JITEL 2017
XIII Jornadas de Ingeniería Telemática
27-29 Septiembre 2017, Valencia (España)

Actas de las XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia (España), 27-29 de Septiembre de 2017

ISBN: 978-84-9048-595-8
DOI: http://dx.doi.org/10.4995/JITEL2017.2017.6951

Mecanismos de nivel de transporte para la optimización de envíos en base al ancho de banda estimado sobre Long Fat Networks

Alan Briones, Guillermo Dobao, Ramon Martín de Pozuelo, Agustín Zaballos, Guiomar Corral
Grupo de Investigación en Internet Technologies y Storage (GRITS)
Universidad Ramon Llull – La Salle
08022
<abriones, gdobao, ramonmdp, zaballos, guiomar> @salleurl.edu

***Resumen*- Este artículo investiga los mecanismos de diferentes protocolos de transporte en transferencias sobre redes de alta capacidad y alto retardo, conocidas como *Long Fat Networks* (LFNs), para un envío eficiente de los datos. *Transport Control Protocol* (TCP) presenta limitaciones de rendimiento y flexibilidad. En la literatura se pueden encontrar diferentes propuestas de variantes del comportamiento de TCP, protocolos como *Stream Control Transmission Protocol* (STCP) o soluciones que proporcionan una comunicación confiable y mecanismos de control de congestión sobre *User Datagram Protocol* (UDP). En este artículo se presentan una serie de mecanismos de nivel de transporte para la optimización de transferencias de datos sobre redes LFN. Estos mecanismos ofrecen un rendimiento elevado utilizando todo el ancho de banda disponible del enlace mediante un proceso de cálculo del estado de la red y un control de congestión activo para la utilización de todo el *bandwidth*, a la vez que reactivo en caso de producirse pérdidas para evitar congestiones en la red. El objetivo es demostrar la eficiencia de dichos mecanismos, así como su adaptabilidad y *aggressive friendliness* respecto a otros protocolos de transporte mediante el despliegue de una serie de pruebas expuestas en este artículo.**

***Palabras Clave*- protocolo de transporte, long fat networks, pérdidas de paquetes, ancho de banda, retraso, control de congestión.**

I. Introducción

Internet ha ido cambiando a lo largo de los años. En un mundo cada vez más interconectado, en el que los usuarios requieren y consumen cada vez más información y la necesitan de manera inmediata, se ha producido un cambio de paradigma respecto a cómo se concibieron inicialmente las redes y su uso.

Este aumento de uso está ligado en gran parte al incremento de la oferta de servicios multimedia. El amplio abanico de contenidos, tanto por la cantidad ofrecida como por el tamaño de los mismos, ha evidenciado la necesidad de disponer de redes con mayor ancho de banda para conexiones *end-to-end*, donde los equipos finales están separados por grandes distancias.

Dentro de la clasificación de enlaces de alta capacidad, existe un tipo de red conocida como *Long Fat Network* (LFN) [1]. La principal característica que define este tipo de redes es su alto *Bandwidth* (BW) y unos valores elevados de *Round Trip Time* (RTT). Una red es considerada LFN si su *Bandwidth-Delay Product* (BDP) es mayor a 12500 bytes (10^5 bits). Por ejemplo, un enlace de 1 Gbps y 1 ms de RTT, obtiene un BDP de 10^6 bits, siendo clasificada como LFN.

Estas características de las LFNs provocan que *Transmission Control Protocol* (TCP), el protocolo de transporte más utilizado en la red, no obtenga un buen rendimiento, lo que estimuló la definición de una extensión del protocolo [2]. Las principales problemáticas surgen debido al propio diseño de TCP, como la limitación de la ventana de congestión, la cual solo permite una ventana máxima de 65 KB debido a que su campo en la cabecera es de 16 bits. Otra problemática es el elevado valor de *Round Trip Time* (RTT) y la acumulación de mensajes de confirmación debido a los tiempos de *timeout* y retransmisión en el caso de producirse pérdidas.

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

Para solventar las problemáticas mencionadas anteriormente, se han propuesto diferentes mecanismos para TCP, así como nuevos protocolos que permitan extraer el máximo rendimiento de las redes LFN.

En este paper se propone otra alternativa mediante un mecanismo de control de congestión que incluye el cálculo del estado del enlace, permitiendo al protocolo adaptar su comportamiento y maximizar el uso del canal de manera casi inmediata. Finalmente, se muestra una prueba de concepto del funcionamiento del protocolo propuesto.

La organización del artículo queda de la siguiente manera. En la Sección II se repasa brevemente el Estado del Arte de los protocolos de nivel de transporte más destacados. En la Sección III se presenta los mecanismos propuestos. En la sección IV se explican una serie de directrices para la implementación dichos mecanismos, así como el Testbed utilizado, las pruebas realizadas y sus resultados. Finalmente, en la sección V se extraen las conclusiones y se presentan las líneas de futuro.

II. Estado del Arte

Este Estado del Arte se focaliza en mostrar los mecanismos de control de congestión de los diferentes protocolos de transporte más destacados para larga distancia [3].

