



Universidad
Rey Juan Carlos

Escuela Técnica Superior
Ingeniería de Telecomunicación

TECNOLOGÍAS DE TELEVISIÓN EN INTERNET

PRÁCTICA 1: ANÁLISIS DE TRÁFICO EN REDES IP

Yolanda Lillo Mata

DNI: 20616264-F

Correo: y.lillo.2016@alumnos.urjc.es

Aunque no sea una pregunta de la memoria, antes de empezar vamos a ver los pasos que hemos ido realizando para ponernos en contexto, lo primero que hacemos es iniciar una sesión en VNC y otra en SSH a través de la página de los laboratorio docentes de la ETSIT. Una vez tenemos la sesión iniciada en cada terminal, usamos el comando *ifconfig* o *ip address show* para consultar las ips de servidor y cliente.

```
ylillo@f-l3210-pc10:~$ ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s31f6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    group default qlen 1000
    link/ether 48:4d:7e:cb:65:02 brd ff:ff:ff:ff:ff:ff
    inet 212.128.255.190/23 brd 212.128.255.255 scope global dynamic enp0s31f6
        valid_lft 21447sec preferred_lft 21447sec
    inet6 fe80::4a4d:7eff:feeb:6502/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    group default
    link/ether 02:42:83:03:65:33 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
ylillo@f-l3210-pc10:~$
```

Imagen 1. Obtener IP cliente en VNC

```
ylillo@f-l3208-pc01:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:4f:7a:1a:7e3 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s31f6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 212.128.255.65 netmask 255.255.254.0 broadcast 212.128.255.255
    inet6 fe80::529a:4c:ff:fe00:410d prefixlen 64 scopeid 0x20<link>
    ether 50:9a:4c:00:41:0d txqueuelen 1000 (Ethernet)
    RX packets 6884207 bytes 8059875617 (8.0 GB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2205630 bytes 1583981466 (1.5 GB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 20 memory 0xf7100000-f7120000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Bucle local)
    RX packets 1510972 bytes 882595652 (882.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1510972 bytes 882595652 (882.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Imagen 2. Obtener IP servidor en SSH

También probamos a enviar un ping desde el cliente al servidor para ver como creíamos se puede establecer la conexión.

```

ylillo@f-l3210-pc10:~$ ping 212.128.255.65
PING 212.128.255.65 (212.128.255.65) 56(84) bytes of data.
64 bytes from 212.128.255.65: icmp_seq=1 ttl=64 time=0.531 ms
64 bytes from 212.128.255.65: icmp_seq=2 ttl=64 time=0.531 ms
64 bytes from 212.128.255.65: icmp_seq=3 ttl=64 time=0.533 ms
64 bytes from 212.128.255.65: icmp_seq=4 ttl=64 time=0.541 ms
64 bytes from 212.128.255.65: icmp_seq=5 ttl=64 time=0.537 ms
^C
--- 212.128.255.65 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4081ms
rtt min/avg/max/mdev = 0.531/0.534/0.541/0.004 ms
ylillo@f-l3210-pc10:~$

```

Imagen 3. Ping del cliente hacia el servidor

Una vez hecho estos pasos empezamos el desarrollo de la práctica.

1. Comunicaciones UDP:

a. ¿Cómo se configura el Servidor? Indica los comandos utilizados.

Configuramos el servidor con el comando `iperf -s -u -i 1`, donde, `-s` quiere decir que es un servidor, `-u` nos indica que estamos en una conexión UDP y por último, `-i` nos indica en este caso que cada 1 segundo se envían informes de la red.

```

ylillo@f-l3208-pc01:~$ iperf -s -u -i 1
-----
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.65 port 5001 connected with 212.128.255.190 port 44853
[ ID] Interval      Transfer    Bandwidth   Jitter    Lost/Totl  Datagrams
[ 3] 0.0- 1.0 sec   114 MBytes  953 Mbits/sec  0.024 ms  0/81054 (0%)
[ 3] 1.0- 2.0 sec   113 MBytes  952 Mbits/sec  0.024 ms  0/80955 (0%)
[ 3] 2.0- 3.0 sec   114 MBytes  953 Mbits/sec  0.030 ms  0/81048 (0%)
[ 3] 3.0- 4.0 sec   114 MBytes  953 Mbits/sec  0.027 ms  0/81024 (0%)
[ 3] 4.0- 5.0 sec   113 MBytes  950 Mbits/sec  0.025 ms 161/80977 (0.2%)
[ 3] 5.0- 6.0 sec   113 MBytes  952 Mbits/sec  0.026 ms 182/81114 (0.22%)
[ 3] 6.0- 7.0 sec   114 MBytes  954 Mbits/sec  0.019 ms  0/81101 (0%)
[ 3] 7.0- 8.0 sec   114 MBytes  953 Mbits/sec  0.026 ms  0/81051 (0%)
[ 3] 8.0- 9.0 sec   114 MBytes  953 Mbits/sec  0.029 ms  0/81061 (0%)
[ 3] 9.0-10.0 sec   114 MBytes  953 Mbits/sec  0.063 ms  0/80999 (0%)
[ 3] 0.0-10.0 sec  1.11 GBytes  953 Mbits/sec  0.026 ms 343/810431 (0.042%)

```

Imagen 4. Configuración servidor UDP y resultado

b. ¿Cómo se configura el Cliente? Indica los comandos utilizados.

Configuramos el cliente con el comando `iperf -c 212.128.255.65 -u -b 2g -i 1`, donde `-c` nos indica que es un cliente, seguido de la ip del servidor que en este caso es 212.128.255.65, `-u` indica que es una conexión UDP, `-b` para indicar el ancho de banda, y por último `-i` que nos indica que cada 1 segundo se envían informes de la red.

```

ylillo@f-l3210-pc10:~$ iperf -c 212.128.255.65 -u -b 2g -i 1
-----
Client connecting to 212.128.255.65, UDP port 5001
Sending 1470 byte datagrams, IPG target: 5.88 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.190 port 44853 connected with 212.128.255.65 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec   114 MBytes  953 Mbits/sec
[ 3] 1.0- 2.0 sec   113 MBytes  952 Mbits/sec
[ 3] 2.0- 3.0 sec   114 MBytes  953 Mbits/sec
[ 3] 3.0- 4.0 sec   114 MBytes  953 Mbits/sec
[ 3] 4.0- 5.0 sec   114 MBytes  953 Mbits/sec
[ 3] 5.0- 6.0 sec   114 MBytes  954 Mbits/sec
[ 3] 6.0- 7.0 sec   114 MBytes  953 Mbits/sec
[ 3] 7.0- 8.0 sec   114 MBytes  953 Mbits/sec
[ 3] 8.0- 9.0 sec   114 MBytes  953 Mbits/sec
[ 3] 9.0-10.0 sec   114 MBytes  954 Mbits/sec
[ 3] 0.0-10.0 sec  1.11 GBytes  953 Mbits/sec
[ 3] Sent 810431 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec  1.11 GBytes  953 Mbits/sec  0.025 ms 343/810431 (0.042%)

```

Imagen 5. Configuración cliente UDP y resultado

c. ¿Cuál es el ancho de banda máximo para transmitir en la red?

El ancho de banda máximo para transmitir en la red es de 954 Mbits/sec por mucho valor que le demos al ancho de banda esto es lo máximo que podrá enviar (ver imágenes 4 y 5).

d. Indica los valores de ancho de banda, jitter y pérdida de paquetes.

El ancho de banda, como se dice en el apartado c, es de 954 Mbits/sec, el jitter tiene un valor de 0.063 segundos y tenemos una pérdida de paquetes de 0,22 % (ver imagen 4)

e. ¿Qué pasa si se aumenta y disminuye el tamaño del buffer? Indica las variaciones en las medidas. Analiza los resultados obtenidos y describe las conclusiones a las que llegas.

Si ponemos el buffer tanto en cliente como en servidor al mismo número, no cambia el jitter pero si tenemos pérdidas. Esto puede ser causado por la gran cantidad de datos que se envían.

Si aumentamos el tamaño del buffer en el cliente, no vemos ningún cambio en el ancho de banda, nuestra red no satura, y el jitter aumenta. Si el servidor tiene un buffer pequeño hace que se sature, esto puede ser debido al tamaño de los paquetes.

Si el buffer del cliente lo cambiamos a uno más pequeño, el jitter no cambia pero si se modifica el ancho de banda ajustándose al buffer para evitar las pérdidas

de paquetes, si cambiamos también el del servidor a uno más pequeño se puede ver que si tenemos pérdidas y disminuye el ancho de banda.

Si solo cambiamos el buffer en el servidor se reduce el ancho de banda y aumenta la pérdida de paquete y jitter.

f. ¿Qué pasa si se aumenta y disminuye el tamaño del paquete? Indica las variaciones en las medidas. Analiza los resultados obtenidos y describe las conclusiones a las que llegas.

Si ponemos un tamaño pequeño, el jitter disminuye, si también cambiamos el servidor, aumenta las pérdidas de paquetes, en el ancho de banda no se nota mucho el cambio.

Si solo aumentamos el servidor, no vemos ningún cambio notable y disminuimos el servidor se reduce el ancho de banda.

g. Adjunta la captura de Wireshark y describe brevemente los paquetes pertenecientes al flujo de la comunicación que se ha desarrollado entre el cliente y el servidor.

ip.addr == 212.128.255.65 and ip.addr == 212.128.255.190

No.	Time	Source	Destination	Protocol	Length	Info
1636	4.6210952..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1637	4.6211113..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1638	4.6211189..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1639	4.6211265..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1640	4.6211341..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1641	4.6211415..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1642	4.6211488..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470
1643	4.6211559..	212.128.255.190	212.128.255.65	UDP	1512	43588 → 5001 Len=1470

» Frame 1639: 1512 bytes on wire (12096 bits), 1512 bytes captured (12096 bits) on interface enp0s3if6, id 0
 » Ethernet II, Src: Dell_cb:65:02 (48:4d:7e:cb:65:02), Dst: Dell_00:41:0d (50:9a:4c:00:41:0d)
 » Internet Protocol Version 4, Src: 212.128.255.190, Dst: 212.128.255.65
 » User Datagram Protocol, Src Port: 43588, Dst Port: 5001
 Source Port: 43588
 Destination Port: 5001
 Length: 1470
 Checksum: 0xadd9 [unverified]
 [Checksum Status: Unverified]
 [Stream index: 0]
 » [Timestamps]
 » Data (1470 bytes)
 Data: 000000030037c0290805af480000000000000000030313233..
 [Length: 1470]

Imagen 6. Captura wireshark comunicación UDP

Como es un servicio no orientado a conexión, no vemos ningún mensaje de comienzo o finalización cuando filtramos los paquetes UDP. En la imagen podemos ver tanto el puerto origen como el puerto destino, y la longitud de los paquetes con la cabecera da 8 bytes. En total tenemos 1470 bytes.

2. Comunicaciones TCP:

a. ¿Cómo se configura el Servidor? Indica los comandos utilizados.

Para configurar el servidor utilizamos el comando `iperf -s -i 1`, donde `-i` sirve para dar un informe de la red cada segundo, `-s` indica que es el servidor y como no ponemos nada, por defecto se considera que es TCP.

b. ¿Cómo se configura el Cliente? Indica los comandos utilizados.

Para configurar el cliente utilizamos el comando `iperf -c 212.128.255.65 -i 1`, donde `-c` nos indica que es un cliente y después vemos la ip del servidor y `-i` que nos da un informe de la red cada segundo, al igual que antes, como no ponemos nada por defecto se considera TCP.

c. ¿Cuál es el ancho de banda máximo para transmitir en la red? ¿Ese ancho de banda es distinto que con UDP? Razona tu respuesta.

