Análisis de Prestaciones de un Planificador de Tráfico sobre Plataformas de Propósito General

Eduardo Magaña, Edurne Izkue y Jesús Villadangos

Universidad Pública de Navarra

Departamento de Automática y Computación
Campus de Arrosadía, 31006 Pamplona, SPAIN
Tel. +34 948 169853

eduardo.magana@unavarra.es, edurne@izkue.com, jesusv@unavarra.es

RESUMEN

En el presente trabajo se analizan las prestaciones de un planificador que se ejecuta sobre una plataforma de propósito general, en concreto un PC con sistema operativo Linux. El planificador se configura para manejar una disciplina de cola CBQ (*Class Based Queueing*) y mediante una serie de experimentos se comprueba que las limitaciones del sistema son debidas a la precisión del reloj. Una vez modificado dicho factor se analizan nuevamente las prestaciones del planificador. Esto permite comprobar que las prestaciones del planificador mejoran en gran medida a costa de incrementar escasamente (entorno al 10%) el uso de CPU del sistema.

Palabras clave

Planificador, calidad de servicio, DiffServ, CBQ, TBF, Linux, ventana de congestión, precisión de reloj.

1. INTRODUCCIÓN

Las redes de datos transportan paquetes que corresponden a diferentes tipos de servicios. Uno de los criterios habituales para clasificarlos es su requerimiento temporal. En tal caso, se distinguen los servicios que tienen unos requisitos temporales muy estrictos, llamados de tiempo real, y el resto de servicios. Un ejemplo del primer tipo de servicios es la transmisión de voz sobre IP (VoIP) y el vídeo bajo demanda (VoD), mientras que el servicio de correo electrónico y la transferencia de ficheros mediante FTP son un ejemplo del segundo tipo. Además, la calidad de los servicios de tiempo real es muy sensible a las pérdidas de paquetes, ya que puede dar lugar a cortes en la reproducción de la información.

SIGCOMM – Latin America & Caribbean 4/01 San Jose, Costa Rica

En definitiva, la provisión de servicios de tiempo real, de calidad y atractivos para el usuario, implica necesariamente que el proveedor tiene que garantizar una calidad de servicio suficiente para dichos servicios, tratando de evitar la pérdida de paquetes, minimizar el retardo y las variaciones del retardo (*jitter*) de los mismos.

En la actualidad, en Internet no se garantiza ninguna calidad de servicio ya que se basa en el mecanismo Best Effort. Esto quiere decir que el flujo que más paquetes transmita, más posibilidades tendrá de hacer llegar sus paquetes al destino. El protocolo de Internet, IP, no provee por tanto ninguna garantía de calidad de servicio.

La idea de poder garantizar calidad de servicio hace pensar en la necesidad de un planificador de tráfico, siguiendo la propuesta de diferenciación de servicios (DiffServ) del IETF [2].

En este trabajo se estudian las posibilidades de uso de una arquitectura de propósito general como plataforma de planificación. Esto refleja la situación de un proveedor de servicios de Internet (ISP) que quiere proveer servicios de tiempo real garantizándoles una adecuada calidad de servicio. Una de las principales características de los ISPs es que, habitualmente y para maximizar beneficios, contratan un ancho de banda menor que el que podrían requerir todos sus abonados simultáneamente. Este hecho puede dar lugar a situaciones en las que los servicios de tiempo real se vean afectados por el uso de los recursos de red del resto de usuarios. Por tanto, los ISPs deben utilizar sistemas de planificación para limitar las tasas de los usuarios y para garantizar una tasa mínima a los servicios de tiempo real.

Se ha tratado de observar el mayor número de detalles diferentes que se producen con el uso de un planificador y en los siguientes apartados se muestran los resultados. En primer lugar, en la sección 2 se presentan las disciplinas de cola utilizadas CBQ y TBF. En la sección 3 se describe el entorno experimental. En la sección 4, se presenta el efecto de la planificación cuando los tamaños de los paquetes y los tiempos entre llegadas siguen diferentes distribuciones. A continuación la sección 5 muestra el efecto del planificador sobre los flujos UDP y TCP. La sección 6 se dedica a mostrar el efecto de modificar la precisión del reloj del

sistema. En este apartado se analizan las mejoras que se obtienen con una mayor precisión de reloj y el coste computacional que ello supone. Finalmente, en la sección 7 se presentan trabajos relacionados y en la sección 8 las conclusiones obtenidas.

2. CONTROL DE TRÁFICO EN LINUX

El sistema operativo Linux incorpora en las versiones recientes de kernel soporte para control de tráfico que aporta las funcionalidades básicas de disciplinas de colas, clases y filtros para poder poner en marcha un planificador de calidad de servicio. El sistema de control de tráfico en Linux se esquematiza en la Figura 1 [5].

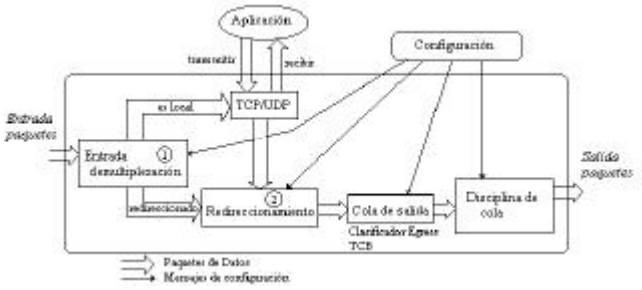

Figura 1. Esquema de funcionamiento de la parte de red del kernel de LINUX.

Los procesos del sistema operativo pueden transmitir y recibir paquetes haciendo uso de funcionalidades de red a nivel de kernel. Una de estas funcionalidades se encargada del control de tráfico en Linux. La Figura 1 muestra su estructura de bloques.

La entrada demultiplexación ① examina los paquetes de llegada y determina si el paquete es para la máquina local o hay que encaminarlo a la siguiente máquina. Si el paquete es para la propia máquina, se envía a la capa superior para ser procesado. Si no, se pasa al bloque de redireccionamiento ②. Al bloque de redireccionamiento ② también le llegan paquetes a transmitir por la red generados por la propia máquina en capas superiores.

El redireccionamiento ② observa las tablas de rutas, busca el siguiente salto, hace selección de la interfaz de salida adecuada para esa ruta y encapsula el paquete. Una vez hecho todo esto y tras determinar el nodo destino, almacena el paquete en la cola de salida, es decir, el buffer de salida adecuado a ese servicio, ruta o tipo de información, para ser transmitido en cuanto pueda, tras el acceso a la red (que depende de la topología de la red, por ejemplo, si es 802.2 Ethernet será por contienda con el protocolo CSMA/CD, y si es Token Ring esperará a coger el testigo). Y aquí es dónde actúa el control de tráfico de Linux. Entre otras cosas decide si el paquete se encola o se tira (por ejemplo, porque las colas tengan longitud limitada y ya estén llenas, o porque el tráfico excede algún límite de velocidad de transmisión). El control de tráfico puede decidir en qué orden se envían los paquetes, dando por ejemplo prioridad a ciertos flujos frente a otros, y puede retardar la transmisión de los paquetes, por ejemplo, limitando la velocidad de tráfico de salida a la red con un Token Leaky Bucket.

El control de tráfico que ofrece Linux se puede usar para construir combinaciones complejas de disciplinas de cola, clases y filtros útiles para aplicar disciplinas de QoS sobre los paquetes que se mandan a algún interfaz de salida.

En el interfaz de salida se puede también configurar un dispositivo de *shaper*, es decir, un moldeador de tráfico que determina el tipo de tráfico que sale a la red. También se puede hacer un enrutamiento explícito mediante el etiquetado de los paquetes con el comando *ipchains* [8]. La cola del interfaz de salida sigue una disciplina de cola bien CBQ, FIFO, del tipo limitador de tasa como sería TBF (Token Bucket Filter), etc.

En la base del funcionamiento del control de tráfico en Linux hay tres bloques principales:

- *Qdisc*: Disciplina de cola, lo que sería la cola que recoge los paquetes y los saca según la disciplina determinada. Existe por lo menos una en el interfaz de salida. Se puede especificar una disciplina de cola por cada clase final (en la que acaban los paquetes tras ser clasificados).

- *Class*: La clase determina de qué tipo de tráfico se trata. Más concretamente dice a qué tipo de servicio o a qué clasificación corresponde el paquete dentro del planificador, para el posterior tratamiento adecuado de estos paquetes que la clase recoge. Cada clase tiene sus propias características referentes al tipo de disciplina de cola que se le quiera asociar.

- *Classifiers o filters*: el filtro o el clasificador, dónde se determina el criterio para la discriminación de paquetes, bien sea por las direcciones origen y/o destino como por el tipo de información que lleva (según el puerto origen y/o destino), u otros campos de las cabeceras, con total flexibilidad.

Estos tres bloques son la base fundamental para entender en qué se basa un controlador de tráfico en Linux. La gestión del ancho de banda es jerárquica, es decir, que las clases siguen una jerarquía que guarda relación con el ancho de banda a determinar en cada clase.

Sólo puede existir una *qdisc* raíz por dispositivo. Esta *qdisc* se asocia al dispositivo de salida, la cual es dueña de todo el ancho de banda que éste ofrece.

