TP 4 - Calidad de Servicio (QoS) / Traffic Shaping

Fecha de Entrega: 02/10/2023.

Objetivo: Conocer los mecanismos de control de tráfico bajo entornos Linux, cómo definir Traffic Shaping Policies y cómo simular las condiciones adversas de una red WAN mediante el módulo *netem*. Categorías ISO: F**C**A**P**S.

Bibliografía

- EVANS, J., FILSFILS, C., 2007, *Deploying IP and MPLS QoS for Multiservice Networks: Theory & Practice.* Morgan Kaufmann.
  - Capítulo 1. "QOS Requirements and Service Level Agreements" (hasta pág. 24)
  - Capítulo 2. "Introduction to QOS Mechanics and Architectures" (hasta pág. 141)
- MEDHI, D., RAMASAMY, K., 2007, *Network Routing Algorithms Protocols and Architectures.* Morgan Kaufmann.
  - Capítulo 14. "Router Architectures"
  - Capítulo 22. "Packet Queuing and Scheduling"
  - Capítulo 23. "Traffic Conditioning"

Experiencia en el laboratorio

Se realizará una transferencia digital de audio "en bruto" (*raw*) para apreciar el efecto que causan algunas características no deseadas (pérdidas, delay, reordenamiento) que suelen observarse en redes WAN. Para ello, los sonidos capturados por el micrófono de una PC se enviarán por la red, sobre datagramas UDP, hasta un segundo host que reproducirá la señal de audio. Se utilizarán para ello los comandos `nc` , `arecord` y `aplay` (paquete `alsa-utils`). El router intermedio entre estos equipos implementará el emulador *netem* en su interfaz de salida hacia el host receptor.

1. Si el sonido se captura con una frecuencia de 44100 muestras por segundo (hz), cada muestra se representa como un entero de 16 bits y hay dos canales en simultáneo (izquierdo y derecho), ¿qué tasa de transferencia será requerida para esta aplicación de "voz sobre IP punto a punto"?

2. Configure la red del laboratorio para que uno de los hosts actúe como router entre los extremos de la interlocución. Recuerde que para habilitar ruteo en un host con sistema Linux debe ejecutar:

   ```
   echo 1 > /proc/sys/net/ipv4/ip_forward
   ```

   Debido a las condiciones del laboratorio, donde todos los hosts están sobre el mismo enlace (son adyacentes), es necesario impedir que se comuniquen directamente sin pasar por el router. Para ello, ejecute los siguientes comandos (en los hosts y routers):

   ```
   echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects
   echo 0 > /proc/sys/net/ipv4/conf/INTERFAZ/send_redirects
   echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
   echo 0 > /proc/sys/net/ipv4/conf/INTERFAZ/accept_redirects
   echo 0 > /proc/sys/net/ipv4/conf/all/shared_media
   echo 0 > /proc/sys/net/ipv4/conf/INTERFAZ/shared_media
   ```

3. En un equipo con parlantes, inicie un servicio que reproduzca lo recibido por la red:

   ```
   nc -u -l PUERTO | aplay -t raw -f cd -V stereo
   ```

(donde `-t` es tipo de archivo, `-f` es formato de audio y PUERTO es el puerto UDP donde se recibe el flujo)

4. En un equipo con micrófono, inicie el envío de audio a través de la red:

   ```
   arecord -t raw -f cd | nc -u IP PUERTO
   ```

   (deben coincidir -t, -f y PUERTO en nc)

5. Durante la comunicación, valide el cálculo realizado en el punto 1 mediante `iptraf` .

6. Utilizando el emulador netem, modifique las características de la red para simular delay y pérdida de paquetes.

   ```
   # para agregar latencia
   tc qdisc add dev eth0 root netem delay 200ms

   # para agregar jitter variable
   tc qdisc change dev eth0 root netem delay 200ms 35ms

   # para simular perdida de paquetes
   tc qdisc change dev eth0 root netem loss 0.1%
   ```

7. Repita el ejercicio desde el punto 3 utilizando TCP como protocolo de transporte. Compare la calidad percibida de la recepción bajo condiciones de pérdida o reordenamiento respecto de la prueba con UDP.

8. Sobre la misma topología, utilice 'tc' para aplicar una política de control de tasa de transerencia de 100 KB/s y repita la experiencia.

9. Modifique la anterior política y cambie la limitación de la tasa de transferencia a 3mbit/s máxima. Luego asignele al flujo de datos del streaming de audio una tasa mínima de 180kb/s y una máxima de 200 kb/s; por otro lado asignele 1mbit/s mínimo y 2 mbit/s máximo para el flujo web HTTP; y finalmente 1 mbit máximo para los demás flujos de información. Se recomienda revisar para ello el apartado "Conceptos fundamentales".

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.

   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.

   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).

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.

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.

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

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.

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

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. 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`).

