Sistemas Telemáticos para Medios Audiovisuales
Práctica 5: Calidad de Servicio en Linux

Departamento de Teoría de la Señal y Comunicaciones
y Sistemas Telemáticos y Computación
(GSyC)

Noviembre de 2018

1. Control de tráfico

Descomprime el fichero que contiene el escenario de NetGUI lab-tc.tgz para realizar la práctica de control de tráfico en Linux.

1.1. Sin control de tráfico ni a la entrada ni a la salida

El *router* `r1` no tiene activado el control de tráfico en ninguna de sus interfaces.

1.1.1. Un flujo de datos

Arranca `iperf` en modo servidor UDP en `pc3` y arranca `iperf` en modo cliente UDP en `pc1` para que envíe tráfico a 3 Mbit durante 10 segundos a `pc3`.

Observa en el lado servidor, el informe del tráfico recibido en el sentido `pc1` → `pc3`.

1.1.2. Dos flujos de datos

- Arranca `iperf` en modo servidor UDP en `pc4`.
- Arranca otro `iperf` en modo servidor UDP en `pc3`.
- Inicia una captura de tráfico en la interfaz `eth1` de `r1`.
- Escribe (todavía sin ejecutar) el comando que arranca `iperf` en modo cliente UDP en `pc1` para que envíe 3 Mbit al servidor `pc3` en el sentido `pc1` → `pc3` durante 10 segundos.
- Escribe (todavía sin ejecutar) el comando que arranca `iperf` en modo cliente UDP en `pc2` para que envíe 3 Mbit al servidor `pc4` en el sentido `pc2` → `pc4` durante 10 segundos.
- Ejecuta los dos comandos anteriores uno a continuación de otro (lo más rápidamente que puedas) para que su ejecución se realice de forma simultánea.
- Interrumpe la captura aproximadamente 10 segundos después de que arrancaras `iperf`.

A continuación analiza los resultados obtenidos:

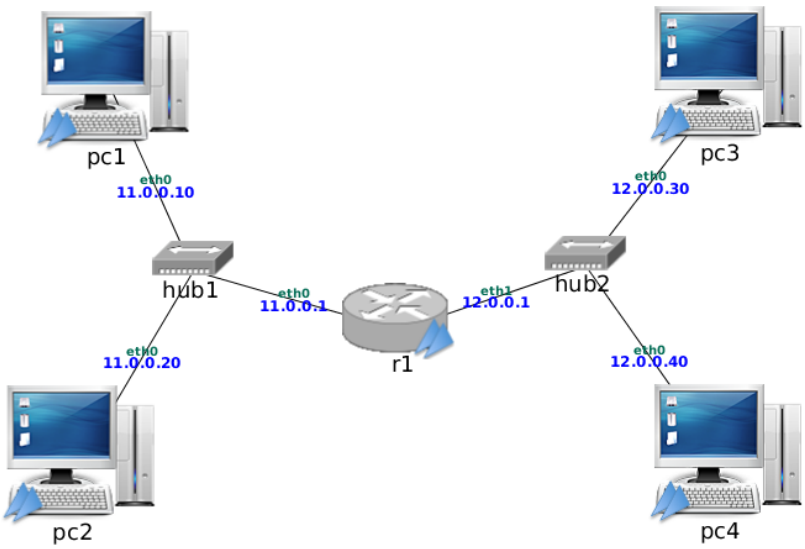

Figura 1: Escenario para control de tráfico.

