Trabajo Práctico

TP 5 - Calidad de Servicio (QoS) / Traffic Shaping

Fecha de entrega: 28/10/2021

Franco, Juan Martín 149.615

juanmartin_franco@hotmail.com

Experiencia en el laboratorio

Se realizarán pruebas con utilidades estándar de Linux para apreciar el efecto que causan algunas características no deseadas (pérdidas, delay, reordenamiento) que suelen observarse en redes WAN.

El router intermedio entre estos equipos implementará el emulador netem en su interfaz de salida hacia el host receptor.

En primera instancia, se debe descargar el laboratorio 'QoS' para Netkit desde este Link y ubicarlo ~/netkit/labs/netkit-lab_qos.tar.gz .

Luego se debe descomprimir en ~/netkit/netkit-lab_qos para su uso.

La red del laboratorio es la siguiente:

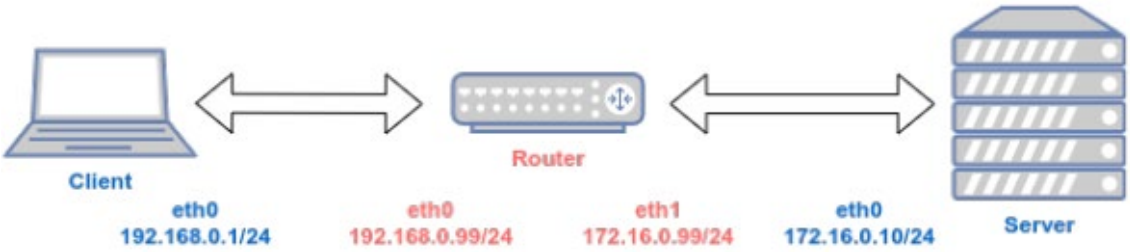
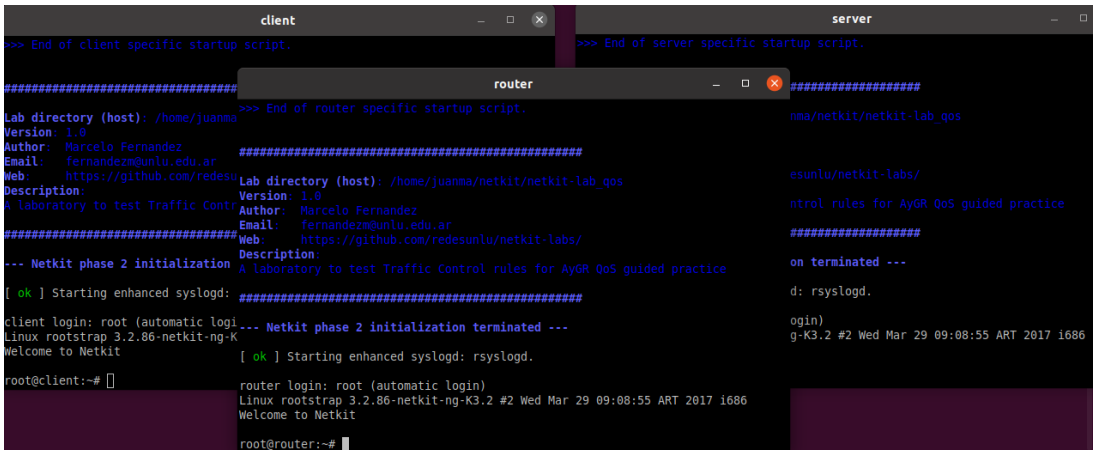


Figura 1: Topología laboratorio Netkit

1. Posicionarse en el directorio ~/netkit/netkit-lab_qos/
2. Lanzar el laboratorio con el comando lstart .

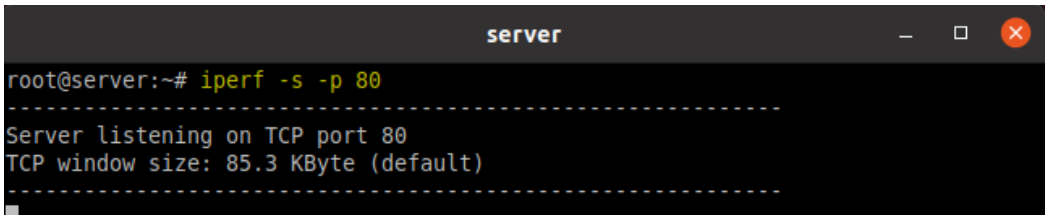
Una vez posicionado sobre la carpeta del laboratorio, ejecuto el comando

```
lstart
```

