Universidad Politécnica de Cartagena
E.T.S. de Ingeniería de Telecomunicación

Espacio-Tele□o n^{0}2 (2011)
Revista de la ETSIT-UPCT

Medidas de capacidad en entornos P2P usando técnicas de dispersión de paquetes

David Montoro Mouzo, Jose María Malgosa Sanahuja

Departamento de Tecnologías de la Información y las Comunicaciones
Universidad Politécnica de Cartagena

josem.malgosa@upct.es;david.montoro@upct.es

Resumen

En el presente artículo se recopilan y analizan los trabajos realizados en el seno de nuestro grupo de investigación en el campo de la medición de capacidades en redes *peer-to-peer* utilizado técnicas de dispersión de paquetes. Así, se analizará la importancia que este tipo de mediciones tiene asociadas y se expondrán las técnicas y fundamentos necesarios para llevarlas a cabo. En este punto se detallará la herramienta de medición implementada así como la técnica de medición desarrollada. Igualmente, se expondrán los resultados obtenidos y se analizará la bondad de nuestra herramienta de medición mediante el análisis de los diferentes escenarios posibles en una implementación real de una red *peer-to-peer* típica hecha a tal efecto. Se finalizará exponiendo las líneas futuras de investigación que se abren tras la realización de estos trabajos.

Proyecto/Grupo de investigación: Grupo de Ingeniería Telemática. Plan Nacional I+D+i "Characterization, evaluation, planning and improvement of key technologies for the future Internet: knowledge and transfer (CALM)". TEC2010-21405-C02-02.

Líneas de investigación: *Redes Overlay*; *Aplicaciones peer-to-peer*.

1. Introducción

Las motivaciones existentes en la búsqueda de la obtención de una herramienta que posibilite la medición del ancho de banda de cuello de

botella en una red *peer-to-peer* surgen de dos puntos principales. Por un lado, hoy en día Internet está formada por la interconexión de gran cantidad de redes IP que siguen una política de enrutamiento independiente y que se denominan Sistemas Autónomos (o AS, del inglés *Autonomous Systems*). Por norma general, un sistema autónomo está controlado por una única entidad o corporación (típicamente un proveedor de servicios de comunicaciones) que proporciona un servicio a los usuarios finales de la red. La conexión entre usuarios finales pertenecientes a AS diferentes es muy común, de tal manera que la necesidad interconexión entre diferentes AS entre si. Esta interconexión puede realizarse directamente entre los AS implicados o vía un tercer AS que actúa como intermediario. En el segundo caso, el AS intermedio cobra una tarifa a los otros AS interconectados por utilizar una determinada capacidad de su red. Esta situación a potenciado la investigación en herramientas fiables que sean capaces de monitorizar que el servicio dado por el AS intermediario se hace acorde a contrato. En este sentido, aquí se va a proponer un sistema capaz de utilizar los *peers* presentes en los AS interconectados como sondas de medida que permitan auditar si el enlace que los interconecta tiene la capacidad contratada.

Por otra parte, es un hecho constatado que la arquitectura de Internet está avanzando cada vez más hacia las redes *overlay* con el propósito de proveer servicios adicionales de manera más eficiente y con una mejor *QoS*. Las redes *overlay*, como por ejemplo las redes *P2P*, son redes de computadores que están implementadas sobre otra red utilizando enlaces lógicos. En este tipo de redes, el enrutamiento es una cuestión más compleja debido a la complejidad de los caminos entre los pares. Esta heterogeneidad de caminos crea la necesidad de estimar el ancho de banda de cuello de botella de esos caminos (es decir, el mínimo ancho de banda de un salto en el camino entre esos pares de *peers* -que será el que limite la velocidad de transmisión entre ellos-) con la finalidad de obtener un enrutamiento lo más óptimo posible. Como un primer paso de cara a utilizar la información relativa a la capacidad para mejorar los algoritmos de enrutamiento *P2P* actuales, se ha desarrollado una herramienta capaz de acumular esta información a lo largo de un camino utilizando técnicas de dispersión de paquetes.