```
ylillo@f-13208-pc01:~$ iperf -s -i 1
-----
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
-----
[ 4] local 212.128.255.65 port 5001 connected with 212.128.255.190 port 36658
[ ID] Interval           Transfer     Bandwidth
[ 4] 0.0- 1.0 sec      92.8 MBytes  779 Mbits/sec
[ 4] 1.0- 2.0 sec      93.0 MBytes  780 Mbits/sec
[ 4] 2.0- 3.0 sec      92.7 MBytes  778 Mbits/sec
[ 4] 3.0- 4.0 sec      92.6 MBytes  777 Mbits/sec
[ 4] 4.0- 5.0 sec      93.1 MBytes  781 Mbits/sec
[ 4] 5.0- 6.0 sec      92.5 MBytes  776 Mbits/sec
[ 4] 6.0- 7.0 sec      92.8 MBytes  778 Mbits/sec
[ 4] 7.0- 8.0 sec      92.4 MBytes  775 Mbits/sec
[ 4] 8.0- 9.0 sec      90.4 MBytes  758 Mbits/sec
[ 4] 9.0-10.0 sec      90.3 MBytes  757 Mbits/sec
[ 4] 0.0-10.0 sec      924 MBytes  774 Mbits/sec
```

Imagen 7. Servidor TCP

Vemos como el máximo ancho de banda es 780 Mbits/sec disminuye su valor respecto a UDP porque TCP se adapta a la red para así no saturar y evitar que se pierdan paquetes.

d. Indica los valores de ancho de banda, jitter y pérdida de paquetes.

Como estamos en TCP y se trata de una conexión fiable no tendremos pérdida de paquetes, ni tampoco jitter, el ancho de banda máximo es el mismo que hemos indicado en el apartado anterior.

e. ¿Qué pasa si se aumenta y disminuye el tamaño del buffer? Indica las variaciones en las medidas. Analiza los resultados obtenidos y describe las conclusiones a las que llegas.

No se observan cambios ni aumentando o disminuyendo el buffer de cliente o servidor ni de ambos a la vez.

f. ¿Qué pasa si se aumenta y disminuye el tamaño de ventana? Indica las variaciones en las medidas. Analiza los resultados obtenidos y describe las conclusiones a las que llegas.

Si aumentamos el tamaño de ventana del cliente, se envía todo el ancho de banda, lo mismo pasa si a la vez aumentamos el servidor.

Si disminuimos el cliente, el ancho de banda disminuye y si a la vez disminuimos el servidor, disminuye aún más el ancho de banda.

Si disminuyo el tamaño de ventana del servidor, disminuye el ancho de banda, si a la vez elevamos el tamaño de ventana del cliente no hay cambios en el ancho de banda.

Por último si se aumenta el tamaño de ventana del servidor, se ocupa todo el ancho de banda, si a al vez se disminuye el del cliente se disminuye también su ancho de banda.

g. Adjunta la captura de Wireshark y describe brevemente los paquetes pertenecientes al flujo de la comunicación que se ha desarrollado entre el cliente y el servidor, así como los parámetros del protocolo.

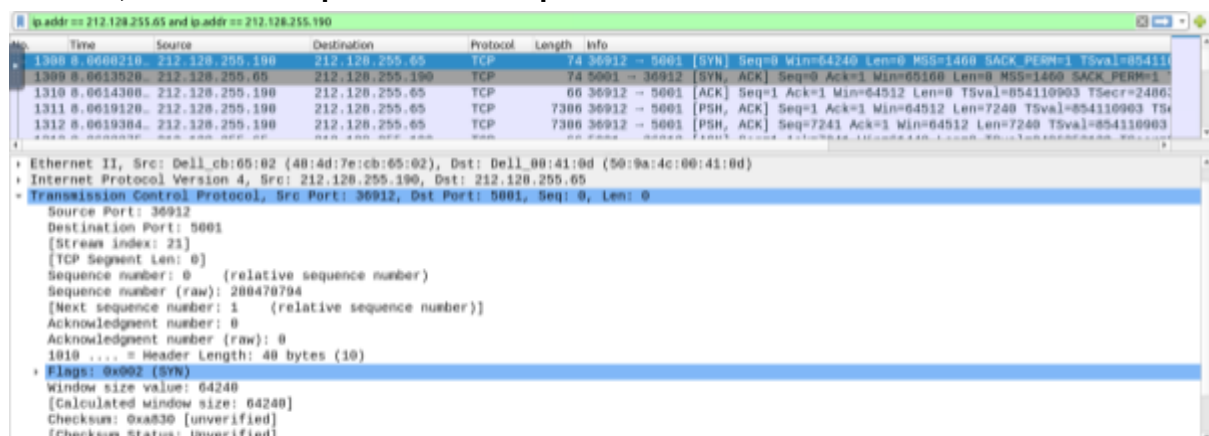


Imagen 8. Captura 1 wireshark comunicación TCP (SYN)

Cuando empieza a capturar paquetes vemos que se manda un mensaje SYN desde el cliente y el servidor contesta con un SYN ACK para confirmar la recepción y así poder empezar a comunicarse, esto se debe a que TCP está orientado a conexión.

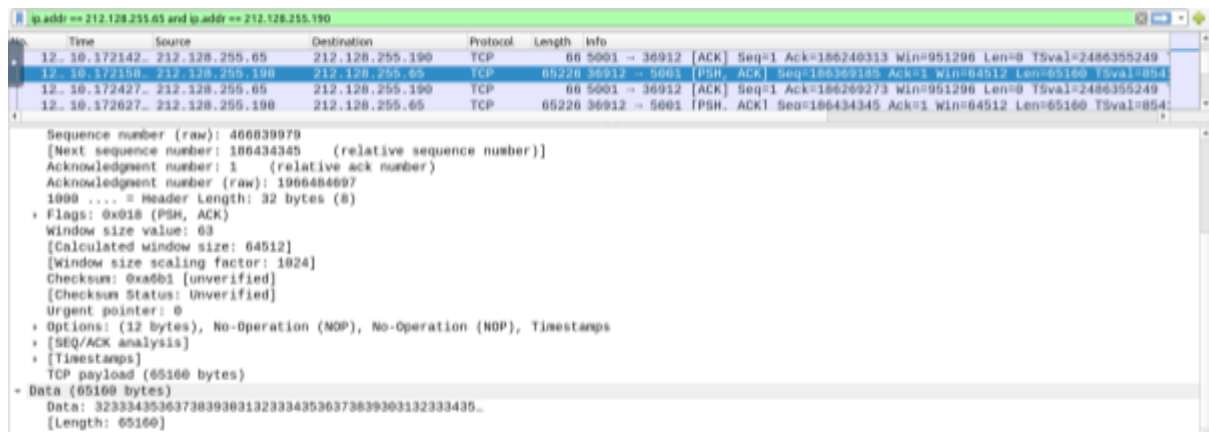


Imagen 9. Captura 2 wireshark comunicación TCP (PSH,ACK)

En los paquetes PSH,ACK, paquetes donde se envían datos vemos como en la cabecera está el número de secuencia, ACK, puerto origen y destino, unos flags, tamaño de ventana, checksum...luego tiene 12 Bytes en la cabecera de opcionales donde una es el timestamp. Y por último vemos que el tamaño de los datos es de 65160 bytes

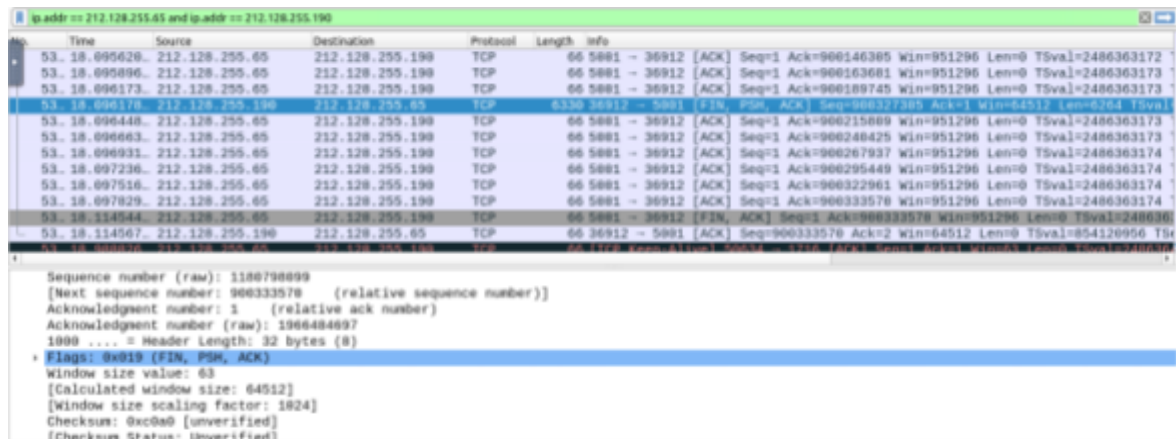


Imagen 10. Captura 3 wireshark comunicación TCP (FIN)

Después de enviar todos los datos, para llevar a cabo el cierre de la conexión el cliente le envía un mensaje FIN lo que el servidor contesta con un FIN ACK y por último el cliente vuelve a enviar otro ACK para confirmar que le ha llegado y así cerrar la comunicación.

Comunicación entre dos clientes y un servidor

Cambio IPs de cliente y servidor con respecto a los ejercicios anteriores por hacer poder realizar las capturas con wireshark en VNC.

3. Comunicaciones UDP:

a. ¿Cómo se configura el Servidor? Indica los comandos utilizados.

Se configura con el mismo comando que cuando solo tenemos un cliente (ver apartado 1.a)