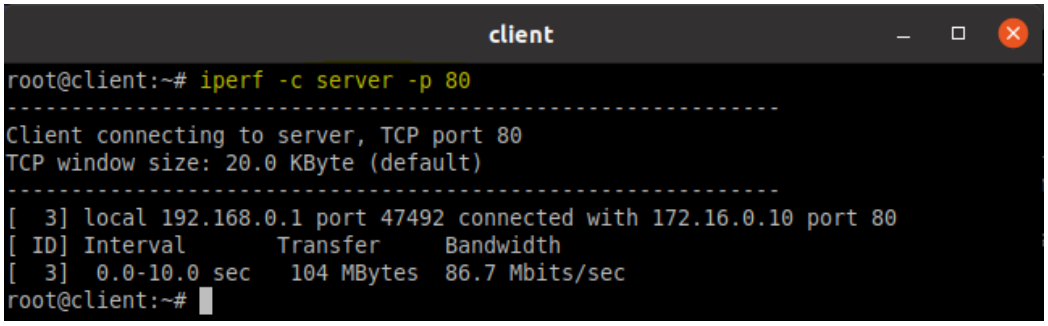


3. Evaluar tasa de transferencia por defecto.

En el server, ejecutar: iperf -s -p 80



En el cliente, ejecutar: iperf -c server -p 80

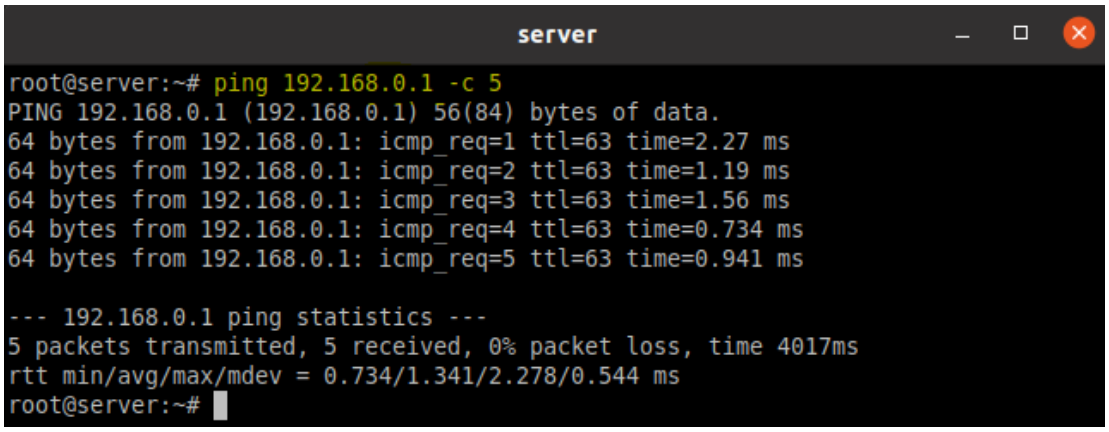


Con eso se tiene una idea de la tasa de transferencia que hay entre cliente y servidor por medio del router.

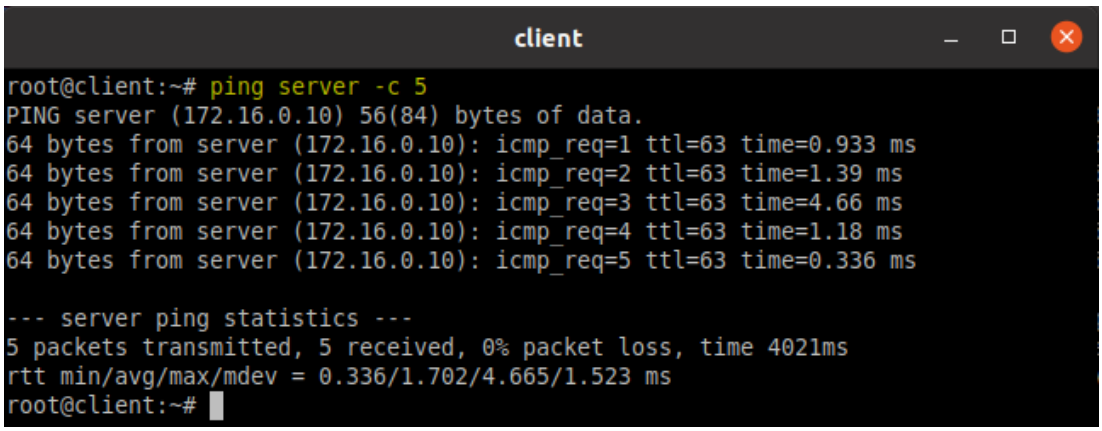
La tasa de transferencia entre el cliente y el servidor por medio del router es de 86.7 Mbits/sec.

4. Evaluar latencia (RTT) por defecto.

En el server, ejecutar: ping 192.168.0.1



En el cliente, ejecutar: ping server



Con eso se tiene una idea de la latencia que hay entre cliente y servidor por medio del router.

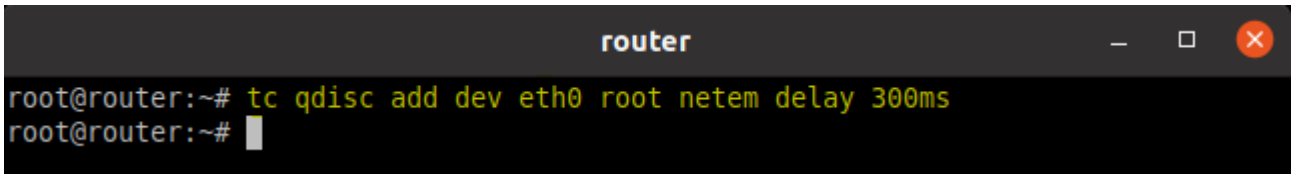
5. En el router, utilice la disciplina de encolado netem para agregar artificialmente latencia y/o pérdida de paquetes. Tras cada comando, vuelva a evaluar con ping las características de la red entre cliente y servidor y viceversa, y determine si se condicen con lo establecido mediante netem.

 # para agregar latencia en el enlace saliente (outbound)

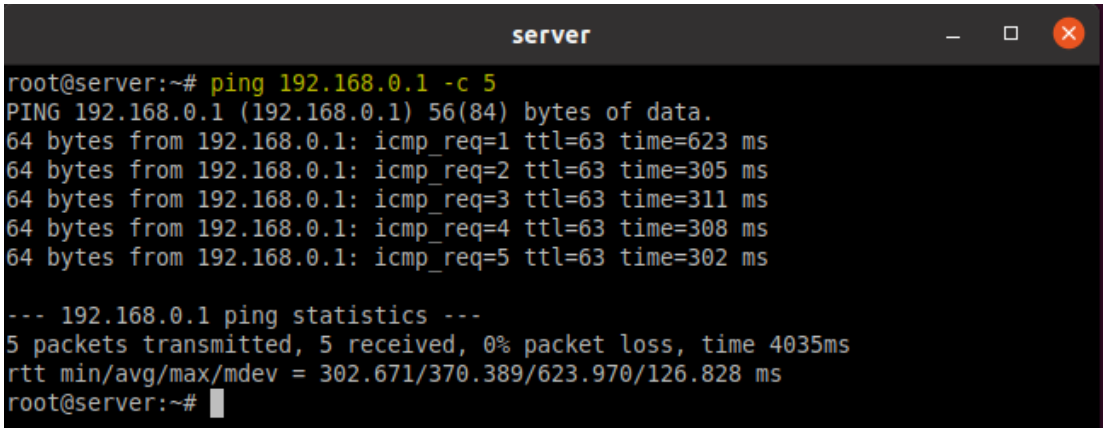
tc qdisc add dev eth0 root netem delay 300ms

