left.

Paquete 23: El cliente envía una solicitud para el archivo “favicon.ico” utilizando una solicitud HTTP GET.

Paquete 24: El servidor reconoce la solicitud con un TCP ACK.

Paquete 25: El servidor responde con un HTTP 200 OK, comenzando la transferencia del “favicon.ico”.

Paquete 26: La transferencia se completa y el cliente reconoce la recepción de los datos del favicon.

k) In one sentence describe what the user was doing (Reading email? Accessing a web page? FTP? Other?).

El usuario estaba accediendo a una página web, de todas las solicitudes HTTP GET para recursos como “logo.gif” y “favicon.ico”, seguidas de las respuestas HTTP 200 OK del servidor.

Exercise Two

Open “Wireshark”, then use the “File” menu and the “Open” command to open the file “WS-Ex02.pcap”. You should see 176 packets listed.

a) In the first few packets, the client machine is looking up the common name (cname) of a web site to find its IP address. What is the cname of this web site? Give two IP addresses for this web site.

El usuario está buscando el sitio web llamado yahoo.com, algunas direcciones IP para el sitio web son 216.109.117.106 o 216.109.117.109.

b) How many packets/frames does it take to receive the web page (the answer to the first http get request only)?

Hay un total de 15 tramas entre la solicitud HTTP GET y la respuesta HTTP 200 OK.

c) Does this web site use gzip to compress its data for sending? Does it write cookies? In order to answer these questions, look under the payload for the reassembled packet that represents the web page. This will be the last packet from question b above. Look to see if it has “Content-Encoding” set to gzip, and to see if it has a “Set-Cookie” to write a cookie.

Sí, está utilizando gzip para la codificación de contenido y escribe cookies.

d) What is happening in packets 26 and 27? Does every component of a web page have to come from the same server? See the Hint to the left.

Frame 26 realiza una consulta estándar mientras que el 27 es la respuesta a dicha solicitud. No todos los componentes de una página web tienen que provenir del mismo servidor, una página web puede incluir recursos de múltiples servidores.

e) In packet 37 we see another DNS query, this time for us.i1.yimg.com. Why does the client need to ask for this IP address? Didn’t we just get this address in packet 26? (This is a trick question; carefully compare the two common names in packet 26 and 37.)

El nombre común en ambos tiene dígitos diferentes.
Frame 26 = a321.yimg[...]
Frame 37 = a943.yimg[...].
Está tratando de resolver un subdominio diferente del que se resolvió en el paquete 26.

f) In packet 42 we see a HTTP “Get” statement, and in packet 48 a new HTTP “Get” statement. Why didn’t the system need another DNS request before the second get statement? Click on packet 42 and look in the middle window. Expand the line titled “Hypertext Transfer Protocol” and read the “Host:” line. Compare that line to the “Host:” line for packet 48.

El sistema no necesitaba otra solicitud DNS antes de la segunda declaración HTTP GET en el paquete 48 porque se está accediendo al mismo host en ambos paquetes. La línea Host en el paquete 42 y la comparas con la línea Host en el paquete 48, si ambas líneas especifican el mismo dominio, indica que el cliente aún se está comunicando con el mismo servidor.

g) Examine packet 139. It is one segment of a PDU that is reassembled with several other segments in packet 160. Look at packets 141, 142, and 143. Are these three packets also part of packet 160? What happens if a set of packets that are supposed to be reassembled do not arrive in a continuous stream or do not arrive in the proper order?

Almacenamiento en buffer: La pila TCP receptora almacena en buffer paquetes fuera de orden hasta que lleguen los que faltan.

Reordenación: TCP utiliza números de secuencia para reordenar los paquetes correctamente antes de entregarlos a la capa de aplicación.

Solicitudes de retransmisión: Si se pierden paquetes o no se reciben dentro de un tiempo de espera, TCP solicitará la retransmisión de los paquetes faltantes.