```

Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.190 port 5001 connected with 212.128.254.42 port 49156
[ ID] Interval      Transfer      Bandwidth      Jitter    Lost/Total Datagrams
[ 3] 0.0- 1.0 sec   113 MBytes   949 Mbits/sec   0.028 ms    6/80672 (0.0074%)
[ 4] local 212.128.255.190 port 5001 connected with 212.128.255.65 port 44580
[ 3] 1.0- 2.0 sec   75.9 MBytes  637 Mbits/sec   0.039 ms   25624/79785 (32%)
[ 4] 0.0- 1.0 sec   58.0 MBytes  487 Mbits/sec   0.068 ms   39391/80766 (49%)
[ 3] 2.0- 3.0 sec   56.6 MBytes  475 Mbits/sec   0.048 ms   40694/81079 (50%)
[ 4] 1.0- 2.0 sec   58.0 MBytes  486 Mbits/sec   0.040 ms   39708/81065 (49%)
[ 3] 3.0- 4.0 sec   55.9 MBytes  469 Mbits/sec   0.053 ms   40462/80354 (50%)
[ 4] 2.0- 3.0 sec   55.7 MBytes  468 Mbits/sec   0.061 ms   41326/81093 (51%)
[ 3] 4.0- 5.0 sec   56.9 MBytes  477 Mbits/sec   0.051 ms   40507/81079 (50%)
[ 4] 3.0- 4.0 sec   58.4 MBytes  490 Mbits/sec   0.051 ms   39300/80950 (49%)
[ 3] 5.0- 6.0 sec   57.4 MBytes  481 Mbits/sec   0.068 ms   39794/80731 (49%)
[ 4] 4.0- 5.0 sec   55.6 MBytes  467 Mbits/sec   0.065 ms   41373/81068 (51%)
[ 3] 6.0- 7.0 sec   57.6 MBytes  483 Mbits/sec   0.042 ms   39818/80897 (49%)
[ 4] 5.0- 6.0 sec   55.8 MBytes  468 Mbits/sec   0.036 ms   41288/81097 (51%)
[ 3] 7.0- 8.0 sec   55.1 MBytes  462 Mbits/sec   0.063 ms   40726/79996 (51%)
[ 4] 6.0- 7.0 sec   59.1 MBytes  496 Mbits/sec   0.037 ms   38903/81058 (48%)
[ 3] 8.0- 9.0 sec   55.8 MBytes  468 Mbits/sec   0.042 ms   40955/80766 (51%)
[ 4] 7.0- 8.0 sec   56.2 MBytes  471 Mbits/sec   0.078 ms   40985/81077 (51%)
[ 3] 9.0-10.0 sec   57.3 MBytes  481 Mbits/sec   0.040 ms   39911/80795 (49%)
[ 3] 0.0-10.3 sec   642 MBytes  525 Mbits/sec  15.636 ms  348636/806452 (43%)
[ 4] 8.0- 9.0 sec   75.8 MBytes  636 Mbits/sec   0.033 ms   26999/81104 (33%)
[ 4] 9.0-10.0 sec   114 MBytes  952 Mbits/sec   0.035 ms   188/81159 (0.23%)
[ 4] 0.0-10.0 sec   647 MBytes  542 Mbits/sec   0.060 ms  349461/810653 (43%)

```

Imagen 11. Resultado servidor UDP con dos clientes

b. ¿Cómo se configura cada Cliente? Indica los comandos utilizados.

Los dos clientes se configuran igual que cuando teníamos uno (ver apartado 1.b)

```

ylillo@f-13109-pc02:~$ iperf -c 212.128.255.190 -u -b 2g -i 1
-----
Client connecting to 212.128.255.190, UDP port 5001
Sending 1470 byte datagrams, IPG target: 5.88 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.254.42 port 49156 connected with 212.128.255.190 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 1.0 sec   113 MBytes   949 Mbits/sec
[ 3] 1.0- 2.0 sec   112 MBytes   942 Mbits/sec
[ 3] 2.0- 3.0 sec   114 MBytes   953 Mbits/sec
[ 3] 3.0- 4.0 sec   113 MBytes   946 Mbits/sec
[ 3] 4.0- 5.0 sec   114 MBytes   954 Mbits/sec
[ 3] 5.0- 6.0 sec   113 MBytes   949 Mbits/sec
[ 3] 6.0- 7.0 sec   113 MBytes   951 Mbits/sec
[ 3] 7.0- 8.0 sec   112 MBytes   941 Mbits/sec
[ 3] 8.0- 9.0 sec   113 MBytes   949 Mbits/sec
[ 3] 9.0-10.0 sec   113 MBytes   949 Mbits/sec
[ 3] 0.0-10.0 sec   1.10 GBytes  948 Mbits/sec
[ 3] Sent 806451 datagrams
[ 3] Server Report:
[ 3] 0.0-10.3 sec   642 MBytes  525 Mbits/sec  15.636 ms 348636/806452 (43%)

```

Imagen 12. Configuración y resultado cliente 1 comunicación UDP


```

ylillo@f-13208-pc01:~$ iperf -c 212.128.255.190 -u -b 2g -i 1
-----
Client connecting to 212.128.255.190, UDP port 5001
Sending 1470 byte datagrams, IPG target: 5.88 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.65 port 44580 connected with 212.128.255.190 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec    114 MBytes  954 Mbits/sec
[ 3] 1.0- 2.0 sec    114 MBytes  953 Mbits/sec
[ 3] 2.0- 3.0 sec    114 MBytes  953 Mbits/sec
[ 3] 3.0- 4.0 sec    113 MBytes  952 Mbits/sec
[ 3] 4.0- 5.0 sec    114 MBytes  954 Mbits/sec
[ 3] 5.0- 6.0 sec    114 MBytes  954 Mbits/sec
[ 3] 6.0- 7.0 sec    114 MBytes  953 Mbits/sec
[ 3] 7.0- 8.0 sec    114 MBytes  954 Mbits/sec
[ 3] 8.0- 9.0 sec    114 MBytes  953 Mbits/sec
[ 3] 0.0-10.0 sec    1.11 GBytes 953 Mbits/sec
[ 3] Sent 810653 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec    647 MBytes  542 Mbits/sec  0.059 ms 349461/810653 (43%)

```

Imagen 13. Configuración y resultado cliente 2 comunicación UDP

c. ¿Cuál es el ancho de banda máximo para transmitir en la red? ¿Es distinto si se utiliza un solo cliente? Razona tu respuesta.

El ancho de banda máximo para cada cliente es de aproximadamente 477 Mbit/s, si se aumenta este ancho de banda tendría lugar la pérdida de paquetes. Como tenemos dos clientes enviando, el ancho de banda total será de 954 Mbit/s aproximadamente de ahí que cada uno de los clientes tenga la mitad. Un cliente no podrá usar el máximo del ancho de banda porque sino el otro cliente tendría poquísimo ancho de banda y se produciría la pérdida de paquetes por el aumento de congestión.

d. Indica los valores de ancho de banda, jitter y pérdida de paquetes.