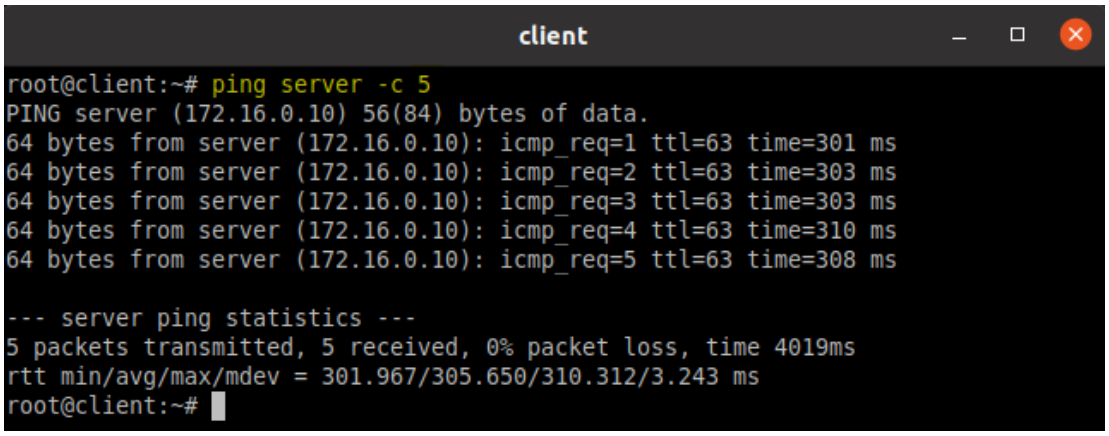
Para agregar latencia en el enlace saliente ejecuto el comando:

```
tc qdisc add dev eth0 root netem delay 300ms
```



Ahora, pruebo realizar ping entre cliente y servidor nuevamente:





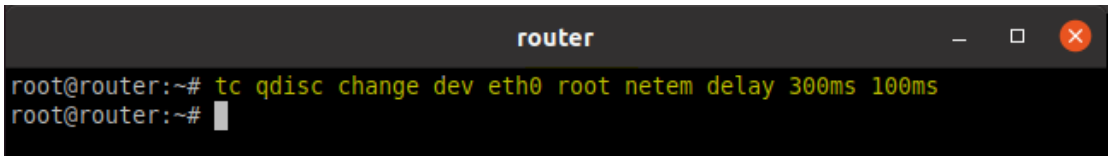
Efectivamente, se puede apreciar la diferencia notable entre los ms de los últimos pings y los ms de los primeros pings.

 # para agregar jitter variable

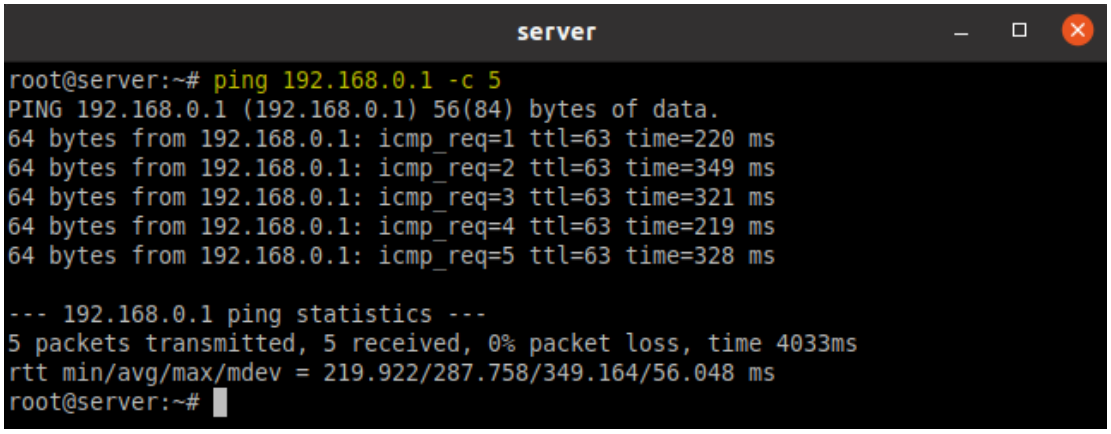
tc qdisc change dev eth0 root netem delay 300ms 100ms

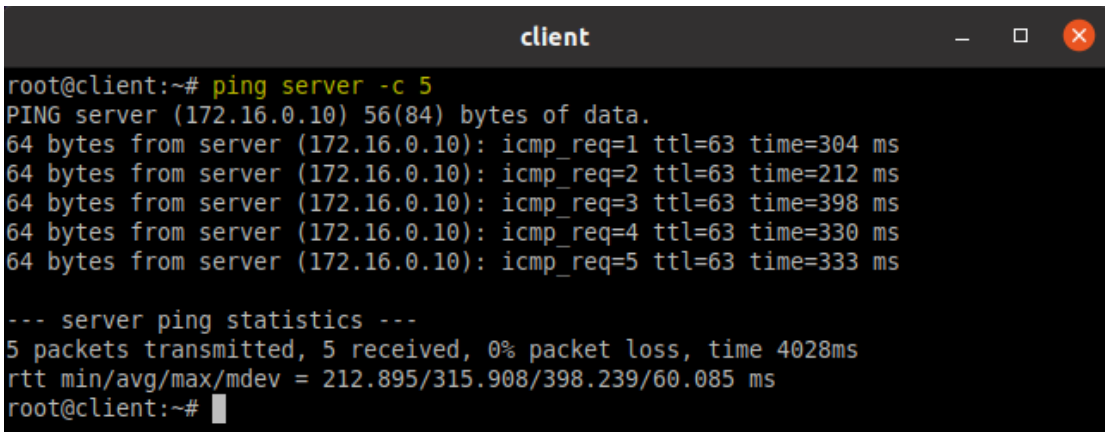
Ahora, para agregar jitter variable ejecuto el comando:

```
tc qdisc change dev eth0 root netem delay 300ms 100ms
```



Nuevamente, realizo pings entre cliente y servidor para corroborar el jitter:





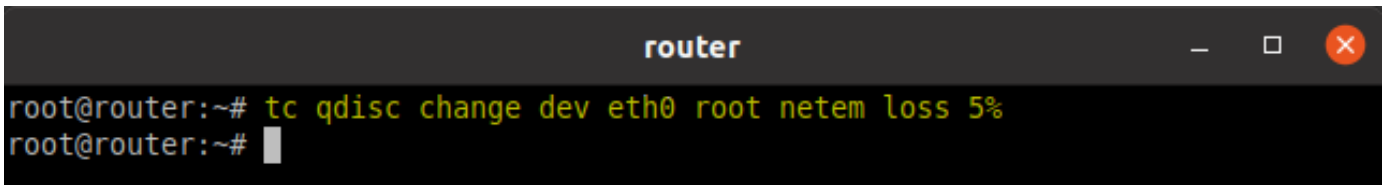
Se puede apreciar la fluctuación del retardo.

