UNIVERSIDAD SAN CARLOS DE GUATEMALA

CENTRO UNIVERSITARIO DE OCCIDENTE

DIVISIÓN CIENCIAS DE LA INGENIERÍA

CARRERA DE INGENIERÍA CIENCIAS Y SISTEMAS

LABORATORIO DE REDES DE COMPUTADORAS 2

ING.: FRANCISCO ROJAS

ESTUDIANTE: Sergio Daniel Cifuentes Argueta - 201730705

Manejo de Ancho de Banda y Protocolos de Red

FECHA: 20 de Septiembre de 2021

Introducción

El ancho de banda es la capacidad máxima y la cantidad de datos que se pueden transmitir a través de una conexión. En ocasiones existen máquinas conectadas a una red que acaparan el ancho de banda que existe. Esto puede ocasionar problemas de velocidad de bajada y/o de subida para los demás usuarios. Para poder solucionar este problema se planea hacer una controlador que esté entre el ISP (Proveedores de acceso a Internet) y los equipos. Este controlador tendrá que manejar el ancho de banda que se les transmite a los equipos conectados según su MAC.

Además de controlar el ancho de banda, también se intentará manejar los protocolos con los cuales se van a conectar los equipos. También se le limitaran los puertos a los que tienen acceso. Esto para que el usuario solo tenga acceso a un cierto puerto con un cierto protocolo de red.

Todos estos accesos serán de un intervalo de tiempo, dependiendo de lo indicado en el manejador. En el siguiente proyecto se intentará hacer todo esto y realizar pruebas con máquinas conectadas al manejador con el objetivo de medir y monitorear los aspectos que se le limitan en la red.

Descripción de proyecto

El proyecto coniste en limitar ancho de banda, protocolos y puertos en un intervalo de tiempo a usuarios conectados al manejador. Esto se hara segun las entradas que se le proporciona. Estas entradas se dividen en cuatro:

- enlace.conf
- modo.conf
- usuario_BW.conf
- usuario_Proto.conf

Archivos de entrada

enlace.conf

En este archivo de configuración se indicará la velocidad de bajada y de subida en Mbps.

```
down=5
up=1
```

modo.conf

Para este archivo existiran dos posibilidades que son 1 y 0. Si es 1 significa que el ancho de banda es fijo y si es 0 el ancho de banda es variable

```
modalidad=1
```

usuario_BW.conf

En este archivo contiene las configuraciones del ancho de banda con la mac del usuario el up, down, tiempo inicial y tiempo final.

```
MAC1,30,30,3:35,4:00
MAC2,30,30,3:35,4:00
MAC3,10,10,3:35,4:00
MAC1,25,25,4:00,5:00
MAC2,25,25,4:00,5:00
MAC3,25,25,4:00,5:00
```

usuario_Proto.conf

En este archivo contiene las configuraciones del ancho de banda con la mac del usuario el protocolo, el puerto, tiempo inicial y tiempo final.

```
MAC1,icmp,3:35,4:00
MAC2,icmp,3:35,4:00
MAC3,tcp,80, 3:35,4:00
```

Requisitos

Sistema Operativo (Manejador) : Debian 11

Sistema Operativo (Máquinas de prueba): Centos 7 minimal

Protocolos
UDP
TCP
ICMP

Lenguaje: Bash

Marco Teórico

HTB (Hierarchical Token Bucket)

Token bucket es un algoritmo usado en redes de computadoras de paquetes conmutados y redes de telecomunicaciones. Se puede usar para verificar que las transmisiones de datos, en forma de paquetes, se ajusten a límites definidos en el ancho de banda y la ráfaga (una medida de la irregularidad o variaciones en el flujo de tráfico). También se puede utilizar como un algoritmo de planificación para determinar el tiempo de las transmisiones que cumplirán con los límites establecidos para el ancho de banda y la ráfaga: véase planificador de red.