Para desarrollar esta herramienta se ha partido de los trabajos realizados por Harfoush en [3], incluyendo esta un módulo instalable del *kernel* de Linux, aplicaciones en el espacio de usuario de Linux y de procesado de datos y decisión en base a las cuales estimar la capacidad.

1.1. Conceptos básicos en la medida de anchos de banda utilizando técnicas de dispersión de paquetes

De cara a que el lector pueda comprender de manera clara lo que resta de artículo, se definirán de manera unívoca diversos conceptos. Estos conceptos puede clasificarse en tres categorías: topológicos, relacionados con el ancho de banda y relacionados con las técnicas de dispersión de paquetes. Primero, en cuanto a los conceptos topológicos se refiere, se ha de tener en cuenta

que se busca medir el ancho de banda en los enlaces que componen el camino que une dos usuarios finales (o *end-to-end path*) . En este contexto se considerará segmento a cada uno de los enlaces punto a punto físicos (capa de enlace de datos) que conforman ese camino; y se considerará salto a cada uno de los enlaces en la capa IP (típicamente compuestos por varios segmentos y conectados por routers). La herramienta implementada puede medir de forma precisa la capacidad de los diversos saltos que conforman el camino extremo a extremo.

Relacionadas con el ancho de banda existen diversas métricas definidas. El término *throughput* denota el número de bytes transferidos en un *path* de una red durante una cantidad determinada de tiempo. En lo referente a este artículo, las capacidades de los enlaces se definen como el máximo *throughput* alcanzable en ese enlace. En cuanto a nomenclatura, a lo largo de este documento para referirse a la capacidad del enlace i-ésimo de un *path* se usará b_i. Por ancho de banda de cuello de botella de un path (o *bottleneck bandwidth*) se entenderá el máximo *throughput* que se puede idealmente obtener a lo largo de ese *path*, y estará marcado por la capacidad del enlace más lento. Lo denotaremos por b_{bw}.

En el contexto de la medida de la capacidad de los saltos utilizando técnicas de dispersión de paquetes, se entiende por sonda (o *probe*) a una secuencia de uno o más paquetes transmitidos desde un origen común con la finalidad de obtener información de los tiempos de llegada (de todos o solo de algunos de ellos) a un origen. Normalmente son transmitidos *back-to-back*, sin separación temporal entre las transmisiones de los paquetes individuales. Si todos los paquetes de la sonda son iguales se dirá que se está ante una sonda uniforme. De lo contrario, se referirá a la sonda como una sonda no uniforme. En cuestión de nomenclatura, denominaremos *s(p)* al tamaño del paquete p. Por tiempo entre llegadas de un par de paquetes en un enlace (o dispersión) nos estaremos refiriendo al tiempo transcurrido entre la llegada a ese enlace del último *byte* del primer paquete y la llegada del último *byte* del segundo paquete. Nos referiremos a la dispersión producida en el enlace i-ésimo como t_i, y será el resultado del cociente entre el tamaño del paquete y la capacidad de dicho enlace.

2. Técnicas de medida de dispersión de paquetes utilizadas

2.1. Técnica Packet Pair

Las técnicas utilizadas por la herramienta desarrollada son una generalización de las técnicas utilizadas en [3]. Estas técnicas están basadas en la técnica *packet pair*, propuesta originalmente por Keshav [5], cuyo propósito es obtener el ancho de banda de cuello de botella del camino extremo a extremo mediante el envío de una sonda uniforme de dos paquetes ($s(P_1) = s(P_2)$) enviada *back-to-back*. Así, si se planteara un escenario con tres saltos en el cual $b_1 = b_3 > b_2$, la dispersión de los paquetes a lo largo del *path* sería la de la Figura 1. Entonces, el b_{bw} del *path* se obtendrá a partir de la dispersión de los paquetes en el enlace de destino haciendo uso de la Ecuación 1. Mediante la