 # para simular perdida de paquetes

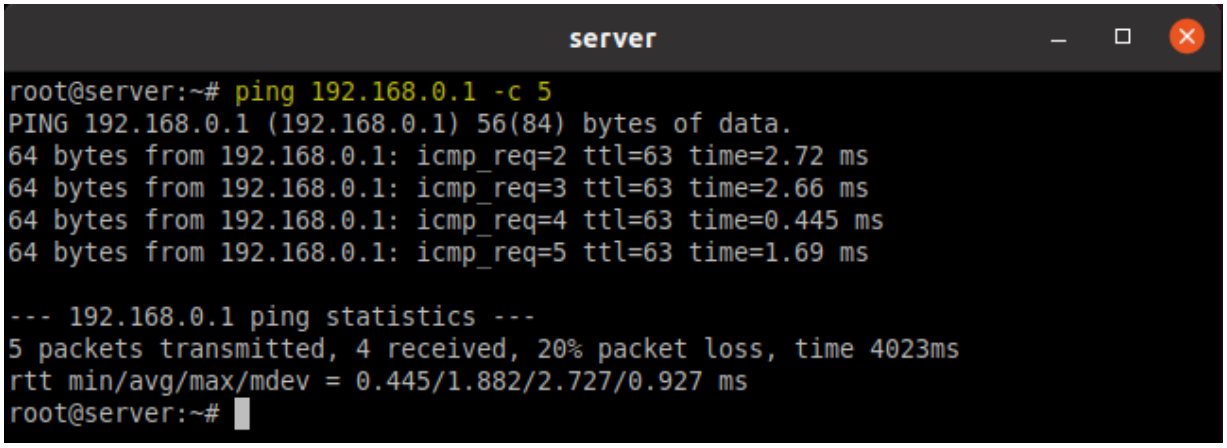
tc qdisc change dev eth0 root netem loss 5%

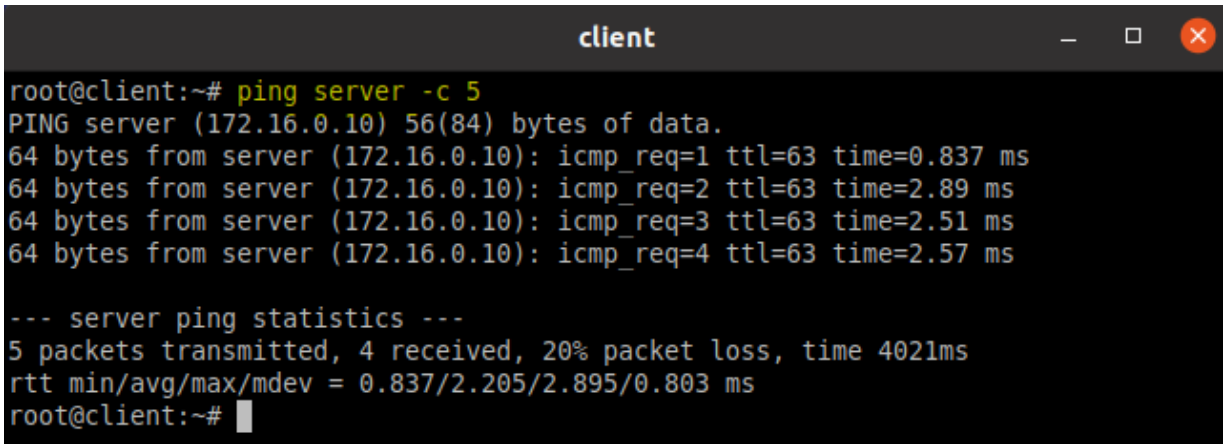
Por último, para simular la pérdida de paquetes, utilizo el comando:

```
tc qdisc change dev eth0 root netem loss 5%
```



Por último, realizo los pings entre cliente y servidor para verificar cómo se comportan con el comando que simula la pérdida de paquetes:





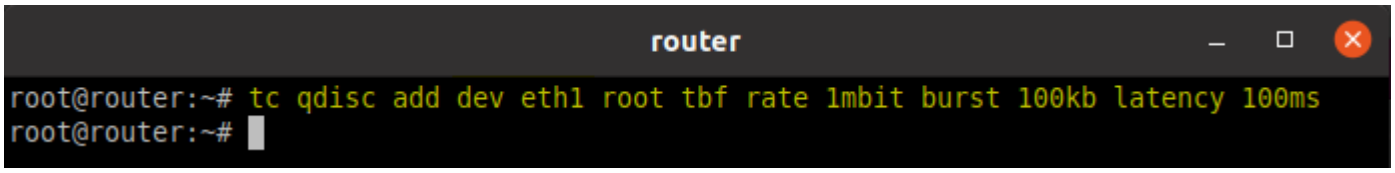
Efectivamente, se puede visualizar que en ambos casos, se perdió un paquete de los 5 enviados, lo que implica un 20% de pérdida de paquetes.

Recordar que al aplicarlo sobre la interfaz eth0 del router, este tendrá efecto sobre la interfaz que apunta la "LAN", o sea, el cliente.

6. Sobre la misma topología, utilizamos TBF (Token Bucket Filter), una disciplina de encolado filtro classless con tc para aplicar una política de control de tasa de transferencia de 1Mb/s:

 # Limitamos la interfaz eth1 del router a 1Mbps, balde de 100 KB

tc qdisc add dev eth1 root tbf rate 1mbit burst 100kb latency 100ms



El comando utilizado me permite limitar la interfaz eth1 (definida posterior al parámetro dev) utilizando la disciplina Token Bucket Filter (por eso el TBF).