A continuación vamos a revisar las disciplinas de colas utilizadas en el presente trabajo: CBQ y TBF.

2.1 Planificador tipo CBQ

CBQ (*Class Based Queueing*) [6][7][16] es un mecanismo de control de tráfico basado en compartir el ancho de banda de un enlace para así poder tener un mantenimiento de los recursos. Tiene la característica de permitir compartir el enlace entre múltiples agentes, familias de protocolos o tipos de tráfico.

Ofrece una estructura jerárquica por cada enlace, en la cual se encuentran las clases, que son las estructuras correspondientes a algún tipo de agregado de tráfico. La propia estructura jerárquica especifica la política deseada para ese enlace expresada en

porcentajes del ancho de banda en periodos de congestión. La reserva de este ancho de banda puede ser estática (mediante la asignación del administrador) o dinámica (por el uso de un protocolo de reserva de recursos como podría ser RSVP [3]). Todos los paquetes que pasan por el router son asociados a una clase que es una hoja del árbol, de la estructura jerárquica, como los nodos D, E, B o C de la Figura 2. Los nodos intermedios del árbol no transmiten ningún paquete como podría ser el caso del nodo A de la Figura 2. Estos nodos sólo dan información de cómo se asocia el ancho de banda sobrante y guardan estadísticas de los nodos que cuelgan de ellos, ya que los paquetes que estos nodos hoja transmiten pasan por ellos.

Como objetivo principal de CBQ hay que destacar que una clase recibe por lo menos el ancho de banda que se le asoció, incluso en situación de congestión. Por lo tanto si es una clase con mucha demanda recibe como mínimo el ancho de banda asociado, pudiendo recibir más si las reglas así lo permitieran como se explica más adelante. Si una clase no tiene ancho de banda asociado, CBQ no garantiza ningún ancho de banda en caso de congestión. El ancho de banda asociado depende del mecanismo de planificación de paquetes del router, como se verá más adelante. Una correcta selección de anchos de banda asociados hace más útil un planificador desde el punto de vista del usuario.

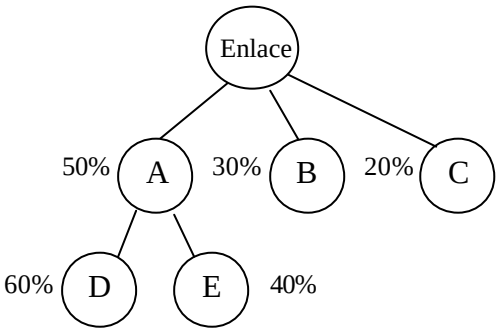

Figura 2. Ejemplo de configuración con CBQ.

Otra característica importante de CBQ es el préstamo de ancho de banda. La redistribución del ancho de banda que alguna clase no usa no es arbitraria. Cogiendo como ejemplo la configuración de la Figura 2, el nodo D debe coger por lo menos el 60% del ancho de banda asociado que tiene el nodo A, y el nodo E el 40%. La notación de un mínimo ancho de banda se debe al hecho de poder permitir que una clase coja el ancho de banda sobrante de la clase superior de la que cuelga. De modo que si el nodo D estuviera superando el ancho de banda asociado, es decir necesitara más del 60% del ancho de banda asociado de A, su padre, y éste se lo permitiera, cogería de su ancho de banda asociado sobrante para cubrir la falta de ancho de banda de D. El nodo E tendría el ancho de banda sobrante del nodo A. Si la utilización de ancho de banda de los nodos E y D no superara el ancho de banda del nodo A, este ancho de banda sobrante se repartiría siguiendo los porcentajes con los nodos B y C si el nodo superior, enlace, así lo permitiera.

Cada clase puede tener asociada una disciplina de cola de salida del router. La única clase que debe tener una disciplina de cola de salida es la de la raíz, la que directamente está ligada a la tarjeta de red de salida. En cada clase se definen dos planificadores de paquetes, el genérico y el de compartición de enlace:

- El planificador genérico se encarga de transmitir los paquetes cuando el ancho de banda que requiere el flujo de esa clase no supera el ancho de banda asociado. De este modo necesita un planificador apropiado desde el punto de vista del usuario. Por ejemplo, en el caso de tener tráfico de tiempo real se necesitaría un planificador que logre redireccionar este flujo sin superar el retardo máximo.
- El planificador de compartición del enlace es aquel que se encarga de transmitir los paquetes cuando el ancho de banda del flujo de esa clase supera el ancho de banda asociado. Este planificador tiene que tener las mismas capacidades que el planificador general, pero también hay que sumarle la tarea de limitar el ancho de banda para poder acomodar el ancho de banda de la clase al ancho de banda asociado, e incluso debe tener incorporada alguna estrategia para descartar paquetes si esto fuera necesario.

Con el limitador de ancho de banda (que hace uso del estimador de ancho de banda que se explica después), se puede garantizar que ninguna clase utilizará ancho de banda sobrante que le corresponda a otra clase que lo requiera. Se puede concluir que CBQ garantiza un ancho de banda mínimo a las clases que así lo tienen prefijado.

Los planificadores de paquetes usan los conceptos de selector y retardador. El selector define la clase que en cada instante tiene permiso para enviar el próximo paquete. Por ejemplo, si se clasifican las clases por prioridades y se comienza con la de mayor prioridad y se sigue con las demás clases mediante un round-robin es el caso en el que se dice que CBQ es un planificador del tipo WRR (Weight Round Robin). El retardador calcula el momento en el que aquella clase que está superando el ancho de banda asociado transmite el próximo paquete. La clase que se retrasa tiene limitada su tasa al ancho de banda asociado en la jerarquía.

CBQ necesita también de los siguientes mecanismos:

- Clasificador de paquetes, para clasificar los paquetes que pasan por el router en la clase apropiada del enlace de salida. A este mecanismo CBQ no le especifica ningún requisito especial. El clasificador define la funcionalidad del planificador para el usuario, asociando los flujos con clases.
- Estimador del ancho de banda, es aquel que estima el ancho de banda que usa cada clase en un determinado intervalo de tiempo, para determinar si está recibiendo o no el ancho de banda asociado a esa clase. El intervalo de tiempo es un parámetro fundamental con el que se determina la precisión con la que el router cumple las relaciones jerárquicas del reparto del ancho de banda del enlace.

Entre las clases existen tres tipos especiales, la *exempt* (exenta), clase que no tiene restricciones de ancho de banda a consumir, la *bounded* (delimitada), clase que no coge el ancho de banda sobrante de su nodo padre, aquel ancho de banda que las clases contiguas no consumen, y la *isolated* (aislada), clase que no da ancho de banda prestado a los nodos hijos, haciendo que cada clase hija sea del tipo *bounded*.

2.2 Planificador tipo TBF

El control de tráfico de Linux ofrece la posibilidad de colocar una disciplina de cola en cada clase. Entre ellas está la más típica FIFO, la de prioridades, la ya conocida CBQ y el TBF. El TBF (*Token Bucket Flow*) es un limitador de tasa basado en pasar paquetes a la red con un Token. Estos Tokens se generan con una tasa constante, tasa a la que se quiere limitar la salida del flujo.

Por tanto su aplicación es limitar la tasa máxima de salida de los flujos sin garantizar ningún ancho de banda mínimo o préstamo entre clases.

3. ENTORNO EXPERIMENTAL

En este trabajo se estudian las prestaciones de un planificador mediante el siguiente montaje experimental. Dos redes A y B de 100 Mbps y 10 Mbps respectivamente, se unen por medio de un router en el cual se configura un planificador, como se observa en la Figura 3.

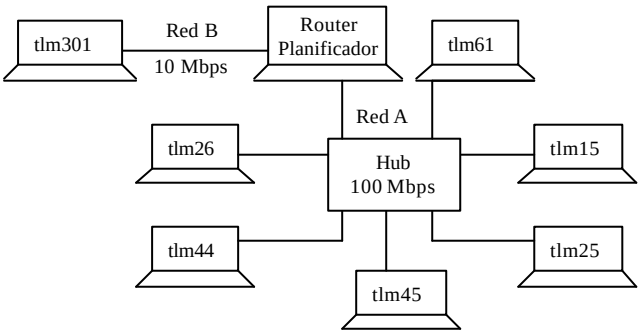

Figura 3. Entorno de trabajo.

Este planificador se implementa sobre un PC que realiza funciones de router. Se ejecuta sobre una plataforma PC Pentium III a 550 MHz, sistema operativo Linux RedHat v6.1 y un kernel versión 2.2.12-20. El planificador se implementa con el control de tráfico que provee Linux *iproute2-2.2.4* [1] [13] que se configura mediante la herramienta *tc* [7].

El planificador trata con flujos procedentes de máquinas de la red A y los encamina a la máquina destino, tlm301, colocada en la Red B. Debido a que esta Red B es de menor capacidad, el planificador tendrá que priorizar unos flujos frente a otros, garantizando calidad de servicio. Para distinguir los flujos en el planificador se ha utilizado el número de puerto de destino, asociando a cada uno un tipo de servicio (de tiempo real, retardo mínimo, máxima fiabilidad, etc.).