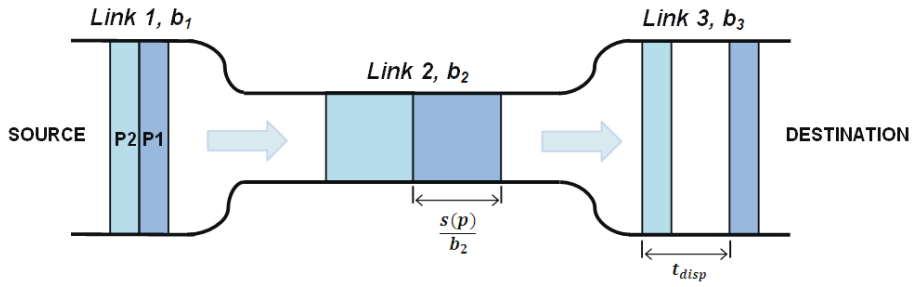

Figura 1: Ejemplo esquemático de la técnica Packet Pair (adaptación de [5]).

técnica *packet pair* se conoce el ancho de banda de cuello de botella del camino pero no se puede saber qué enlace del camino es el cuello de botella ni cuál es la capacidad de los distintos enlaces, por lo que es necesario mejorar y generalizar esta técnica de cara a obtener la capacidad de resolver los problemas detallados en la introducción.

$$b_{bbw} = \frac{s(P2)}{t_{disp}} \tag{1}$$

2.2. Técnicas de dispersión generalizadas

Con la finalidad de obtener herramientas suficientes para lograr determinar la capacidad de los saltos que conforman un camino entre dos usuarios finales se ha de generalizar la técnica *packet pair*. Para ello se ha de ser capaz de realizar tres tipos de medidas utilizando las técnicas de dispersión de paquetes (para lo que hay que definir tres tipos de sondas). En [3] se detalla un procedimiento para obtener la capacidad de todos los saltos de un camino utilizando estos tres tipos de sondas, por lo que para la herramienta implementada se partió de las técnicas ahí detalladas y se realizaron modificaciones con la finalidad de poder obtener una implementación práctica viable. A continuación se expondrán las bases de los tres tipos de sondas y se detallará cómo usarlas de cara a la medición de la capacidad de un enlace específico.

2.2.1. Medidas extremo a extremo: técnica *packet pair*

Las medidas extremo a extremo siguen el procedimiento general de *packet pair*. Se puede generalizar (y en este caso así se ha hecho) enviando un mayor número de paquetes y tomando la dispersión entre el primero y el último. Esto permite la obtención de resultados más fiables al ser los tiempos de dispersión mayores.

2.2.2. Medidas de prefijos: *técnica cartouche*

Se entiende por prefijo una serie de saltos del camino extremo a extremo que empieza en el *host* origen pero que no llega al *host* destino. El reto que presentan este tipo de medidas es conservar la dispersión producida al final del prefijo hasta el destino. Suponiendo que se envié una prueba *packet pair* al uso, la dispersión temporal en el host final solamente reflejaría el b_{bw} del prefijo si la capacidad de todos los saltos restantes es mayor que éste. Si este requisito no se cumple, la dispersión temporal del par de paquetes del prefijo se verá aumentada en el resto del camino (al encontrarse con enlaces más lentos) y originará que la medida sea incorrecta. Para solventar este hecho se ha de aumentar de manera controlada la dispersión al final del prefijo, de manera que esta sea lo suficientemente grande para no verse afectada por los enlaces posteriores más lentos. Para esto se utiliza un tipo de sonda no uniforme denominada *cartouche*.

