White Paper

Medida del *throughput* sobre TCP

Octubre 2011
Rev. A2

A la hora de medir la tasa máxima de transmisión que puede ofrecer un enlace aerDOCSIS se suele recurrir a herramientas similares a las que miden la velocidad de descarga desde Internet, que basan su estimación en contar el tiempo que el usuario necesita para descargar o enviar un fichero de tamaño conocido a través de una conexión TCP. A menudo, los resultados obtenidos por estas herramientas difieren mucho de lo esperado, con lo que puede parecer que el enlace aerDOCSIS no está ofreciendo todo el throughput que debería. En cambio, cuando se usan herramientas basadas en el protocolo UDP, el throughput estimado es similar al valor teórico esperado.

Este documento pretende analizar el funcionamiento de estas herramientas para demostrar que los resultados no están relacionados con la tecnología aerDOCSIS sino con la configuración de determinados parámetros del enlace aerDOCSIS y del propio protocolo de transporte TCP.

En primer lugar se realizará una breve descripción de los protocolos TCP y UDP. A continuación, se profundizará en relación entre estos protocolos y la medida del throughput. Finalmente se propondrá una metodología para optimizar la configuración de conexiones TCP sobre enlaces aerDOCSIS.

Protocolos TCP y UDP

Tanto TCP como UDP son protocolos del nivel de transporte y se encuentran por tanto entre la capa de red y la capa de aplicación. El funcionamiento de ambos protocolos es muy diferente tal y como se explica a continuación.

TCP (*Transport Control Protocol*)

Este protocolo se define en RFC 793 y sus principales características son:

- **Orientado a conexión:** los sistemas de los dos extremos se sincronizan para controlar el flujo de paquetes y adaptarse a la congestión de la red. Se establece un *circuito virtual* en cada sentido de la comunicación.
- **Comunicación fiable:** garantía de entrega sin errores y en orden.
  - Confirmación de entrega
  - Solicitud de retransmisiones
  - Detección y corrección de errores (*checksum* en la cabecera TCP)
- **Control de errores**: si el paquete se recibe correctamente, el receptor envía una confirmación de entrega. Si no, pide la retransmisión.
- **Control de Flujo**: el protocolo define mecanismos para reducir la tasa de transmisión cuando se detectan pérdidas de paquetes y para incrementar la tasa hasta la capacidad máxima cuando dejan de detectarse errores en la comunicación.
- **Comunicación Punto a Punto**
- **Aplicaciones típicas**: aquellas en donde prime la Fiabilidad o las que necesiten tasas de transmisión altas: HTTP, FTP, SSH, SMTP, POP, Telnet,…

UDP (*User Datagram Protocol*)

El protocolo UDP se define en RFC 768, y sus características más importantes son:

- **No orientado a la conexión**: un extremo puede mandar datos al otro extremo de forma unidireccional y sin esperar a que se establezca ninguna conexión. La cabecera UDP contiene la mínima información de direccionamiento para no requerir el establecimiento de una conexión.
- **Comunicación NO fiable**: ninguna garantía.
  - Sin confirmación de entrega
  - Sin solicitud de retransmisiones
  - Únicamente detección de errores (*checksum* en la cabecera UDP)
- **Basado en datagramas**: cabeceras pequeñas (8 bytes) para el transporte más simple posible.
- **Ausencia de control de flujo:** los datagramas pueden llegar desordenados.
- **Comunicación Punto a Punto, o Punto a Multipunto** (*Multicast*, *Broadcast*,…)
- **Aplicaciones típicas**: aquellas en donde prima la velocidad y el bajo retardo (*Streaming* de Audio o Video en tiempo real), o las que transmiten poca información y no merece la pena que establezcan una conexión extremo a extremo: DHCP, DNS, NTP, NFS, RCP, TFTP, BOOTP...

Protocolos de transporte y medida del *throughput*

Protocolo TCP

Los mecanismos de control de flujo y de control de congestión del protocolo TCP pueden hacer que la tasa binaria máxima medida entre dos extremos difiera de su valor esperado. Por lo tanto es importante conocer estos mecanismos y su funcionamiento para poder realizar e interpretar estas medidas de forma correcta.