Se recomienda que para la realización de este ejercicio se trabaje con una máquina virtual (Virtualbox, VMWare, etc.) con alguna distribución de Linux (Ubuntu, Debian, etc.), de manera tal de poder instalar los paquetes mencionados (git, graphviz, python3, etc.). De paso, dicha VM podría servir para los TPs "a distancia" que se necesiten en el futuro.

Consignas a resolver

1. Calcule la tasa de transferencia mínima requerida en bps (o múltiplos) y en bytes/seg para las siguientes aplicaciones. Contemple la tasa de bits en crudo, sin compresión alguna y omita el overhead de capas inferiores. ⌂

a. Aplicación de voz sobre IP punto a punto, donde se transfiere sonido en un solo canal (mono) con una frecuencia de muestreo de 8000 Hz (muestras por segundo) y cada valor es un entero de 8 bits.

Frecuencia de muestreo: 8000Hz (8000 muestras por segundo)

Suponiendo que transfiero 1 segundo de audio → 8000 muestras por c/segundo que pasa

Sabiendo que cada muestra pesa 8 bits → 8000 muestras x 8 bits = 64000bps

Esto equivale a 64kbits/seg.

b. Estación meteorológica remota que recopila valores de temperatura, humedad, dirección y velocidad del viento, posición (gps) y remite dichos valores en una PDU a nivel de aplicación de 246 bytes sobre procolo TCP con una periodicidad de 5 segundos. Contemple aquí el overhead de capas inferiores.

Para resolver esto, debo encontrar una tasa de transferencia mínima la cual me permita enviar la PDU completa (es decir, los 246 bytes de aplicación + los headers restantes) en 5 segundos o menos, ya que en caso contrario se estarían solapando.

Para lograr esto, primero calculo el tamaño total de la PDU:

Header Ethernet = 14 bytes.

Header IP = 20 bytes.

Header TCP = 20 bytes.

Nivel de aplicación = 246 bytes.

Total PDU = 300 bytes → Pasado a bits → 300 x 8 = 2400 bits.

Estos 2400 bits deben poder transmitirse en 5 segundos, por lo que la tasa mínima es:

2400 bits por segundo / 5 segundos = 480bps.

c. Aplicación de transmisión de sonido punto a punto con una frecuencia de muestreo de 44100 Hz, donde cada valor de la muestra es un entero entre 0 y 65.535 y se envían dos canales de sonido en simultáneo (izquierdo y derecho).

Frecuencia de muestreo: 44100Hz (44100 muestras por segundo).

Suponiendo que transfiero 1 segundo de audio → 44100 muestras por c/segundo que pasa.

Sabiendo que cada muestra “pesa” entre 0 y 65535 bits, esto me indica que son 16 bits por cada muestra.

44100 (muestras por segundo) x 16 (tamaño de muestra) = 705,6Kbps

Esto equivale a 705600bps y, como tengo 2 canales de sonido en simultaneo (estéreo):

705,6Kbps x 2 canales = 1411,2Kbps = 1.411.200bps

d. Cámara IP de videovigilancia con una resolución SIF (352 x 288 pixeles) a 25 cuadros por segundo (fps) de video en escala de grises.

Resolución SIF = 352 x 288 = 101.376 pixeles.

Como el video es en escala de grises, cada pixel se representa con 8 bits.

101.376 (pixeles) x 8 (bits) = 811.008bits

Es decir, un solo fotograma por segundo necesitaría 811.008bits como mínimo.

Como se necesita transmitir 25 fotogramas por segundo:

811.008 (bits por fotograma) x 25 (cantidad de fotogramas por segundo) = 20.275.200bits, su equivalente en bytes = 20.275.200 / 8 = 2.534.400bytes.

e. Aplicación de broadcast (radio en línea), con una frecuencia de muestreo de 22050 Hz, resolución de 16 bits, sonido estéreo y hasta 10 receptores posibles en simultáneo.

Frecuencia de muestreo = 22050Hz (22050 muestras por segundo).

Suponiendo que transfiero 1 segundo = 22050 muestras por c/segundo que pasa.

Sabiendo que cada muestra “pesa” 16 bits:

22050 (muestras por segundo) x 16 (tamaño de muestra) = 352,8Kbps

Como tengo sonido estéreo, es decir, 2 canales:

352,8Kbps x 2 (cantidad de canales) = 705,6Kbps

Como tengo hasta 10 receptores posibles en simultáneo:

705,6Kbps x 10 (cantidad de receptores) = 7056Kbps (0.7056 Mbps).

f. Aplicación de videollamada simultánea con resolución de video SIF en color de 24 bits (RGB) y sonido con calidad de CD.

Sonido con calidad de CD:

Frecuencia de muestreo = 44,1Khz = 44100Hz (44100 muestras por segundo).

Sabiendo que cada muestra “pesa” 16 bits:

44100 (muestras por segundo) x 16 (tamaño de muestra) = 705,6Kbps = 722.534,4bps

Video con resolución SIF en color RGB:

Como la resolución es SIF (352 x 288):

352 x 288 = 101.376 pixeles.

Al ser video en color, cada pixel es representado por 24bits, por lo que:

101.376 (pixeles) x 24 (bits para representar un pixel) = 2.433.024bits

Ahora, debo sumar la cantidad de bits que necesito para el video + la cantidad de bits necesarias para el audio:

2.433.024 (bits necesarios para el video) + 722.534,4 (bits necesarios para el audio) =

3.155.558,4 bits por segundo = 24.075Mbits por segundo

g. Servidor de almacenamiento de sonido en un estudio de grabación, almacenando audio a una frecuencia de muestreo de 96 Khz, resolución de 24 bits y 16 canales en simultáneo.

Frecuencia de muestreo: 96KHz = 96000Hz (96000 muestras por segundo)

Suponiendo que transfiero 1 segundo = 96000 muestras por c/segundo que pasa.

