

1. Configuración de función policing y marcado de tráfico en DSCP

Utiliza la herramienta tc para garantizar que el tráfico que entra en r1 cumple las siguientes características:

La red diffServ deberá garantizar a la entrada los siguientes anchos de banda para pc1, descartando el tráfico sobrante:

- Flujo 1: máximo 1.2 mbit con ráfaga 10k para el tráfico dirigido a pc4, marcado con calidad1. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 3.
- Flujo 3: máximo de 600kbit y ráfaga 10k, marcado con calidad3. Si se supera este ancho de banda, el tráfico será descartado definitivamente en r1.

La red diffServ deberá garantizar a la entrada los siguientes anchos de banda para pc2, descartando el tráfico sobrante:

- Flujo 2: máximo 300kbit con ráfaga 10k para el tráfico dirigido a pc5, marcado con calidad2. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 4.
- Flujo 4: máximo de 400kbit y ráfaga 10k, marcado con calidad4. Si se supera este ancho de banda, el tráfico será descartado definitivamente en r1.

//En R1

```
#!/bin/bash
tc qdisc del dev eth0 ingress

tc qdisc add dev eth0 ingress handle ffff:
tc filter add dev eth0 parent ffff: \
    protocol ip prio 1 u32 \
    match ip src 2.2.0.2/32 \
    police rate 1.2mbit burst 10k continue flowid :1

tc filter add dev eth0 parent ffff: \
    protocol ip prio 2 u32 \
    match ip src 2.2.0.3/32 \
    police rate 600kbit burst 10k continue flowid :2

tc filter add dev eth0 parent ffff: \
    protocol ip prio 3 u32 \
    match ip src 2.2.0.2/32 \
    police rate 1mbit burst 10k drop flowid :3

tc filter add dev eth0 parent ffff: \
    protocol ip prio 4 u32 \
    match ip src 2.2.0.3/32 \
    police rate 1mbit burst 10k drop flowid :4
```

Utiliza la herramienta tc para garantizar que el tráfico que entra en r2 cumple las siguientes características:

La red diffServ deberá garantizar a la entrada los siguientes anchos de banda para pc3, des- cartando el tráfico sobrante:

- Flujo 5: máximo 400kbit con ráfaga 10k dirigido a pc6, marcado con calidad2. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 6.
- Flujo 6: máximo 300kbit con ráfaga 10k dirigido a pc6, marcado con calidad3. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 7.

- Flujo 7: máximo 100kbit con ráfaga 10k, marcado con calidad4. Si se supera este ancho de banda, el tráfico será descartado definitivamente en r2.

```
#!/bin/bash
tc qdisc del dev eth0 ingress

tc qdisc add dev eth0 ingress handle fffff:
tc filter add dev eth0 parent fffff: \
    protocol ip prio 2 u32 \
    match ip src 4.2.0.2/32 \
    police rate 400kbit burst 10k continue flowid :1

tc filter add dev eth0 parent fffff: \
    protocol ip prio 3 u32 \
    match ip src 4.2.0.2/32 \
    police rate 300kbit burst 10k continue flowid :2

tc filter add dev eth0 parent fffff: \
    protocol ip prio 4 u32 \
    match ip src 4.2.0.2/32 \
    police rate 100kbit burst 10k drop flowid :3
```

```
#!/bin/bash
tc qdisc del dev eth1 root

tc qdisc add dev eth1 root handle 1:0 dsmark indices 8

tc class change dev eth1 classid 1:1 dsmark mask 0x3 value 0x28
tc class change dev eth1 classid 1:2 dsmark mask 0x3 value 0x48
tc class change dev eth1 classid 1:3 dsmark mask 0x3 value 0x68
tc class change dev eth1 classid 1:4 dsmark mask 0x3 value 0x88

tc filter add dev eth1 parent 1:0 protocol ip prio 1 \
    handle 1 tcindex classid 1:1
tc filter add dev eth1 parent 1:0 protocol ip prio 1 \
    handle 2 tcindex classid 1:2
tc filter add dev eth1 parent 1:0 protocol ip prio 1 \
    handle 3 tcindex classid 1:3
tc filter add dev eth1 parent 1:0 protocol ip prio 1 \
    handle 4 tcindex classid 1:4
```

1. Realiza scripts para r1 y otro para r2 dónde se configuren estos perfiles de tráfico. Incluye dichos scripts en la memoria.