El algoritmo token bucket se basa en una analogía de un cubo de capacidad fija en el que los tokens, normalmente representando una unidad de bytes o un único paquete de tamaño predeterminado, son añadidos a un ritmo fijo. Cuando se debe verificar que un paquete se ajusta (es conforme) a los límites definidos, se inspecciona el cubo para ver si contiene suficientes tokens en ese momento. Si es así, el número apropiado de tokens, por ejemplo equivalente a la longitud del paquete en bytes, se eliminan ("se consumen"), y el paquete se pasa, por ejemplo, para su transmisión. El paquete no es conforme si no hay suficientes tokens en el cubo, y el contenido del cubo no se modifica. Los paquetes no conformes se pueden tratar de varias maneras:

- Pueden ser borrados.
- Pueden ser puestos en cola para su posterior transmisión cuando se hayan acumulado suficientes tokens en el cubo.

- Pueden ser transmitidos, pero marcados como no conformes, posiblemente para ser borrados posteriormente si la red está sobrecargada.

Por lo tanto, un flujo conforme puede contener tráfico con un ritmo promedio hasta el ritmo al que se añaden tokens al cubo, y tiene una ráfaga determinada por la profundidad del cubo.

Ancho de banda

El ancho de banda se mide como la cantidad de datos que se pueden transferir entre dos puntos de una red en un tiempo específico. Normalmente, el ancho de banda se mide en bits por segundo (bps) y se expresa como una tasa de bits. El ancho de banda denota la capacidad de transmisión de una conexión y es un factor importante al determinar la calidad y la velocidad de una red.

Hay varias formas diferentes de medir el ancho de banda. Algunas se utilizan para calcular el flujo de datos en un momento dado, mientras que otras miden el flujo máximo, el flujo típico o lo que se considera un buen flujo.

El ancho de banda también es un concepto clave en muchas otras áreas tecnológicas. Por ejemplo, en el procesamiento de señales se usa para describir la diferencia entre las frecuencias superior e inferior en una transmisión como una señal de radio, y se mide típicamente en hercios (Hz).

Se puede comparar el ancho de banda con el agua que fluye a través de una tubería. El ancho de banda sería la velocidad a la que el agua (los datos) atraviesa la tubería (la conexión) bajo diversas circunstancias. En lugar de bits por segundo, podríamos medirla en litros por minuto. La cantidad de agua que posiblemente pueda fluir a través de la tubería representa el ancho de banda máximo, mientras que la cantidad de agua que fluye en un momento dado a través de la tubería representa el ancho de banda actual.

Enlaces Simetricos y Asimetricos

Una conexión simétrica es la que tiene capacidad de enviar y recibir información al mismo tiempo y a la misma velocidad sin afectar a la eficacia de la señal de conexión. Tanto la descarga como la subida de datos se realizan a la misma velocidad, sin interferir entre ellas.

Muchos servicios de internet ofrecen conexiones asimétricas, lo cual permite una gran capacidad de descarga de datos pero la velocidad de subida es mucho más limitada. Por ejemplo, si tienes contratado un servicio de internet de 100 Mb/s, es más que probable que ese dato sólo se refiera a la velocidad de descarga y que apenas un 10% de esa velocidad esté destinada a la subida de datos.

Las conexiones asimétricas son las más tradicionales porque la actividad de los usuarios normalmente se centra en la descarga de datos. Un usuario particular, por regla general, consume mucha más información de la que produce y sube a internet.

Protocolos de red

Un protocolo de red designa el conjunto de reglas que rigen el intercambio de información a través de una red de computadoras.

Este protocolo funciona de la siguiente forma, cuando se transfiere información de un ordenador a otro, por ejemplo mensajes de correo electrónico o cualquier otro tipo de datos esta no es transmitida de una sola vez, sino que se divide en pequeñas partes.

TCP

TCP tiene un funcionamiento muy sencillo que consta de tres fases. En la primera, se establece la conexión con la autorización de ambas partes. Entonces, se produce un procedimiento denominado ‘negociación en tres pasos’. Para, después, iniciarse la transferencia de la información. Aquí, se establecen cada uno de los parámetros para un

intercambio ordenado, correcto y, sobre todo, seguro. Por último, mediante una ‘negociación en cuatro pasos’ se finaliza la conexión entre cliente y servidor.

UDP

UDP o Protocolo de Datagrama de Usuario (User Datagram Protocol) es un protocolo que permite la transmisión de datos sin conexión previa; de esta manera, es posible enviar información de una forma muy rápida, sin necesidad de confirmar la conexión, y esperar la respuesta de que los paquetes fueron recibidos correctamente.