Para demostrar la eficacia del planificador desarrollado se han implementado un conjunto de herramientas. Se desea comprobar la eficacia de los filtros activados en el planificador, bien por el tipo de tráfico que sea (TCP o UDP), como por el puerto al que accede ese tráfico o bien por el tipo de servicio activado en el octeto TOS (Tipo de Servicio) o incluso por las direcciones IP origen y/o destino. Primeramente es necesario generar un flujo de paquetes en el que poder elegir la distribución tanto el tamaño del paquete como el tiempo entre paquetes. Para hacer esto posible se ha realizado un programa que permite generar tráfico según diferentes distribuciones estadísticas (Poisson, Pareto o Determinista) y también a partir de un fichero de traza procedente de una captura de tráfico real. En la máquina destino se ha implementado un servidor para recibir y analizar el tráfico generado.

Entre los análisis realizados se ha comprobado la independencia del planificador ante diferentes tipos de distribuciones, tanto para el tamaño de paquete como para el tiempo entre paquetes. A continuación se comprueba el efecto de la planificación cuando interfieren flujos TCP y UDP, mostrándose a su vez, la evolución del tamaño de la ventana de congestión. Finalmente, se modifica la precisión de reloj del sistema. Esta modificación permite obtener las principales aportaciones del trabajo, ya que se comprueba que el planificador mejora sus prestaciones sin aumentar de manera ostensible el consumo de CPU del sistema. Es decir, el sistema puede realizar tareas de planificación de gran precisión sin necesidad de monopolizar el uso de CPU ni para las tareas de planificación, ni para las tareas de control de tiempos.

4. EFECTO DEL TAMAÑO Y DISTRIBUCIÓN DE LOS PAQUETES

El tráfico que circula por las redes de datos tiene gran variabilidad en su patrón de comportamiento, por ello se analiza en este apartado el comportamiento del planificador ante diferentes combinaciones en las distribuciones de los flujos en cuanto al tiempo entre paquetes y al tamaño de los mismos. En la literatura un modelo típico utilizado para el tiempo entre paquetes ha sido la distribución de Pareto como distribución *heavy-tailed* más simple. Sin embargo, modelos autosimilares modelan mejor el tráfico de datos [9][12]. De todos modos, una distribución tipo Pareto en escalas pequeñas de segundos a minutos tiene apariencia autosimilar. En este caso se analiza el sistema considerando que el tiempo entre paquetes sigue una distribución tipo Pareto y un tamaño de los paquetes que sigue una distribución Exponencial.

Generalmente los servicios con requerimientos de Calidad de Servicio (VoIP, video, …) son servicios bajo protocolo de transporte UDP y por eso se ha realizado la simulación con flujos UDP. En la Figura 4, entre los segundos 0 y 200, se muestran un conjunto de flujos desde 300 Kbps, 500 Kbps, 750 Kbps y 1 Mbps, con ancho de banda garantizado a esas mismas tasas, sobre los que interfiere un flujo también UDP de 12 Mbps. Estos flujos parecen independientes del flujo interferente de 12 Mbps, puesto que no se nota reducción alguna en la tasa de los flujos cuando

interfiere este flujo UDP de 12 Mbps. Los flujos son independientes de la interferencia del flujo de 12 Mbps y no sufren pérdida alguna excepto el de 12 Mbps por la limitación de ancho de banda de la red destino B y por no tener un ancho de banda garantizado.

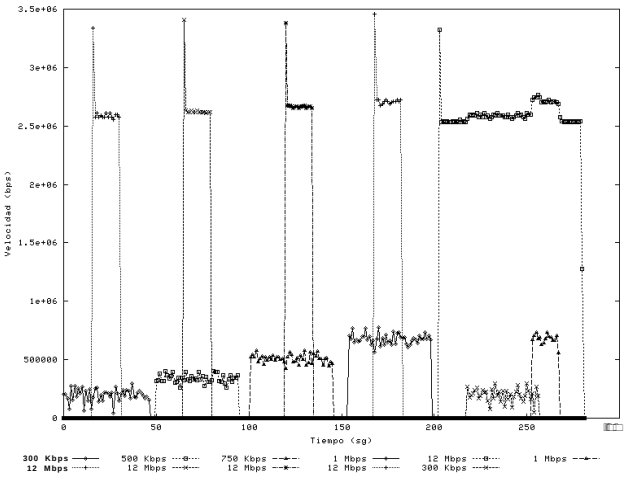

Figura 4. Tiempo entre paquetes Pareto y tamaño Exponencial, con planificador tipo CBQ.

También se puede apreciar en la Figura 4, entre los segundos 200 y 300 aproximadamente, como aún existiendo un flujo interferente de 12 Mbps sin ancho de banda garantizado, se pueden realizar transferencias de flujos de tasas menores, de 300 Kbps y 1 Mbps con ancho de banda garantizado, nuevamente independientes de la carga de red gracias al planificador tipo CBQ del router.

La planificación CBQ funciona bien principalmente para bajas velocidades y, ciertamente, a bajas velocidades es independiente de la distribución que pueda tener el flujo a tratar. Esto es debido, como ya se demostrará más adelante, a la falta de precisión de reloj del sistema del planificador.

Se ha realizado la prueba con las 9 combinaciones posibles de las distribuciones determinista, exponencial y pareto, del tamaño de paquete y del tiempo entre paquetes, para apreciar mejor la posible dependencia del resultado con las distribuciones escogidas. Se concluye que las diferentes combinaciones, tanto en la distribución del tiempo entre paquetes como en la del tamaño de paquetes, no afectan al planificador tipo CBQ del router, ya que siempre se obtienen los mismos resultados.

4.1 Distribución del tiempo entre paquetes antes y después del planificador

Un aspecto que preocupa mucho en los servicios de tiempo real es el retardo que puede introducir un router y sobre todo el efecto del *jitter*, la variación del retardo entre paquetes.

Puesto que tenemos la posibilidad de ver el tiempo entre paquetes en origen y en destino, se ha querido hacer hincapié en la posible variación que sufre tras pasar por un router con una planificación de este tipo CBQ.

Se toma el caso en el que el tiempo entre paquetes sigue una distribución de Pareto y el tamaño del paquete por su parte sigue una distribución Exponencial. Se puede ver en la Figura 5 que la distribución del tiempo entre paquetes ha cambiado levemente.

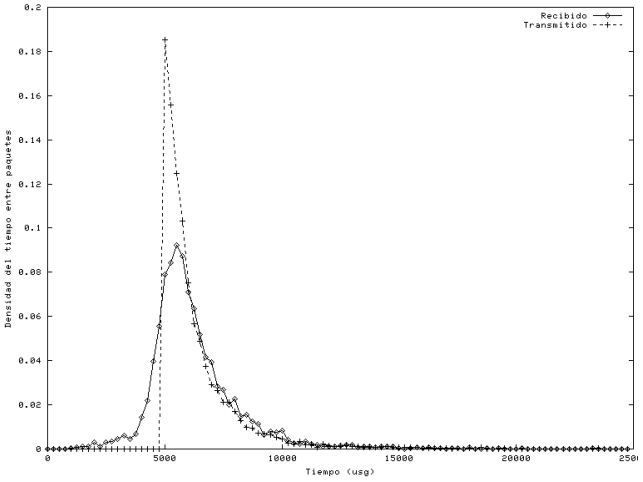

Figura 5. Función de densidad del tiempo entre paquetes del flujo de 1 Mbps en transmisión y en recepción.

El experimento consiste en la transimisión de un flujo UDP de 1 Mbps y en la mitad de su transferencia se le ha añadido un flujo de 12 Mbps. Este flujo de 12 Mbps está limitado a una tasa menor de 2 Mbps. El flujo de 1 Mbps no ha reducido su velocidad de transmisión ni ha tenido pérdidas durante el tiempo en que se ha estado transmitiendo simultáneamente el flujo de 12 Mbps, por lo tanto, sólo queda una forma de ver la razón de este cambio en la distribución del tiempo entre paquetes. La siguiente Figura 6 muestra el tiempo entre paquetes frente al número de secuencia de los paquetes transmitidos. Se puede apreciar claramente como la distribución que sigue el tiempo entre paquetes cambia cuando aparece el flujo interferente de 12 Mbps, entre los paquetes 2200 y 4800. Esta reducción del tiempo entre paquetes es la que produce el cambio en la función de densidad de 0 a 5000 μsg, afectando igualmente en la función de distribución. A la vez que unos valores del tiempo entre paquetes se reducen, otros aumentan, y eso también se deja notar en la función de densidad, en 10 msg de la Figura 5 por ejemplo.

El tiempo entre paquetes transmitidos coincide con el tiempo entre paquetes recibidos en esos momentos en los que no interfiere el flujo de 12 Mbps. Cuando incide el flujo de 12 Mbps, en algunos casos el tiempo entre paquetes en recepción resulta menor que el tiempo entre paquetes con el que se ha transmitido a costa del retraso en cola de paquetes anteriores, y esto indica que los paquetes son almacenados antes de ser transmitidos. CBQ mantiene unas colas para cada clase y al tener dos clases transmitiendo trata de dar el ancho de banda asociado mínimo a cada clase. Por lo tanto, los transmite con un tiempo entre paquetes tal que resulte la velocidad de transmisión deseada.

