Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679

DOCUMENTACIÓN PRÁCTICA 1

- **Analizad un protocolo de los fotogramas capturados. Cuáles son, y cuáles son los niveles del modelo OSI que le podemos asociar?**

```
7 0.007664      142.250.200.110      10.80.136.163      TCP      54 443 → 51215 [ACK] Seq=1 Ack=321 Win=1269 Len=0
```

Este es el frame que hemos escogido para estudiar y analizar los diferentes protocolos que aparecen. Como vemos en la imagen el protocolo de este frame es TCP y pertenece al nivel de transporte, es decir, a la capa 4 del modelo OSI.

La TCP (Protocolo de Control de Transmisión) es una conexión segura, ordenada y que chequea si ocurren errores a la hora de enviar los paquetes a través de aplicaciones ejecutadas en hosts comunicados vía IP posterior a haber realizado una conexión de tipo 3-way handshaking. Por lo tanto, el objetivo del protocolo TCP es garantizar la entrega segura y sin errores de los paquetes, este protocolo tiene control de concurrencia. Como hemos podido observar en wireshark el esquema de un paquete TCP es:

<table>
<tr><td colspan="8">Source port</td><td colspan="4">Destination port</td></tr>
<tr><td colspan="12">Sequence number</td></tr>
<tr><td colspan="12">Acknowledgment number (if ACK set)</td></tr>
<tr><td colspan="2">Data offset</td><td colspan="2">Reserved
0 0 0</td><td>N
S</td><td>C
W
R</td><td>E
C
E</td><td>U
R
G</td><td>A
C
K</td><td>P
S
H</td><td>R
S
T</td><td>S
Y
N</td><td>F
I
N</td><td colspan="4">Window Size</td></tr>
<tr><td colspan="8">Checksum</td><td colspan="4">Urgent pointer (if URG set)</td></tr>
<tr><td colspan="12">Options (if data offset > 5. Padded at the end with "0" bytes if necessary.)
…</td></tr>
</table>

En el paquete seleccionado su información propia, es la que se muestra en la siguiente imagen:

```
Source Port: 443
Destination Port: 51215
[Stream index: 0]
[Conversation completeness: Incomplete (12)]
[TCP Segment Len: 0]
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 167580934
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 321    (relative ack number)
Acknowledgment number (raw): 2860250107
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
Window: 1269
[Calculated window size: 1269]
[Window size scaling factor: -1 (unknown)]
Checksum: 0x393e [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
```

Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679

- **Ahora centraros en el nivel más alto de los protocolos de los fotogramas capturados. ¿Cuáles son, cuáles son sus misiones?**

  El nivel de protocolo más alto como se puede ver en el archivo adjunto es TLSv1.2, seguidos del TCP y el ARP.
  La misión del TLSv1.2 es un protocolo que permite la conexión segura a través de encriptar la información que se envía a través de internet. Se parece mucho al protocolo SSL, pero con agregar HTTPS al navegador. Este protocolo como es un protocolo criptográfico pertenece a la capa 6 del modelo OSI.

- **¿Qué sucede cuando intentas abrir la URL, habéis llegado directamente a los recursos?**

  En el caso de la página web de la UPF a la hora de realizar la búsqueda en web hemos observado que se redirigía a una búsqueda de tipo https habiendo introducido http://. Es decir, se redirige a una página segura.
  En el otro caso, es decir, el de google, esta redirección no ocurre por lo cual los resultados obtenidos serán directamente de http.

- **¿Cuántas conversaciones TCP habéis obtenido? Escoge un http y clica en "Follow TCP Stream". ¿Qué habéis conseguido ahora?**

  A partir de poner el filtro de display se observan dos para el caso UPF y tres para el caso de Google. Escogiendo un http y clicando en "Follow TCP Stream", observamos también dos conversaciones en UPF y tres en Google, tal y como se muestra en las siguientes imágenes.