ICMP

Para intercambiar datos de estado o mensajes de error, los nodos recurren al Internet Control Message Protocol (ICMP) en las redes TCP/IP. Concretamente, los servidores de aplicaciones y las puertas de acceso como los routers, utilizan esta implementación del protocolo IP para devolver mensajes sobre problemas con datagramas al remitente del paquete. Aspectos como la creación, la funcionalidad y la organización dentro de la amplia gama de protocolos de Internet se especificaron en 1981 en la RFC 792. En el caso de la sexta versión del Internet Protocol (IP), la implementación específica ICMPv6 fue definida en la RFC 4443.

Traffic Control (tc)

Linux ofrece un conjunto muy rico de herramientas para administrar y manipular la transmisión de paquetes. La comunidad de Linux más grande está muy familiarizada con las herramientas disponibles en Linux para la manipulación de paquetes y el firewall (netfilter, y antes de eso, ipchains), así como con cientos de servicios de red que pueden ejecutarse en el sistema operativo. Pocos dentro de la comunidad y menos fuera de la comunidad Linux son

conscientes del tremendo poder del subsistema de control de tráfico que ha crecido y madurado bajo los kernels 2.2 y 2.4.

Este CÓMO pretende introducir los conceptos de control de tráfico, los elementos tradicionales (en general), los componentes de la implementación del control de tráfico de Linux y proporcionar algunas pautas. Este CÓMO representa la recopilación, fusión y síntesis del CÓMO LARTC, la documentación de proyectos individuales y, lo que es más importante, la lista de correo LARTC durante un período de estudio.

El alma impaciente, que simplemente desea experimentar en este momento, es recomendada al Control de Tráfico usando tcng y HTB HOWTO y LARTC HOWTO para una satisfacción inmediata.

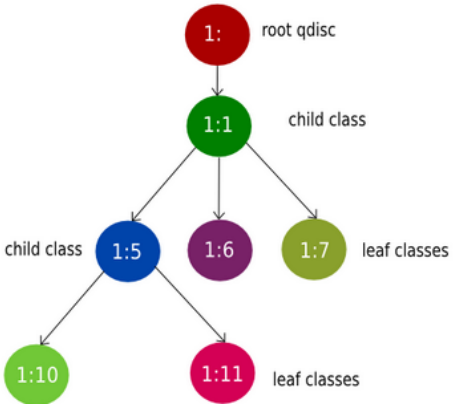
Disciplinas sencillas de hacer cola sin clases

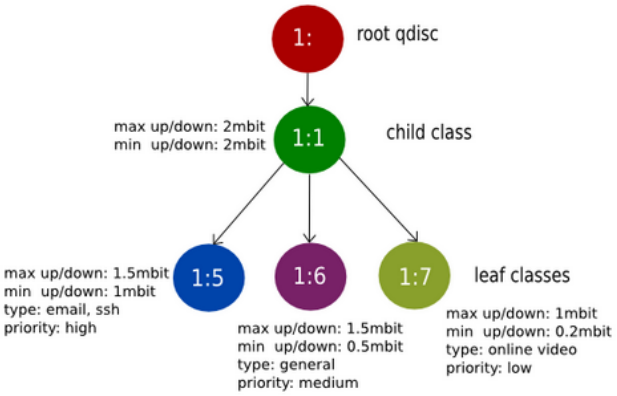
No tiene subdivisiones internas configurables. Las disciplinas de colas sin clases aceptan datos, luego reprograman, retrasan o descartan según las disciplinas de colas (qdisc).

- pfifo_fast (Primero en entrar, primero en salir) - no configurable, hardware predeterminado
- Token Bucket Filter (TBF): ralentiza una interfaz
- Cola de equidad estocástica (SFQ): round robin, cada sesión tiene la oportunidad

Disciplinas de cola con clase

- Class Based Queueing (CBQ) -Una cola con clase (vieja, compleja)
- Hierarchical Token Bucket (HTB) - otra cola con clase





Comando at

Mientras trabajamos en distribuciones GNU/Linux, generalmente utilizamos crontab para programar trabajos en general. Hay otra utilidad muy útil e interesante para programar tareas únicas. Con at podemos leer comandos de entrada estándar o scripts que deseamos que se ejecuten más tarde, una única vez. Si queremos que un comando o un script se ejecute más de una vez, debemos programarlo mediante crontab, tal y como hemos indicado en el párrafo anterior.