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en `wireshark` y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

1.2. Control de admisión para el tráfico de entrada

Vamos a configurar `r1` para restringir el tráfico de entrada para 2 flujos de datos que recibe `r1`:

- Flujo 1: origen 11.0.0.10 se va a restringir a una velocidad de 1Mbit y una cubeta de 10k.
- Flujo 2: origen 11.0.0.20 se va a restringir a una velocidad de 2Mbit y una cubeta de 10k.
- Utiliza `tc` para definir esta configuración en la interfaz `eth0` de `r1` que es la interfaz de entrada de `r1` para los flujos 1 y 2. Ten en cuenta que se aplique primero el filtro del flujo número 1 y después el del número 2. Guarda esta configuración en un fichero de *script*, por ejemplo con el nombre `tc-ingress.sh` que deberá contener las instrucciones que ejecutarías en la línea de comandos:

```
#!/bin/sh

# Esto es un comentario

echo "Borrando la disciplina de cola ingress en la interfaz eth0"
tc qdisc del dev eth0 ingress

echo "Creando la disciplina de cola ingress en la interfaz eth0"
tc qdisc add ...
...
```

Una vez creado el *script* debes darle permisos de ejecución con la orden:

```
chmod 755 tc-ingress.sh
```

Y por último, para ejecutarlo, debes escribir[1]:

```
./tc-ingress.sh
```

- Inicia una captura de tráfico en la interfaz `eth1` de `r1`.
- Arranca dos clientes y 2 servidores tal y como lo hiciste en el apartado 1.1.2.
- Interrumpe la captura aproximadamente 10 segundos después de que arrancaras `iperf`, cuando los servidores hayan terminado de recibir todo el tráfico.

A continuación analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en `wireshark` y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

1.3. Disciplinas de colas para el tráfico de salida

1.3.1. Token Bucket Filter (TBF)

Mantén la configuración del tráfico de entrada en `r1` que has realizado en el apartado anterior en el *script* `tc-ingress.sh`.

- Define en `r1` para su interfaz `eth1` una disciplina TBF de salida con ancho de banda 1.5 Mbit, latencia 10 ms y tamaño de cubeta 10k y guarda la configuración en un nuevo *script* `tc-egress-tbf.sh`.
- Inicia una captura de tráfico en la interfaz `eth1` de `r1`.
- Arranca dos clientes y 2 servidores tal y como lo hiciste en el apartado 1.1.2.
- Interrumpe la captura aproximadamente 10 segundos después de que arrancaras `iperf`.

A continuación analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en `wireshark` y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

[1]Si es la primera vez que ejecutas este *script*, el *router* no tendrá configurada ninguna disciplina de cola en esa interfaz y al usar la instrucción de borrado se mostrará un error, pero la ejecución del *script* continuará y se aplicarán el resto de los cambios que hayas configurado.

Modifica la configuración de TBF de salida para que ahora tenga una latencia de 20 segundos y realiza la misma prueba que antes [2]. Interrumpe la captura al menos cuando hayan pasado 20 segundos desde que comenzaste a enviar tráfico desde los clientes. A continuación analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en `wireshark` y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

1.3.2. Prioridad (PRIO)

Mantén la configuración del tráfico de entrada en `r1` que has realizado en el apartado anterior en el *script* `tc-ingress.sh`. Borra la disciplina de cola de salida configurada en la interfaz `eth1` de `r1`.

La configuración TBF en el apartado 1.3.1 permite gestionar el ancho de banda de salida para que no supere el valor configurado, en nuestro caso 1.5Mbit. Toma como punto de partida esta configuración para que ahora se atienda el tráfico de salida según diferentes prioridades, configurando una disciplina de cola con prioridad que sea hija de la disciplina TBF.

- Escribe un *script* en `r1`, `tc-egress-prio.sh`, para configurar TBF con los siguientes parámetros: ancho de banda 1.5Mbit, cubeta 10k y latencia 20s. Crea una disciplina de cola hija con prioridad de tal forma que se asignen las siguientes prioridades:
  - Prioridad 1 (más prioritario): tráfico de la dirección IP origen 11.0.0.10
  - Prioridad 2 (prioridad intermedia): tráfico de la dirección IP origen 11.0.0.20
  - Prioridad 3 (menos prioritario): no lo vamos a definir.
- Inicia una captura de tráfico en la interfaz `eth1` de `r1`.
- Arranca 2 servidores para recibir los dos flujos de datos tal y como se hizo en el apartado 1.1.2.
- Arranca dos clientes y 2 servidores tal y como lo hiciste en el apartado 1.1.2.
- Interrumpe la captura aproximadamente 35 segundos después de que arrancaras `iperf`.