La misión del control de congestión es detectar la congestión de la red antes de que ésta se colapse y actuar en consecuencia [4][5].

A. Principales causas que producen congestión en la red

Una de las principales causas de congestión es la incapacidad del *host* destino de procesar toda la información recibida, provocando que se descarten parte de los datos entrantes.

Otra de las causas de congestión es debida a que algún dispositivo intermedio no es capaz de tratar toda la información recibida, por lo que empieza a encolar los datos recibidos, generando retrasos (*delay*). En casos de saturación total existe la posibilidad de que llegue a descartarlos, generando pérdidas.

B. Estrategias para poder detectar y mitigar la congestión en la red

Para evitar que la red se sature, es necesario que el protocolo sea capaz de interpretar el estado de la red y actuar en consecuencia para extraer el máximo rendimiento de ella sin que se produzca un deterioro de la comunicación o pérdidas de información.

Por un lado, se plantean técnicas preventivas, como el control de admisión. Se limita el número de usuarios, se monitoriza que el flujo no exceda un límite fijado o se regula el tráfico en el acceso de la red. Para poder aplicar estos mecanismos se debe diseñar la red adecuadamente y tener el control total de ella.

Por otro lado, existen técnicas reactivas que resuelven la congestión una vez ya se ha detectado o cuando está a punto de producirse.

Para su detección existen dos clases:

- Realimentación directa: Los nodos intermedios de la red señalizan a los extremos la existencia de congestión o si hay riesgo de que se produzca. Se indica mediante el marcado de paquetes o el envío de paquetes especiales.
- Realimentación indirecta: Los extremos de la comunicación detectan la congestión basándose en las pérdidas de paquetes, retrasos y la variabilidad en los tiempos de recepción (*jitter*).

C. Protocolos TCP para larga distancia

En la actualidad existen diferentes propuestas de protocolos TCP para la transmisión de datos a larga distancia.

Parallel TCP Reno (P-TCP)

P-TCP [6] tiene un funcionamiento similar a TCP Reno y su control de congestión consiste en la transmisión de varios streams de datos en paralelo.

Un inconveniente es que el número de streams puede variar según la red, el tipo de información, etc. Se calcula, aproximadamente, 16 streams para redes de larga distancia, no permitiendo priorización entre tráficos.

Scalable TCP Reno (S-TCP)

S-TCP [7] basa su control de congestión en una mejora de TCP Reno. En lugar de utilizar *Additive Increase*, utiliza un incremento exponencial. Además, aplica un decremento multiplicativo para la reducción de la ventana, siendo un protocolo menos agresivo en caso de pérdidas.

High Speed TCP (HS-TCP)

HS-TCP [6] tiene un comportamiento como TCP Reno en una ventana de congestión pequeña. A partir de que su *Congestion Window* (CW) supere una cantidad de paquetes determinada, un *flag* es activado, modificando su CW en función de una tabla predefinida por el propio protocolo.

El hecho de utilizar una tabla predefinida no permite que sea un protocolo dinámico ni flexible.

H-TCP

H-TCP [8] está basado en HS-TCP. En lugar de utilizar una tabla, utiliza un algoritmo de AIMD heterogéneo que permite adaptarse a las capacidades del canal de forma eficiente.

Basa su comportamiento en las pérdidas producidas y el RTT, mejorando la eficiencia de TCP adaptando sus parámetros dinámicamente.

Es un control de congestión complejo en cuanto a cálculos e implementación.

HSTCP-LP

HSTCP-LP está basado en HS-TCP y TCP-LP [6]. Utiliza solo el ancho de banda residual que no es utilizado por otros flujos. En el caso de haber varios flujos, todos tienen la misma prioridad.

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

Al tratarse de un protocolo no intrusivo, en redes congestionadas, ve afectado su *throughput*.

D. Otro protocolos para larga distancia

A parte de protocolos basados en TCP, existen otras propuestas como ahora *Stream Control Transmission Protocol* (SCTP) [9] o protocolos basados en UDP que proponen soluciones ante las ineficiencias propias de TCP.

Stream Control Transmission Protocol (SCTP)

SCTP aporta una serie de funcionalidades y mecanismos adicionales de los que TCP carece. Propone un sistema de cabeceras para la gestión de los flujos. Además, este protocolo provee de mayor seguridad en el establecimiento de conexión mediante un proceso de *4-way handshake,* en vez del *3-way handshake* utilizado en TCP, para prevenir ataques de denegación de servicio (DoS).

A pesar de utilizar las mismas variables que TCP, como la CW y de utilizar un control de congestión similar al de TCP (*slow-start*, *congestion avoidance*) con ligeras modificaciones, SCTP basa su funcionamiento en la ventana de recepción (RWND) para evitar que se sature el receptor.

Además, introduce otras características como la posibilidad de la recepción desordenada de paquetes, así como el *multihoming* y el *multistreaming*.