El comando at, nos puede ser útil para apagar el sistema a una hora especifica, realizar una copia de seguridad única, enviar un correo electrónico como recordatorio a la hora especificada, entre otras muchas cosas.

No suelo utilizar este comando, pero quizás a vosotros os puede ser de utilidad en algún momento. El programa at forma parte del temario del temario de la certificación de LPIC 102, en su versión 500

Los comandos principales son:

- at: ejecuta comandos a la hora especificada.
- atq: enumera los trabajos pendientes de los usuarios.
- atrm: borra trabajos por su número 1.de trabajo.

Proyecto

Para facilitar el uso del programa al usuario, se implemento un menu en donde se puede realizar acciones para el mejor uso del software. Esto consiste en un simple bash con un select.

```
cho "MANEJADOR DE ANCHO DE BANDA"
hile true
o
   options=("Limpiar iptables y at jobs" "Instalar herramientas- crear nodos htb
        "Programar BW y Proto" "Manual de configuraciones" "Salir")
   select opt in "${options[@]}"
   do
       case $opt in
           "Limpiar iptables y at jobs") ./clear.sh
           break;;
           "Instalar herramientas- crear nodos htb") ./create-nodes.sh
           break;;
           "Programar BW y Proto") ./exe2.sh
           break;;
           "Manual de configuraciones") ./manual.sh
           break;;
           "Salir") exit ;;
           *) echo "Opcion Invalido";;
       esac
  done
one
```

Cada opción ejecuta otro archivo .sh, esto es para que el codigo no se encuentre todo junto y esté más ordenado.

Lo primero que debemos de lograr es eliminar todas las reglas en iptables existentes. Esto con el objetivo de asignar las reglas de manera más fácil. Esto se logra con clear.sh en donde se elimina todo:

```
iptables -F
```

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

Después de limpiar las reglas debemos de crear los nodos de htb necesarios para el manejador y los usuarios, para esto es el archivo create-nodes.sh.

Nodo padre(id = 1:1)

```
#NODO RAIZ id: 1:1
tc qdisc add dev $DEV root handle 1: htb default 0xA
```

Nodo hijos(id=1:2, 1:3, 1:4)

```
#NODO HIJO id: 1:2
tc class add dev $DEV parent 1:1 classid 1:2 htb rate 1Kbit
#NODO HIJO id: 1:3
tc class add dev $DEV parent 1:1 classid 1:3 htb rate 1Kbit
#NODO HIJO id: 1:4
tc class add dev $DEV parent 1:1 classid 1:4 htb rate 1Kbit
```

los nodos hijos por defecto tendrán 1Kbit.

Y por último debemos de leer los archivos para finalmente formar los comandos at y iptables.

Todo esto se hace en el archivo exe2.sh con la ayuda de at-jobs.sh y iptables.sh.

at

```
  #args:
  #[1]mac, [2] up, [3] down, [4] start_time, [5] end time
```

```
        #Job de inicio
    ./at-job.sh ${start_time[1]} ${start_time[0]} $BWTT ${args[0]} $CEIL
        #Job al terminar
    ./at-job.sh ${end_time[1]} ${end_time[0]} 0 ${args[0]}
```

```
DEV=enp1s0

# 12:15
if [ $4=="MAC1" ]; then
    echo "/sbin/tc class change dev $DEV parent 1:1 classid 1:2 htb rate ${3}Kbit $5"| at $2:$1

fi
if [ $4=="MAC2" ]; then
    echo "/sbin/tc class change dev $DEV parent 1:1 classid 1:3 htb rate ${3}Kbit $5"| at $2:$1

fi
if [ $4=="MAC3" ]; then
    echo "/sbin/tc class change dev $DEV parent 1:1 classid 1:4 htb rate ${3}Kbit $5"| at $2:$1
fi
```

iptables

```
  #args:
  #[1]mac, [2] protocolo, [3] puerto, [4] start_time, [5] end time
    IFS=','
```

```
if [ ${#puertos[@]} -eq 2 ]; then
  ./iptable.sh 1 ${args[1]} ${args[0]} ${args[3]} ${args[4]} ${puertos[0]} $
else
  ./iptable.sh 1 ${args[1]} ${args[0]} ${args[3]} ${args[4]} ${args[2]}
fi
```

```
if [ $1 -eq 0 ]; then
    iptables -I FORWARD 1 -p icmp -m mac --mac-source $MAC -m time --timestart $4 --timestop $5 -j ACCEPT
    iptables -I FORWARD 1 -p icmp -m state --state RELATED,ESTABLISHED -m time --timestart $4 --timestop $5 -
j ACCEPT
fi
```