El *throughput* de una conexión TCP está limitado por dos ventanas: la ventana de recepción y la ventana de congestión. La primera trata de no superar la capacidad del receptor para procesar datos (control **de flujo**) y la segunda trata de no exceder la capacidad de la red (**control de congestión**). A continuación se explica el funcionamiento básico de estos dos mecanismos de control.

Control de Flujo en TCP

TCP emplea un protocolo de control de flujo extremo a extremo. Su objetivo es regular la tasa de envío de paquetes para evitar saturar al receptor. Si el receptor se saturara comenzaría a descartar paquetes que precisarían reenvío, haciendo menos eficiente el uso de la red.

El protocolo de control de flujo se basa en el uso de una ventana deslizante (***Sliding Window***). En cada ACK, el receptor indica al transmisor en el campo ***Receive Window*** la cantidad de bytes que hay disponibles en su *buffer.* Si el transmisor envía esa cantidad de bytes deberá esperar a que el receptor le informe de que tiene espacio en el *buffer* antes de transmitir más datos. Mientras el receptor esté saturado enviará una ventana de cero bytes, y en el momento en el que tenga espacio disponible en su *buffer* enviará una ventana mayor que cero.

→ **Escalado de ventana en TCP**

El campo de *Receive Window* está limitado a 16 bits, lo que supone un tamaño de ventana de 65535 bytes. Este tamaño de ventana puede limitar la eficiencia en las redes con gran ancho de banda.

Para solucionar este problema se usa un factor de escalado que permite incrementar el tamaño de la ventana desde 65535 bytes a 1 Gigabyte. Este factor de escalado se envía en el campo "***TCP Window Scale"*** de la cabecera TCP, se negocia durante el establecimiento de la conexión, y permanece fijo una vez establecido.

Control de Congestión en TCP

Los mecanismos de control de congestión tienen como objetivo evitar saturar la red. Estos mecanismos regulan la cantidad de datos que el transmisor introduce en la red, reduciendo el flujo de datos en caso de detectar algún problema.

Para detectar problemas en la red el transmisor se basa en la ausencia de ACKs. Cuando el transmisor envía un paquete se activa un temporizador de retransmisión, que alerta de la ausencia del ACK cuando se supera el RTT (***Round Trip Time***).

Básicamente, estos mecanismos se basan en controlar el tamaño de la ventana de transmisión. La ventana de transmisión se define como la cantidad de bytes que puede transmitir un extremo antes de recibir confirmación. Al iniciarse la comunicación la ventana de transmisión es de sólo un paquete, y va aumentando conforme se reciben ACKs correctamente. Si la comunicación se realiza sin problemas la ventana de transmisión acabará siendo igual que la de recepción. En cambio si se detecta algún problema se reducirá de nuevo la ventana de transmisión.

A la hora de realizar medidas de *throughput*, hay que tener en cuenta que al comienzo de la conexión la capacidad de la misma estará por debajo del *throughput* real del enlace. Más adelante se explicarán las precauciones que hay que tomar cuando se quieren hacer medidas de *throughput* en una celda aerDOCSIS.

Protocolo UDP

En el caso del protocolo UDP no existen mecanismos de gestión de flujo ni tampoco mecanismos de control de congestión. No introduce retardos para establecer ninguna conexión y no realiza ningún seguimiento del estado de la misma. La información simplemente se envía desde el transmisor, que confía en que llegue lo mejor posible al destinatario.

Con esta filosofía de transporte "lo más simple posible", UDP entrega los datos al destino de forma muy rápida y con el menor retardo, haciendo que las medidas de *througput* para tráfico de aplicaciones sobre UDP resulte mucho más sencillo que las medidas para tráfico sobre TCP.