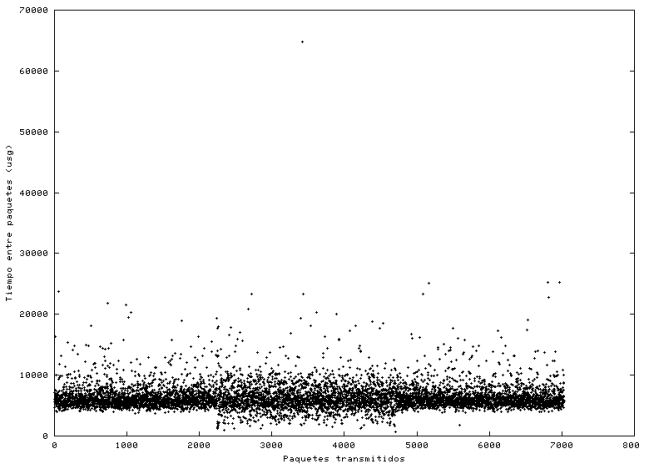

Figura 6. Tiempo entre paquetes recibidos, transmitidos con una distribución de Pareto a 1 Mbps y pasando por un planificador junto con un flujo de 12 Mbps en medio de la transmisión.

También se observa un pico, en el paquete 3500 aproximadamente, que no es de la transmisión sino generado en el planificador del router debido posiblemente a alguna tarea adicional del sistema operativo. Con el flujo interferente, el planificador debe atender las dos colas, y con la falta de precisión de reloj, tiene una granularidad que hace que algún paquete del flujo de 1 Mbps tenga que esperar más tiempo en cola del deseado produciendo un aumento del tiempo entre paquetes con el anterior paquete transmitido. Si el siguiente paquete de ese flujo se transmite en su momento, el tiempo entre paquetes con el anterior es menor que el tiempo entre paquetes con el que llegaron al planificador, creando tiempos entre paquetes menores en recepción respecto a los tiempos entre paquetes en transmisión. De este modo se aprecia una diferencia en la función de densidad del tiempo entre paquetes en recepción respecto en transmisión.

Con los otros flujos con los que se ha realizado la prueba, de 300 Kbps, 500 Kbps y 750 Kbps, ocurre el mismo efecto, pero no resulta tan marcado como en el flujo de 1 Mbps.

5. EFECTO DE LA PLANIFICACIÓN SOBRE FLUJOS UDP y TCP

La detección de congestión en los flujos TCP se basa en la pérdida de paquetes y por tanto en falta de asentimientos. Esto se traduce en una reducción a cero del tamaño de la ventana de transmisión del emisor, interrumpiendose temporalmente la transmisión de datos [15]. Se comprueba con la siguiente realización: se transmite un flujo TCP de 1 Mbps, al que consecutivamente se añaden flujos UDP interferentes de 2 Mbps, 1,5 Mbps, 3 Mbps y finalmente un flujo UDP de alta tasa, 5 Mbps. Todos estos flujos transmitidos simultáneamente requieren de un ancho de banda de 12,5 Mbps y por lo tanto congestionan la red B. La Figura 7 muestra el efecto de congestión en la red B tras el router sin ningún planificador configurado. Más adelante se compara con la respuesta del router con una planificación del tipo CBQ.

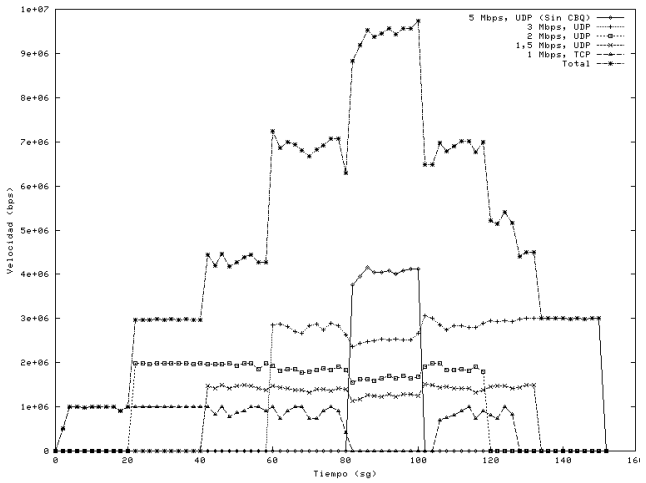

Figura 7. Flujo TCP con congestión sin planificador.

La ausencia del planificador hace que el flujo TCP no tenga una tasa garantizada y, por tanto, al detectar congestión en la red el emisor reduzca su ventana de transmisión a cero deteniendo la transmisión. En la Figura 7 se comprueba que se alcanza una velocidad máxima de 9,5 Mbps, a partir del segundo 82, limitada por la red B. En este punto la red está congestionada y por eso el flujo TCP cesa su transmisión para proseguirla cuando ya no detecte congestión, en el segundo 102 aproximadamente. El flujo TCP se continuará transmitiendo a la vez que reciba los asentimientos de los paquetes ya enviados.

En el caso de tener un tráfico de entrada al router que supera el ancho de banda de la red de salida (12,5 Mbps frente a 10 Mbps de la red), se puede apreciar como el comportamiento es Best Effort, comportamiento por defecto de Internet, resultando con mayor velocidad de transmisión el flujo cuya tasa de salida es mayor, el flujo UDP de 5Mbps.

Con una configuración en el router de un planificador tipo CBQ, que le asegura una tasa de 1 Mbps al flujo TCP de 1 Mbps, mediante la limitación del flujo UDP de 5 Mbps a una tasa de salida de 500 Kbps, la conexión TCP no se interrumpe a pesar de detectar congestión, como se observa en la Figura 8. En este caso la velocidad máxima que se alcanza es de 7,5 Mbps debida a los flujos de 1 Mbps, 5 Mbps (limitado a 500Kbps), 1,5 Mbps (limitado a 1 Mbps), 2 Mbps y 3 Mbps. El flujo TCP no transmite a tasa constante porque realmente detecta cierta congestión. Esto nuevamente hace dudar de la eficiencia del planificador CBQ de cumplir sus requisitos con la configuración actual.

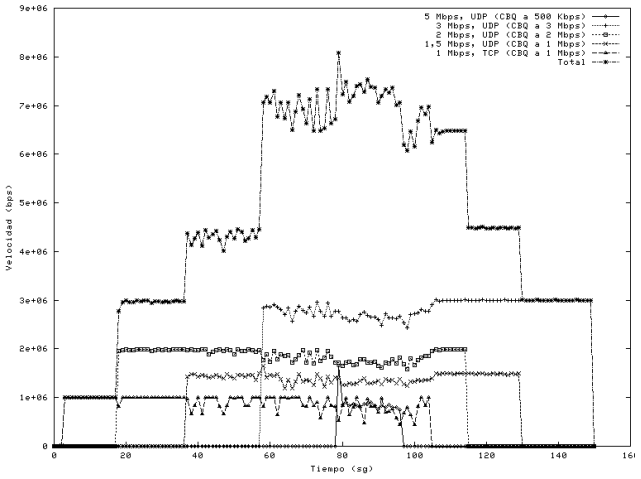

Figura 8. Flujo TCP con cogestión y planificador.

En la Figura 8 se puede apreciar como el flujo TCP no ha detenido su transferencia y los flujos UDP no han reducido su velocidad de transmisión por el hecho de llegar un nuevo flujo interferente de 5 Mbps que congestionaría toda la red de 10 Mbps. Por lo tanto los efectos que se observan son debidos al planificador y la limitación de ancho de banda de la red B de 10 Mbps.

5.1 Efecto de la planificación sobre la ventana de congestión en flujos TCP

En este caso se analiza la evolución del tamaño de la ventana de congestión de una conexión TCP, es decir, el número de paquetes que en cada momento transmite el protocolo de transporte sin esperar asentimiento. Este valor se obtiene directamente del kernel del sistema operativo del ordenador que transmite los datos con protocolo de transporte TCP. Para obtener dicho valor se ha modificado el kernel del sistema operativo consiguiendo un fichero en el directorio /proc del ordenador en el cual se va a medir la ventana de congestión. En el fichero creado se almacenan los tamaños de la ventana de congestión de todos los sockets TCP abiertos en la máquina, instantáneamente, para luego poder representar y analizar dichos datos. Para poder añadir este fichero se han tenido que modificar varios ficheros del protocolo TCP y el sistema proc del kernel.

Se ha comprobado como los flujos TCP detienen su transmisión cuando hay un flujo interferente de alta tasa (por ejemplo, en la Figura 7, entre los segundos 80 y 102, se muestra el efecto de un flujo interferente de 5 Mbps que congestiona la red B). Al colocar un planificador del tipo CBQ en dicho router se aprecia (en la Figura 8, entre los segundos 80 y 100) como el flujo TCP, con ancho de banda garantizado, no se detiene y mantiene su velocidad de transmisión. Resultaría interesante poder observar la ventana de control de congestión en ambas situaciones.