Entrega de datos: Los datos completos solo se entregan a la aplicación una vez que se han recibido y ordenado correctamente todos los segmentos.

h) Return to examine frames 141 and 142. Both of these are graphics (GIF files) from the same source IP address. How does the client know which graphic to match up to each get statement? Hint: Click on each and look in the middle window for the heading line that starts with “Transmission Control Protocol”. What difference do you see in the heading lines for the two files? Return to the original “Get” statements. Can you see the same difference in the “Get” statements?

Números de Secuencia TCP: Cada segmento TCP tiene un número de secuencia único que indica el orden en el que se deben ensamblar los paquetes. Los números de secuencia para tramas 141 y 142 serán diferentes, reflejando el orden en el que los paquetes fueron enviados y recibidos.

Declaraciones GET: Cuando se vuelve a las declaraciones originales de "GET" en las solicitudes HTTP, es probable que las solicitudes para cada gráfico también tengan identificadores o parámetros distintos (como diferentes rutas o nombres de archivo). Esto ayuda al servidor y al cliente a rastrear qué recurso específico se está solicitando.

Exercise Three

Open “Wireshark”, then use the “File” menu and the “Open” command to open the file “WS-Ex03.pcap”. You should see 22 packets listed.

These packets represent two different requests for web pages. Packets 1-7 involve the request for the web page www.yahoo.com. Packets 8-22 involve the request for the web page my.usf.edu.

a) Compare the destination port in the TCP packet in frame 3 with the destination port in the TCP packet in frame 12. What difference do you see? What does this tell you about the difference in the two requests?

Frame 3: Tiene un puerto de destino de 80, indica una solicitud HTTP.

Frame 12: Tiene un puerto de destino 443, indica una solicitud HTTPS.

The following table compares the two requests for web pages. For example, row i) shows that frames 1-2 and frames 8-9 represent the DNS lookups for each of the web requests.

Row	www.yahoo.com frames	my.usf.com frames	Brief Explanation of Activity
i)	1-2	8-9	DNS Request to find IP address for common name & DNS Response
ii)	3-5	10-12	Three-way handshake
iii)	--	13-20	
iv)	6	21	“Get” request for web page
v)	7	22	First packet from web server with web page content.

b) Explain what is happening in row “iii” above. Why are there no frames listed for yahoo in row “iii"?

Fue un período de inactividad o que el cliente no envió ninguna solicitud ni recibió paquetes del servidor durante este tiempo.

c) Look at the “Info” column on frame 6. It says: “GET / HTTP / 1.1. What is the corresponding Info field for the my.usf.com web request (frame 21)? Why doesn’t it read the same as in frame 6?

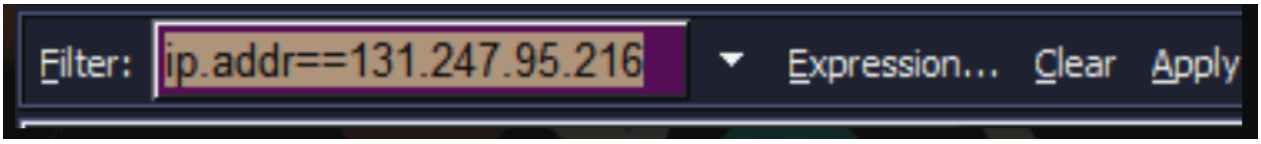
"Info" difiere porque el frame 21 solicita un recurso específico de my.usf.com, mientras que el frame 6 solicita el recurso raíz de un servidor diferente.

Exercise Four

In this exercise, you are going to capture live traffic from your computer. Open up Wireshark and use the “Capture” menu to save live traffic. The Wireshark “QuickStart” guide distributed with these exercises contains more instructions on using Wireshark.

Start capturing data, visit a live web site (os.ecci.ucr.ac.cr/ci0121) using your standard Internet browser, and stop capturing data.