Herramientas de Medida del *Throughput*

Actualmente existen numerosas herramientas para estimar el *throughput* de una determinada conexión. Como ya se ha explicado anteriormente, estas herramientas basan su estimación en medir el tiempo necesario para transmitir una cantidad de bytes conocida a través de una conexión TCP. Así, dividiendo los bytes transmitidos entre el tiempo invertido en su descarga, se calcula la tasa media de recepción/ transmisión que tuvo esa conexión y este valor se emplea como estimación del *throughput* disponible.

Al analizar los resultados de estas herramientas hay que tener en cuenta varios detalles, como si están contando o no todos los bytes de cabeceras (IP, TCP, etc.), o si en ese momento hay otra aplicación haciendo uso de los recursos de la conexión, por ejemplo.

En cualquier caso la principal fuente de error en las medidas obtenidas con estas herramientas están ligadas a la configuración del protocolo TCP. La configuración de las ventanas TCP en los extremos, la latencia del enlace, y el tamaño de la descarga pueden limitar la capacidad de la conexión, obteniendo de este modo un *throughput* menor que el esperado.

Influencia de la latencia y de la ventana TCP

A continuación se va a proponer un ejemplo del comportamiento de una conexión TCP sobre un enlace **aerDOCSIS**. El objetivo es mostrar cómo influye la latencia y el tamaño de ventana sobre la capacidad de la conexión TCP.

El escenario de ejemplo se compone de dos ordenadores conectados a través de un enlace aerDOCSIS que puede dar hasta 30 Mbps, configurado con un tiempo de trama de 10ms. Cada ordenador se conecta a un extremo del enlace a través de una conexión *Ethernet* de 100Mbps, por lo que parece claro que el *thoughput* de la comunicación extremo a extremo estará limitado por el *throughput* del enlace aerDOCSIS.

Entre los dos ordenadores se establece una conexión **FTP** (transportada sobre **TCP**) en la que un extremo ejerce de cliente y se descargará un fichero ubicado en el otro ordenador, en donde estará instalado el servidor FTP. El tamaño de ventana TCP en ambos ordenadores es de **64KB**, que es la ventana por defecto en la mayoría de los sistemas operativos.

Que la ventana sea de 64KB implica que el cliente puede enviar hasta 64KB sin esperar el ACK del servidor. El tiempo mínimo que tardará en llegar el ACK desde que se comienza a enviar el primer paquete es el *Round Trip Time* (RTT), que es el tiempo que tarda un paquete en ir y volver del cliente al servidor.

En un enlace aerDOCSIS el RTT es aproximadamente cuatro veces el tiempo de trama, con lo que en este caso tenemos:

RTT ~ 4*10ms = **40ms**

En resumen, con la configuración del ejemplo se pueden enviar 512Kbits (64KB) cada 40ms. Por lo tanto, el máximo *throughput* medible por conexión TCP cumple la siguiente relación:

$$Caudal = \frac{\text{Ventana}}{RTT}$$

Y en este ejemplo sería:

CAUDAL = 512kbit/40msg = ***12,8 Mbps***

Si no existiera la limitación de la ventana TCP se podrían enviar paquetes mientras se espera el ACK, es decir durante todo el RTT. La cantidad de bits que se pueden enviar durante este tiempo es el ***Bandwidth Delay Product*** (BDP):

BDP = C*RTT = 30Mbps*40ms = 1,2Mbits

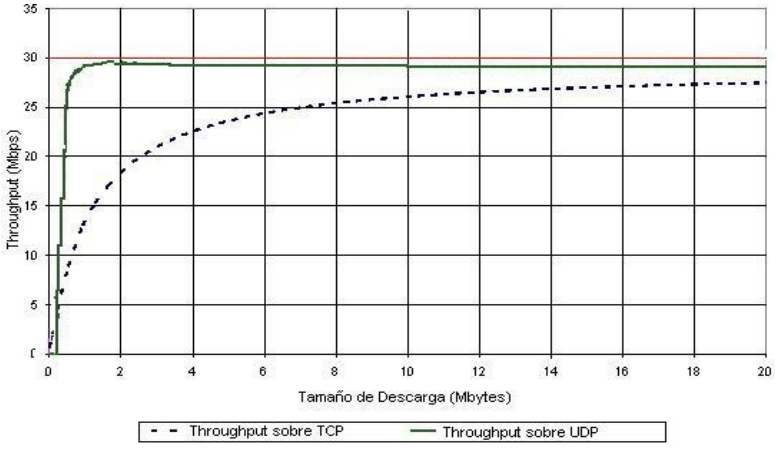
La conclusión es que en esta configuración la conexión TCP está enviando 512Kbits cada RTT cuando podría estar enviando 1,2Mbits, es decir, está aprovechando un 42,67% del *throughput* que ofrece el enlace aerDOCSIS.