En la siguiente prueba, sin ningún tipo de planificador en el router, se ha transmitido un flujo TCP de 1,5 Mbps, y en mitad de esta transmisión se le ha añadido un flujo UDP de 12 Mbps. Al coincidir con el flujo UDP, en el segundo 17[1] de la Figura 9, el control de congestión del flujo TCP detiene la transmisión y la prosigue una vez acabado el flujo UDP, en el segundo 42 de la Figura 9. Debido al flujo UDP interferente de 12 Mbps el flujo TCP ha detenido su transmisión y por lo tanto, aparece la zona plana en el crecimiento de la ventana de congestión entre los segundos 27 y 52 de la Figura 10. A pesar de que el flujo UDP finaliza su transmisión en el segundo 33 de la Figura 9, el flujo TCP no vuelve a transmitir hasta el segundo 42 de la Figura 9. Esto es debido a que antes de continuar transmitiendo, debe recibir los asentimientos que no recibió debidos a la congestión de la red y detectar que ya no existe tal congestión.

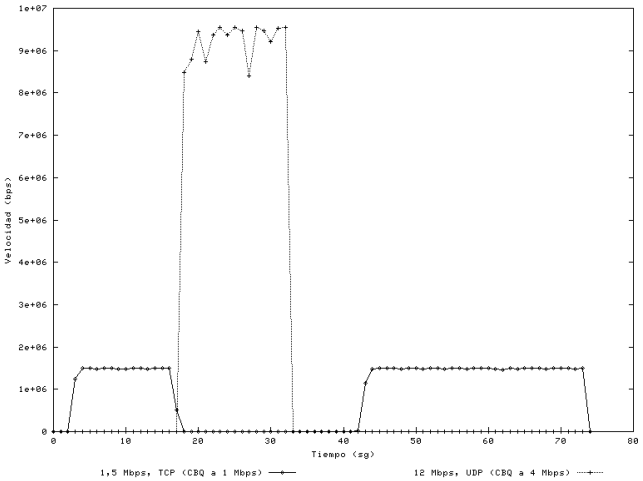

Figura 9. Flujos TCP y UDP sin planificador.

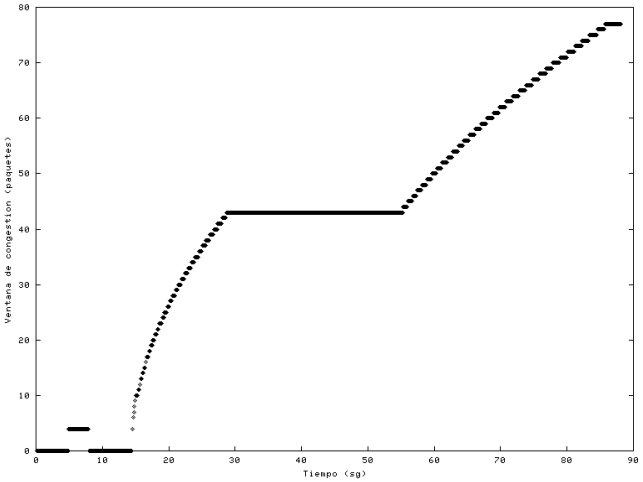

Figura 10. Ventana de congestión sin planificador.

Al observar la evolución del tamaño de la ventana de congestión se puede apreciar el efecto de la interferencia del flujo UDP. En la

[1] La Figura 9 y Figura 10 (al igual que la Figura 12 y Figura 11) tienen diferente temporización debido a que se han realizado en diferentes ordenadores, pero corresponden a la misma experiencia.

Figura 10 se observa como el valor de la ventana de congestión va aumentando con el transcurso de la transmisión (debido al efecto Slow Start) y se detiene cuando coincide con el flujo UDP interferente entre los segundos 27 y 52. El valor de la ventana de congestión no decrece hasta un valor nulo en estos momentos puesto que indica el número de paquetes enviados en espera de asentimiento por parte del receptor. Al coincidir con el flujo UDP interferente los paquetes enviados se pierden o los asentimientos se pierden o no se pueden transmitir y por eso el flujo TCP deja de transmitir más paquetes y sigue teniendo igual número de paquetes sin asentir.

Con la misma situación y usando un planificador CBQ que garantiza la velocidad de transmisión TCP de 1,5 Mbps a 1 Mbps y el flujo UDP de 12 Mbps a 4 Mbps, se transmite en el mismo orden que en el caso anterior.

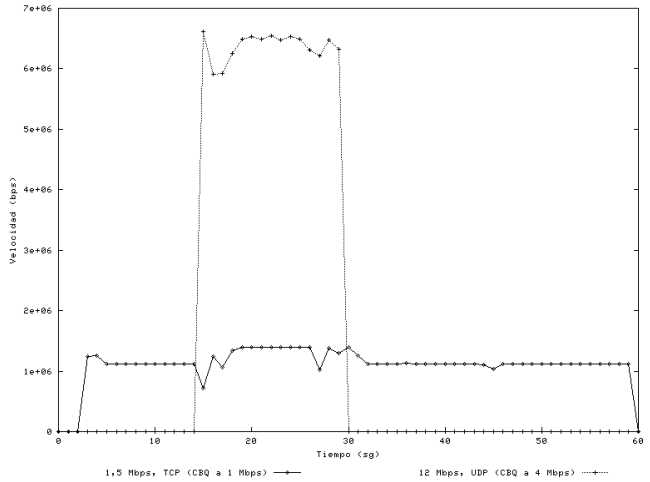

Figura 11. Flujos TCP y UDP con planificador.

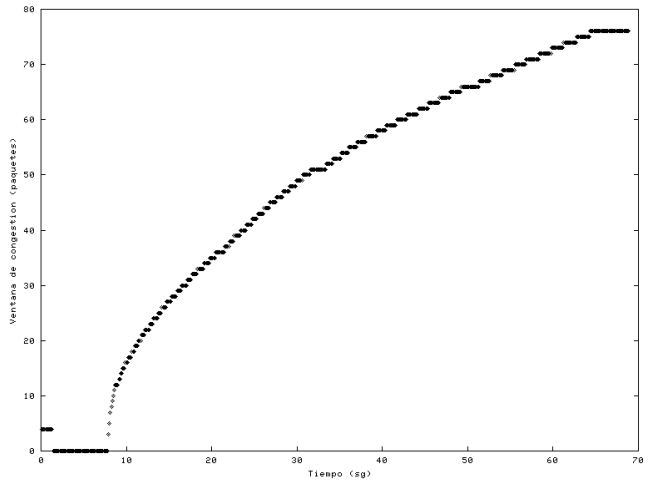

Figura 12. Ventana de congestión con planificador.

En esta nueva situación, el flujo TCP no detiene su transmisión a pesar de la interferencia del flujo UDP de 12 Mbps. En la Figura 12 se observa el valor de la ventana de congestión que corresponde a esta prueba. No se detecta el efecto del flujo interferente UDP sobre la ventana de congestión como tampoco en la velocidad de transmisión como se observa en la Figura 11.

El análisis de los resultados inducen a considerar un conjunto de modificaciones sobre el sistema dónde se ha implementado el planificador para garantizar una cierta calidad de servicio. Estas modificaciones y el efecto de las mismas se presentan a continuación.

6. EFECTO DE LA PRECISIÓN DE RELOJ EN LA PLANIFICACIÓN DE SERVICIOS DE RED

En este apartado se plantean un conjunto de modificaciones sobre el sistema de planificación que se relacionan con la precisión de reloj del sistema operativo Linux. Estas modificaciones tienen un efecto positivo sobre la planificación de servicios, pero reducen el uso de CPU disponible para el planificador. Sin embargo, como se muestra en este apartado, el consumo de CPU no es grande y no afecta en gran medida al planificador, el cual, sin embargo, aumenta su precisión a la hora de limitar y garantizar tasas a los diferentes flujos, mejorando estas garantías para flujos de tasas mayores.

Para la modificación de la precisión de reloj se hace uso de dos variables propias del kernel que permiten el ajuste de la precisión de reloj. Estas variables son PSCHED_CLOCK_SOURCE y HZ.

El parámetro PSCHED_CLOCK_SOURCE puede tomar tres valores:

- PSCHED_JIFFIES (valor que toma por defecto): Se basa la precisión del reloj en intervalos pequeños de tiempo (jiffies). En arquitecturas PC tiene una granularidad de reloj con HZ=100, de centésimas de segundos, y no es suficiente para tareas en tiempo real. Esta granularidad es, por ejemplo, con la que funciona el planificador de procesos del sistema.

- PSCHED_GETTIMEOFDAY: Realiza la petición de hora necesitando para ello varias instrucciones de ensamblador para rellenar una estructura no natural que separa segundos de microsegundos.

- PSCHED_CPU: Es un valor que no todos los procesadores pueden tomar, únicamente los procesadores Alfa y Pentium de última generación. Realiza la petición de la hora con una sola instrucción de ensamblador, resultando ser el valor que mejor precisión de reloj ofrece.

En cuanto a la variable HZ, ésta indica el número de interrupciones que se realizan en el kernel en un segundo. Por defecto tiene un valor de 100. Obviamente cuanto mayor sea este valor mayor será la precisión obtenida. Esto implica, a su vez, un mayor consumo de CPU por parte del sistema operativo para el control de tiempos e interrupciones.

