

Ejercicios de captura de tráfico

Ejercicio 1:

Hemos abierto dos consolas, una para ejecutar Wireshark con permisos de superusuario y la segunda para realizar el ping más adelante.

Ejecutamos Wireshark con el comando:

```
$ sudo wireshark-gtk
```

Se inicia Wireshark y realizando click derecho en la sección de las columnas elegimos la opción “Column Preferences...”

Se nos despliega un menú en el que añadimos una nueva columna con nombre PO de tipo Src port (unresolved) y PD de tipo Dest port (unresolved).

Empezamos a capturar el tráfico desde Wireshark y realizamos el ping en la segunda consola con el comando:

```
$ sudo hping3 -S -p 80 www.uam.es
```

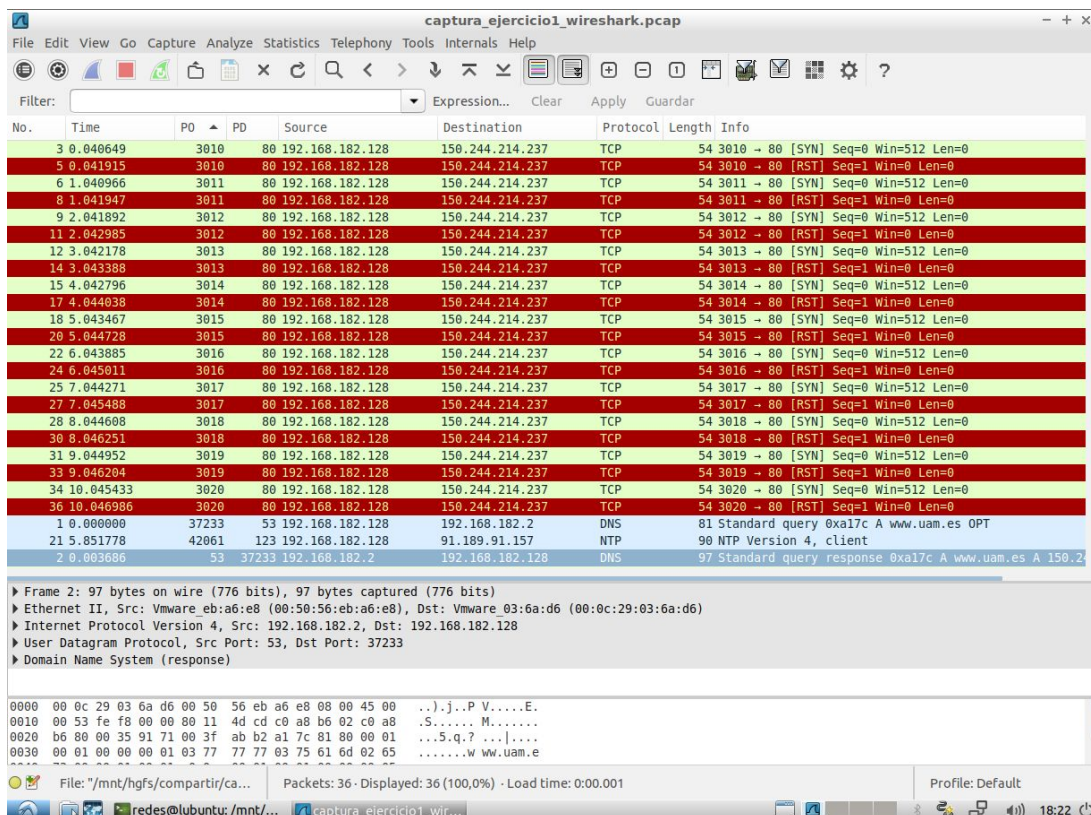
Detenemos la captura de tráfico y empezamos a analizarlo.

Guardamos la captura en formato “pcap”.

Cerramos Wireshark y volvemos a lanzarlo desde su correspondiente consola.

Abrimos la captura anteriormente guardada y comprobamos que efectivamente se guardó correctamente.

Ordenamos los paquetes según el campo PO anteriormente establecido y observamos que hay un único paquete con valor 53 en el campo PO.