Influencia del tamaño de descarga

Como ya se ha comentado, al inicio de la conexión TCP el tamaño de la ventana es de un solo paquete, y se incrementará progresivamente hasta alcanzar el valor máximo en función del *throughput* disponible. Cuanto mayor sea el *throughput* del enlace más tiempo tardará el tamaño de ventana en converger. Esto en la práctica quiere decir que durante los primeros segundos de una medición de *throughput*, éste irá creciendo de forma progresiva, y por lo tanto si no medimos hasta que el valor se normaliza podemos estar obteniendo un *throughput* medido menor del real.

En conclusión: al realizar medidas de *throughput* sobre *TCP* es necesario garantizar que el tiempo de la prueba es suficientemente grande para permitir al protocolo alcanzar el máximo tamaño de ventana y evitar que los resultados se vean sesgados por el tiempo de convergencia.

En la siguiente gráfica se muestran las medidas de *throughput* realizadas sobre un enlace aerDOCSIS de 30Mbps. Cuando la medida se realiza empleando el protocolo TCP son necesarias varias decenas de Mbytes para aproximarse al *throughput* real del enlace. En cambio empleando el protocolo UDP, que



Cómo medir tráfico en un enlace aerDOCSIS

Un problema muy habitual cuando se trabaja por primera vez con tecnología aerDOCSIS es que cuando se realiza una prueba de tráfico simple para medir el *throughput* potencial que puede ofrecer este sistema, se observan tasas de transferencia mucho menores de las que en teoría, y con las modulaciones existentes, se deberían poder conseguir ¿A qué se debe esto? ¿El enlace no es capaz de proporcionar la capacidad que prometía? Nada de eso. La causa está en todo lo comentado hasta ahora sobre el protocolo TCP.

Es probable que la prueba de tráfico se haya realizado transfiriendo un fichero de un PC a otro, o descargando un archivo desde internet, o utilizando un "test de velocidad" online. En todos estos casos, lo que se está haciendo es establecer una única conexión TCP, con lo que sufriremos todos los condicionantes que se han expuesto hasta ahora en este documento. ¿Cómo habría que medir por tanto para poder alcanzar toda la capacidad potencial?