En la Figura 2 se observa un *cartouche* para medir el b_{bw} del prefijo compuesto por los tres primeros enlaces ($b_{bw_{1,3}}$) de un escenario hipotético compuesto por cinco enlaces. En este caso, la técnica *packet pair* no nos permitiría medir la capacidad del enlace debido a que la capacidad del último enlace es menor que la capacidad del prefijo. Para solventarlo se utiliza un *cartouche* compuesto por paquetes grandes (p en la figura) y paquetes pequeños (q y m, denominados estos últimos marcadores). Entonces un *cartouche* de tamaño r se define como una secuencia de paquetes pm seguido por $(r-1)$ secuencias de paquetes $[pq]$ y por otra secuencia $[pm]$. Los paquetes marcadores m llegan al host destino mientras que el resto de los paquetes salen al final del prefijo. Esto permite el aumento controlado del tiempo de dispersión que se pretende. En la Figura 2 se observa un *cartouche* de tamaño $r = 2$ para medir el prefijo en cuestión. La relación b_{bw} del prefijo con el b_{bw} del resto del *path* marca un valor mínimo para r (que será función además de los tamaños $s(p)$ y $s(q) = s(m)$), aunque éste puede hacerse arbitrariamente grande para obtener resultados más fiables (de hecho, la herramienta implementada utiliza un r más grande de lo que sería estrictamente necesario con esta finalidad). El b_{bw} del prefijo, siempre que se cumpla la condición de preservación, estará definido por la Ecuación 2.

$$b_{bbw_{prefijo}} = \frac{s(m) + (r-1) \cdot (s(p) + s(q))}{t_{disp}} \tag{2}$$

2.2.3. Medidas de sufijos: técnica *cartouche train*

Para medir sufijos del *path* se utiliza la técnica *cartouche train*. Un ejemplo de esta técnica puede verse en la Figura 2. La idea básica es conseguir medir el b_{bw} del sufijo aún cuando no es el b_{bw} del *path* completo (la velocidad de alguno o todos los enlaces del *path* es mayor que el b_{bw} extremo a extremo). En este caso, se debe de ser capaz de lograr variaciones controladas en la dispersión de los paquetes a lo largo del sufijo de tal manera que sea posible estimar las capacidades de los enlaces que lo componen.

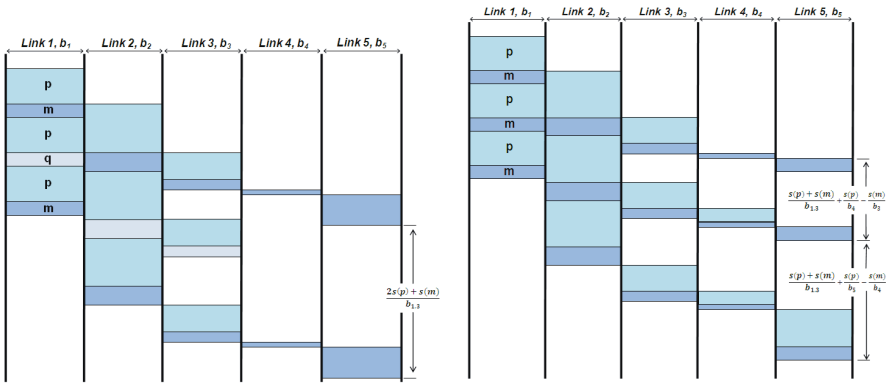

Figura 2: Ejemplo de *cartouche* (izqda., con $b_5 < b_2 < b_1 < b_3 < b_4$), ejemplo de *cartouche − train* (dcha., con $b_2 < b_1 < b_5 < b_3 < b_4$) (adaptación de [4]).

En la técnica *cartouche train* esto se hace enviando un número de *cartouche* como los del apartado anterior igual al número de enlaces que conforma el sufijo. Los paquetes que no son marcadores abandonan incrementalmente el camino, de tal manera que los paquetes p y q del primer *cartouche* abandonan el camino tras el primer enlace del sufijo, los del segundo tras el segundo y así sucesivamente. Si los *cartouches* tienen un tamaño suficientemente grande, el espaciado entre los paquetes marcadores m de cada *cartouche* se vera modificado por última vez en el enlace que en el que el resto de paquetes del cartouche abandonan el *path*. Así, si se conoce el b_{bw} del prefijo se conoce la separación con la que los marcadores entran al sufijo y se puede estimar de manera incremental la capacidad de los enlaces del sufijo en función de la dispersión de los marcadores medida al final del camino. Sin embargo, hay relaciones de capacidad de los enlaces que hacen inviable esta técnica. En la Figura 2 puede verse un ejemplo de un *cartouche train* para estimar $b_{bw_{4,5}}$ formado por *cartouches* de tamaño $r = 1$. Por cuestiones de espacio, no se plasmará aquí la matemática detrás de este tipo de sondas, pero toda ella se encuentra disponible en [4].