A continuación analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en wireshark y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

[2] Ten en cuenta que ahora el tráfico quedará en la cola de la disciplina TBF esperando a ser cursado según el ancho de banda que hemos configurado. El cliente terminará de enviar a los 10 segundos y esperará a recibir el informe del servidor. Sin embargo, el servidor no acabará de recibir (y por tanto no enviará el informe) hasta que TBF no termine de atender el tráfico de la cola de salida, que será más de 10 segundos. Al no recibir el cliente el informe del servidor, terminará imprimiendo un `Warning`. De la misma forma cuando el servidor haya terminado de recibir y envíe el informe al cliente, éste ya habrá terminado su ejecución e imprimirá un mensaje indicando que no ha podido enviar el informe al cliente: `Connection refused`.

1.3.3. Hierarchical token Bucket (HTB)

Mantén la configuración del tráfico de entrada en `r1` que has realizado en el apartado anterior en el *script* `tc-ingress.sh`. Borra la disciplina de cola de salida configurada en la interfaz `eth1` de `r1`.

- Escribe un *script* en `r1`, `tc-egress-htb.sh`, para configurar en su interfaz `eth1` una disciplina HTB de salida con ancho de banda 1.2 Mbit. Reparte el ancho de banda de esta interfaz de salida de la siguiente forma:
  - 700 kbit para el tráfico con origen en `pc1`, `ceil` 700kbit.
  - 500 kbit para el tráfico con origen en `pc2`, `ceil` 500kbit.
- Inicia una captura de tráfico en la interfaz `eth1` de `r1`.
- Arranca 2 servidores para recibir los dos flujos de datos tal y como se hizo en el apartado 1.1.2.
- Arranca dos clientes y 2 servidores tal y como lo hiciste en el apartado 1.1.2.
- Interrumpe la captura aproximadamente 35 segundos después de que arrancaras `iperf`.

A continuación analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en wireshark y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

Modifica la configuración de `ceil` en cada uno de los flujos para que puedan utilizar 1.2Mbit. Realiza la misma prueba que antes y analiza los resultados obtenidos:

1. Explica las estadísticas que muestran los servidores.
2. Carga la captura en wireshark y muestra cada uno de los flujos de forma gráfica. Explica el ancho de banda medido para cada uno de los flujos.

2. DiffServ

Descomprime el fichero que contiene el escenario de NetGUI lab-diffServ.tgz para realizar la práctica de diffServ en Linux.

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

En el escenario de la figura 2 se va a configurar la red para que el tráfico desde `pc1`, `pc2` y `pc3` envíen paquetes a `pc4`, `pc5` y `pc6` atravesando una red diffServ. Se distinguirán 4 calidades diferentes con los códigos: EF, AF31, AF21 y AF11.

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:

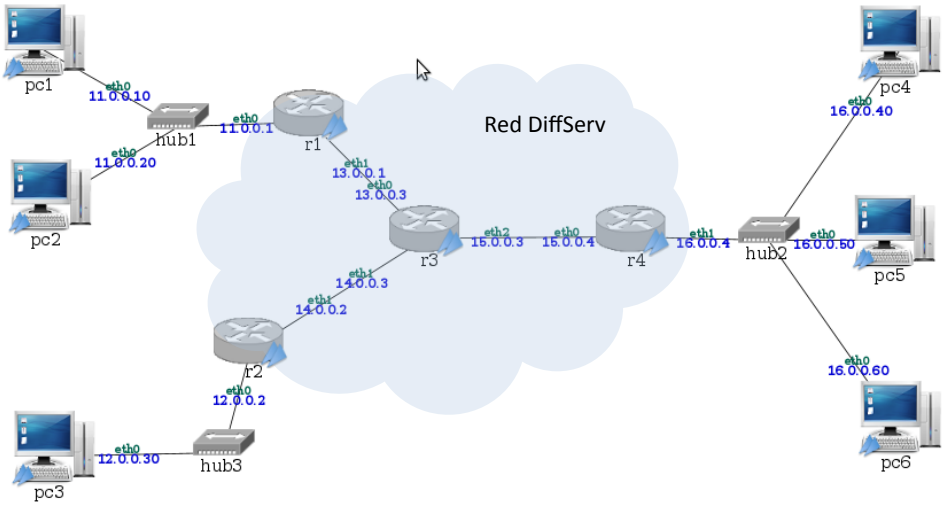

Figura 2: Escenario para diffServ.

- Flujo 1: máximo 1.2mbit con ráfaga 10k para el tráfico dirigido a `pc4`, marcado con calidad EF. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 2.
- Flujo 2: máximo de 300kbit y ráfaga 10k, marcado con calidad AF31. 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 3: máximo 600kbit con ráfaga 10k para el tráfico dirigido a `pc5`, marcado con calidad AF21. 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 calidad AF11. Si se supera este ancho de banda, el tráfico será descartado definitivamente en `r1`.

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`, descartando el tráfico sobrante:
  - Flujo 5: máximo 400kbit con ráfaga 10k dirigido a `pc6`, marcado con calidad AF31. 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 calidad AF21. Si se supera este ancho de banda, el tráfico quedará clasificado dentro del flujo 7.
  - Flujo 7: máximo 100kbit con ráfaga 10k, marcado con calidad AF11. Si se supera este ancho de banda, el tráfico será descartado definitivamente en `r2`.

Realiza un *script* para `r1` y otro para `r2` donde se configuren estos perfiles de tráfico.

1. Ejecuta en tu escenario el envío "simultáneo" de:
   - Desde el `pc1` 2Mbit a `pc4`

- Desde el `pc2` 1.5Mbit a `pc5`
- Desde el `pc3` 1Mbit a `pc6`

2. Realiza una captura en la subred 15.0.0.0/24 y 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.
3. 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:
   - Tráfico de calidad EF
   - Tráfico de calidad AF31
     - Total
     - Con origen en `pc1`.
     - Con origen en `pc3`.
   - Tráfico de calidad AF21
     - Total
     - Con origen en `pc2`.
     - Con origen en `pc3`.
   - Tráfico de calidad AF11
     - Total
     - Con origen en `pc2`.
     - Con origen en `pc3`.

   Explica los resultados obtenidos.

2.2. Tratamiento de tráfico en función del marcado DSCP

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:

- Calidad EF: HTB 1mbit como mínimo y 1Mbit como máximo.
- Calidad AF31: HTB 500kbit como mínimo y 500kbit como máximo.
- Calidad AF21: HTB 600kbit como minimo y 600kbit como máximo.
- Calidad AF11: HTB 200kbit como mínimo y 200kbit como máximo.

Realiza un *script* para `r3` donde se configure esta disciplina de cola según el marcado de los paquetes.

1. Ejecuta en tu escenario el envío "simultáneo" de:
   - Desde el `pc1` 2Mbit a `pc4`

- Desde el `pc2` 1.5Mbit a `pc5`
- Desde el `pc3` 1Mbit a `pc6`

2. Realiza una captura en la subred 15.0.0.0/24, espera al menos 1 minuto después de que el haya terminado de enviarse el tráfico de `pc1`, `pc2` y `pc3` y comprueba que el resultado es el esperado, es decir, el tráfico sigue el perfil indicado en las especificaciones anteriores.

3. 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:

   - Tráfico de calidad EF
   - Tráfico de calidad AF31
   - Tráfico de calidad AF21
   - Tráfico de calidad AF11

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

Modifica la configuración de HTB en `r3` para que si algún flujo no está utilizando el ancho de banda que tiene garantizado lo puedan usar el resto de flujos y vuelve a hacer la misma captura de tráfico en la subred 15.0.0.0/24. Explica los resultados obtenidos.