A continuación, y haciendo uso de las modificaciones expuestas anteriormente, se analizan las prestaciones del planificador en cuanto al efecto sobre el retardo extremo a extremo, las ventajas

sobre la limitación de la tasa de salida del router y se determina el bajo coste que supone para la CPU proporcionar una mayor precisión mejorando el funcionamiento del planificador.

6.1 Retardo extremo a extremo

Por medio de una función análoga al ping se puede realizar un estudio del tiempo extremo a extremo. Para ello se activa el puerto 7 (de echo) en el receptor y se le mandan paquetes con el time-stamp de transmisión para compararlo con el tiempo de recepción. Se transmite este paquete cada segundo transmitiéndose tantos paquetes como segundos pasan.

Una vez iniciada la transmisión de paquetes para determinar el retardo extremo a extremo (RTT), diez segundos después se añade un flujo UDP interferente de 1 Mbps. Pasados otros diez segundos se añade otro flujo UDP interferente de 1 Mbps, teniendo así un total de 2 Mbps de flujo interferente. Se sigue así sucesivamente, cada 10 segundos, hasta tener simultáneamente 10 flujos UDP de 1 Mbps cada uno, produciendo un flujo interferente total de 10 Mbps (límite de congestión de la red B).

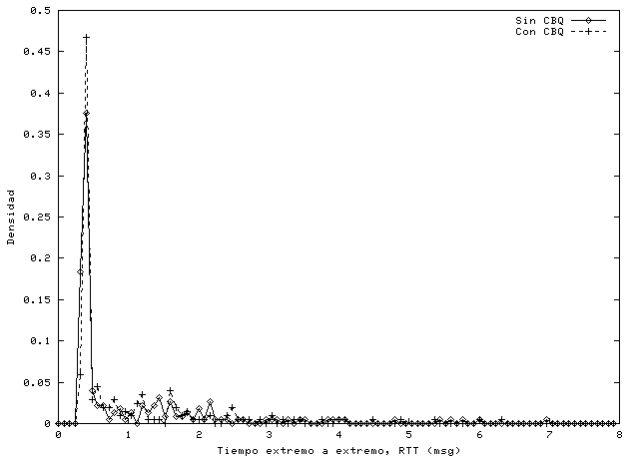

Figura 13. Función de densidad del tiempo extremo a extremo con y sin planificador con flujo interferente.

En la Figura 13 se tiene la función de densidad de los valores de tiempo extremo a extremo, bien sin ningún tipo de planificación como con un planificador del tipo CBQ. Se puede observar como apesar de tener un valor de tiempo extremo a extremo predominante de 0,4 msg, existe una notable variación que muestra un aumento del tiempo extremo a extremo debido a los flujos interferentes. Se comprueba que el uso del planificador mejora muy poco el valor del tiempo extremo a extremo, siendo ambas curvas muy parecidas.

El planificador CBQ no introduce gran mejora en el tiempo extremo a extremo a no ser que se mejore la precisión de reloj del sistema en el que se implementa. Para ello sólo hay que ver los datos representados en la Tabla 1.

Tabla 1. Valores medios de RTT con diferentes precisiones de reloj.

PSCHED_CLOCK_SOURCE	PSCHED_JIFFIES	PSCHED_CPU
HZ=100	3,14 msg	1,82 msg
HZ=1024		1,60 msg
HZ=10000		0,85 msg

Se observa como el colocar el planificador CBQ mejora muy poco el RTT debido a la falta de precisión de reloj del planificador. La modificación sobre la precisión del reloj consigue alcanzar unos valores medios para el RTT que se presentan en la Tabla 1. Este valor medio ha sido calculado en iguales condiciones junto con un flujo interferente de 12 Mbps al que se le garantiza un ancho de banda mínimo de 2 Mbps mediante una disciplina de cola CBQ.

Se puede apreciar como el primer cambio de la variable PSCHED_CLOCK_SOURCE de un valor inicial de PSCHED_JIFFIES a un valor de PSCHED_CPU, mejora considerablemente la media, bajando de 3,14 msg valor medio de RTT a 1,82 msg. Después, con el cambio del valor de la variable HZ se aprecia que el valor medio del tiempo extremo a extremo, RTT, se ve menos afectado por el flujo interferente. Al mejorar la precisión de reloj en el sistema del planificador, los paquetes que llegan al puerto 7 cada segundo son tratados más rápidamente en el planificador, aún cuando está interfiriendo un flujo de 12 Mbps. Por lo tanto se puede asegurar que el planificador del tipo CBQ cumple mejor sus requisitos cuanto mayor es la precisión de reloj, haciendose notable la mejoría en la calidad de servicio implementando el planificador CBQ en el router.

6.2 La precisión de reloj y la planificación

Como ya se ha visto, el control de tráfico del sistema operativo Linux ofrece la posibilidad de colocar una disciplina de cola en la interfaz de red. Si esta disciplina de cola es del tipo CBQ se puede seleccionar otra disciplina de cola diferente en cada clase de salida. Entre las disciplinas de cola que se pueden añadir están la disciplina FIFO, la de prioridades, la del tipo CBQ y el TBF.

6.2.1 Limitador de tasa TBF

En esta sección se analiza el efecto de la precisión de reloj a la hora de limitar la tasa de salida máxima que ofrece una disciplina de cola tipo TBF en Linux.

En una primera prueba, para conocer las prestaciones del planificador, la precisión de reloj en el sistema operativo es la que tiene por defecto y por lo tanto se tienen los valores de PSCHED_CLOCK_SOURCE con PSCHED_JIFFIES y HZ igual a 100 [10][11]. Para esta simulación se han transmitido una serie de flujos UDP a 1 Mbps, que van a ir a diferentes clases, dentro de una primera disciplina de cola CBQ acabando cada una de las colas de cada clase con una disciplina de cola TBF. Cada clase tiene una limitación de tasa diferente, empezando por 100 Kbps, e

incrementando cada vez 100 Kbps, hasta llegar a 900 Kbps. Esto quiere decir que deben salir limitados los flujos a 100 Kbps, 200 Kbps, 300 Kbps, ... hasta 900 Kbps. También se transmiten 3 flujos UDP a 1,4 Mbps para limitar su tasa de salida a 1 Mbps, 1,1 Mbps y 1,2 Mbps, respectivamente. Por último se transmite un flujo de 3 Mbps para limitarlo a 2 Mbps para observar la limitación a mayores tasas de salida. Cada flujo limitado es una ráfaga de la Figura 14.

La limitación de tasa hasta 400 Kbps ha sido satisfactoria, pero no se ha podido limitar a 500 Kbps, sino que lo ha limitado a una tasa menor de 400 Kbps. Lo mismo ocurre con las limitaciones de 700 Kbps, 800 Kbps, 900 Kbps y 1 Mbps, que se han limitado a un poco menos de 600 Kbps. El tamaño de los paquetes, es constante de 1400 bytes para todos los flujos, para así necesitar tiempos entre paquetes no muy reducidos para tasas altas. Con estos datos se aprecia que la elección de estas velocidades a las que se limitan los flujos son debidas a la precisión de reloj del sistema en el que se tiene el planificador. Estas tasas de salida son siempre con tiempos entre paquetes múltiplos de 0,01 sg (HZ = 100), y por el hecho de transmitir a un tamaño de paquete constante para todos los flujos se obtiene el resultado visto.

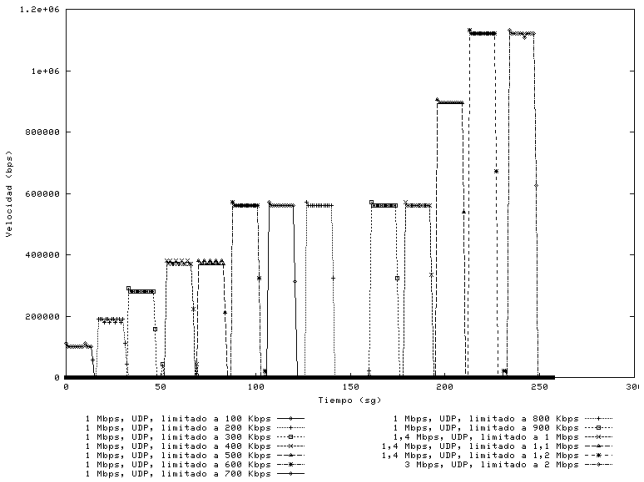

Figura 14. Limitación de tasa por TBF.

Los valores de tasas de salida máxima que proporciona la disciplina de cola TBF se ven mejorados con una mayor precisión de reloj como se aprecia en la Figura 15.

Cambiando la variable PSCHED_CLOCK_SOURCE del valor PSCHED_JIFFIES (valor por defecto) al PSCHED_CPU (ya se nota cierta mejora, pero no toda la deseada, corroborando con la Tabla 1) y el valor de la variable HZ de 100 a 1024 se puede apreciar en la Figura 15 como se consiguen las tasas de salida deseadas. Todavía el flujo que debería haber alcanzado una tasa de 2 Mbps no la logra quedándose con una tasa de 1,9 Mbps. Este flujo también consigue su tasa de salida deseada de 2 Mbps con un nuevo cambio en la variable HZ a 10000.