2.2.4. Procedimiento de media

Para determinar la capacidad de un determinado subconjunto de saltos $[i, j]$ ($b_{bw_{i,j}}$) de un camino de n saltos, se utiliza la implementación concreta de los tres tipos de sondas explicados con anterioridad para seguir el procedimiento definido en [3]. Este proceso secuencial es el siguiente:

Paso 1. Utilizar la técnica *packet-pair* para medir $b_{bw_{1,n}}$. En base a este determinar el tamaño r de los *cartouche* utilizados en el siguiente paso.

Paso 2. Utilizando *cartouche* de un r apropiado, medir $b_{bw_{1,i-1}}$ y $b_{bw_{1,j}}$ según la Ecuación 2. Entonces se presentan dos posibilidades:

· Si $b_{bw_{1,i-1}} > b_{bbw_{1,j}}$ entonces $b_{bw_{i,j}} = b_{bbw_{1,j}}$ y el proceso de medida queda finalizado en este punto.

· En otro caso, proceder al Paso 3.

Paso 3. Usando un *cartouche train* de longitud $l = j - i + 1$ con enlace de salida inicial i, para estimar $b_{bw_{1,j}}$ siguiendo el Apartado *2.2.3*. Entonces se vuelven a presentar dos posibilidades:

· Si la condición de viabilidad se cumple, se toma esta estimación como $b_{bw_{i,j}}$

· En caso contrario no es posible determinar esta capacidad usando esta herramienta.

3. Herramienta de medición implementada

El sistema implementado puede apreciarse en la figura 3. En esta figura se aprecia además un esquema secuencial del funcionamiento del sistema que se detallara posteriormente. La estructura del sistema está dividida entre los espacios de aplicación y *kernel* del sistema operativo Linux. Esto es debido a que para que la herramienta de medición implemente fielmente las técnicas aquí detalladas debían cumplirse principalmente dos premisas: los paquetes tienen que ser inyectados en la red *back-to-back* y la medida de tiempos debe ser lo suficientemente precisa. Dado que la gestión de los paquetes se realiza a nivel de *kernel*, para conseguir estos dos propósitos todas la funcionalidad de la herramienta relacionada con la transmisión de paquetes deberá estar a nivel de *kernel*. Además, para poder tomar tiempos de manera precisa no es razonable esperar a que el *kernel* procese el paquete y lo pase al nivel de aplicación, ya que este cambio de contexto tiene aparejado un retardo variable que haría las medidas poco precisas. Por tanto, la toma de tiempos deberá hacerse a nivel de *kernel* y lo más próximo posible a la recepción del paquete. Sin embargo, por las restricciones de memoria que las implementaciones *in-kernel* traen consigo, la herramienta no puede implementarse por completo a nivel de *kernel*; por lo que una parte de ella se encuentra en el nivel de aplicación.

Entrando en el detalle de los módulos que componen la herramienta, en la Figura 3 el módulo implementado a nivel de *kernel* se denomina como *BWSocket*. Esta parte de la herramienta está implementada como un módulo instalable del *kernel* de Linux y es el fruto de la creación de una nueva familia de *sockets* a partir de la familia de *socket* de propósito general de Linux [6]. Así, a esta familia se le ha añadido tanto la funcionalidad del envío de las diversas sondas como de la toma de tiempos.