Sabiendo que cada muestra “pesa” 24 bits:

96000 (muestras por segundo) x 24 (tamaño de muestra) = 2304Kbps

Al tener 16 canales en simultáneo:

2304Kbps x 16 (canales en simultáneo) = 36.864Kbps

h. Streaming de audio y video digital punto a punto en tiempo real, sin compresión, con resolución de video Resolución Full HD en color RGB (3 x 8 bits) a 60 frames por segundo y audio en 44.1 Khz, 5.1 canales.

Primero, calculo la cantidad de bits necesarios para el sonido:

Frecuencia de muestreo: 44.1Khz = 44100Hz (44100 muestras por segundo)

En tiempo real, transfiero cada segundo = 44100 muestras por c/segundo que pasa.

Sabiendo que cada muestra “pesa” 24 bits:

44100 (muestras por segundo) x 24 (tamaño de muestra) = 1058,4Kbps

Como es audio 5.1:

1058,4 x 6 (cantidad de canales) = 6350,4Kbps = 6.502.809,6 bits

Ahora, continuando con el video:

Resolución Full HD = 1.920 x 1.080 = 2.073.600 pixeles

Como el Streaming es en color RGB, para cada pixel necesito 24bits:

2.073.600 (pixeles) x 24 (bits por cada pixel) = 49.766.400bits

Como la cantidad de frames por segundo es 60:

49.766.400 (bits por fotograma) x 60 (cantidad de fotogramas por segundo) = 2.985.984.000 bits necesarios para el video.

Ahora, sumando ambos valores:

2.985.984.000 (bits para el video) + 6.502.809,6 (bits para el audio) = 2.992.486.809,6 bits por segundo = 2.7869Gbits por segundo como mínimo

Corolario: los valores que surgen de estos cálculos corresponden a flujos en crudo. En las aplicaciones modernas se utilizan codecs de audio y video que optimizan la comunicación, reduciendo notablemente la tasa de transferencia necesaria para las mismas. Sobre esta temática se ampliará en la clase referida a Voz sobre IP.

2. Considerando las siguientes aplicaciones: ⌂

a. Videojuego con multijugador online. Ejemplo: Overwatch, Call of Duty.

b. Aplicación de servicio de archivos en la nube: Ejemplo Google Drive, Dropbox

c. Streaming de video. Ejemplo: Netflix, Flow.

d. Servicio de DNS.

e. Servicio de mensajería instantánea. Ejemplo: WhatsApp, Messenger.

f. Repositorio FTP de imágenes ISO.

Piense en cómo las aplicaciones se ven afectadas por éstas variables (tasa de transferencia baja/alta, delay, pérdida, duplicación y reordenamiento), así también como por otras de las métricas vistas en clase.

A partir de allí realice, mediante una tabla, una comparación donde establezca cuáles parámetros o características deberían exigirse para tener una buena calidad de servicio (Ej alta tasa de transferencia, bajo delay, baja perdida, indistinta duplicación, indistinto re ordenamiento).

Justifique sus decisiones en cada caso.

Aplicación	Tasa de Transferencia	Delay	Pérdida de Paquetes	Duplicación	Reordenamiento
Videojuego con multijugador Online	Alta	Bajo	Baja	Bajo	Bajo
Servicio de archivos en la nube	Media	Indistinto	Baja	Bajo	Alto
Streaming de Video	Alta	Bajo	Baja	Bajo	Bajo
Servicio de DNS	Baja	Bajo	Indistinto	Indistinto	Indistinto
Servicio de mensajería instantánea	Media	Bajo	Baja	Bajo	Alto
Repositorio FTP de imágenes ISO	Alta	Indistinto	Baja	Baja	Alto

Motivo de las decisiones:

- Videojuego con Multijugador Online: se ha optado por una alta tasa de transferencia y un bajo delay, pérdida de paquetes, duplicación y reordenamiento, ya que en estos casos los juegos en tiempo real “castigan” mucho un simple retardo ya que los demás jugadores pueden sacar provecho de este delay.

- Servicio de archivos en la nube: se ha optado por una tasa de transferencia media ya que no es una aplicación que requiera de gran ancho de banda, suponiendo que los archivos que se van a subir no van a ser muy pesados (en caso de serlo, optaría por una alta tasa).
  Se espera una baja pérdida de paquetes y un alto reordenamiento

- Streaming de Video: se ha optado por una alta tasa de transferencia y un bajo delay, baja pérdida de paquetes, baja duplicación y bajo reordenamiento ya que lo ideal es que no haya ningún lag en la transmisión (ya que es en tiempo real).

- Servicio de DNS: se ha optado por una tasa de transferencia baja ya que no es una aplicación que demande mucha tasa de transferencia, aunque si me parece necesario que el delay sea bajo ya que el hecho de que la resolución de nombres de dominio sea un proceso lento no es algo que los usuarios quieran experimentar.

- Servicio de mensajería instantánea: se ha optado por una tasa de transferencia media, ya que, la aplicación de mensajería no parece consumir mucho. Se eligió un delay bajo ya que los mensajes deben llegar lo antes posible (de manera inmediata).

- Repositorio FTP de imágenes ISO: se ha optado por una alta tasa de transferencia ya que, las imágenes ISO suelen ser muy pesadas y la transferencia de dichos archivos por medio de FTP requerirá que dicha tasa no sea tan limitada.

3. Se requiere configurar una política de control de tráfico para el uso de un enlace de red que limite la tasa de transferencia independientemente del tipo de tráfico cursado por el enlace. Para este trabajo es necesario configurar una PC como router donde se aplica la política y un cliente y servidor para la generación del tráfico (puede utilizar la herramienta nc).

a. Detalle la configuración realizada para permitir ruteo y la configuración de la política de control aplicada en cada una de las interfaces de red. Verifique la aplicación de la política utilizando las herramientas de medición de tráfico vistas en clase (por ejemplo mediante iperf e iptraf).

Para emular el laboratorio donde se aplicarán las políticas de control de tráfico, se utilizará uno provisto por netkit, mas precisamente el laboratorio de QoS.