3. Una organización requiere configurar una política de control de tráfico que priorice los servicios de correo electrónico, web y curse el resto del tráfico sin prioridad. El enlace existente en la organización es de 2 Mbps, y se requiere que el correo pueda llegar a operar hasta 1.5 Mbps y el tráfico web hasta 1 Mbps de tasa de transferencia.

   a. Proponga una solución utilizando algunas de las disciplinas de encolado vistas. Describa someramente su implementación y valídela con las herramientas de monitoreo ya utilizadas, generando tráfico sintético para los puertos de los servicios involucrados. Documente las correspondientes salidas.

   b. Mediante el software *tcviz* y el paquete *graphviz*, obtenga un gráfico (llamado *politica-tc.png*) que represente la política de control de tráfico establecida.
   Adjunte dicho gráfico en su respuesta.

```
apt install git graphviz python3
git clone https://github.com/ze-phyr-us/tcviz
cd tcviz
./tcviz.py INTERFAZ-DE-RED | dot -Tpng > politica-tc.png
```

4. 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.

DESAFÍO: Utilizando el componente `tc netem` realice cuatro simulaciones de una red de área amplia (WAN) que reproduzcan los siguientes escenarios: ⌂

a. *Delay variable:* Aplique una regla de delay, definiendo el valor que este adoptará, su variación y un valor de correlación. Detalle la configuración establecida. Utilice una herramienta de diagnóstico de red para evaluar empíricamente la latencia existente y grafique, a partir de las pruebas realizadas, el RTT percibido en función del tiempo.

b. *Pérdida:* Simule un escenario de pérdida de paquetes, luego utilice la herramienta `ping` para verificar que efectivamente las reglas fueron correctamente definidas en el host. Detalle la configuración establecida. Realice una captura para visualizar el comportamiento y documente lo hallado.

c. *Duplicación:* Simule una situación de duplicación de paquetes, luego utilice la herramienta `ping` para verificar que efectivamente las reglas fueron correctamente definidas en el host. Detalle la configuración establecida. Realice una captura para visualizar el comportamiento y documente lo hallado.

d. *Reordenamiento:* Simule una situación de reordenamiento de paquetes, donde un 25% (con una correlación del 50%) sean enviados inmediatamente mientras que el resto tenga un retraso de 10 ms. Pruebe estableciendo una sesión ssh y verifique el comportamiento a través de una captura.

DESAFÍO 2: Una empresa de transporte terrestre de larga distancia desea proveer conectividad WIFI de calidad a sus pasajeros. Para ello ha instalado un router wireless en cada vehículo que tiene conexión a Internet mediante un modem celular de tecnología 5G incorporado. Bajo su criterio, ¿qué tasa de transferencia mínima garantizada debería tener la conexión celular para garantizar que al menos la mitad de sus pasajeros puedan visualizar videos de YouTube sin interrupciones? Detalle su respuesta e incluya el procedimiento que realizó para calcularla, indicando además los factores que tuvo en cuenta para ello.

DESAFÍO 3: La final de la Copa Mundial de Fútbol 2023 fue presenciada por 89.000 espectadores en el Estadio de Lusail. Suponiendo que 1 de cada 30 espectadores filmaron y transmitieron en vivo la definición por penales con la aplicación Instagram (o Twitter o la que ud prefiera). ¿Qué cantidad de datos por segundo estima usted que debieron procesar los servidores de tal red social para garantizar que todos los flujos de video en vivo fueran vistos sin cortes por los usuarios de las redes? ¿Qué factores tuvo en cuenta para dicho cálculo?

Referencia

- Páginas del manual
  `man tc` , `tc-netem` , `tc-drr` , `tc-htb` , ...
  http://man7.org/linux/man-pages/man8/tc.8.html
- Cheat Sheet sobre QoS de PacketLife.net
  http://packetlife.net/media/library/19/QoS.pdf
- Traffic Control: Control de tráfico en Linux
  https://www.ctr.unican.es/asignaturas/dec/Doc/dec_seminario_TrafficControl.pdf
- Manual tc Packet Filtering and netem. Ariane Keller. ETH Zurich. 2006
  http://tcn.hypert.net/tcmanual.pdf
- Hertzog, R., Mas, R., Capítulo 10.4. Calidad del servicio en “Debian 11: El manual del administrador de Debian”, 2015
  https://debian-handbook.info/browse/es-ES/stable/sect.quality-of-service.html
- Vita Smid, Visualize your Linux traffic control (TC) configuration. (*tcviz*)
  https://github.com/vitawasalreadytaken/tcviz

Conceptos fundamentales

De acuerdo a su manual, TC maneja **tres** conceptos fundamentales:

QDISCS *qdisc* es la abreviatura de **disciplina de encolado** y es el componente principal para entender el control del tráfico. Siempre que se solicita al sistema operativo enviar un paquete a una interfaz, el kernel lo encola en la *qdisc* que dicha interfaz tiene definida. Inmediatamente después, el kernel trata de desencolar tantos paquetes como sea posible de la *qdisc*, para dárselos al driver de la placa de red.
Una *qdisc* sencilla es la denominada `PFIFO` , que no hace procesamiento y es una cola puramente First-In, First-Out. Desde luego, la qdisc `pfifo` sí almacena el tráfico en caso de que la interfaz de red momentáneamente no lo pueda enviar.