- **Utilizar UDP:** al ser un protocolo no orientado a la conexión, UDP se limita a cursar los datagramas. No controla ni el flujo ni la congestión extremo a extremo, con lo que cuando se inyecta cierta tasa de tráfico UDP en un extremo se cursa sin más hacia el destino. Si al medir con UDP se consigue saturar en enlace, es la explicación clara de que el "problema" lo estaba provocando el protocolo TCP, no el enlace aerDOCSIS.
- **Establecer varias conexiones TCP en paralelo:** ya que TCP incluye una limitación por conexión, parece obvio que creando varias conexiones en paralelo el *throughput* agregado total cursado aumentará. Además, esta aproximación se asemeja más a un funcionamiento real, dado que una única conexión TCP ocupando todo el ancho de banda es bastante artificial. No hay que olvidar que siempre que hablamos de una limitación de TCP hablamos de limitación por conexión.
- **Realizar tests unidireccionales:** cuando se realiza un test unidireccional, se genera un pequeño tráfico de feedback (ACK) en sentido contrario para poder controlar el medio (el transmisor detecta problemas en la red en base a la ausencia de ACKs). Si el sentido contrario se encuentra vacío, los ACK se cursarán sin ningún problema (no compiten con nadie). En un test bidireccional, en cambio, los ACKs que proporcionan los servidores TCP compiten en su flujo de servicio con el tráfico de datos que se está transportando en el sentido contrario, por lo que cabe la posibilidad de que se retrasen en las colas y/o se descarten por saturación del flujo de servicio. En este momento entrarán en juego los algoritmos de control de congestión de TCP y provocarán una reducción en la capacidad del enlace observada. Por este motivo, los tests de velocidad online primero prueban la bajada y luego la subida
- **Aumentar el tiempo de la medida**: ya se ha visto que TCP comienza desde una velocidad de transmisión baja y la va incrementando hasta llegar a detectar congestión, con lo que es conveniente que los tests duren un tiempo suficiente para que el protocoloTCP puede llegar a alcanzar la máxima velocidad.
- **Aumentar el tamaño de la ventana TCP**.
- **Reducir la latencia extremo a extremo**: el tiempo de convergencia de una conexión TCP está directamente relacionado con la "distancia" entre cada extremo de la conexión, es decir, con su retardo. Es deseable que el servidor que se esté utilizando en las pruebas esté lo más "cercano" posible al cliente. Como ya se ha expuesto, la latencia extremo a extremo en una celda aerDOCSIS se puede mejorar reduciendo el tiempo de trama configurado en la BS.

Herramientas para medir *throughput*

Actualmente existen numerosas herramientas para estimar el *throughput* de una determinada conexión. Una de las más simples y utilizadas es "**Iperf**", un programa Cliente-Servidor muy sencillo que permite medir la velocidad máxima que alcanzan 2 máquinas conectadas en una red local (o en este caso, a través de una conexión aerDOCSIS). Está aplicación está disponible tanto para sistemas Windows como GNU/ Linux. Existe también la versión con GUI llamada "**jperf**" y basada en Java. La gran ventaja de esta aplicación es su versatilidad y su sencillez.

Para inyectar tráfico con Iperf se necesitan únicamente dos PCs con la aplicación instalada, asegurándonos de que ambas máquinas son accesibles entre ellas (hacer un ping previo). Uno de los PCs hará de Servidor (el que escucha) y otro hará de Cliente (el que genera tráfico). Hay que ejecutar primero el Servidor, si no el cliente no podrá enviar nada. Un ejemplo de sintaxis sencilla podría ser la siguiente:

- **Servidor**: iperf -s -i 1
- **Cliente**: iperf -c "IP_Servidor" -i 1 -t 60

Se puede añadir complejidad a este comando añadiendo opciones, como por ejemplo:

-u = inyectar tráfico UDP (por defecto es TCP)
-b = en tráfico UDP, cuánto tráfico queremos generar (5M = 5Mbps)
-i = cada cuantos segundos queremos ver resultados por pantalla
-t = cuantos segundos queremos generar tráfico (por defecto son 10s)
-p = cambiar el puerto (para poder realizar varias conexiones en paralelo)
-d = hacer el test bidireccional´
-f = para cambiar las unidades de medida (-f MB para medir en Megabytes)
-w = cambiar el tamaño de ventana (ej: -w 256K)

NOTA: En TCP no se especifica el throughput; el Iperf se supone que saca todo lo que se puede hasta llenar el canal. En cambio en UDP podemos elegir cuanto tráfico queremos inyectar.

Cómo optimizar el *throughput de* una conexión TCP

Hasta ahora se ha tratado de comprender cómo pueden influir en la medida del *throughput* ciertos parámetros de TCP. En este apartado se explicará cómo mejorar el comportamiento de las conexiones TCP sobre un enlace aerDOCSIS.

Existen tres maneras de aumentar el *throughput* máximo de tráfico TCP a través de un enlace aerDOCSIS:

A) **Reducir la latencia RTT del sistema**. Con esto se conseguirá reducir el tiempo de llegada de los ACK, lo que mejorará el rendimiento con ventanas TCP pequeñas. Como se ha comentado, la forma más sencilla de conseguir esto es reduciendo la longitud de trama aerDOCSIS (parámetro "***Frame Duration***"). Seleccionando un tiempo de trama más pequeño se conseguirá reducir el RTT a costa de reducir también ligeramente el *throughput* potencial máximo del enlace aerDOCSIS, pero conseguiremos que el throughput para cada conexión TCP aumente.