A nivel de aplicación, en la Figura 3 se aprecian tres módulos: *Logic*, *Request Manager* y *Probe Manager*. Los módulos *Request Manager* y *Probe Manager* son los módulos que respectivamente gestionan con el *kernel* el envío y la recepción de una sonda concreta (de las tres detalladas). El módulo *Probe Manager* obtiene los datos de configuración del módulo de lógica, se comunican con el *kernel* para especificarle el tipo de prueba que ha de enviar utilizando llamadas

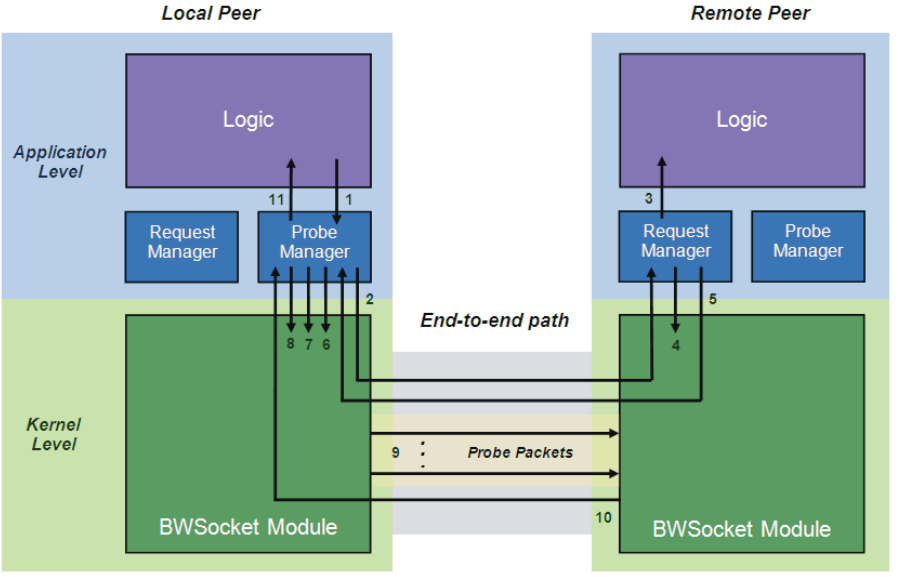

Figura 3: Esquema de la herramienta de medición implementada (bloques) y secuencia de medición (números y flechas).

IOCTL [7], y cuando obtienen los resultados provenientes del *kernel* los devuelve al módulo de lógica. El módulo *Request Manager* se ocupa de configurar el *kernel* del host remoto para la recepción y toma de tiempos para esa medida concreta en base a la información que la instancia de *Request Manager* le facilita desde el otro extremo. El módulo de lógica implementa el procedimiento de medida del Apartado 2.2.4. iniciando instancias de los dos módulos anteriores según sea necesario.

3.1. Funcionamiento del sistema

El procedimiento secuencial utilizado por el sistema para la realización de una prueba es el mostrado mediante los números presentes en la Figura 3. Entonces, para realizar una estimación de capacidad de un enlace o subconjunto de enlaces determinado, el módulo de lógica seguirá el proceso descrito en el Apartado 2.2.4. iniciando instancias de los módulos de medición y siguiendo estos pasos para cada sonda enviada. Además, hay que puntualizar que las mediciones no implican la toma de un único tiempo. En lugar de esto se toman muchos tiempos mediante el envío repetido de los paquetes que compondrían la sonda individual y se realizan histogramas, todo esto de cara a evitar posibles efectos del tráfico cruzado [2]. Así, los pasos para enviar cada una de estas sondas son:

1. El módulo de lógica ejecuta una instancia de *ProbeManager*, indicándole

el tipo de medida a realizar y proporcionándole la información necesaria (como por ejemplo la dirección IP del *peer* remoto, el final del prefijo y r en medidas con *cartouche*, etc.). En el *peer* remoto el módulo lógica tiene permanentemente ejecutada una instancia de *RequestManager*.

2. *ProbeManager* en el host local consulta a *RequestManager* en el host remoto la viabilidad de la realización de la prueba.