Algunas de las disciplinas de encolado disponibles (`qdisc`) son:

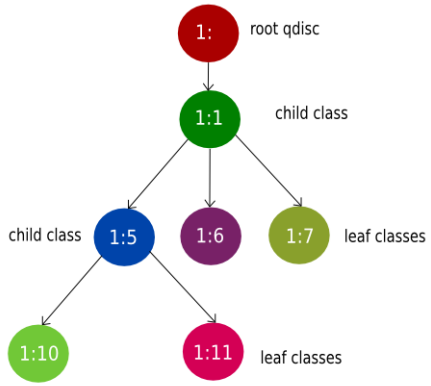
- Simples, sin posibilidad de definir clases (classless)
  - First In, First Out (FIFO)
    - Packet limited First In, First Out (pfifo)
    - Three-band First In, First Out (pfifo_fast) - por defecto en linux antiguos
    - con múltiples colas
    - basadas en hash
      - Stochastic Fairness Queuing (SFQ) - round robin, cada flujo udp o sesión tcp tiene su chance
      - Fair Queuing with Controlled Delay (fq_codel) - por defecto en linux modernos
  - Token Bucket Filter (TBF) - ralentiza una interfaz, hace shaping o policing según los parámetros
  - Generalized Random Early Detection (GRED) - hace Random Early Detection para evitar congestión
  - …
- Con posibilidad de definir clases (classfull)
  - Hierarchical Token Bucket (HTB)
  - Deficit Round Robin (DRR)
  - Class Based Queueing (CBQ) - antigua y compleja
  - …

CLASES Algunas *qdisc* pueden contener **clases**, que a su vez pueden contener más qdiscs - el tráfico puede entonces encolarse en cualquiera de las qdisc internas, que están dentro de las clases. Cuando el kernel intenta desencolar un paquete de una qdisc con clases, el mismo puede provenir de cualquiera de las clases hijas (dependiendo de la política). Una qdisc puede, por ejemplo, dar prioridad a ciertos tipos de tráfico al tratar de desencolar paquetes de una cierta clase antes de los otros.

FILTROS Finalmente, un **filtro** es utilizado por una qdisc con clases para determinar en qué clase se encola (o a qué clase corresponde) un paquete que egresará del host. Siempre que el tráfico llega a una clase con subclases, *DEBE ser clasificado*. Se pueden emplear varios métodos, entre los cuales están los filtros. (…) Es importante destacar que los filtros no son globales para toda la interfaz, sino que *siempre residen en una qdisc*.

Teoría de operación

Las clases forman un árbol, donde cada clase tiene un solo padre, pero puede contener varias clases o disciplinas (*qdisc*) hijas. Algunas disciplinas de encolado (por ejemplo `CBQ` y `HTB`) permiten la adición de más clases, mientras que otras (`PRIO`) se crean con un número fijo de clases hijas. Aquellas qdiscs que permiten agregar dinámicamente clases pueden tener cero o más subclases sobre las cuales se puede encolar tráfico.



Por otra parte, cada **clase** contiene una **qdisc** hoja que por defecto es `PFIFO` o `PFIFO_FAST`, aunque se puede conectar otra qdisc en reemplazo (y así sucesivamente).

Cuando un paquete entra en una qdisc con clases, se puede clasificar en alguna de las clases internas. Hay tres formas distintas de clasificar paquetes, aunque no todas las qdisc las implementan:

- Mediante los **filtros de tc**: si se asocian filtros de tc a una clase, son utilizados primero para determinar lo sucesivo. Los filtros pueden considerar cualquier campo de la cabecera de un paquete [utilizando `u32` o similares], como también las marcas que se ponen por iptables (utilizando `iptables -t mangle ... -j MARK --set-mark NN`).
- Mediante el **tipo de servicio**: algunas qdisc tienen criterios de clasificación propios que se basan en el campo TOS de IP.
- Mediante **skb->priority**: las aplicaciones de usuario que operan contra un socket pueden indicar una id de clase para cada paquete (que tiene vida sólo en el sistema operativo, hasta que el paquete deja el equipo) mediante la opción `SO_PRIORITY`.

Sintaxis de los comandos

```
# para ver configuración actual y estadísticas (tokens disponibles, paquetes descartados, reord
tc -s [ qdisc | class | filter ] show dev INTERFAZ

# para agregar / cambiar / quitar / ver disciplinas de encolado
tc qdisc [ add | change | del ] dev INTERFAZ root nombre_qdisc parametros_qdisc
# para agregar / cambiar / quitar / ver clases de trafico
tc class [ add | change | del ] dev INTERFAZ parent id_qdisc_padre nombre_qdisc
# para agregar / cambiar / quitar / ver reglas de filtrado de trafico
tc filter [ add | change | del ] dev INTERFAZ root protocolo proto prio prioridad \
   tipo_filtro parametros_filtro

# para volver todo a la normalidad
tc qdisc del dev eth0 root
```