2. Inicia capturas:

diffServ-01.cap en la subred5, diffServ-02.cap en la subred6 y diffServ-03.cap en la subred3

```
2910 packets dropped by kernel
r3:~# tcpdump -i eth0 -w /hosthome/diffser01.cap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
[2]+ Stopped                  tcpdump -i eth0 -w /hosthome/diffser01.cap
r3:~# tcpdump -i eth0 -w /hosthome/diffser02.cap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
2986 packets captured
2986 packets received by filter
0 packets dropped by kernel
r3:~# fg
tcpdump -i eth0 -w /hosthome/diffser01.cap
66 packets captured
2986 packets received by filter
2920 packets dropped by kernel
r3:~#
```

```
r4:~# tcpdump -i eth0 -w /hosthome/dif-serv03.cap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
3663 packets captured
3663 packets received by filter
0 packets dropped by kernel
r4:~#
```

para que capture el tráfico que se genera en tu escenario por el envío "simultáneo" de:

Desde el pc1 2M a pc4

```
pc1:~# iperf -u -c 5.0.0.2 -p 5001 -b 2mbit
-----
Client connecting to 5.0.0.2, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 108 KByte (default)
-----
[ 3] local 2.2.0.2 port 32768 connected with 5.0.0.2 port 5001
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 2.39 MBytes 2.00 Mbits/sec
[ 3] Sent 1702 datagrams
[ 3] Server Report:
[ 3] 0.0- 9.9 sec 2.37 MBytes 2.00 Mbits/sec 0.302 ms 7/ 1701 (0.41%)
[ 3] 0.0- 9.9 sec 1 datagrams received out-of-order
```

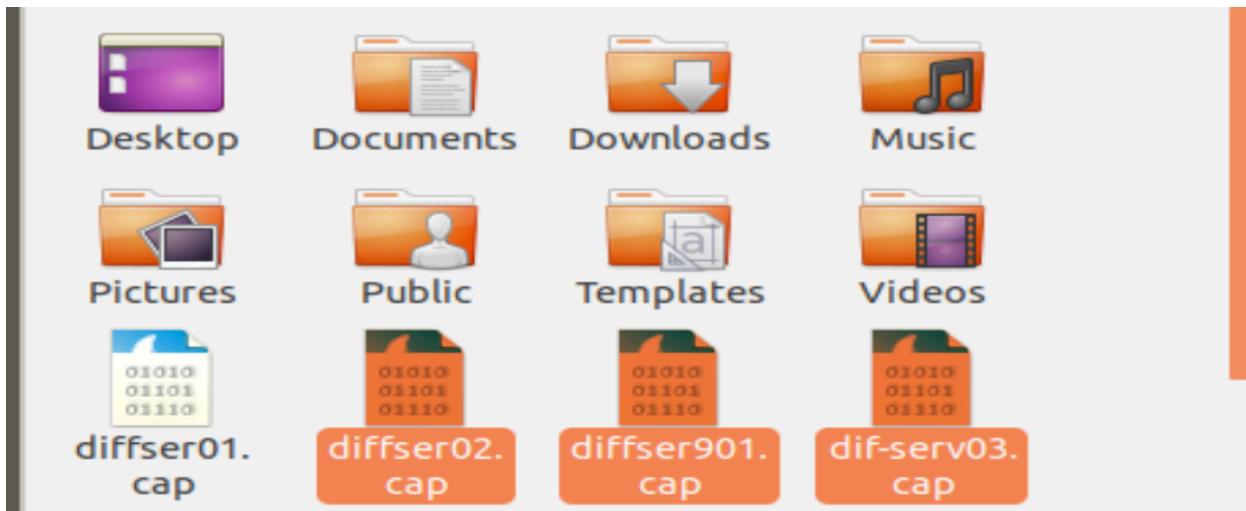
Desde el pc2 1.5M a pc5

```
pc2:~# iperf -u -c 5.0.0.3 -p 5001 -b 1.5mbit
-----
Client connecting to 5.0.0.3, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 108 KByte (default)
[ 3] local 2.2.0.3 port 32768 connected with 5.0.0.3 port 5001
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.79 MBytes 1.50 Mbits/sec
[ 3] Sent 1277 datagrams
[ 3] Server Report:
[ 3] 0.0- 9.9 sec 1.77 MBytes 1.50 Mbits/sec 0.239 ms 13/ 1277 (1%)
pc2:~#
```

Desde el pc3 1M a pc6

```
pc3:~# iperf -u -c 5.0.0.4 -p 5001 -b 1mbit
-----
Client connecting to 5.0.0.4, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 108 KByte (default)
[ 3] local 4.2.0.2 port 32768 connected with 5.0.0.4 port 5001
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.19 MBytes 999 Kbits/sec
[ 3] Sent 852 datagrams
[ 3] Server Report:
[ 3] 0.0- 9.8 sec 946 KBytes 794 Kbits/sec 0.166 ms 193/ 852 (23%)
pc3:~#
```

3. Interrumpe las capturas, al menos 1 minuto después de que la transmisión haya terminado. Comprueba que el resultado es el esperado:

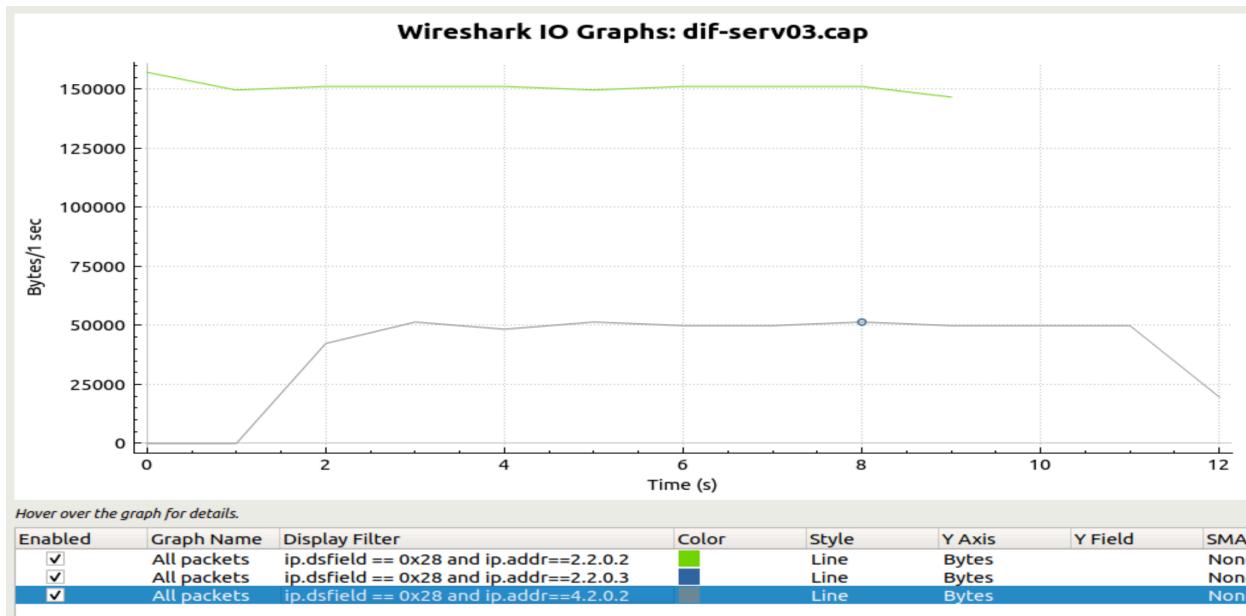


El tráfico que entra en la red diffServ es el que se ha especificado en el control de admisión. El tráfico está marcado según las especificaciones anteriores.

Para ello, consulta las gráficas IO graphs de Wireshark aplicando los filtros sobre las marcas DSCP de tal forma que se muestre cada calidad marcada de cada una de las fuentes:

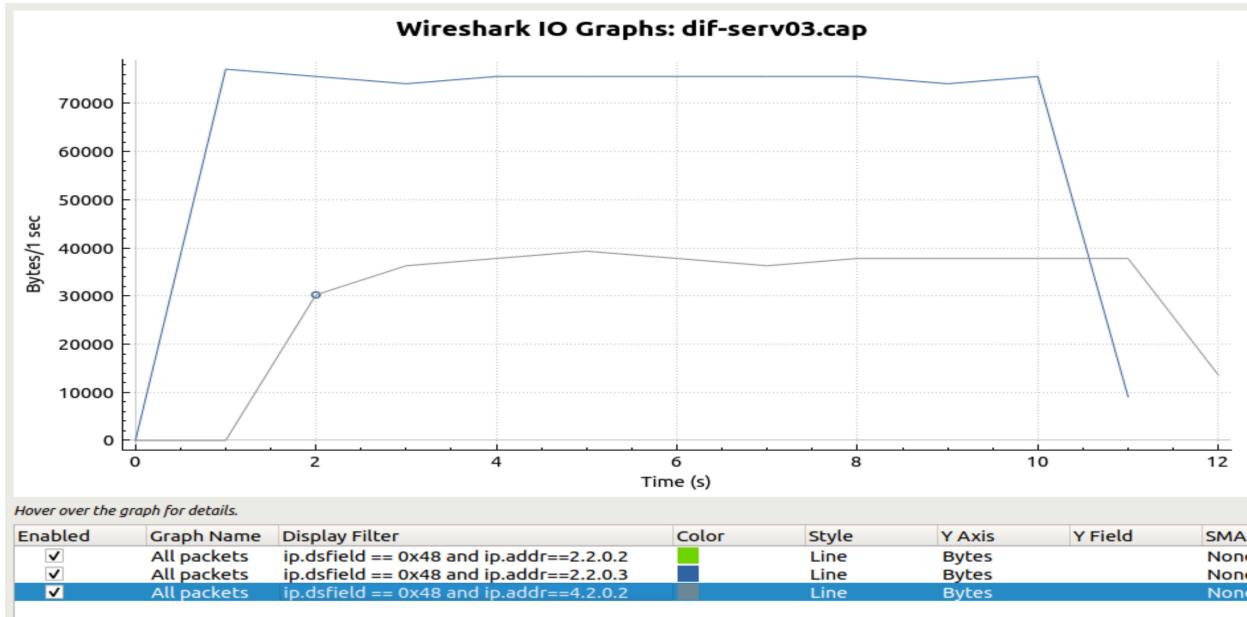
```
Match destination: ip.dst == x.x.x.x
Match source: ip.src == x.x.x.x
Match either: ip.addr == x.x.x.x
--The Archetypal Paul
```