3. En el host remoto, *RequestManager* consulta con el módulo de lógica la viabilidad (si hay otra medida en curso el módulo del *kernel* se encuentra bloqueado y no es posible la realización de esta medida)

4. Suponiendo que la prueba es viable, *RequestManager* enviará una llamada IOCTL al módulo del *kernel* del host remoto indicándole el tipo de prueba a recibir y el número de repeticiones de la prueba.

5. En este punto, *RequestManager* envía una confirmación a *ProbeManager* de que la prueba es viable.

6. Entonces, *ProbeManager* registra en el *kernel* local la información referente a la prueba utilizando nuevamente una llamada IOCTL.

7. Tras el registro de la información de la prueba, se almacena en el módulo del *kernel* la información del destinatario de la prueba con otra llamada IOCTL.

8. Entonces, se envía al módulo del *kernel* en el host local la orden de comenzar el envío de la prueba usando también llamada IOCTL.

9. Ya en el espacio de *kernel*, el módulo *BWSocket* del host local bloquea el *kernel* y comienza el envío de los paquetes constituyentes de la prueba tantas veces como repeticiones se le haya indicado. Como esto lo hace sin mediación ni interrupción de ningún tipo, los paquetes son enviados *back-to-back*.

10. Cada vez que un paquete de la prueba llega al módulo del *kernel* del host remoto, éste conserva el tiempo en el que se recibió. Para medir tiempos de manera precisa se utiliza el reloj de CPU. Cuando se han recibido todos los paquetes que se esperaban, este módulo envía al *ProbeManager* del host local un paquete con los tiempos por cada repetición de la prueba realizada.

11. En el host local *ProbeManager* pasa al módulo de lógica los tiempos. En este módulo se calcula la capacidad en cuestión usando la fórmula que sea necesaria en cada caso para cada repetición de la prueba. Con las diversas repeticiones se elabora un histograma y en base a él se decide la capacidad estimada final en la prueba. En función de este resultado y del punto del flujo del procedimiento de medida del Apartado 2.2.4. en que se encuentre el sistema, puede procederse a realizar nuevas pruebas empezando nuevamente por el primer punto de este procedimiento.

4. Experimento de medida y resultados

Para comprobar el correcto funcionamiento de la herramienta de medida se implementó en el laboratorio una maqueta que representara el escenario típico en el que se quiere aplicar. Este escenario, que puede verse en al figura 4, representa el caso de estudio en el que dos usuarios finales del sistema se encuentran en AS diferentes que se encuentran interconectados entre sí. Así, en este caso tendremos tres saltos: L_1 (a $100Mbps$), L_2 (a $10Mbps$) y L_3 (a $100Mbps$). En este escenario, y como ya se ha comentado en el apartado introductorio, podrían interesar dos medidas: el $b_{bbw_{1,3}}$ y la capacidad del enlace que interconecta los AS.

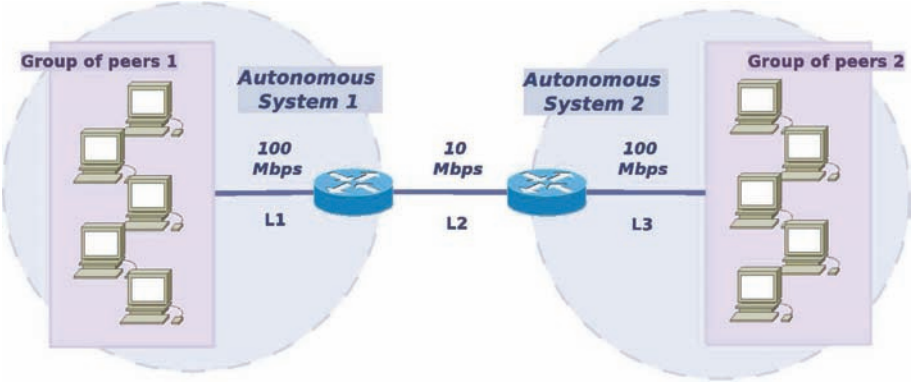

Figura 4: Escenario de estudio.