  - UPF:

Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679

- Google:

- **Analizad el primer fotograma de cada sesión http. ¿Cuál es la dirección IP de destino? ¿Y cuál es la dirección MAC de los fotogramas? ¿Cuál es vuestra interpretación de lo que veis?**

  En el caso de la UPF, la IP destino es: 104.16.111.107 y la MAC de origen es: 4c:eb:bd:a3:6e:c3 y la de destino es: 00:09:0f:09:0a:8b.

  En el caso de Google, la IP destino es: 142.250.201.68 y la MAC de origen son las mismas por el hecho de que las máquinas encargadas de la conexión son la misma.

- **Describe los protocolos que encontráis en la captura. ¿Cuál es el propósito de cada fase de ellas?**

  Primeramente observamos que al hacer Discover de DHCP nos damos cuenta que hay un DHCP por lo que le solicitamos una IP y nos la da mediante el proceso Offer-Request-ACK. El protocolo ARP dada la IP 192.168.1.1 nos informa de la MAC que corresponde a esa IP. Posteriormente, buscamos una página web por nombre y el protocolo de DNS nos da su IP para poder navegar a esa página. Hecho esto, establecemos conexión desde 192.168.1.89 a 104.16.112.107, esto lo podemos ver por las flags del frame 9 y 10 las cuales son SYN y SYN ACK. Nos conectamos a la página en cuestión mediante los protocolos TCP y HTTP. Luego, encontramos muchos protocolos relacionados con la encriptación y paquetes TCP con la flag de ACK los cuales indican que los paquetes están llegando correctamente y en el orden adecuado. Finalmente vemos cómo finaliza el proceso de TCP devolviendo un ACK con la flag de FIN.

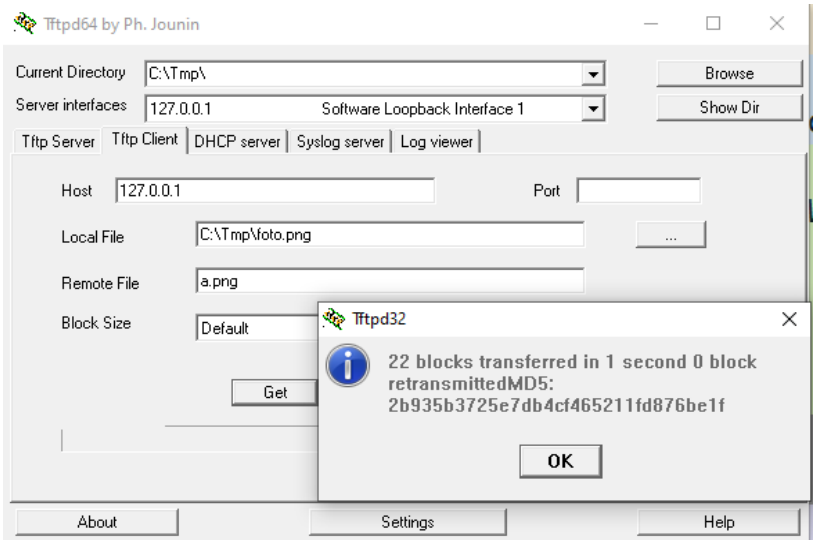
Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679

- **¿Cuál es la dirección “127.0.0.1”, puedes averiguarlo?**

  Esta dirección IP corresponde a la dirección estándar para el IPv4 loopback traffic. Esta dirección se refiere también al localhost de un equipo, es decir que estamos en la red local y esta dirección IP hace referencia al equipo en cuestión.

- **Desde el “Tftp Client”, configure "Host" en 127.0.0.1, "Remote File" en el nombre de su archivo, "Local File" en un nombre diferente y haga clic en "Get". El archivo debe transferirse y almacenarse también en C:\Tmp.**

  Realizando este proceso hemos obtenido los resultados esperados como se muestra en la siguiente imagen.



- **Elimine el archivo transferido antes de solicitar una nueva transmisión, ahora capturando el tráfico con Wireshark seleccionando el "Adaptador para captura de tráfico de loopback" de la lista de interfaces. Transfiere el archivo dos veces. Utiliza un filtro de captura para eliminar el tráfico que no corresponda a tu transferencia Tftp.**

  Para poner el filtro de captura nos hemos basado en que cualquier transferencia mediante TFTP utiliza como protocolo de transporte el protocolo UDP y en Wireshark no hay ningún filtro de captura que solo capture los paquetes TFTP, pero si existe en el caso de los UDP, con lo cual mirando los paquetes UDP podemos ver todos los TFTP.

Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679

- **Volved a cargar el archivo de captura en Wireshark y descubrid qué protocolos en las capas de enlace, red, transporte y aplicación veis:**

```
Frame 85: 548 bytes on wire (4384 bits), 548 bytes captured (4384 bits) on interface \Device\NPF_Loopback, id 0
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 50938, Dst Port: 50937
Trivial File Transfer Protocol
Data (512 bytes)
```

  Cargando la captura en Wireshark que hemos tenido que realizar en el apartado anterior, donde hemos establecido un filtro para solo ver los paquetes de nuestra transferencia TFTP. Podemos observar que cada paquete usa los protocolos Null/Loopback, IPv4, UDP y TFTP, donde Null/Loopback son dos protocolos de la capa más baja, es decir de la capa de enlace, IPv4 es un protocolo de red, UDP de transporte y TFTP de aplicación.

  - **¿Qué puerto se usó en el lado del servidor y del cliente en cada transferencia? ¿Puede decir qué puertos se utilizarían en una próxima transferencia?**

    Para la primera transferencia el puerto de origen es el 50935 y el de destino el 50936. En cambio, para la segunda transferencia el puerto de origen es el 50937 y el de destino el 50938. Como era de esperar, en ambos casos en los paquetes de ACK los puertos se invierten. Para hacer el “request” de la conexión se usa el puerto 69. En una próxima transferencia se usaría el siguiente par de puertos, es decir, el 50939 y el 50940 respectivamente.

  - **¿Eres capaz de explicar desde la captura cómo funciona este protocolo de aplicación?**

    Primeramente se realiza una read request, después se envía un ack conforme se acepta la petición y se puede empezar a hacer la transferencia de datos, esta se hace mediante paquetes de 548 bytes hasta llegar al último, que en nuestro caso hay menos.

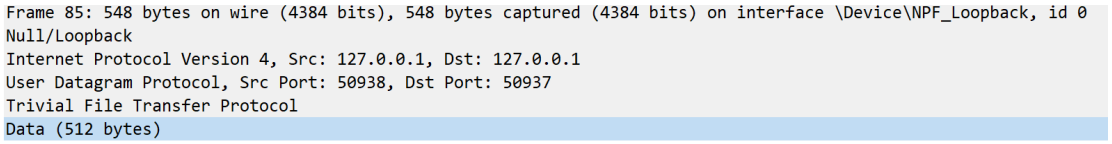
  - **¿Cuántos bytes se han transferido al cliente y al servidor en esta transferencia?**

    Cliente - Servidor: 548 bytes * 21 paquetes + 216 bytes (último paquete) = 11724 bytes.

    Servidor - Cliente: 36 bytes * 22 paquetes = 792 bytes.

  - **¿Cuántos bytes de carga útil se transfieren en cada cuadro? ¿La suma coincide con el tamaño del archivo?**

Sara Soriano - 240007
Rubén Vera - 241456
Eneko Treviño - 241679



Como se puede observar en la imagen, de los 548 bytes, de carga útil solo tenemos en cada paquete 512, ya que consideramos carga útil los bytes del paquete excluyendo los bytes de la cabecera. Por lo tanto, en cada paquete solo obtendremos 512 bytes de carga útil y 36 bytes de cabecera.

Esto significa que de los 11724, 36 bytes * 22 paquetes = 792 bytes corresponden a cabecera y, el resto, 11724 - 792 bytes = 10932 bytes son los bytes de carga útil. El resultado de sumar la carga útil de todos los paquetes es bastante aproximado al tamaño del archivo. Ya que en nuestro caso, el archivo pesa 11000 bytes y la suma nos da 10932. Por otra parte, vemos que añadiendo los bytes de la cabecera nos pasamos del tamaño.