Traffic de calidad1



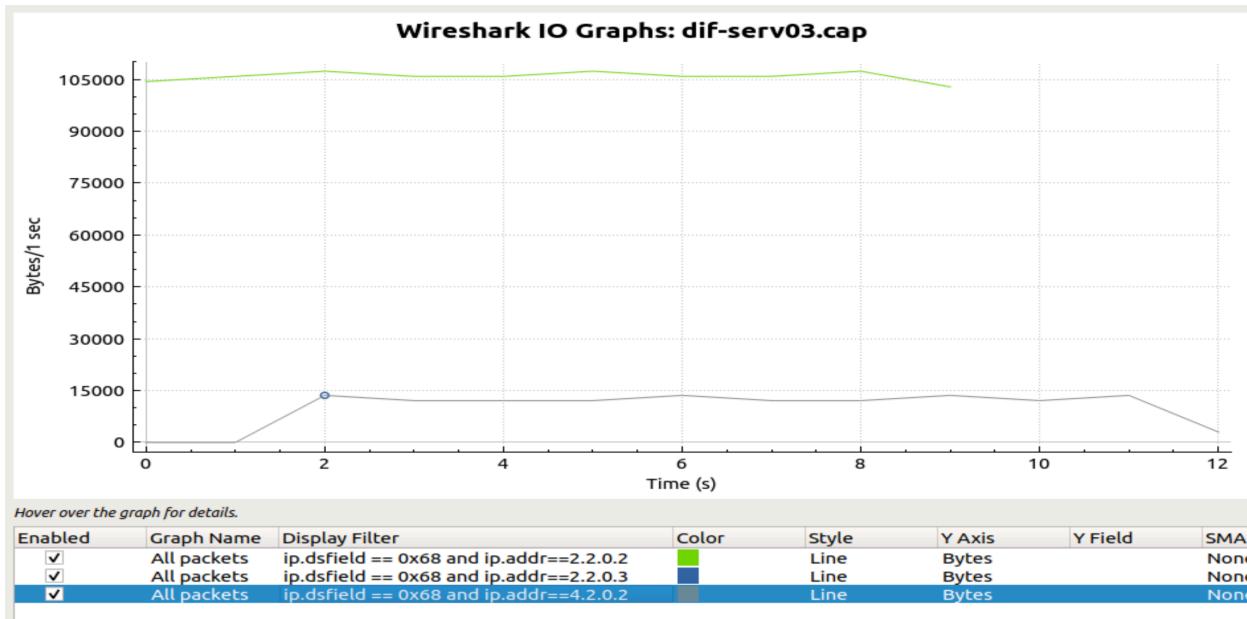
Trafico de calidad2

- Total
- Con origen en pc2.
- Con origen en pc3.