Entonces, usando la herramienta implementada para medir el ancho de banda extremo a extremo así como las capacidades de los enlaces individuales, se obtienen los resultados mostrados en los histogramas de la Figura 5.

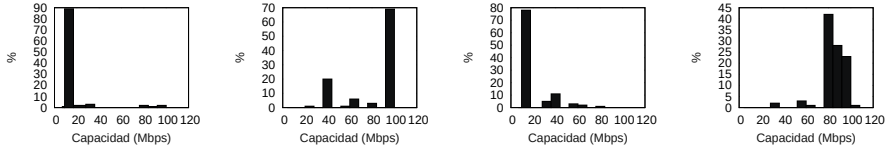

Figura 5: Resultados de la prueba. De izqda. a dcha.: $b_{bbw_1,3}$, b_1, b_2, b_3.

5. Conclusiones y trabajos futuros

Como se puede ver, la herramienta implementada estima de manera fiel la capacidad de enlaces individuales así como el ancho de banda de cuello de botella

de los diversos segmentos del *path* extremo extremo, tal y como se pretendía. Esto abre diversas posibilidades en dos campos principales: la mejora de la herramienta implementada y la aplicación de las medidas de capacidad. Con respecto a la mejora de la herramienta ya hay trabajos avanzados en dos líneas. La primera es conseguir obtener medidas precisas de capacidades en el orden de $1Gbps$, ya que actualmente no es posible medir de manera precisa tiempos en esa escala debido a cómo funciona la planificación y ejecución de tareas en el *kernel* de Linux. Sin embargo, se está ya próximo a la solución de este problema utilizando técnicas de tiempo real. La segunda es integrar la herramienta de medición en un cliente *P2P*, de tal manera que a la postre se pueda utilizar de manera más efectiva la información que proporciona. En este sentido, hay trabajos realizados en la integración de la herramienta en *Vuze* [8] (cliente de *BitTorrent* anteriormente conocido como *Azureus*). En lo que respecta a la aplicación de las posibilidades que ofrece la herramienta se está trabajando en dos vertientes. La primera de ellas es utilizar las efectos conocidos del tráfico cruzado sobre las sondas detalladas en este documento [2] para la implementar un analizador de tráfico en enlaces remotos. La segunda de ellas es utilizar la información sobre el ancho de banda extremo a extremo proporcionada por la herramienta para mejorar los algoritmos *P2P* actuales (ya sea mediante la creación de redes *overlay* más robustas en sistemas *P2P* des-estructurados o mediante una distribución más eficiente de claves y contenidos en sistemas *P2P* estructurados [1]).

Referencias

[1] Neil Daswani, Hector Garcia-Molina, Beverly Yang; Open Problems in Data-Sharing Peer-to-Peer Systems; Proceedings of the 9th International Conference on Database Theory, Vol. 2572, Pages: 1 - 15 (2003).

[2] Constantinos Dovrolis, Parameswaran Ramanathan, David Moore; Packet Dispersion Techniques and Capacity Estimation; IEEE/ACM Transactions on Networking (2001).

[3] Khaled Harfoush, Azer Bestavros, and John Byers; Measuring Bottleneck Bandwidth of Targeted Path Segments; Proceedings of IEEE INFOCOM (2003).

[4] Khaled Harfoush, Azer Bestavros, and John Byers; Measuring Bottleneck Bandwidth of Targeted Path Segments; Technichal Report, Boston University (2001).

[5] Srinivasan Keshav; Packet-Pair Flow Control; IEEE/ACM Transactions on Networking (1994).

[6] Codigo fuente del tipo de sockets PF_PACKET, en el sitio web Linux Cross Reference; http://lxr.free-electrons.com/source/net/packet/pf_packet.c/.

[7] Código fuente relativo a IOCTL del *kernel* de Linux, en el sitio web Linux Cross Reference; http://lxr.free-electrons.com/source/Documentation/ioctl/.

[8] Sitio web del proyecto Vuze; http://www.vuze.com/