```

Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.190 port 5001 connected with 212.128.254.42 port 49156
[ ID] Interval      Transfer    Bandwidth    Jitter    Lost/Total Datagrams
[ 3] 0.0- 1.0 sec    113 MBytes  949 Mbits/sec  0.028 ms  6/80672 (0.0074%)
[ 4] local 212.128.255.190 port 5001 connected with 212.128.255.65 port 44580
[ 3] 1.0- 2.0 sec    75.9 MBytes  637 Mbits/sec  0.039 ms 25624/79785 (32%)
[ 4] 0.0- 1.0 sec    58.0 MBytes  487 Mbits/sec  0.068 ms 39391/80766 (49%)
[ 3] 2.0- 3.0 sec    56.6 MBytes  475 Mbits/sec  0.048 ms 40694/81079 (50%)
[ 4] 1.0- 2.0 sec    58.0 MBytes  486 Mbits/sec  0.040 ms 39708/81065 (49%)
[ 3] 3.0- 4.0 sec    55.9 MBytes  469 Mbits/sec  0.053 ms 40462/80354 (50%)
[ 4] 2.0- 3.0 sec    55.7 MBytes  468 Mbits/sec  0.061 ms 41326/81093 (51%)
[ 3] 4.0- 5.0 sec    56.9 MBytes  477 Mbits/sec  0.051 ms 40507/81079 (50%)
[ 4] 3.0- 4.0 sec    58.4 MBytes  490 Mbits/sec  0.051 ms 39300/80950 (49%)
[ 3] 5.0- 6.0 sec    57.4 MBytes  481 Mbits/sec  0.068 ms 39794/80731 (49%)
[ 4] 4.0- 5.0 sec    55.6 MBytes  467 Mbits/sec  0.065 ms 41373/81068 (51%)
[ 3] 6.0- 7.0 sec    57.6 MBytes  483 Mbits/sec  0.042 ms 39818/80897 (49%)
[ 4] 5.0- 6.0 sec    55.8 MBytes  468 Mbits/sec  0.036 ms 41288/81097 (51%)
[ 3] 7.0- 8.0 sec    55.1 MBytes  462 Mbits/sec  0.063 ms 40726/79996 (51%)
[ 4] 6.0- 7.0 sec    59.1 MBytes  496 Mbits/sec  0.037 ms 38903/81058 (48%)
[ 3] 8.0- 9.0 sec    55.8 MBytes  468 Mbits/sec  0.042 ms 40955/80766 (51%)
[ 4] 7.0- 8.0 sec    56.2 MBytes  471 Mbits/sec  0.078 ms 40985/81077 (51%)
[ 3] 9.0-10.0 sec    57.3 MBytes  481 Mbits/sec  0.040 ms 39911/80795 (49%)
[ 3] 0.0-10.3 sec    642 MBytes  525 Mbits/sec 15.636 ms 348636/806452 (43%)
[ 4] 8.0- 9.0 sec    75.8 MBytes  636 Mbits/sec  0.033 ms 26999/81104 (33%)
[ 4] 9.0-10.0 sec    114 MBytes  952 Mbits/sec  0.035 ms 188/81159 (0.23%)
[ 4] 0.0-10.0 sec    647 MBytes  542 Mbits/sec  0.060 ms 349461/810653 (43%)

```

Imagen 11. Resultado servidor UDP con dos clientes

	Ancho de banda	Jitter	Pérdida de paquetes
CLIENTE 1	483 Mbits/sec	0.068 s	50 %
CLIENTE 2	496 Mbits/sec	0.078 s	50 %

e. ¿Qué ocurre cuando hay dos clientes transmitiendo datos?

Cuanto tenemos dos clientes transmitiendo datos a la vez el ancho de banda máximo se reduce porque ambos clientes tienen que compartirlo. Cuando ambos envían vemos que el servidor se satura y que hay pérdida de paquetes.

f. Prueba a lanzar el servidor y un solo cliente ocupando todo el ancho de banda. A mitad de la transferencia, lanza el segundo cliente ocupando también todo el ancho de banda. ¿Cuáles son los resultados obtenidos? ¿Qué conclusiones obtienes?

Podemos ver como el primer cliente transmite ocupando todo el ancho de banda, como si fuera un único cliente, cuando se une el segundo cliente intenta ocupar también todo el ancho de banda y eso no puede ser, por tanto, vemos como ambos cliente reducen el ancho de banda a la mitad y tienen unas pérdidas de un 50 %.

Podemos concluir que la comunicación UDP no controla cuando llegan dos clientes a la red transmitiendo todo el ancho de banda lo que lleva a que se congestione y se reduzca el ancho de banda a la mitad y por tanto, se eleven las pérdidas, es decir, se produce una sobrecarga de la red.

g. Prueba a lanzar el servidor y un solo cliente ocupando la mitad del ancho de banda. A mitad de la transferencia, lanza el segundo cliente ocupando también la mitad del ancho de banda. ¿Cuáles son los resultados obtenidos? ¿Qué conclusiones obtienes?

Cuando los dos clientes emiten a la mitad de ancho de banda, disminuye el ancho de banda que llega al servidor, sigue habiendo pérdidas pero muy inferiores a cuando los dos envían con el máximo ancho de banda.

Podemos llegar a la conclusión de que como ahora solo utilizan la mitad de ancho de banda cada uno de los clientes no llega a producirse una saturación de la red, es decir, no se supera el ancho de banda máximo aunque puede que alguna vez al sumar los anchos de banda de los clientes si supere el ancho de banda máximo de ahí las pequeñas pérdidas que se producen.

h. Adjunta la captura de Wireshark y describe brevemente los paquetes pertenecientes al flujo de la comunicación que se ha desarrollado entre el cliente y el servidor.

No.	Time	Source	Destination	Protocol	Length	Info
14..	7.7631735..	212.128.255.65	212.128.255.190	UDP	1512	44580 → 5001 Len=1470
14..	7.7631735..	212.128.254.42	212.128.255.190	UDP	1512	49156 → 5001 Len=1470
14..	7.7631841..	212.128.255.65	212.128.255.190	UDP	1512	44580 → 5001 Len=1470
14..	7.7631841..	212.128.254.42	212.128.255.190	UDP	1512	49156 → 5001 Len=1470
14..	7.7631841..	212.128.255.65	212.128.255.190	UDP	1512	44580 → 5001 Len=1470
14..	7.7631841..	212.128.254.42	212.128.255.190	UDP	1512	49156 → 5001 Len=1470
14..	7.7634149..	212.128.254.42	212.128.255.190	UDP	1512	49156 → 5001 Len=1470
14..	7.7634150..	212.128.255.65	212.128.255.190	UDP	1512	44580 → 5001 Len=1470
14..	7.7634150..	212.128.254.42	212.128.255.190	UDP	1512	49156 → 5001 Len=1470

* Frame 147007: 1512 bytes on wire (12096 bits), 1512 bytes captured (12096 bits) on interface enp0s31f6, id 0
 * Ethernet II, Src: Dell_00:41:0d (50:9a:4c:00:41:0d), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 * Internet Protocol Version 4, Src: 212.128.255.65, Dst: 212.128.255.190
 * User Datagram Protocol, Src Port: 44580, Dst Port: 5001
 Source Port: 44580
 Destination Port: 5001
 Length: 1470
 Checksum: 0x0440 [unverified]
 [Checksum Status: Unverified]
 [Stream index: 7]
 * [Timestamps]
 * Data (1470 bytes)

Imagen 12. Captura wireshark comunicación UDP con dos clientes

Vemos que no se producen cambios respecto a la conexión con un único cliente, solo vemos como ahora hay dos clientes enviando datos al servidor mediante UDP, y vemos que sigue sin haber mensajes de inicio y fin al igual que pasaba en la primera conexión UDP de esta práctica. Podemos destacar que el tamaño de los mensajes es el mismo, cambian los puertos origen y el valor de checksum.

4. Comunicaciones TCP:

a. **¿Cómo se configura el Servidor? Indica los comandos utilizados.**

Se configura con el mismo comando que cuando solo tenemos un cliente (ver apartado 2.a)

b. **¿Cómo se configura cada Cliente? Indica los comandos utilizados.**

Ambos clientes se configuran con el mismo comando que cuando teníamos solo un cliente (ver apartado 2.b)

c. **¿Cuál es el ancho de banda máximo para transmitir en la red? ¿Es distinto que con UDP? ¿Es distinto si se utiliza un solo cliente TCP? Razona tu respuesta.**

