Control de Tráfico y Administración de Ancho de Banda en Linux con *tcng*

Federico Fapitalle Javier Echaiz* Jorge R. Ardenghi

Laboratorio de Investigación en Sistemas Distribuidos (LISiDi)
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur – Bahía Blanca, Argentina
T.E.: +54 291–4595135 Fax: +54 291–4595136
e-mail: ffapi@infovia.com.ar {je, jra}@cs.uns.edu.ar

Resumen

En estas líneas de trabajo se pretende introducir algunas nociones de control de tráfico, presentando una herramienta que provee los medios para lograr la tarea de configuración de las políticas para el mismo. Posteriormente, se presentarán algunos resultados y planes acerca de trabajos futuros.
Palabras Clave: control de tráfico, ancho de banda.

1 Introducción

El término *control de tráfico* hace referencia al subsistema de colas de paquetes en una red o dispositivo de red. El control de tráfico consiste de diversas operaciones [1]. *Clasification* es el mecanismo por el cual se identifican los paquetes y se los coloca en flujos o clases individuales. *Policing* es el mecanismo por el cual se limita la cantidad de paquetes o bytes en un flujo que corresponde a una clasificación particular. *Scheduling* es el proceso de decisión a través del cual se ordenan los paquetes para ser transmitidos. *Shaping* es el proceso a través del cual los paquetes son demorados y transmitidos para producir un flujo predecible.

Estas características del sistema de control de tráfico pueden combinarse de tal forma de reservar un determinado ancho de banda para un flujo de datos determinado (o una aplicación), o para limitar el ancho de banda disponible para un flujo o aplicación en particular.

El siguiente listado no pretende ser una lista exhaustiva de las soluciones disponibles para los usuarios mediante el uso de control de tráfico, pero introduce los tipos de problemas que pueden ser sorteados usando los mecanismos de control de tráfico para maximizar la usabilidad de un enlace de red.

- Limitar el ancho de banda total disponible a un valor fijo conocido.
- Limitar el ancho de banda de un usuario, servicio o cliente en particular.
- Maximizar el *throughput* de tráfico TCP en un enlace asimétrico.

*Becario del Consejo Nacional de Investigaciones Científicas y Técnicas, República Argentina.

- Reservar ancho de banda para un usuario, servicio o cliente en particular.
- Priorizar tráfico sensible a la latencia.
- Realizar una administración y distribución equitativa del ancho de banda disponible.
- Filtrar un tipo particular de tráfico. Este caso muestra que el control de tráfico está relacionado con el concepto de *firewalling*, y que existen casos donde el primero puede realizar parcialmente las tareas del segundo[1].

1.1 El Control de Tráfico en Linux

El diseño del subsistema de control de tráfico en Linux es altamente modular, con una vasta elección de lo que se denomina *disciplinas de colas* (una combinación entre *queuing* y *scheduling*) y clasificadores. Entre las disciplinas de cola encontramos [4]:

- *Drop-tail FIFO*, que descarta los paquetes entrantes cuando alcanza su tamaño máximo.
- *Random Early Detection (RED) FIFO*, que descarta los paquetes entrantes antes de alcanzar el tamaño máximo de cola, de manera tal que protocolos capaces de controlar la congestión (por ej. TCP) pueden demorar la transmisión de paquetes.
- *Token Bucket Filter (TBF)*, que emite paquetes a un rate fijo y constante.
- *Priority Scheduler*, que emite los paquetes de las clases de mayor prioridad antes que los paquetes de las clases con menor prioridad.
- *Hierarchical Token Bucket (HTB)*, que (entre otras cosas) permite asignar cierto porcentaje del ancho de banda disponible a cada clase.

2 tcng - Traffic Control Next Generation

tcng (*Traffic Control Next Generation*) es una revisión de la infraestructura de control de tráfico de red de Linux. El objetivo es superar las desventajas de la arquitectura existente, y hacer la misma más fácilmente extensible.

Esta infraestructura define un nuevo lenguaje de configuración, mucho más amigable, y provee un compilador (`tcc`) para trasladar la configuración a un conjunto de lenguajes de bajo nivel, entre ellos `C` y `tc`. Su sintaxis es similar a la de lenguajes conocidos como `C` o `perl`. Permite a los usuarios definir sus propias construcciones del lenguaje a través de macros y variables.

tcng incluye un simulador (`tcsim`) que utiliza código real de kernel para simular el subsistema de control de tráfico.

El detalle de la sintaxis y elementos del lenguaje utilizado por *tcng* puede encontrarse en [3].

[1] También es factible el caso inverso.

2.1 Un Ejemplo

El siguiente ejemplo ilustra los elementos encontrados en una configuración típica *tcng*:

```
#include "fields.tc"
#include "ports.tc"

dev "eth0" {
    egress {
        /* clasificacion */
        class (<$high>) if tcp_dport == PORT_HTTP;
        class (<$low>)  if 1;

        /* queuing */
        prio {
            $high = class (1) {
                fifo (limit 20kB);
            }
            $low = class (2) {
                fifo (limit 100kB);
            }
        }
    }
}
```

Las primeras 2 líneas incluyen las definiciones para los campos de cabecera y números de puertos.