El laboratorio está compuesto de los siguientes dispositivos e interfaces:

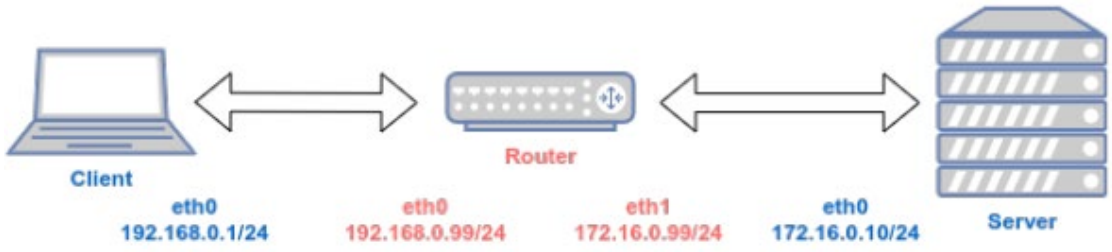


Figura 1: Topología laboratorio Netkit

Ahora bien, para configurar las políticas de control de tráfico es necesario aplicarlas sobre las interfaces del router.

Por ejemplo, se puede aplicar un Token Bucket sobre la interfaz eth0 del router, en el cual se aplicará Policing.

Para realizar esto se utilizará el comando:

```
tc qdisc add dev eth1 root tbf rate 3mbit burst 200kb latency 0ms
```

Suponiendo que queremos aplicar el algoritmo de Token Bucket, con un tamaño de balde (CIR) de 3mbit y un tamaño de ráfaga de 200kb, y al ser Shaping la latencia debe ser de 0ms para que los paquetes no tengan chance de ser encolados.

En este caso, se utiliza 1ms ya que no se permite utilizar 0ms.

```
router
root@router:~# tc qdisc add dev eth1 root tbf rate 3mbit burst 600kb latency 1ms
root@router:~#
```

Luego, para comprobar que se aplicó correctamente la política, utilizamos el comando:

```
ip a s
```

Obteniendo lo siguiente:

```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc tbf state UNKNOWN qlen
 1000
    link/ether 62:f9:7a:a1:63:3c brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.99/24 scope global eth1
    inet6 fe80::60f9:7aff:fea1:633c/64 scope link
       valid_lft forever preferred_lft forever
root@router:~#
```

Ahora, solo resta comprobar los resultados utilizando una herramienta como por ejemplo iperf:

```
client
root@client:~# iperf -c server -p 80 -t 60 -w 5mbit -i 5
------------------------------------------------------------
Client connecting to server, TCP port 80
TCP window size:  320 KByte (WARNING: requested 4.77 MByte)
------------------------------------------------------------
[  3] local 192.168.0.1 port 52515 connected with 172.16.0.10 port 80
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 5.0 sec  2.12 MBytes  3.57 Mbits/sec
[  3]  5.0-10.0 sec  1.62 MBytes  2.73 Mbits/sec
[  3] 10.0-15.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 15.0-20.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 20.0-25.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 25.0-30.0 sec  1.62 MBytes  2.73 Mbits/sec
[  3] 30.0-35.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 35.0-40.0 sec  1.62 MBytes  2.73 Mbits/sec
[  3] 40.0-45.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 45.0-50.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3] 50.0-55.0 sec  1.62 MBytes  2.73 Mbits/sec
[  3] 55.0-60.0 sec  1.75 MBytes  2.94 Mbits/sec
[  3]  0.0-60.7 sec  21.0 MBytes  2.90 Mbits/sec
```

En la imagen de arriba, se puede observar como en el primer instante de tiempo (es decir, en el intervalo desde el segundo 0 al 5, se aprovecha el burst y luego, una vez agotado, no se supera la tasa (CIR) definida anteriormente, es decir 3Mbit).

b. Indicar la configuración de tc necesaria para aplicar policing.

Para aplicar Policing mediante tc es necesario utilizar la siguiente configuración:

```
tc qdisc add dev eth0 root tbf rate 1mbit burst 100kb latency 0ms
```

siendo:

rate 1mbit = cantidad (o tasa) de arribo de tokens al balde (por segundo), en este caso 1000000 tokens por segundo.

burst 100kb = cantidad máxima de bytes que se pueden enviar “de golpe”, en este caso 100000 bytes.

latency 0ms = tiempo máximo que puede esperar un paquete a que lleguen tokens antes de ser descartado, en este caso es 0 porque se está realizando policing.

c. Indicar la configuración de tc necesaria para aplicar shaping.

Para aplicar Shaping mediante tc es necesario utilizar la siguiente configuración:

```
tc qdisc add dev eth0 root tbf rate 1mbit burst 100kb latency 100ms
```

siendo:

rate 1mbit = cantidad (o tasa) de arribo de tokens al balde (por segundo), en este caso 1000000 tokens por segundo.

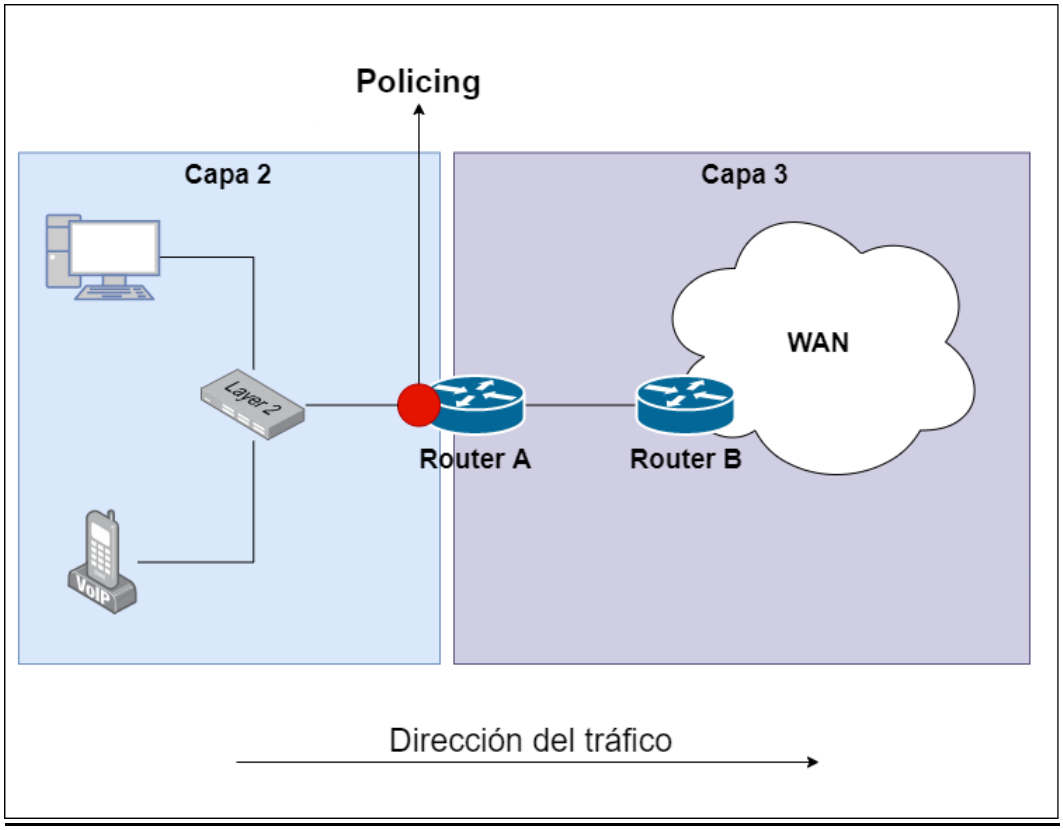
burst 100kb = cantidad máxima de bytes que se pueden enviar “de golpe”, en este caso 100000 bytes.