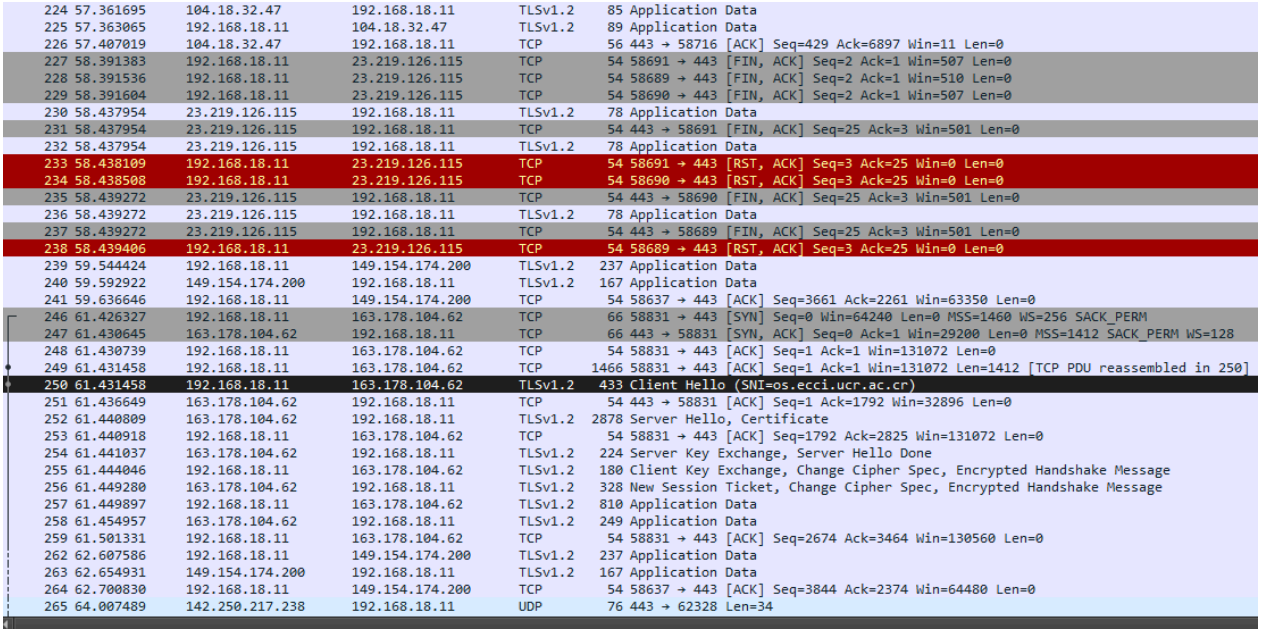
If you have a large amount of network traffic, the relevant data may be hidden among a lot of broadcast messages. To focus on just the key frames, you can set a display filter like this.



For the IP number enter the IP number of your client machine. Type it as shown (ip.addr==your.ip.address) in the graphic above. Then click on “Apply”.

Using an approach similar to the approach in Exercise One, describe the set of frames that you captured.

- For this description think of this as a conversation – every discussion starts with a question and follows with an answer.
- For example, two of the frames will contain the DNS request for an IP address for the web site, and the DNS answer with the IP number.
- Remember that some answers may take several frames if they need to be reassembled from segmented packets.