Las siguientes 2 líneas determinan qué es lo que se está configurando, en este caso, el tráfico saliente (`egress`) del dispositivo de red identificado como `eth0`.

La configuración consiste de dos partes: la clasificación y el sistemas de colas. En el ejemplo. se utiliza un *priority scheduler* con dos clases para las prioridades alta (`$high`) y baja (`$low`).

Los paquetes con destino al puerto TCP 80 (`PORT_HTTP`) son enviados a la clase de mayor prioridad, mientras que el resto de los paquetes (`if 1`) son enviados a la clase de menor prioridad. Es recomendable finalizar la sección de clasificación con una regla de selección que se resuelva siempre en verdadero.

La sección de colas define una disciplina de colas para prioridades estáticas, en dos clases. Dentro de la clase de alta prioridad, existe otra disciplina de colas, una cola FIFO con capacidad de 20 kilobytes. Análogamente, la clase de baja prioridad contiene una cola FIFO de 100 kilobytes.

2.2 Simulación y Resultados

Esta sección presenta la forma de trabajo con el simulador `tcsim`[2], para poder ver los resultados de la configuración realizada sin interferir en ningún ambiente de producción.

Asumimos que una configuración ya ha sido escrita *off-line* y sólo resta ser compilada y activada. Para ello, asumimos una configuración sencilla como la que se muestra debajo, y se asume que se encuentra en un archivo llamado `config.tc`.

[2]Actualmente se trabaja en otro simulador denominado `umlsim`, basado en UML (*User Mode Linux*).

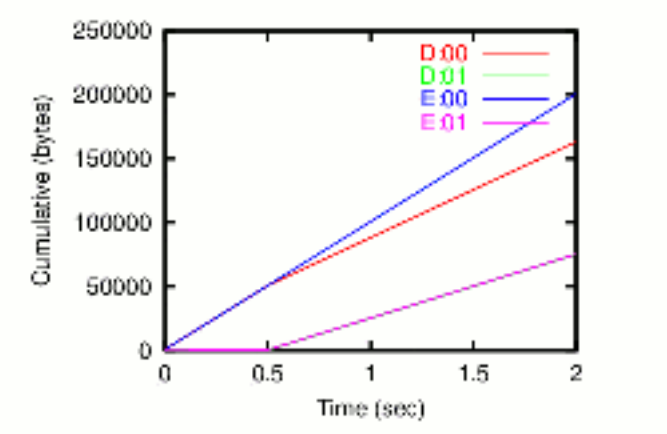

Figura 1: Resultado de la simulación

```
dev "eth0" {
      egress {
            class(<$flujo1>) if ip_tos == 0;
            class(<$flujo2>) if 1;

            prio {
                 $flujo2 = class(1);
                 $flujo1 = class(2);
            }
      }
}
```

Para realizar la simulación, definimos el siguiente archivo para `tcsim`, que lo llamaremos `config.tcsim`

```
dev "eth0" 1Mbps {

      #include "config.tc"

}

every 0.008s send 0 x 800 /* 125 pck/sec, 8*800 bits c/u */
time 0.5s
every 0.008s send 1 x 400 /* 125 pck/sec, 8*400 bits c/u */
time 2s
```

Finalmente, realizamos la simulación de la configuración realizada y obtenemos un gráfico del tráfico como se ve en la Figura 1.

3 Conclusiones y Trabajos Futuros

tcng nace como una alternativa para simplificar las tareas de configuración para control de tráfico en Linux. A través de una interfaz simple y homogénea, provee los medios para realizar una configuración sencilla de control de tráfico, evitado inconsistencias y redundancias.

tcng brinda la posibilidad de testear, utilizando las herramientas `tcsim` y `umlsim`, las configuraciones realizadas, ajustar parámetros, etc., antes de ponerlas en práctica en ambientes de producción.

En un futuro próximo, se plantearán un conjunto de configuraciones típicas para entornos de producción, como por ejemplo cyber-cafés, *server farms*, laboratorios informáticos en entornos académicos, etc. Se definirán las políticas que mejor se adecúen para cada entorno. Se realizarán las configuraciones necesarias y las pruebas correspondientes, tanto en un ambiente simulado con las herramientas provistas, como en un ambiente "real". A tal fin se acondicionará el laboratorio informático disponible.

Bibliografía

[1] Martin A. Brown: *Traffic Control HOWTO*, v1.0.1, Noviembre 2003.

[2] Martin A. Brown: *Traffic Control using tcng and HTB HOWTO*, v1.0, Abril 2003.

[3] Werner Almesberger: *Traffic Control - Next Generation. Reference Manual*, Junio 2003.

[4] Epfl Ica y Werner Almesberger: *Linux Traffic Control - Implementation Overview*, Diciembre 1998.

[5] Epfl Ica, Jamal Hadi Salim y Werner Almesberger: *Differentiated Services On Linux*, Junio 2001.