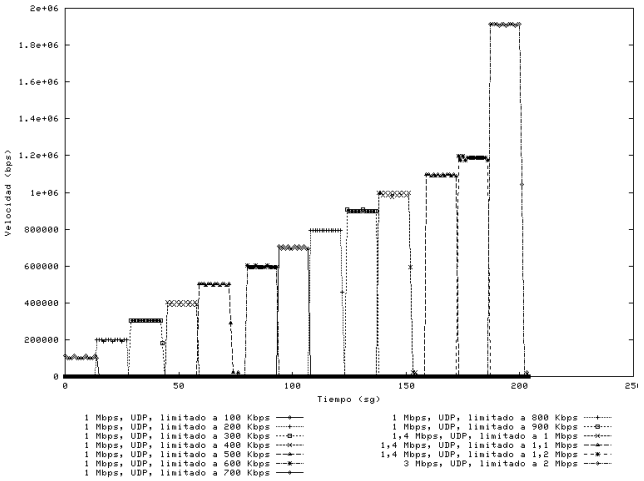

Figura 15. Tasas de salida TBF con PSCHED_CPU y HZ=1024.

Con estas figuras se puede concluir que la falta de precisión de reloj es lo que hace que no se consigan las tasas de salida máximas deseadas, independientemente de los cálculos y procedimientos internos del planificador. De modo que se puede asegurar que el limitador de tasas TBF es sensible a la precisión de reloj, pero que cumple perfectamente sus requisitos si se aumenta esta precisión.

6.2.2 Ancho de banda mínimo de CBQ

En la Figura 16 se tiene precisión de reloj con los valores de las variables en PSCHED_CPU y HZ=10000 y se observa la limitación de tasa conseguida con CBQ. En ella se transmite un flujo UDP a 1.5 Mbps limitado con CBQ a 1 Mbps siendo interferido por otro flujo UDP de 12 Mbps limitado a 6 Mbps. Se observa como al interferir el segundo flujo entre los segundos 14 y 30, el flujo de 1.5 Mbps aumenta su tasa por encima del Mbps al que estaba limitado.

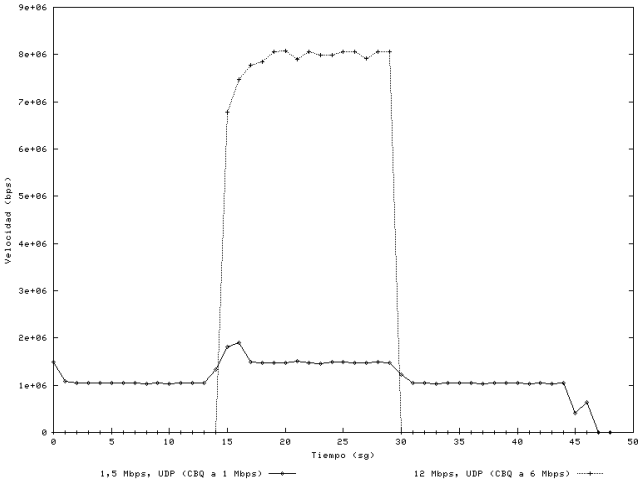

Figura 16. Limitación de tasa de salida con CBQ (PSCHED_CPU y HZ=10000)

Al contrario que con el planificador de tipo TBF, la limitación de tasa máxima no se mejora en una planificación del tipo CBQ con una mejor precisión de reloj. Esto es debido a que la característica del planificador CBQ es garantizar un ancho de banda mínimo y no la de limitar la tasa de salida máxima.

Los flujos con un tamaño de paquete pequeño requieren de un tiempo entre paquetes también pequeño para obtener una determinada velocidad de transmisión. Por lo tanto, al tener mejor precisión de reloj, a este tipo de flujo con tamaño de paquetes pequeño se le consigue asegurar mejor su ancho de banda mínimo con el planificador tipo CBQ. Para demostrarlo se puede ver la Figura 17 en la que la precisión de reloj viene determinada por los valores PSCHED_CPU y HZ=100. El flujo UDP de 2 Mbps es el que tiene un tamaño de paquetes pequeño y es el flujo que más se ve perjudicado con la presencia del nuevo flujo UDP de 5 Mbps. Este fenómeno se observa en la Figura 17 entre los segundos 40 y 61, dónde el flujo de 2 Mbps reduce su tasa a casi 1 Mbps al aparecer en ese intervalo el flujo de 5 Mbps limitado a 2 Mbps.

Con la variable HZ=1024, la precisión es mayor y se consigue garantizar con el planificador CBQ el ancho de banda asociado a este flujo de 2 Mbps, con tamaño de paquete pequeño, de 2 Mbps. Se puede observar, entre los segundos 40 y 60 de la Figura 18, como el flujo UDP de 2 Mbps no reduce su velocidad de transmisión, como se quería garantizar.

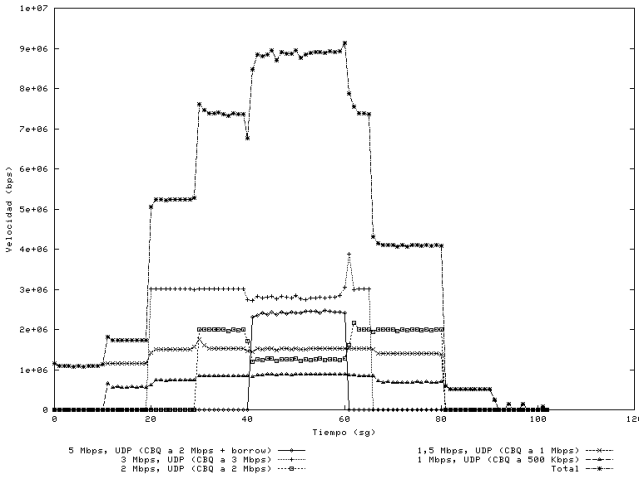

Figura 17. Diversos flujos con CBQ (PSCHED_CPU y HZ=100).

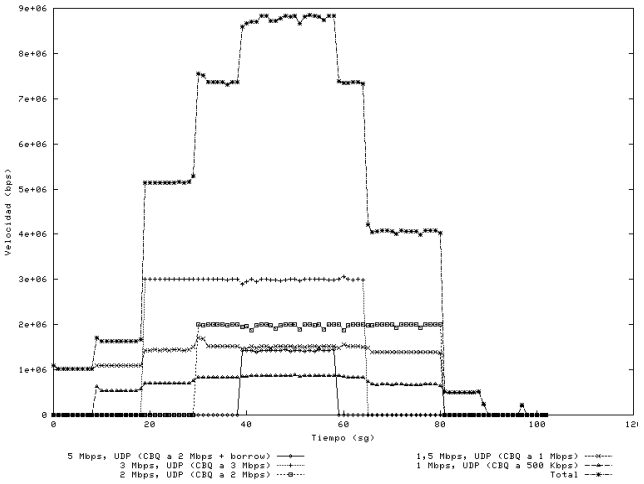

Figura 18. Diversos flujos con CBQ (PSCHED_CPU y HZ=1024)

También hay que comentar el incremento de velocidad de transmisión que se aprecia en los primeros flujos UDP de 1,5 Mbps (con 1 Mbps garantizado y limitado con CBQ) y 1 Mbps (con 500 Kbps garantizado y limitado) con cada flujo nuevo añadido. Estos incrementos se aprecian en los segundos 10, 20 y 30 aproximadamente de la Figura 17, y con la precisión de reloj mejorada también en la Figura 18. Es un efecto que la disciplina de cola CBQ no consigue controlar apesar de cambiar la precisión de reloj. Se puede concluir que la disciplina de cola CBQ garantiza un ancho de banda mínimo que sí depende de la precisión de reloj, pero no limita la velocidad máxima de transmisión de salida.

6.3 Precisión de reloj y consumo de CPU

Una vez comprobado que las prestaciones del planificador (bien sea del tipo TBF como CBQ) mejoran al aumentar la precisión de reloj, se trata de determinar en que medida afecta al consumo de CPU esta mejora.

Tabla 2. Porcentaje de uso de CPU sin planificación.

PSCHED_CLOCK_SOURCE	PSCHED_JIFFIES	PSCHED_CPU
HZ=100	①	②=①+0,6%
HZ=1024	①+1,2%	②+0,9%
HZ=10000		②+11,8%

Para obtener una mayor precisión de reloj se utilizan valores de HZ mayores y estos requieren de mayor uso de CPU, como es de esperar. Con objeto de medir el consumo de CPU se toma como punto de referencia el consumo de CPU debido a las tareas de administración del propio sistema operativo; es decir, sin ningún proceso de usuario en ejecución y siendo la variable HZ=100 y con PSCHED_CPU, se toma el valor de referencia llamado ② (Tabla 2).