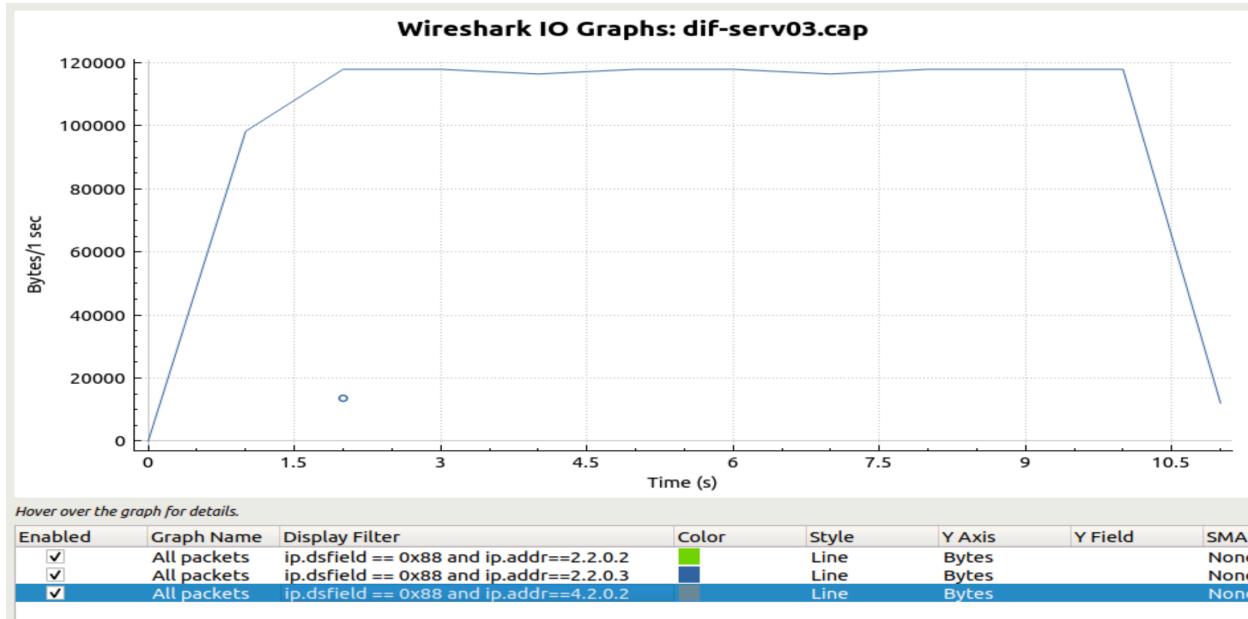
Trafico de calidad3

- Total
- Con origen en pc1.
- Con origen en pc3.



Trafico de calidad4

- Total
- Con origen en pc2.
- Con origen en pc3.



Explica los resultados obtenidos e incluye todas las gráficas que consideres necesarias en la memoria.

En los resultado de las gráficas Wireshark IO Graphs, utilice la captura de la red sub3, donde se muestran todos los paquetes de las demás subredes, obteniendo el máximo de Bytes/sec de cada IP con destino a la red sub4, teniendo un control de trafico de entrada, luego en el de salida aplique un DSCP marcando adecuadamente el campo DS, para luego realizar su tratamiento segú'n diferentes disciplinas de cola.

2. Tratamiento de tráfico en función del marcado DSCP

		DSCP (6bits)	DS (8 bits)	Prioridad de descarte de tráfico
Clase AF1	AF11	001 010	0010 1000 – 0x28	Baja
	AF12	001 100	0011 0000 – 0x30	Media
	AF13	001 110	0011 1000 – 0x38	Alta
Clase AF2	AF21	010 010	0100 1000 – 0x48	Baja
	AF22	010 100	0101 0000 – 0x50	Media
	AF23	010 110	0101 1000 – 0x58	Alta
Clase AF3	AF31	011 010	0110 1000 – 0x68	Baja
	AF32	011 100	0111 0000 – 0x70	Media
	AF33	011 110	0111 1000 – 0x78	Alta
Clase AF4	AF41	100 010	1000 1000 – 0x88	Baja
	AF42	100 100	1001 0000 – 0x90	Media
	AF43	100 110	1001 1000 – 0x98	Alta

		DSCP	DSCP HEX	DS (8 bits)
Case AF1	AF11	001 010	0x0A	0010 1000 0x28
	AF12	001 100	0x0C	0011 0000 0x30
	AF13	001 110	0x0E	0011 1000 0x38
Case AF2	AF21	010 010	0x12	0100 1000 0x48
	AF22	010 100	0x14	0101 0000 0x50
	AF23	010 110	0x16	0101 1000 0x58
Case AF3	AF31	011 010	0x1A	0110 1000 0x68
	AF32	011 100	0x1C	0111 0000 0x70
	AF33	011 110	0x1E	0111 1000 0x78
Case AF4	AF41	100 010	0x22	1000 1000 0x88
	AF42	100 100	0x24	1001 0000 0x90
	AF43	100 110	0x26	1001 1000 0x98

Mantén la configuración realizada en r1, r2.

Se establecen los siguientes parámetros de calidad dentro del router del núcleo diffServ (r3) para cada una de las calidades definidas. Configura HTB con ancho de banda 2.4Mbit para compartir entre todos los flujos con el siguiente patrón:

Calidad1: HTB 1Mbit como mínimo y 1Mbit como máximo.

Calidad2: HTB 500kbit como mínimo y 500kbit como máximo.

Calidad3: HTB 400kbit como mínimo y 400kbit como máximo.

Calidad4: HTB 200kbit como mínimo y 200kbit como máximo.

```
r3
GNU nano 2.0.7          File: htb.sh

tc qdisc del dev eth2 root
tc qdisc add dev eth2 root handle 1:0 dsmark indices 8 set_tc_index
tc filter add dev eth2 parent 1:0 protocol ip prio 1 \
    tcindex mask 0xfc shift 2
tc qdisc add dev eth2 parent 1:0 handle 2:0 htb
tc class add dev eth2 parent 2:0 classid 2:1 htb rate 2.4Mbit
tc class add dev eth2 parent 2:1 classid 2:10 htb rate 1Mbit ceil 1Mbit
tc class add dev eth2 parent 2:1 classid 2:11 htb rate 500kbit ceil 500kbit
tc class add dev eth2 parent 2:1 classid 2:12 htb rate 400kbit ceil 400kbit
tc class add dev eth2 parent 2:1 classid 2:13 htb rate 200kbit ceil 200kbit

#####1
#####1 trafico af11 => ds=x28 = 0010 1000 -0x28 => dscp = 001 010 = 0x0a
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x0a tcindex classid 2:10

#####
#####2
#####2 trafico af21 => ds=x48 = 0100 1000 -0x48 => dscp = 010 010 = 0x12
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x12 tcindex classid 2:11

#####
#####3
#####3 trafico af31 => ds=x68 = 0110 1000 -0x68 => dscp = 011 010 = 0x1a
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x1a tcindex classid 2:12

#####
#####4
#####4 trafico af41 => ds=x88 = 1000 1000 -0x88 => dscp = 100 010 = 0x22
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x22 tcindex classid 2:13
```