The screenshot shows the Wireshark interface with a packet capture named 'captura_ejercicio1_wireshark.pcap'. The packet list is sorted by the 'PO' column, showing a single packet with PO 53. The packet details pane shows the structure of the packet: Ethernet II, Internet Protocol Version 4, User Datagram Protocol, and Domain Name System (response).

No.	Time	PO	PD	Source	Destination	Protocol	Length	Info
3	0.040649	3010	80	192.168.182.128	150.244.214.237	TCP	54	3010 → 80 [SYN] Seq=0 Win=512 Len=0
5	0.041915	3010	80	192.168.182.128	150.244.214.237	TCP	54	3010 → 80 [RST] Seq=1 Win=0 Len=0
6	1.040966	3011	80	192.168.182.128	150.244.214.237	TCP	54	3011 → 80 [SYN] Seq=0 Win=512 Len=0
8	1.041947	3011	80	192.168.182.128	150.244.214.237	TCP	54	3011 → 80 [RST] Seq=1 Win=0 Len=0
9	2.041892	3012	80	192.168.182.128	150.244.214.237	TCP	54	3012 → 80 [SYN] Seq=0 Win=512 Len=0
11	2.042985	3012	80	192.168.182.128	150.244.214.237	TCP	54	3012 → 80 [RST] Seq=1 Win=0 Len=0
12	3.042178	3013	80	192.168.182.128	150.244.214.237	TCP	54	3013 → 80 [SYN] Seq=0 Win=512 Len=0
14	3.043388	3013	80	192.168.182.128	150.244.214.237	TCP	54	3013 → 80 [RST] Seq=1 Win=0 Len=0
15	4.042796	3014	80	192.168.182.128	150.244.214.237	TCP	54	3014 → 80 [SYN] Seq=0 Win=512 Len=0
17	4.044038	3014	80	192.168.182.128	150.244.214.237	TCP	54	3014 → 80 [RST] Seq=1 Win=0 Len=0
18	5.043467	3015	80	192.168.182.128	150.244.214.237	TCP	54	3015 → 80 [SYN] Seq=0 Win=512 Len=0
20	5.044728	3015	80	192.168.182.128	150.244.214.237	TCP	54	3015 → 80 [RST] Seq=1 Win=0 Len=0
22	6.043885	3016	80	192.168.182.128	150.244.214.237	TCP	54	3016 → 80 [SYN] Seq=0 Win=512 Len=0
24	6.045011	3016	80	192.168.182.128	150.244.214.237	TCP	54	3016 → 80 [RST] Seq=1 Win=0 Len=0
25	7.044271	3017	80	192.168.182.128	150.244.214.237	TCP	54	3017 → 80 [SYN] Seq=0 Win=512 Len=0
27	7.045488	3017	80	192.168.182.128	150.244.214.237	TCP	54	3017 → 80 [RST] Seq=1 Win=0 Len=0
28	8.044608	3018	80	192.168.182.128	150.244.214.237	TCP	54	3018 → 80 [SYN] Seq=0 Win=512 Len=0
30	8.046251	3018	80	192.168.182.128	150.244.214.237	TCP	54	3018 → 80 [RST] Seq=1 Win=0 Len=0
31	9.044952	3019	80	192.168.182.128	150.244.214.237	TCP	54	3019 → 80 [SYN] Seq=0 Win=512 Len=0
33	9.046204	3019	80	192.168.182.128	150.244.214.237	TCP	54	3019 → 80 [RST] Seq=1 Win=0 Len=0
34	10.045433	3020	80	192.168.182.128	150.244.214.237	TCP	54	3020 → 80 [SYN] Seq=0 Win=512 Len=0
36	10.046986	3020	80	192.168.182.128	150.244.214.237	TCP	54	3020 → 80 [RST] Seq=1 Win=0 Len=0
1	0.000000	37233	53	192.168.182.128	192.168.182.2	DNS	81	Standard query 0xa17c A www.uam.es OPT
21	5.851778	42061	123	192.168.182.128	91.189.91.157	NTP	90	NTP Version 4, client
2	0.003686	53	37233	192.168.182.2	192.168.182.128	DNS	97	Standard query response 0xa17c A www.uam.es A 150.2