B) **Abrir múltiples conexiones TCP simultáneas**. Si existe cualquier tipo de limitación de *throughput* motivada por el protocolo TCP, esta será siempre para cada conexión. Esto quiere decir que, si en el ejemplo planteado anteriormente, la transmisión por FTP de un fichero entre dos ordenadores se realiza a un máximo de 1,5 Mbps (por la naturaleza de la conexión TCP), si se realizan tres transmisiones simultáneas cada una de ellas puede ir a una velocidad de hasta 1,5 Mbps, con lo que el *throughput* total del enlace aerDOCSIS aumenta hasta los 4,5 Mbps.

C) **Modificar la configuración de TCP en ambos extremos de la comunicación**. Los parámetros que se pueden configurar son varios:

- Aumentar el tamaño de las ventanas deslizantes tanto en el emisor como en el receptor.
- Ajustar la cantidad de memoria de sistema que pueden emplear las conexiones TCP (***Maximum Buffer Space***).
- Configurar el límite de memoria para cada conexión TCP que establezca el sistema (***Socket Buffer Sizes***).
- Habilitar la opción de **TCP *Large Window Extensions***, que activa el uso del TCP *Window Scale* y TCP *Time Stamps*. El primero permite emplear ventanas TCP de más de 64KB. El segundo facilita una medida más precisa del RTT y evita errores.
- Seleccionar la opción **TCP *Selective Acknowledgments*** (SACK), que permite al protocolo TCP indicar al transmisor cuáles son exactamente los paquetes que se han perdido y necesitan ser retransmitidos.
- El equipo debe usar la MTU (*Maximum Transfer Unit*) más grande soportada por la red, para lo que puede ser necesario activar la opción de ***Path MTU Discovery***.

El proceso de modificación de estos parámetros depende del sistema operativo sobre el que se trabaje, por lo que será necesario recurrir a la documentación del mismo.

Conclusiones

*En este documento se han explicado los problemas que pueden aparecer al realizar medidas de throughput neto en una red aerDOCSIS, y se ha comprobado la importancia que tiene el protocolo de transporte que se esté utilizando. Mientras que el protocolo **UDP** proporciona resultados fiables y acordes con la capacidad neta esperada, las aplicaciones que se transportan sobre el protocolo **TCP** pueden ver reducida su tasa de transmisión máxima debido a los mecanismos de control de flujo y congestión de TCP, así como por la forma en la que se realicen las mediciones (cantidad de bytes descargados, tiempo de duración de las medidas,…).*

*Es importante destacar que no se trata de que la tecnología aerDOCSIS y el protocolo TCP sean incompatibles, sino que es un comportamiento relacionado con la configuración de TCP en redes de banda ancha. Hay que ser conscientes de que esto puede pasar y sobre todo hay que saber la forma en la que podemos evitar estos problemas. **TCP sólo será capaz de aprovechar al máximo el throughput que puede ofrecer el enlace aerDOCSIS si ambos extremos tienen una configuración de ventana máxima suficientemente holgada,** teniendo en cuenta que a mayor RTT mayor deberá ser la ventana para alcanzar una determinada velocidad de conexión TCP.*

Lo que se recomienda es que si se quiere utilizar un enlace aerDOCSIS para transportar datos sobre TCP, se conozcan la problemática que se presenta y las medidas que se pueden adoptar para mejorar el funcionamiento, como pueden ser: reducir la latencia RTT, que para el caso de una red aerDOCSIS se consigue seleccionando un tiempo de trama menor, configurar el protocolo TCP adecuadamente en ambos extremos, o abrir múltiples conexiones TCP de forma simultánea para conseguir un mayor throughput neto total en el enlace aerDOCSIS.