Resultados

Creacion de modulos

```
1) Limpiar iptables y at jobs
2) Instalar herramientas- crear nodos htb
3) Programar BW y Proto
4) Manual de configuraciones
5) Salir
#? 2
```

```
class htb 1:4 root prio 0 quantum 1000 rate 1Kbit ceil 1Kbit linklayer ethernet
burst 1600b/1 mpu 0b cburst 1600b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000

class htb 1:2 root prio 0 quantum 1000 rate 1Kbit ceil 1Kbit linklayer ethernet
burst 1600b/1 mpu 0b cburst 1600b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000

class htb 1:3 root prio 0 quantum 1000 rate 1Kbit ceil 1Kbit linklayer ethernet
burst 1600b/1 mpu 0b cburst 1600b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000
```

Programación de Jobs

```
root@debian:/home/sergio/Documents# atq
267     Mon Sep 20 01:25:00 2021 a root
281     Mon Sep 20 01:30:00 2021 a root
275     Mon Sep 20 01:30:00 2021 a root
288     Mon Sep 20 01:34:00 2021 a root
287     Mon Sep 20 01:30:00 2021 a root
289     Mon Sep 20 01:34:00 2021 a root
268     Mon Sep 20 01:25:00 2021 a root
274     Mon Sep 20 01:30:00 2021 a root
285     Mon Sep 20 01:30:00 2021 a root
263     Mon Sep 20 01:25:00 2021 a root
270     Mon Sep 20 01:32:00 2021 a root
262     Mon Sep 20 01:25:00 2021 a root
276     Mon Sep 20 01:34:00 2021 a root
278     Mon Sep 20 01:34:00 2021 a root
273     Mon Sep 20 01:30:00 2021 a root
283     Mon Sep 20 01:34:00 2021 a root
```

Cambio de ancho de banda

```
root@debian:/home/sergio/Documents# !54
tc -s -d class show dev enp1s0
class htb 1:4 root prio 0 quantum 23037 rate 1843Kbit ceil 1843Kbit linklayer et
hernet burst 1599b/1 mpu 0b cburst 1599b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000

class htb 1:2 root prio 0 quantum 23037 rate 1843Kbit ceil 1843Kbit linklayer et
hernet burst 1599b/1 mpu 0b cburst 1599b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000

class htb 1:3 root prio 0 quantum 29437 rate 2355Kbit ceil 2355Kbit linklayer et
hernet burst 1599b/1 mpu 0b cburst 1599b/1 mpu 0b level 0
 Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
 backlog 0b 0p requeues 0
 lended: 0 borrowed: 0 giants: 0
 tokens: 200000000 ctokens: 200000000
```

Bibliografía

HTB

http://intronetworks.cs.luc.edu/current/html/tokenbucket.html

https://es.wikipedia.org/wiki/Token_bucket

https://man7.org/linux/man-pages/man8/tc-htb.8.html

https://wiki.mikrotik.com/wiki/Manual:HTB#Dual_Limitation

Ancho de banda

https://www.paessler.com/es/it-explained/bandwidth

https://serverfault.com/questions/833862/how-to-limit-speed-for-every-device-per-mac-address-in-the-gateway-via-linux-com

Enlaces

https://www.fibraopticahoy.com/conexiones-simetricas-vs-asimetricas/

https://www.econectia.com/blog/conexion-internet-simetrico-asimetrico

Protocolos

https://es.wikipedia.org/wiki/Anexo:Protocolos_de_red

tc

https://wiki.debian.org/TrafficControl

https://tldp.org/HOWTO/Traffic-Control-HOWTO/intro.html

https://openwrt.org/docs/guide-user/network/traffic-shaping/packet.scheduler.example5