```

Server listening on TCP port 5001
TCP window size: 128 KByte (default)
.....
[ 4] local 212.128.255.190 port 5001 connected with 212.128.254.42 port 36804
[ ID] Interval      Transfer      Bandwidth
[ 4] 0.0- 1.0 sec    111 MBytes    927 Mbits/sec
[ 5] local 212.128.255.190 port 5001 connected with 212.128.255.65 port 50574
[ 4] 1.0- 2.0 sec    90.8 MBytes    762 Mbits/sec
[ 5] 0.0- 1.0 sec    45.3 MBytes    380 Mbits/sec
[ 4] 2.0- 3.0 sec    65.0 MBytes    545 Mbits/sec
[ 5] 1.0- 2.0 sec    50.0 MBytes    419 Mbits/sec
[ 4] 3.0- 4.0 sec    58.3 MBytes    489 Mbits/sec
[ 5] 2.0- 3.0 sec    51.9 MBytes    436 Mbits/sec
[ 4] 4.0- 5.0 sec    62.3 MBytes    523 Mbits/sec
[ 5] 3.0- 4.0 sec    49.0 MBytes    411 Mbits/sec
[ 4] 5.0- 6.0 sec    64.3 MBytes    540 Mbits/sec
[ 5] 4.0- 5.0 sec    45.8 MBytes    384 Mbits/sec
[ 4] 6.0- 7.0 sec    63.3 MBytes    531 Mbits/sec
[ 5] 5.0- 6.0 sec    51.3 MBytes    430 Mbits/sec
[ 4] 7.0- 8.0 sec    61.4 MBytes    515 Mbits/sec
[ 5] 6.0- 7.0 sec    47.5 MBytes    398 Mbits/sec
[ 4] 8.0- 9.0 sec    56.8 MBytes    477 Mbits/sec
[ 5] 7.0- 8.0 sec    54.7 MBytes    459 Mbits/sec
[ 4] 9.0-10.0 sec    65.7 MBytes    552 Mbits/sec
[ 5] 8.0- 9.0 sec    65.5 MBytes    550 Mbits/sec
[ 4] 0.0-10.0 sec    701 MBytes    586 Mbits/sec
[ 5] 9.0-10.0 sec    91.8 MBytes    770 Mbits/sec
[ 4] 0.0-10.0 sec    554 MBytes    464 Mbits/sec

```

Imagen 13. Servidor TCP comunicación con dos clientes.

Podemos ver como al principio cuando aún no se ha iniciado el segundo cliente el primero ocupa todo el ancho de banda, después vamos viendo que el ancho de banda máximo para transmitir en la red cada uno de los clientes es de más o menos 470Mbit/s si sumamos ambos nos da el maximo de ancho de banda que coincide con el mismo que utilizaba un solo cliente, es decir, 927Mbit/s. Por tanto, comparten el ancho de banda total reduciendo a la mitad en cada uno de ellos para así evitar la pérdida de paquetes o congestiones.

d. Indica los valores de ancho de banda, jitter y pérdida de paquetes.

Como estamos en TCP no podemos hablar de jitter ni pérdidas de paquetes como dijimos con anterioridad en la otra conexión TCP con un cliente. Respecto al ancho de banda, el cliente 1 aproximadamente 489 Mbit/s y el cliente 2 aproximadamente 411 Mbit/s como vemos en el envío de los segundos dos al tres en cada uno de ellos.

e. ¿Qué ocurre cuando hay dos clientes transmitiendo datos?

Cuando tenemos dos clientes transmitiendo como TCP tiene protocolos de congestión puede adaptar los anchos de banda de cada uno de los clientes para hacer que no se sature la red. De ahí las variaciones que íbamos viendo en la imagen 13. De este modo, cada cliente ocupa la mitad del ancho de banda máximo.

f. Prueba a lanzar el servidor y un solo cliente ocupando todo el ancho de banda. A mitad de la transferencia, lanza el segundo cliente ocupando también todo el ancho de banda. ¿Cuáles son los resultados obtenidos? ¿Qué conclusiones obtienes?