224	57.361695	104.18.32.47	192.168.18.11	TLSv1.2	85 Application Data
225	57.363065	192.168.18.11	104.18.32.47	TLSv1.2	89 Application Data
226	57.407019	104.18.32.47	192.168.18.11	TCP	56 443 → 58716 [ACK] Seq=429 Ack=6897 Win=11 Len=0
227	58.391383	192.168.18.11	23.219.126.115	TCP	54 58691 → 443 [FIN, ACK] Seq=2 Ack=1 Win=507 Len=0
228	58.391536	192.168.18.11	23.219.126.115	TCP	54 58689 → 443 [FIN, ACK] Seq=2 Ack=1 Win=510 Len=0
229	58.391604	192.168.18.11	23.219.126.115	TCP	54 58690 → 443 [FIN, ACK] Seq=2 Ack=1 Win=507 Len=0
230	58.437954	23.219.126.115	192.168.18.11	TLSv1.2	78 Application Data
231	58.437954	23.219.126.115	192.168.18.11	TCP	54 443 → 58691 [FIN, ACK] Seq=25 Ack=3 Win=501 Len=0
232	58.437954	23.219.126.115	192.168.18.11	TLSv1.2	78 Application Data
233	58.438109	192.168.18.11	23.219.126.115	TCP	54 58691 → 443 [RST, ACK] Seq=3 Ack=25 Win=0 Len=0
234	58.438508	192.168.18.11	23.219.126.115	TCP	54 58690 → 443 [RST, ACK] Seq=3 Ack=25 Win=0 Len=0
235	58.439272	23.219.126.115	192.168.18.11	TCP	54 443 → 58690 [FIN, ACK] Seq=25 Ack=3 Win=501 Len=0
236	58.439272	23.219.126.115	192.168.18.11	TLSv1.2	78 Application Data
237	58.439272	23.219.126.115	192.168.18.11	TCP	54 443 → 58689 [FIN, ACK] Seq=25 Ack=3 Win=501 Len=0
238	58.439406	192.168.18.11	23.219.126.115	TCP	54 58689 → 443 [RST, ACK] Seq=3 Ack=25 Win=0 Len=0
239	59.544424	192.168.18.11	149.154.174.200	TLSv1.2	237 Application Data
240	59.592922	149.154.174.200	192.168.18.11	TLSv1.2	167 Application Data
241	59.636646	192.168.18.11	149.154.174.200	TCP	54 58637 → 443 [ACK] Seq=3661 Ack=2261 Win=63350 Len=0
246	61.426327	192.168.18.11	163.178.104.62	TCP	66 58831 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
247	61.430645	163.178.104.62	192.168.18.11	TCP	66 443 → 58831 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1412 SACK_PERM WS=128
248	61.430739	192.168.18.11	163.178.104.62	TCP	54 58831 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
249	61.431458	192.168.18.11	163.178.104.62	TCP	1466 58831 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=1412 [TCP PDU reassembled in 250]
250	61.431458	192.168.18.11	163.178.104.62	TLSv1.2	433 Client Hello (SNI=os.ecci.ucr.ac.cr)
251	61.436649	163.178.104.62	192.168.18.11	TCP	54 443 → 58831 [ACK] Seq=1 Ack=1792 Win=32896 Len=0
252	61.440809	163.178.104.62	192.168.18.11	TLSv1.2	2878 Server Hello, Certificate
253	61.440918	192.168.18.11	163.178.104.62	TCP	54 58831 → 443 [ACK] Seq=1792 Ack=2825 Win=131072 Len=0
254	61.441037	163.178.104.62	192.168.18.11	TLSv1.2	224 Server Key Exchange, Server Hello Done
255	61.444046	192.168.18.11	163.178.104.62	TLSv1.2	180 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
256	61.449280	163.178.104.62	192.168.18.11	TLSv1.2	328 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
257	61.449897	192.168.18.11	163.178.104.62	TLSv1.2	810 Application Data
258	61.454957	163.178.104.62	192.168.18.11	TLSv1.2	249 Application Data
259	61.501331	192.168.18.11	163.178.104.62	TCP	54 58831 → 443 [ACK] Seq=2674 Ack=3464 Win=130560 Len=0
262	62.607586	192.168.18.11	149.154.174.200	TLSv1.2	237 Application Data
263	62.654931	149.154.174.200	192.168.18.11	TLSv1.2	167 Application Data
264	62.700830	192.168.18.11	149.154.174.200	TCP	54 58637 → 443 [ACK] Seq=3844 Ack=2374 Win=64480 Len=0
265	64.007489	142.250.217.238	192.168.18.11	UDP	76 443 → 62328 Len=34

Visité el sitio web os.ecci.ucr.ac.cr/ci0121 con mi laptop de IP 192.168.18.11, la cual se conecta con el servidor 163.178.104.62 en el puerto 443, el puerto estándar de HTTPS.

Primero se hizo una conexión TCP con handshake de SYN, SYN-ACK y ACK.

Luego se da el protocolo TLS 1.2.

Después el cliente, mi laptop, envía un Client Hello y el servidor responde Server Hello y el certificado digital.