- Realiza un script para r3 donde se configure esta disciplina de cola según el marcado de los paquetes e incluye dicho script en la memoria.
- Inicia una captura (diffServ-04.cap) en la subred3 para que capture el tráfico que se genera en tu escenario por el envío "simultáneo" de:

Desde pc1: 2M a pc4

```
pc1:~# iperf -u -c 5.0.0.2 -b 2Mbit
```

Desde pc2: 1.5M a pc5

```
pc2:~# iperf -u -c 5.0.0.3 -b 1.5Mbit
```

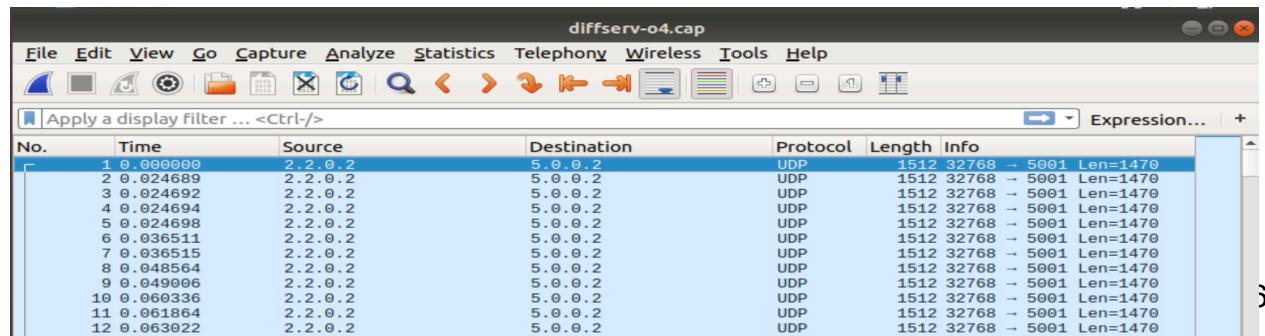
Desde pc3: 1M a pc6

```
pc3:~# iperf -u -c 5.0.0.4 -b 1Mbit
```

Espera al menos 2 minutos después de que haya terminado de enviarse el tráfico de pc1, pc2 y pc3 antes de interrumpir la captura de tráfico.

```
r4:~# tcpdump -i eth0 -w /hosthome/diffserv-o4.cap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
3818 packets captured
3818 packets received by filter
0 packets dropped by kernel
r4:~#
```

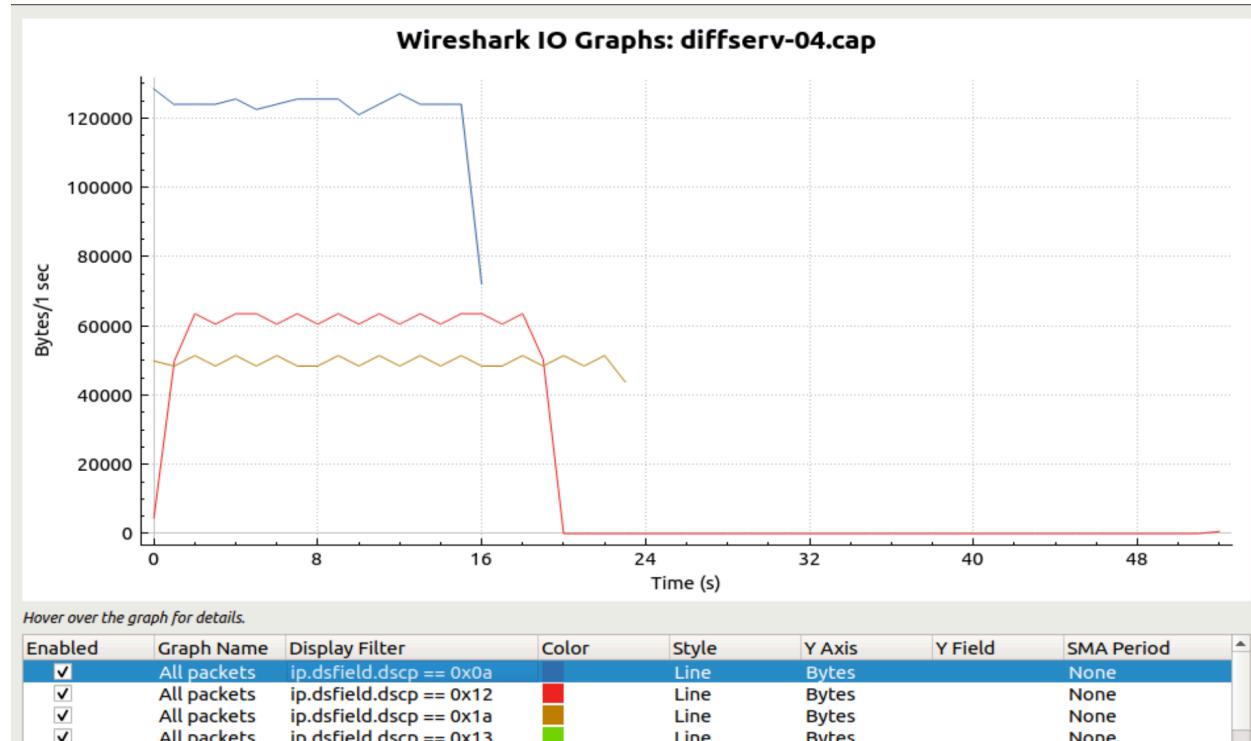
- Comprueba que el resultado es el esperado, es decir, el tráfico sigue el perfil indicado en las especificaciones anteriores. Para ello, consulta las gráficas IO graphs de Wireshark aplicando los filtros sobre las marcas DSCP de tal forma que se muestre cada calidad marcada de cada una de las fuentes incluyendo dichas imágenes en la memoria:



Explica los resultados obtenidos y explica si alguno de los flujos ha encolado tráfico para enviarlo posteriormente a los 10 segundos que dura la transmisión de iperf.

Los de mayor prioridad primero, después va con el del siguiente nivel.

Con el DSCP filtro



4. Modifica la configuración de HTB en r3 para que si algún flujo no esta utilizando el ancho de banda que tiene garantizado lo puedan usar el resto de flujos y vuelve a hacer una captura de tráfico (diffServ-05.cap) en la subred3. Explica qué modificaciones has tenido que hacer en el script.

```
tc qdisc del dev eth2 root
tc qdisc add dev eth2 root handle 1:0 dsmark indices 8 set_tc_index
tc filter add dev eth2 parent 1:0 protocol ip prio 1 \
    tcindex mask 0xfc shift 2
tc qdisc add dev eth2 parent 1:0 handle 2:0 htb
tc class add dev eth2 parent 2:0 classid 2:1 htb rate 2.4Mbit
tc class add dev eth2 parent 2:1 classid 2:10 htb rate 1Mbit ceil 2.4Mbit
tc class add dev eth2 parent 2:1 classid 2:11 htb rate 500kbit ceil 2.4Mbit
tc class add dev eth2 parent 2:1 classid 2:12 htb rate 400kbit ceil 2.4Mbit
tc class add dev eth2 parent 2:1 classid 2:13 htb rate 200kbit ceil 2.4Mbit

#####1
#####1 trafico af11 => ds=x28 = 0010 1000 -0x28 => dscp = 001 010 = 0x0a
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x0a tcindex classid 2:10

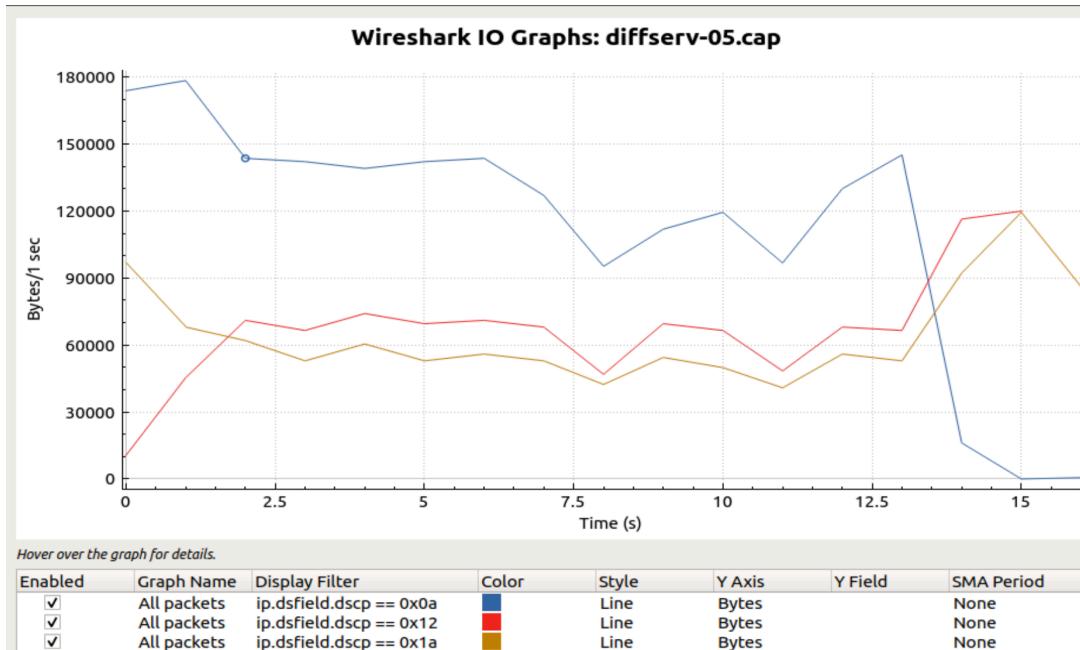
#####2
#####1 trafico af21 => ds=x48 = 0100 1000 -0x48 => dscp = 010 010 = 0x12
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x12 tcindex classid 2:11