latency 100ms = tiempo máximo que puede esperar un paquete a que lleguen tokens antes de ser descartado, en este caso es 100 porque se está realizando Shaping, por lo que los paquetes no son descartados si es que no hay tokens en el balde, sino que son encolados.

d. Detalle un ejemplo en que sea necesario aplicar este tipo de políticas.

Para realizar esto, detallaré dos escenarios:

Escenario N° 1: Policing.



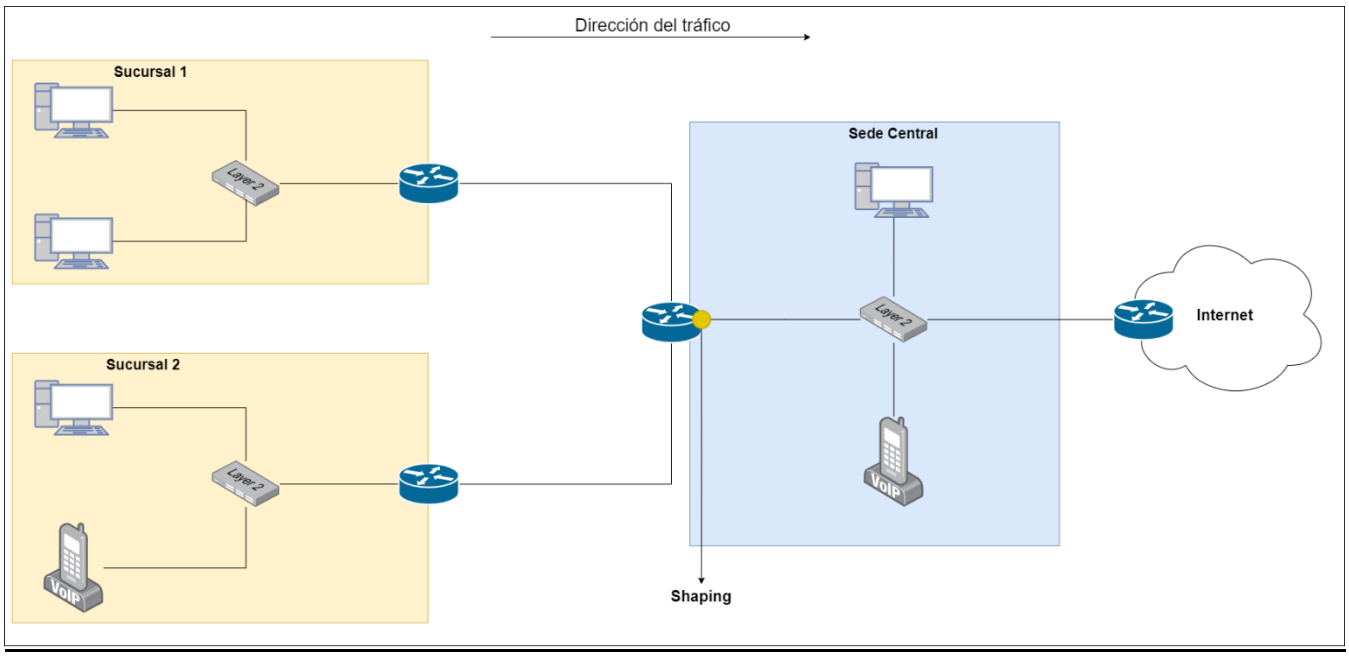
Como se muestra en la figura de arriba, en una red empresarial se transmiten servicios de voz, vídeo, datos, etc.

Cuando una gran cantidad de tráfico entra en el lado de la red, se puede producir una congestión debido a un ancho de banda insuficiente.

Se debe proporcionar un ancho de banda diferente para los servicios de voz, vídeo y datos, separándolos por prioridad.

En esta situación, el policing puede configurarse para proporcionar el mayor ancho de banda garantizado para los paquetes de voz y el menor ancho de banda garantizado para los paquetes de datos (algo que suena bastante lógico, dado que una pérdida de paquetes en un servicio VOIP es algo inaceptable). Esta configuración garantiza que los paquetes referentes al servicio VOIP tengan prioridad.

Escenario N° 2: Shaping



En un escenario supuesto de una red empresarial, la sede central está conectada a las sucursales a través de líneas alquiladas en una red ISP y las sucursales se conectan a Internet a través de la sede central.

Si todas las sucursales se conectan a Internet simultáneamente, una gran cantidad de tráfico web enviado desde la sede a Internet provoca la congestión de la red. Como resultado, parte del tráfico web se descarta.

Como se muestra en la figura de arriba, para evitar la pérdida de tráfico web, se puede configurar el shaping antes de que el tráfico enviado desde las sucursales entre en la sede central, adaptando el tráfico a diferentes velocidades, a cambio de introducir delay y jitter.