```
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
.....
```

ID	Interval	Transfer	Bandwidth
[4]	0.0- 1.0 sec	111 MBytes	931 Mbits/sec
[4]	1.0- 2.0 sec	111 MBytes	932 Mbits/sec
[4]	2.0- 3.0 sec	111 MBytes	931 Mbits/sec
[4]	3.0- 4.0 sec	111 MBytes	931 Mbits/sec
[4]	4.0- 5.0 sec	111 MBytes	930 Mbits/sec
[5]	local 212.128.255.190 port 5001 connected with 212.128.254.42 port 36806		
[4]	5.0- 6.0 sec	93.7 MBytes	786 Mbits/sec
[5]	0.0- 1.0 sec	47.0 MBytes	395 Mbits/sec
[4]	6.0- 7.0 sec	61.9 MBytes	519 Mbits/sec
[5]	1.0- 2.0 sec	51.3 MBytes	430 Mbits/sec
[4]	7.0- 8.0 sec	60.1 MBytes	504 Mbits/sec
[5]	2.0- 3.0 sec	52.4 MBytes	440 Mbits/sec
[4]	8.0- 9.0 sec	56.1 MBytes	470 Mbits/sec
[5]	3.0- 4.0 sec	60.7 MBytes	509 Mbits/sec
[4]	9.0-10.0 sec	51.8 MBytes	435 Mbits/sec
[4]	0.0-10.0 sec	881 MBytes	736 Mbits/sec
[5]	4.0- 5.0 sec	77.2 MBytes	648 Mbits/sec
[5]	5.0- 6.0 sec	90.2 MBytes	756 Mbits/sec
[5]	6.0- 7.0 sec	91.7 MBytes	770 Mbits/sec
[5]	7.0- 8.0 sec	91.8 MBytes	770 Mbits/sec
[5]	8.0- 9.0 sec	91.7 MBytes	769 Mbits/sec
[5]	9.0-10.0 sec	89.9 MBytes	754 Mbits/sec
[5]	0.0-10.0 sec	745 MBytes	624 Mbits/sec

Imagen 14. Servidor TCP ocupando todo el ancho de banda

En los resultado vemos como el primer cliente acaba su conexión con un ancho de banda de 881 Mbit/s y el segundo cliente que se une en mitad de la conexión termina con 624 Mbit/s. Podemos concluir que si el primer cliente está ocupando todo el ancho de banda en un principio cuando llega un nuevo cliente va a intentar también ocupar todo el ancho de

banda posible sin que tengan lugar congestiones por tanto los clientes se unen pero por el mecanismo que tiene TCP de control de la red hace que se disminuya el ancho de banda para el primer cliente que al principio estaba utilizando mucho más ancho de banda.

g. Prueba a lanzar el servidor y un solo cliente ocupando la mitad del ancho de banda. A mitad de la transferencia, lanza el segundo cliente ocupando también la mitad del ancho de banda. ¿Cuáles son los resultados obtenidos? ¿Qué conclusiones obtienes?

Tras ver los resultado se puede ver cómo los clientes envían a la vez con la mitad del ancho de banda, por tanto no se modifica este ancho de banda ya que, la red no está saturada y ambos pueden enviar la mitad sin ningún problema de congestión.

h. Adjunta la captura de Wireshark y describe brevemente los paquetes pertenecientes al flujo de la comunicación que se ha desarrollado entre el cliente y el servidor, así como los parámetros del protocolo.

No.	Time	Source	Destination	Protocol	Length	Info
635	2.1193034	212.128.254.42	212.128.255.190	TCP	66	39510 → 1716 [ACK] Seq=1 Ack=1 Win=63 Len=0 TSval=2095293232 TSecr=2613726
637	2.1193608	212.128.255.190	212.128.254.42	TCP	66	[TCP ACKed unseen segment] 1716 → 39510 [ACK] Seq=1 Ack=2 Win=63 Len=0 TS
740	2.2035727	212.128.255.190	212.128.254.42	TCP	66	[TCP Keep-Alive] [TCP ACKed unseen segment] 1716 → 39510 [ACK] Seq=0 Ack=
777	2.2038707	212.128.254.42	212.128.255.190	TCP	66	[TCP Previous segment not captured] 39510 → 1716 [ACK] Seq=2 Ack=1 Win=63
1028	3.1160632	212.128.254.42	212.128.255.190	TCP	74	36810 → 5001 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=20952
1029	3.1160672	212.128.255.190	212.128.254.42	TCP	74	5001 → 36810 [SYN, ACK] Seq=0 Ack=1 Win=65100 Len=0 MSS=1460 SACK_PERM=1
1030	3.1163915	212.128.254.42	212.128.255.190	TCP	66	36810 → 5001 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=2095294229 TSecr=261
1031	3.1169721	212.128.254.42	212.128.255.190	TCP	7306	36810 → 5001 [PSH, ACK] Seq=1 Ack=1 Win=64512 Len=7240 TSval=2095294230 T
1032	3.1169723	212.128.254.42	212.128.255.190	TCP	7306	36810 → 5001 [PSH, ACK] Seq=7241 Ack=1 Win=64512 Len=7240 TSval=2095294230 T
1033	3.1170403	212.128.255.190	212.128.254.42	TCP	66	5001 → 36810 [ACK] Seq=1 Ack=7241 Win=61440 Len=0 TSval=2613731752 TSecr=
1034	3.1170959	212.128.255.190	212.128.254.42	TCP	66	5001 → 36810 [ACK] Seq=1 Ack=14481 Win=56320 Len=0 TSval=2613731752 TSecr=
1035	3.1173873	212.128.254.42	212.128.255.190	TCP	10202	36810 → 5001 [PSH, ACK] Seq=14481 Ack=1 Win=64512 Len=10136 TSval=2095294
1036	3.1174246	212.128.255.190	212.128.254.42	TCP	66	5001 → 36810 [ACK] Seq=1 Ack=24617 Win=40152 Len=0 TSval=2613731753 TSecr=
1037	3.1177022	212.128.254.42	212.128.255.190	TCP	14546	36810 → 5001 [PSH, ACK] Seq=24617 Ack=1 Win=64512 Len=14480 TSval=2095294
1038	3.1177024	212.128.254.42	212.128.255.190	TCP	4410	36810 → 5001 [PSH, ACK] Seq=30097 Ack=1 Win=64512 Len=4344 TSval=20952942
1039	3.1177025	212.128.254.42	212.128.255.190	TCP	2902	36810 → 5001 [PSH, ACK] Seq=43441 Ack=1 Win=64512 Len=2896 TSval=20952942

* Frame 1028: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface enp0s31f0, id 0
 * Ethernet II, Src: Dell_f8:05:3e (e4:b9:7a:f8:05:3e), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 * Internet Protocol Version 4, Src: 212.128.254.42, Dst: 212.128.255.190
 * Transmission Control Protocol, Src Port: 36810, Dst Port: 5001, Seq: 0, Len: 0

Imagen 15. Captura 1 wireshark comunicación TCP dos clientes (SYN cliente 1)

Time	Source	Destination	Protocol	Length	Info
17..	13.156238	212.128.254.42	212.128.255.190	TCP	2962 36810 → 5001 [PSH, ACK] Seq=782943737 Ack=1 Win=64512 Len=2896 TSval=2095
17..	13.156239	212.128.254.42	212.128.255.190	TCP	2962 36810 → 5001 [PSH, ACK] Seq=782944633 Ack=1 Win=64512 Len=2896 TSval=2095
17..	13.156322	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=782944633 Win=3143680 Len=0 TSval=2613741791
17..	13.156331	212.128.254.42	212.128.255.190	TCP	2962 36810 → 5001 [PSH, ACK] Seq=782949529 Ack=1 Win=64512 Len=2896 TSval=2095
17..	13.156339	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=782949529 Win=3145728 Len=0 TSval=2613741792
17..	13.156361	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=782952425 Win=3145728 Len=0 TSval=2613741792
17..	13.156399	212.128.254.42	212.128.255.190	TCP	13098 36810 → 5001 [ACK] Seq=782952425 Ack=1 Win=64512 Len=13032 TSval=209530426
17..	13.156625	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=782965457 Win=3145728 Len=0 TSval=2613741792
17..	13.156823	212.128.254.42	212.128.255.190	TCP	24682 36810 → 5001 [PSH, ACK] Seq=782965457 Ack=1 Win=64512 Len=24616 TSval=2095
17..	13.156823	212.128.254.42	212.128.255.190	TCP	8754 36810 → 5001 [ACK] Seq=782998073 Ack=1 Win=64512 Len=8688 TSval=209530426
17..	13.156911	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=782998761 Win=3145728 Len=0 TSval=2613741792
17..	13.157112	212.128.254.42	212.128.255.190	TCP	2962 36810 → 5001 [PSH, ACK] Seq=782998761 Ack=1 Win=64512 Len=2896 TSval=2095
17..	13.157113	212.128.254.42	212.128.255.190	TCP	22538 36810 → 5001 [FIN, PSH, ACK] Seq=783001657 Ack=1 Win=64512 Len=22472 TSva
17..	13.157203	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [ACK] Seq=1 Ack=783024130 Win=3145728 Len=0 TSval=2613741792
17..	13.176118	212.128.255.190	212.128.254.42	TCP	66 5001 → 36810 [FIN, ACK] Seq=1 Ack=783024130 Win=3145728 Len=0 TSval=26137
17..	13.176718	212.128.254.42	212.128.255.190	TCP	66 36810 → 5001 [ACK] Seq=783024130 Ack=2 Win=64512 Len=0 TSval=2095304289 T

* Frame 174812: 22538 bytes on wire (180304 bits), 22538 bytes captured (180304 bits) on interface enp0s31f6, id 0
 * Ethernet II, Src: Dell_f8:05:3e (e4:b9:7a:f8:05:3e), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 * Internet Protocol Version 4, Src: 212.128.254.42, Dst: 212.128.255.190
 * Transmission Control Protocol, Src Port: 36810, Dst Port: 5001, Seq: 783001657, Ack: 1, Len: 22472
 * Data (22472 bytes)

Imagen 16. Captura 2 wireshark comunicación TCP dos clientes (FIN cliente 1)

No.	Time	Source	Destination	Protocol	Length	Info
644.2	1.281837	212.128.255.65	212.128.255.190	TCP	60	50534 → 1716 [ACK] Seq=1 Ack=1 Min=63 Len=6 TSval=2488698785 TSecr=856446
645.2	1.282289	212.128.255.190	212.128.255.65	TCP	60	TCP ACKed unseq segment 1716 → 50634 [ACK] Seq=1 Ack=2 Min=63 Len=0 TS
703.2	2.833283	212.128.255.190	212.128.255.65	TCP	60	TCP Keep-Alive [TCP ACKed unseq segment] 1716 → 50634 [ACK] Seq=0 Ack=
754.2	2.836487	212.128.255.65	212.128.255.190	TCP	60	TCP Previous segment not captured 50634 → 1716 [ACK] Seq=2 Ack=1 Min=63
908.2	9.714791	212.128.255.190	212.128.255.65	TCP	60	1716 → 50636 [ACK] Seq=1 Ack=1 Min=63 Len=6 TSval=856457353 TSecr=1488889
990.2	9.719237	212.128.255.65	212.128.255.190	TCP	60	TCP ACKed unseq segment 50636 → 1716 [ACK] Seq=1 Ack=2 Min=63 Len=0 TS