#####3
#####1 trafico af31 => ds=x68 = 0110 1000 -0x68 => dscp = 011 010 = 0x1a
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x1a tcindex classid 2:12

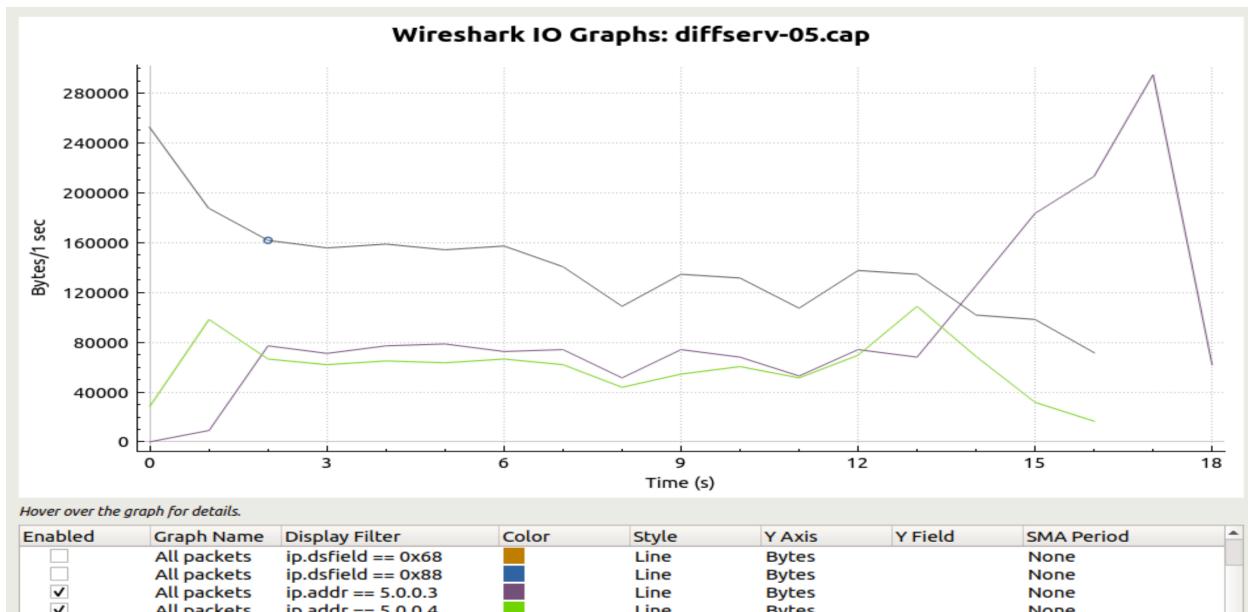
#####4
#####1 trafico af41 => ds=x88 = 1000 1000 -0x88 => dscp = 100 010 = 0x22
tc filter add dev eth2 parent 2:0 protocol ip prio 1 \
    handle 0x22 tcindex classid 2:13
```

Modifique el ceil, para no limitar el ancho de banda de cada filtro, así que cuando este en cola, este puede utilizar el ancho de banda que el otro no este utilizando por el momento.

5. Explica los resultados obtenidos e incluye las gráficas IO graphs que consideres necesarias.



En las gráficas obtenemos que dependiendo del filtro y como estén los demás en cola, este ocupara el max de ancho de banda disponible por el momento que este se este trasmitiendo.



3. Borrado del marcado DSCP

Se desea que una vez el paquete salga de la red DiffServ, en r4, éste no incluya la marca DSCP.

1. Realiza un script para r4 donde se elimine la marca DSCP de todos los paquetes que salen de la red diffServ. Incluye dicho script en la memoria.
2. Utilizando la misma configuración de r1, r2 y r3 que en la sección 1.2, apartado 1, y el mismo patrón de tráfico enviado que el de la sección 1.2, apartado 2, realiza una captura de tráfico (diffServ-06.cap) en la subred4.
3. Explica los resultados obtenidos e incluye las gráficas IO graphs que consideres necesarias.