Frame 2: 97 bytes on wire (776 bits), 97 bytes captured (776 bits)
Ethernet II, Src: Vmware eb:a6:e8 (00:50:56:eb:a6:e8), Dst: Vmware 03:6a:d6 (00:0c:29:03:6a:d6)
Internet Protocol Version 4, Src: 192.168.182.2, Dst: 192.168.182.128
User Datagram Protocol, Src Port: 53, Dst Port: 37233
Domain Name System (response)

0000 00 0c 29 03 6a d6 00 50 56 eb a6 e8 00 00 45 00 ...).j..P V....E.
0010 00 53 fe f8 00 00 00 11 4d cd c0 a8 b6 02 c0 a8 .S.....M.....
0020 b6 00 00 35 91 71 00 3f ab b2 a1 7c 81 00 00 01 ...5;q.7 ...|....
0030 00 01 00 00 00 01 03 77 77 77 03 75 61 6d 02 65W ww.uam.e

File: "/mnt/hgfs/compartir/ca... Packets: 36 - Displayed: 36 (100.0%) - Load time: 0:00.001 Profile: Default

Ejercicio 2:

Empezamos la captura con Wireshark y abrimos un navegador web para generar tráfico de paquetes.

Detenemos la captura y añadimos un filtro escribiendo la condición en la ventana “Filter”:

ip && frame.len > 1000

Este filtro solo capturará paquetes que son de tipo IP y que tengan un tamaño de paquete mayor a 1000 Bytes.

Para guardar únicamente la captura de los paquetes filtrados seleccionamos la opción

File > Export Specified Packets...

Y guardamos seleccionando la opción “Displayed”

Al comparar el tamaño de los primeros paquetes IP con el campo ‘length’ del protocolo IP podemos observar que todos ellos, algunos con distintos tamaños, difieren de 14 Bytes entre el tamaño IP y el campo ‘length’ del protocolo IP:

3432 - 3446, 3793 - 3807, 3541 - 3555, 9777 - 9791, 9777 - 9791.

The screenshot shows the Wireshark interface with the filter 'ip' applied. The packet list shows several packets, with packet 4 selected. The packet details pane for packet 4 shows the following information:

- Frame 1: 3446 bytes on wire (27568 bits), 3446 bytes captured (27568 bits)
- Ethernet II, Src: Vmware eb:a6:e8 (00:50:56:eb:a6:e8), Dst: Vmware 03:6a:d6 (00:0c:29:03:6a:d6)
- Internet Protocol Version 4, Src: 52.215.229.35, Dst: 192.168.182.128
- 0100 ... = Version: 4
- 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 3432
- Identification: 0xffff13 (65299)
- Flags: 0x0000
- Time to live: 128
- Protocol: TCP (6)
- Header checksum: 0x9d58 [validation disabled]
- [Header checksum status: Unverified]
- Source: 52.215.229.35
- Destination: 192.168.182.128
- Transmission Control Protocol, Src Port: 443, Dst Port: 32830, Seq: 1, Ack: 1, Len: 3392
- Secure Sockets Layer

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0010 0d 00 ff 13 00 00 00 06 9d 58 34 d7 e5 23 c0 a8 .h.....X4..#..
0020 05 00 01 bb 80 3e 45 bd 96 0e c9 50 5c e2 50 10 ..->E. ...P..P.
0030 fa f0 9e 7e 00 00 16 03 03 00 59 02 00 00 55 03 ..Y...U.
0040 03 37 35 72 70 f6 d8 ad 5d 36 d8 fc 71 06 59 26 .75rp...}6..q.Y&
```

Creemos que esta diferencia de tamaño se debe a la encapsulación de los paquetes en función a las capas. 14 Bytes estarían reservados a la capa de Enlace, y por lo tanto no se mostrarían en el tamaño del paquete IP (capa de Red).

Ejercicio 3:

Hemos añadido una columna “interarrival” seleccionando el tipo “Delta time”. Para ello hemos hecho click derecho en las columnas y hemos seleccionado la opción “Column Preferences...”, donde se nos despliega el menú para poder añadir nuevas columnas.

Ejercicio 4:

Editamos la columna "Time" realizando click derecho en las columnas y seleccionando la opción "Column Preferences...". Se nos despliega el menú de columnas y cambiamos el tipo de "Time" de "Time (format as specified)" a "Absolute time, as YYYY-MM-DD, and time". Esta opción nos mostrará la fecha y la hora con minutos y segundos de la captura, y también nos mostrará un número que corresponde al tiempo Unix con resolución en segundos. Lo hemos cotejado con el tiempo Unix de nuestra práctica1.py y coinciden los valores.

Ejercicio 5:

Antes de iniciar la captura seleccionamos en las opciones de captura, en la ventana "Capture Filter" introducimos el siguiente comando: "udp"

Iniciamos la captura en Wireshark, y ésta solo captura paquetes de tipo UDP.

Empezamos a generar tráfico a través de un navegador web.

Desde una consola nueva, ejecutamos el siguiente comando:

```
$ sudo hping3 -S -p 80 www.uam.es
```

Volvemos a comprobar los paquetes y observamos que seguimos capturando solo paquetes de tipo UDP (entre ellos el paquete producido por el comando anterior).

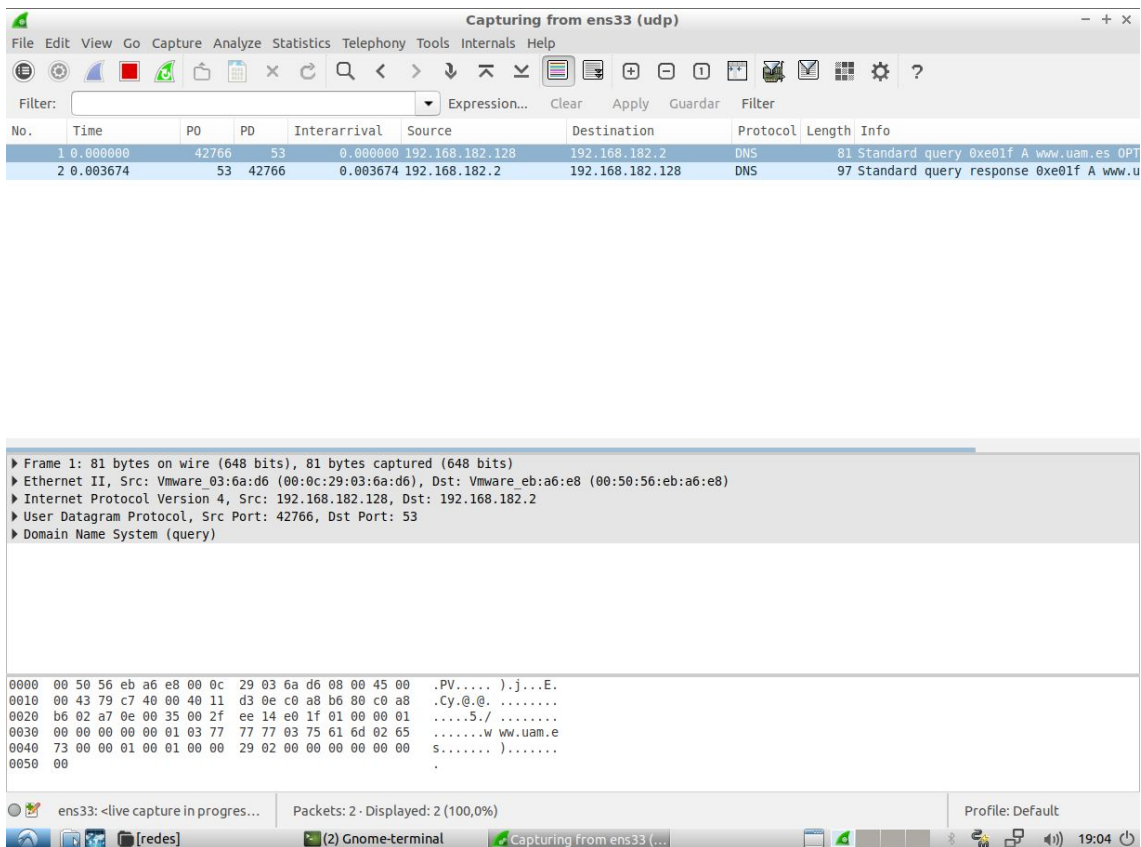


Ilustración 3: Captura de paquetes UDP