1430.3	15.327111	212.128.255.65	212.128.255.190	TCP	60	TCP Keep-Alive [TCP ACKed unseq segment] 50636 → 1716 [ACK] Seq=0 Ack=
1432.3	15.324027	212.128.255.190	212.128.255.65	TCP	60	TCP Previous segment not captured 1716 → 50636 [ACK] Seq=2 Ack=1 Min=63
23.5	6.6145792	212.128.255.65	212.128.255.190	TCP	74	50584 → 50601 [SYN] Seq=8 Win=64512 Len=0 MSS=1460 SACK_PERM=1 TSval=24887
23.5	6.6146442	212.128.255.190	212.128.255.65	TCP	74	50601 → 50584 [SYN, ACK] Seq=8 Ack=1 Min=65108 Len=8 MSS=1460 SACK_PERM=1
23.5	6.6151317	212.128.255.65	212.128.255.190	TCP	60	50584 → 50601 [ACK] Seq=1 Ack=1 Min=64512 Len=8 TSval=2488702272 TSecr=8564
23.5	6.6156799	212.128.255.65	212.128.255.190	TCP	7386	50584 → 50601 [PSH, ACK] Seq=1 Ack=1 Min=64512 Len=7248 TSval=2488702273 TS
23.5	6.6156791	212.128.255.65	212.128.255.190	TCP	7386	50584 → 50601 [PSH, ACK] Seq=7241 Ack=1 Min=64512 Len=7248 TSval=248870227
23.5	6.6157284	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=7241 Min=63448 Len=8 TSval=856460937 TSecr=24
23.5	6.6157316	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=14481 Min=56328 Len=8 TSval=856460937 TSecr=
23.5	6.6162497	212.128.255.65	212.128.255.190	TCP	10282	50584 → 50601 [PSH, ACK] Seq=14481 Ack=1 Min=64512 Len=10136 TSval=2488702

• Frame 23998: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface enp0s31f6, id 0
 • Ethernet II, Src: Dell_00:41:0d (58:9a:4c:00:41:0d), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 • Internet Protocol Version 4, Src: 212.128.255.65, Dst: 212.128.255.190
 • Transmission Control Protocol, Src Port: 50584, Dst Port: 50601, Seq: 8, Len: 0

Imagen 17. Captura 3 wireshark comunicación TCP dos clientes (SYN cliente 2)

No.	Time	Source	Destination	Protocol	Length	Info
56.7	4.273445	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99345833 Min=877568 Len=8 TSval=856461849 TS
56.7	4.275647	212.128.255.65	212.128.255.190	TCP	11650	50584 → 50601 [ACK] Seq=99345833 Ack=1 Min=64512 Len=11584 TSval=2488704086
56.7	4.276811	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99357417 Min=877568 Len=8 TSval=856461849 TS
56.7	4.278148	212.128.255.65	212.128.255.190	TCP	27578	50584 → 50601 [ACK] Seq=99357417 Ack=1 Min=64512 Len=27512 TSval=2488704086
56.7	4.278683	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99384929 Min=877568 Len=8 TSval=856461849 TS
56.7	4.288681	212.128.255.65	212.128.255.190	TCP	1514	50584 → 50601 [PSH, ACK] Seq=99384929 Ack=1 Min=64512 Len=1448 TSval=24887
56.7	4.288681	212.128.255.65	212.128.255.190	TCP	20338	50584 → 50601 [ACK] Seq=99386377 Ack=1 Min=64512 Len=28272 TSval=2488704086
56.7	4.281845	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99406649 Min=877568 Len=8 TSval=856461849 TS
56.7	4.283198	212.128.255.65	212.128.255.190	TCP	4410	50584 → 50601 [PSH, ACK] Seq=99406649 Ack=1 Min=64512 Len=4344 TSval=24887
56.7	4.283191	212.128.255.65	212.128.255.190	TCP	4410	50584 → 50601 [ACK] Seq=99418993 Ack=1 Min=64512 Len=4344 TSval=2488704085
56.7	4.283552	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99415337 Min=877568 Len=8 TSval=856461850 TS
56.7	4.285717	212.128.255.65	212.128.255.190	TCP	13098	50584 → 50601 [ACK] Seq=99415337 Ack=1 Min=64512 Len=13832 TSval=2488704086
56.7	4.286874	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99428369 Min=877568 Len=8 TSval=856461850 TS
56.7	4.288247	212.128.255.65	212.128.255.190	TCP	13098	50584 → 50601 [ACK] Seq=99428369 Ack=1 Min=64512 Len=13832 TSval=2488704086
56.7	4.288591	212.128.255.190	212.128.255.65	TCP	66	50581 → 50584 [ACK] Seq=1 Ack=99441491 Min=877568 Len=8 TSval=856461850 TS
56.7	4.288756	212.128.255.65	212.128.255.190	TCP	23234	50584 → 50601 [PSH, ACK] Seq=99441491 Ack=1 Min=64512 Len=23168 TSval=24887

• Frame 56525: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp0s31f6, id 0
 • Ethernet II, Src: Dell_00:41:0d (58:9a:4c:00:41:0d), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 • Internet Protocol Version 4, Src: 212.128.255.65, Dst: 212.128.255.190
 • Transmission Control Protocol, Src Port: 50584, Dst Port: 50601, Seq: 99384929, Ack: 1, Len: 1448
 • Data (1448 bytes)

Imagen 18. Captura 4 wireshark comunicación TCP dos clientes (ACK cliente 2)

No.	Time	Source	Destination	Protocol	Length	Info
19.15	6.31193	212.128.255.65	212.128.255.190	TCP	17442	50584 → 50601 [PSH, ACK] Seq=684513937 Ack=1 Min=64512 Len=17376 TSval=2488
19.15	6.31193	212.128.255.65	212.128.255.190	TCP	4410	50584 → 50601 [ACK] Seq=684531313 Ack=1 Min=64512 Len=4344 TSval=248871228
19.15	6.31276	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684535657 Min=1431552 Len=8 TSval=856470052
19.15	6.31446	212.128.255.65	212.128.255.190	TCP	20338	50584 → 50601 [ACK] Seq=684535657 Ack=1 Min=64512 Len=20272 TSval=248871228
19.15	6.31531	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684555929 Min=1431552 Len=8 TSval=856470053
19.15	6.31721	212.128.255.65	212.128.255.190	TCP	21786	50584 → 50601 [ACK] Seq=684555929 Ack=1 Min=64512 Len=21728 TSval=248871228
19.15	6.31819	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684577649 Min=1431552 Len=8 TSval=856470053
19.15	6.32001	212.128.255.65	212.128.255.190	TCP	18990	50584 → 50601 [PSH, ACK] Seq=684577649 Ack=1 Min=64512 Len=18824 TSval=2488
19.15	6.32001	212.128.255.65	212.128.255.190	TCP	5858	50584 → 50601 [ACK] Seq=684596473 Ack=1 Min=64512 Len=5792 TSval=248871228
19.15	6.32112	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684602265 Min=1431552 Len=8 TSval=856470053
19.15	6.32254	212.128.255.65	212.128.255.190	TCP	18990	50584 → 50601 [ACK] Seq=684602265 Ack=1 Min=64512 Len=18824 TSval=248871228
19.15	6.32346	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684621889 Min=1431552 Len=8 TSval=856470053
19.15	6.32531	212.128.255.65	212.128.255.190	TCP	14114	50584 → 50601 [FIN, PSH, ACK] Seq=684621889 Ack=1 Win=64512 Len=14048 TSval=
19.15	6.32572	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [ACK] Seq=1 Ack=684635138 Min=1431552 Len=8 TSval=856470054
19.15	6.33089	212.128.255.190	212.128.255.65	TCP	66	50601 → 50584 [FIN, ACK] Seq=1 Ack=684635138 Min=1431552 Len=8 TSval=856470054
19.15	6.330972	212.128.255.65	212.128.255.190	TCP	66	50584 → 50601 [ACK] Seq=684635138 Ack=2 Min=64512 Len=0 TSval=248871229 TS

• Frame 196879: 14114 bytes on wire (112912 bits), 14114 bytes captured (112912 bits) on interface enp0s31f6, id 0
 • Ethernet II, Src: Dell_00:41:0d (58:9a:4c:00:41:0d), Dst: Dell_cb:65:02 (48:4d:7e:cb:65:02)
 • Internet Protocol Version 4, Src: 212.128.255.65, Dst: 212.128.255.190
 • Transmission Control Protocol, Src Port: 50584, Dst Port: 50601, Seq: 684621889, Ack: 1, Len: 14048
 • Data (14048 bytes)

No.	Time	Source	Destination	Protocol	Length	Info
19.15	6.31193	212.128.255.65	212.128.255.190	TCP	17442	50584 → 50601 [PSH, ACK] Seq=684513937 Ack=1 Min=64512 Len=17376 TSval=2488

Source Port: 50584
 Destination Port: 50601
 [Stream index: 00]
 [TCP Segment Len: 14048]
 Sequence number: 684621889 (relative sequence number)
 Sequence number (raw): 2452991568
 [Next sequence number: 684635138 (relative sequence number)]
 Acknowledgment number: 3 (relative ack number)
 Acknowledgment number (raw): 3837493197
 1880 = Header length: 32 bytes (8)
 • Flags: 0x019 (FIN, PSH, ACK)
 Window size value: 63
 [Calculated window size: 64512]
 [Window size scaling factor: 1024]
 Checksum: 0x0f08 [unverified]
 [Checksum status: Unverified]
 Urgent pointer: 0
 • Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
 • [SEQ/ACK analysis]
 • [Timestamps]
 TCP payload (14048 bytes)
 • Data (14048 bytes)

Imagen 19. Captura 5 wireshark comunicación TCP dos clientes (FIN cliente 2)

Al principio podemos ver el inicio de conexión con el primer cliente y también como los parámetros se van ajustando en función de la red y la finalización de esta conexión con este cliente. Después comienza la conexión del segundo cliente con los ACK

correspondientes para confirmar la comunicación. Y por último vemos el fin de la comunicación entre cliente y servidor y así liberar la memoria del servidor.

5. Interacción entre protocolos:

a. ¿Cómo se configura cada Servidor? Indica los comandos utilizados.

Se configura un servidor para el tráfico TCP igual que en el apartado 2 y otro servidor para el tráfico UDP igual que en el punto 1 pero ahora ambos se encuentran en la misma máquina, en mi caso VNC.