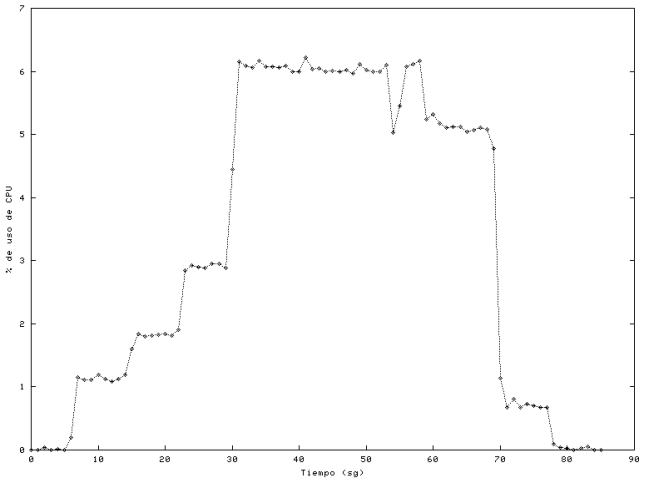

Figura 19. Porcentaje de uso relativo con PSCHED_CPU y HZ=100.

Por el hecho de realizar una planificación, y más concretamente el de la prueba de la Figura 17, el kernel requiere de un consumo de CPU de un 6 % más que en el estado de reposo ② como se muestra en la Figura 19 (cuando coinciden todos los flujos a la vez entre los segundos 40 y 60 de la Figura 17). Respecto de esta referencia ②, por el mero cambio de la variable HZ a un valor de 10000 y sin ningún proceso de usuario corriendo, se requiere de un consumo de CPU de un 11,8 % mayor respecto de ②, sólo para tareas del propio sistema operativo. En este caso de HZ=10000, al realizar el proceso de planificación el pico de máxima utilización de CPU es un 7% mayor respecto el estado en reposo. Con respecto al punto de referencia definido anteriormente ②, el pico de máxima utilización (HZ=10000) supone un incremento de un 19,7% de consumo de CPU, mientras que para HZ=100 era del 6%. Al mejorar la precisión de reloj, aumenta naturalmente el consumo de CPU. Una mejora de precisión de reloj a HZ=10000, 100 veces mayor, supone un aumento del 11,8% para el caso de reposo, lo que es un aumento del CPU totalmente asumible. Mientras que para HZ=100 el planificador supone un consumo máximo en planificación del 6%, para HZ=10000 supone un consumo máximo en planificación del 19,7% respecto al reposo de HZ=100, siendo un 12,9% más respecto el máximo consumo en planificación de HZ=100.

7. TRABAJOS RELACIONADOS

En los estudios sobre planificadores existentes como los trabajos [4][7][14] exponen el funcionamiento de planificadores CBQ sobre sistemas Linux y BSD limitándose a la verificación de sus propiedades sobre diferentes interfaces de red como Ethernet y ATM.

Sin embargo, en el presente trabajo se ha conseguido mejorar el funcionamiento de un planificador sobre un sistema operativo de propósito general como es Linux a costa de aumentar la precisión de reloj con diferentes mecanismos existentes.

Como línea de trabajo queda por comprobar la utilización de relojes externos de alta precisión que cooperen con el sistema operativo sin suponer una carga de CPU extra.

8. CONCLUSIONES

En este trabajo se ha comprobado en primer lugar que el algoritmo de planificación CBQ no es buen limitador de tasa, pero sí garantiza el ancho de banda mínimo. Este ancho de banda se garantiza con mayor eficacia si se tiene una mejor precisión de reloj en el planificador. Los flujos con paquetes pequeños son los más afectados si la precisión de reloj no es muy buena, puesto que para conseguir una determinada velocidad de transmisión requieren de un tiempo entre paquetes más pequeño y por lo tanto mayor precisión de reloj.

El planificador de disciplina de cola tipo CBQ tiene la opción del préstamo de ancho de banda lo cual mejora el rendimiento del enlace. El préstamo de ancho de banda se basa en que si existe ancho de banda disponible en el enlace, no es necesaria la pérdida de paquetes para aquellos flujos que no tienen garantizado completamente el ancho de banda necesario. Supone no restringir la tasa de salida del flujo de una clase determinada, existiendo ancho de banda disponible en la red.

Este planificador CBQ también permite la convivencia de flujos de diferente protocolo de transporte como pueden ser TCP y UDP. También evita la reducción de velocidad de transmisión debida al control de congestión de los flujos TCP garantizándoles un ancho de banda mínimo, frente flujos UDP de alta tasa de transmisión. Se puede asegurar que presenta robustez ante la presencia de flujos interferentes.

La clasificación de flujos en diversas clases se realiza mediante la configuración de filtros y estos pueden basarse en cualquier parámetro de cabecera de los paquetes. Una de las características de CBQ es el filtrado jerárquico. De esta forma se configura un mejor reparto de ancho de banda del enlace, dependiendo de tantos parámetros como se desee.

El limitador de tasas TBF, como disciplina de cola que también ofrece el control de tráfico en Linux, es fuertemente dependiente de la precisión de reloj a la hora de limitar la tasa de salida, principalmente para tasas elevadas. Las velocidades de transmisión de salida deseadas se consiguen con una precisión de reloj mayor. A mayor precisión de reloj del sistema más ajustada es la tasa de salida a la tasa deseada.

La gran problemática encontrada ha sido la precisión de reloj. Para las labores de planificación a alta velocidad la precisión de reloj existente en sistemas operativos de propósito general como Linux, es mala, impidiendo tratar los paquetes con alta precisión temporal para un mejor rendimiento.

Para aumentar esta precisión de reloj, se pueden modificar algunos parámetros del kernel. Los parámetros que se ha probado cambiar han sido la variable PSCHED_CLOCK_SOURCE y la variable HZ. El efecto observado ha sido un pequeño aumento del uso de

CPU, pero una gran mejora respecto a la precisión en la velocidad de transmisión de salida de los flujos. El planificador CBQ garantiza mejor un ancho de banda mínimo y TBF obtiene la tasa de salida más ajustada a la requerida.

Se puede modificar la precisión de reloj y observar diferentes aspectos que las diferentes precisiones de reloj proporcionan. El porcentaje de uso de la CPU no se ha incrementado tanto como se esperaba para una mayor precisión de reloj. Por lo tanto este incremento en la precisión de reloj es muy interesante.

Con las pruebas realizadas y la mejora de la precisión de reloj se ha visto que es posible la implementación de un planificador, para garantizar calidad de servicio, sobre una plataforma de bajo coste como es un PC con Linux, por lo menos a bajas y medias velocidades (10/100 Mbps).

9. REFERENCIAS

[1] W. Almesberger. Linux Network Traffic Control, Implementation Overview. Technical Report SSC/1998/037, EPFL ICA, November 1998. ftp://lrcftp.epfl.ch/pub/people/almesber/pub/tcio-current.ps.gz.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. An Architecture for Differentiated Services. IETF RFC2475, December 1998.

[3] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin. Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification. IETF RFC2205, September 1997.

[4] K. Cho. A Framework for Alternate Queueing: towards Traffic Management by PC-UNIX based routers. In USENIX Annual Technical Conference, New Orleans, Louisiana, 1998.

[5] G. Dhandapani and A. Sundaresan. Netlink Sockets Overview. Technical Report The University of Kansas, September 1999.

[6] S. Floyd and V. Jacobson. Link Sharing and Resource Management Models for Packet Networks. IEEE/ACM Transaction on Networking, Vol.3, No.4, pp 365-386, August 1995.

[7] A. Gómez-Skarmeta and A. López-Toledo. Control de tráfico mediante Class-Based Queueing gestionado por RSVP usando Linux como router con capacidad QoS. In Proceedings of II Jornadas de Ingeniería Telemática JITEL'99, Leganés (Madrid), September 1999.

[8] J. Hadi-Salim. IP Quality of Service on Linux. Technical Report Nortel Network, Ottawa, Canada, July 1999. http://www.davin.ottawa.on.ca/ols

[9] W. Leland, M.S. Taqqu, W. Willinger and D. Wilson. On the self-similar nature of ethernet traffic (extended version). IEEE/ACM Transactions on Networking, Vol.2, No.1, pp 1-15, February 1994.

[10] J. C. Mogul. Efficient Use of Workstations for Passive Monitoring of Local Area Networks. Proceedings of ACM SIGCOMM Symposium on Communications Architectures and Protocols, Philadelphia, PA, September 1990.

[11] V. Paxson, G. Almes, J. Mahdavi and M. Mathis. Framework for IP Performance Metrics, IETF RFC2330, May 1998.

[12] V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. In SIGCOMM '94, pages 257-268, August 1994.

[13] S. Radhakrishnan. Linux-Advanced Networking Overview Versión 1. Technical Report KS 66045-2228 Information and Telecommunications Technology Center, Department of Electrical Engineering & Computer Science, The University of Kansas. Lawrence, September 1999. http://qos.ittc.ukans.edu/howto/howto.html

[14] F. Risso and P. Gevros. Operational and Performance Issues of a CBQ router. Computer Communication Review, Volume 29, Number 5, October 1999.

[15] N.J. Shah. Exploring TCP Stream Rate Control in LAN Environments. PhD Thesis, Universidad de California, Riverside, June 1999.

[16] G. Pawan and M.V. Harrick. Generalized Guaranteed Rate Scheduling Algorithms: A Framework. IEEE/ACM Transactions on Networking, Vol. 5, nº 4, p.56661-71, August 1997.