```
ylillo@f-l3208-pc05:~$ iperf -s -i 1
-----
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
-----
[ 4] local 212.128.255.69 port 5001 connected with 212.128.254.131 port 54104
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0- 1.0 sec   1.47 MBytes 12.3 Mbits/sec
[ 4] 1.0- 2.0 sec   1.15 MBytes 9.67 Mbits/sec
[ 4] 2.0- 3.0 sec   1.35 MBytes 11.3 Mbits/sec
[ 4] 3.0- 4.0 sec   1.17 MBytes 9.82 Mbits/sec
[ 4] 4.0- 5.0 sec   1.35 MBytes 11.4 Mbits/sec
[ 4] 5.0- 6.0 sec   1.44 MBytes 12.1 Mbits/sec
[ 4] 6.0- 7.0 sec   1.29 MBytes 10.9 Mbits/sec
[ 4] 7.0- 8.0 sec   80.2 MBytes 672 Mbits/sec
[ 4] 8.0- 9.0 sec   111 MBytes 934 Mbits/sec
[ 4] 9.0-10.0 sec   110 MBytes 923 Mbits/sec
[ 4] 0.0-10.0 sec   313 MBytes 262 Mbits/sec
```

Imagen 20. Servidor TCP y resultado

```
ylillo@f-l3208-pc05:~$ iperf -s -u -i 1
-----
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.255.69 port 5001 connected with 212.128.254.130 port 55438
[ ID] Interval      Transfer    Bandwidth      Jitter    Lost/Total Datagrams
[ 3] 0.0- 1.0 sec   114 MBytes 953 Mbits/sec  0.025 ms 171368/252433 (68%)
[ 3] 0.0000-1.0000 sec 91 datagrams received out-of-order
[ 3] 1.0- 2.0 sec   113 MBytes 949 Mbits/sec  0.026 ms 348/81003 (0.43%)
[ 3] 2.0- 3.0 sec   112 MBytes 943 Mbits/sec  0.027 ms 912/81074 (1.1%)
[ 3] 3.0- 4.0 sec   113 MBytes 944 Mbits/sec  0.024 ms 815/81090 (1%)
[ 3] 4.0- 5.0 sec   113 MBytes 944 Mbits/sec  0.025 ms 842/81093 (1%)
[ 3] 5.0- 6.0 sec   112 MBytes 942 Mbits/sec  0.045 ms 1002/81067 (1.2%)
[ 3] 6.0- 7.0 sec   112 MBytes 939 Mbits/sec  0.025 ms 1278/81095 (1.6%)
[ 3] 7.0- 8.0 sec   112 MBytes 943 Mbits/sec  0.028 ms 847/81000 (1%)
[ 3] 8.0- 9.0 sec   1012 MBytes 944 Mbits/sec  0.017 ms 178365/900359 (20%)
[ 3] 0.0000-8.9929 sec 91 datagrams received out-of-order
```

Imagen 21. Servidor UDP y resultado

b. ¿Cómo se configura cada Cliente? Indica los comandos utilizados.

Se configura un cliente para el tráfico TCP igual que en el apartado 2 y otro cliente para el tráfico UDP igual que en el punto 1 pero ahora ambos se encuentran distintas máquinas SSH.

```

ylillo@f-l3203-pc03:~$ iperf -c 212.128.255.69 -i 1
-----
Client connecting to 212.128.255.69, TCP port 5001
TCP window size: 144 KByte (default)
-----
[ 3] local 212.128.254.131 port 54104 connected with 212.128.255.69 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 1.0 sec   1.62 MBytes   13.6 Mbits/sec
[ 3] 1.0- 2.0 sec   1.12 MBytes   9.44 Mbits/sec
[ 3] 2.0- 3.0 sec   1.38 MBytes   11.5 Mbits/sec
[ 3] 3.0- 4.0 sec   1.25 MBytes   10.5 Mbits/sec
[ 3] 4.0- 5.0 sec   1.38 MBytes   11.5 Mbits/sec
[ 3] 5.0- 6.0 sec   1.38 MBytes   11.5 Mbits/sec
[ 3] 6.0- 7.0 sec   1.25 MBytes   10.5 Mbits/sec
[ 3] 7.0- 8.0 sec   81.9 MBytes   687 Mbits/sec
[ 3] 8.0- 9.0 sec   111 MBytes    932 Mbits/sec
[ 3] 9.0-10.0 sec   110 MBytes    927 Mbits/sec
[ 3] 0.0-10.0 sec   313 MBytes    262 Mbits/sec
ylillo@f-l3203-pc03:~$

```

Imagen 22. Cliente TCP y resultado, misma máquina

```

ylillo@f-l3203-pc02:~$ iperf -c 212.128.255.69 -u -b 2g -i 1
-----
Client connecting to 212.128.255.69, UDP port 5001
Sending 1470 byte datagrams, IPG target: 5.88 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 3] local 212.128.254.130 port 55438 connected with 212.128.255.69 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 1.0 sec   238 MBytes    2.00 Gbits/sec
[ 3] 1.0- 2.0 sec   115 MBytes    962 Mbits/sec
[ 3] 2.0- 3.0 sec   114 MBytes    953 Mbits/sec
[ 3] 3.0- 4.0 sec   114 MBytes    953 Mbits/sec
[ 3] 4.0- 5.0 sec   114 MBytes    953 Mbits/sec
[ 3] 5.0- 6.0 sec   114 MBytes    954 Mbits/sec
[ 3] 6.0- 7.0 sec   114 MBytes    954 Mbits/sec
[ 3] 7.0- 8.0 sec   114 MBytes    953 Mbits/sec
[ 3] 8.0- 9.0 sec   114 MBytes    952 Mbits/sec
[ 3] 9.0-10.0 sec   114 MBytes    954 Mbits/sec
[ 3] 0.0-10.0 sec   1.23 GBytes    1.06 Gbits/sec
[ 3] Sent 900359 datagrams
[ 3] Server Report:
[ 3] 0.0- 9.0 sec  1012 MBytes   944 Mbits/sec   0.016 ms 178365/900359 (20%)
[ 3] 0.0000-8.9929 sec  91 datagrams received out-of-order
ylillo@f-l3203-pc02:~$

```

Imagen 23. Cliente UDP y resultado, misma máquina

c. ¿Cuál es el comportamiento del tráfico de fondo TCP? Razona tu respuesta.

El tráfico TCP se va adaptando a la red cuando se produce también tráfico de UDP realizando una bajada en su ancho de banda. Como vemos en la imagen 22 al principio utiliza menos ancho de banda porque también se produce UDP y cuando termina vuelve a aumentar.

d. ¿Cuál es el comportamiento de la aplicación UDP? Razona tu respuesta.

El tráfico UDP siempre intenta utilizar todo el ancho de banda posible. (Ver imagen 21).

e. Indica los valores de ancho de banda, jitter y pérdida de paquetes.

Ver imágenes 20 y 21.

UDP → Ancho de banda 944 Mbit/s → Jitter = 0.017 s → pérdidas 20 %

TCP → Ancho de banda 10.9 Mbit/s

f. ¿Qué se ha visto más afectado por los recursos disponibles, la aplicación UDP o el tráfico de fondo TCP?

Se ve más afectado el tráfico de fondo TCP ya que, es el que tiene que cambiar notablemente su ancho de banda.

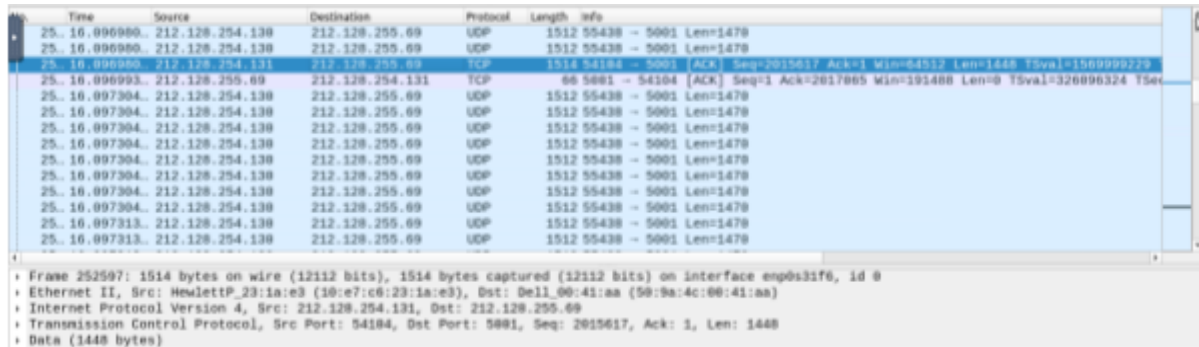
g. ¿Qué ocurre si el tráfico de streaming UDP aumenta de repente en un momento dado y trata de consumir más ancho de banda? Razona tu respuesta.

Se puede ver que el tráfico TCP se va adaptando al ancho de banda que tiene disponible, para así si el tráfico streaming UDP quiere consumir mucho más ancho de banda el tráfico de fondo TCP se reduce.

h. Analiza los resultados obtenidos en las diferentes pruebas con los distintos parámetros y describe las conclusiones a las que llegas.

Tras realizar todos los cambios y analizar los resultado, podemos concluir que el tráfico de fondo TCP siempre se ve más afectado y tiene que cambiar porque siempre intenta evitar la congestión de la red. Por el contrario el tráfico UDP siempre utiliza todo el ancho de banda disponible, el máximo y le da igual que se pierdan paquetes o congestiones.

i. Adjunta la captura de Wireshark y describe brevemente los paquetes pertenecientes al flujo de la comunicación que se ha desarrollado entre el cliente y el servidor, así como los parámetros del protocolo.



No.	Time	Source	Destination	Protocol	Length	Info
25	10.996990	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.996990	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
26	10.996993	212.128.255.69	212.128.254.131	TCP	60	5001 → 54104 [ACK] Seq=1 Ack=2617005 Win=151488 Len=0 TSval=326896324 Tsec=...
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997304	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997313	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478
25	10.997313	212.128.254.130	212.128.255.69	UDP	1512	55438 → 5001 Len=1478

Frame 252597: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp0s31f6, id 0
Ethernet II, Src: HewlettP_23:1a:e3 (10:e7:c6:23:1a:e3), Dst: Dell_00:41:aa (50:9a:4c:00:41:aa)
Internet Protocol Version 4, Src: 212.128.254.131, Dst: 212.128.255.69
Transmission Control Protocol, Src Port: 54104, Dst Port: 5001, Seq: 2615617, Ack: 1, Len: 1448
Data (1448 bytes)

Imagen 25. Parte de la captura de wireshark

Vemos como el análisis de la captura es similar a lo que hemos comentado en los puntos anteriores. Destacar que las cabeceras son iguales tanto en UDP como en TCP cuando solo teníamos un cliente pero cambia la longitud de TCP cuando aparecen los paquetes UDP.