UDP-based protocols

Un conjunto de protocolos basados en UDP apareció tratando de proporcionar un control eficaz de la congestión y funciones de confiabilidad sin ser protocolos orientados a conexión por definición. Entre todos ellos (Tsunami, PA-UDP, SABUL, etc.), *UDP-based Data Transfer* (UDT) es el que presenta una mejor utilización del rendimiento y ofrece una solución prometedora para transferencias de datos pesadas a través de redes de larga distancia, pero también se han identificado algunas deficiencias y dificultades en su implementación real [10][11][12].

III. Especificación de mecanismos

El objetivo principal de los mecanismos propuestos es conseguir el máximo ancho de banda disponible de manera agresiva respecto a otros flujos para el envío de grandes volúmenes de datos *non-real time* sobre redes LFN.

Los mecanismos propuestos se inspiran principalmente en los protocolos SCTP y UDT, siendo planteados para su uso sobre UDP.

Por un lado, se utiliza la estructura de mensajes y cabeceras en SCTP. Principalmente, el mecanismo propuesto se basa en la gestión de streams propuesta por SCTP (la cual su explicación no es objetivo del artículo), así como en el uso del *Selective-ACK* (SACK) para la confirmación de información recibida y solicitud de información perdida; siendo la principal diferencia con SCTP que éste trabaja sobre TCP y el mecanismo propuesto lo hace sobre UDP, cambiando el planteamiento del control de congestión.

Por otro lado, se utiliza un control de congestión basado en el protocolo UDT. El mecanismo basa su fórmula de cálculo de estimación de ancho de banda y control de congestión en la fórmula original de UDT pero modificando el cálculo del incremento del número de paquetes por ráfaga (en base a la eficiencia del envío, haciéndolo más agresivo), así como el cálculo del ancho de banda estimado. Este cálculo de BW es realizado mediante señalización *in-band* durante el envío de mensajes de datos. Todo ello implica una mejora en términos de utilización del rendimiento y flexibilidad.

Este apartado se focaliza en la explicación de los dos mecanismos propuestos; el cálculo del estado de la red y el control de congestión.

A. Estado de la red

Una vez establecida la asociación y negociada la seguridad de la comunicación, se lleva a cabo un proceso de entrenamiento para conocer el estado de la red.

Este cálculo del estado de la red permite al protocolo estimar el ancho de banda máximo de la red, así como el *Round Trip Time* (RTT) y el *Receiving Rate* (RR), o ventana de recepción, de la comunicación.

Funcionamiento

Tal y como se muestra en la *Fig.* 1, este proceso está formado por el envío de 10 ráfagas con el objetivo de tener diferentes muestras representativas a la hora de realizar el cálculo, con bloques ordenados en grupos de 2 (*packet-pair*) a 20 paquetes (*packet-train*) de 9000 bytes (jumboframes), donde se escoge el número de paquetes por bloque dependiendo de la velocidad estimada del enlace en la iteración anterior. Cuantos más paquetes se utilicen, más se reduce la probabilidad de error en la estimación del ancho de banda. Los paquetes a enviar son paquetes de datos, los cuales pueden contener parte de la información de la transferencia o ser paquetes de datos vacíos, según lo requiera la situación de la comunicación.

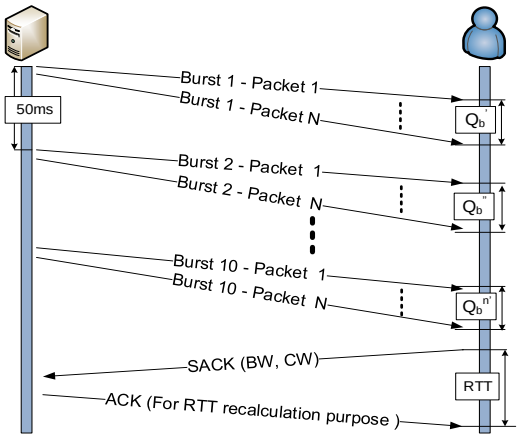


Fig. 1 Estado de la red – Cálculo del BW

Al inicio de esta fase de cálculo del ancho de banda se envían los paquetes de cada bloque de forma consecutiva por parte del emisor hacia el receptor. Por cada bloque, un proceso en el receptor registra el tiempo

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

de llegada del primer paquete y del último, calculando la diferencia de tiempo y el tamaño total (en bits) de los paquetes recibidos (la suma total del tamaño de cada uno de ellos) para realizar un cálculo del ancho de banda.

Con estos datos, se extrae el ancho de banda (BW), mediante la Ec. 1, donde *b* es el número de bits recibidos y *Qb* es el tiempo entre la llegada del primero y el último de estos bits, y que corresponde al tiempo que ha tardado la red en transportar estos datos. De esta forma se extrae el *bottleneck bandwidth*, es decir, el máximo ancho de banda al que podremos enviar por esa red.

$$BW = \frac{b}{Q_b} \qquad \text{(Ec.1)}$$

El servidor realiza, mediante esta fórmula (Ec.1), estimaciones del BW, una para cada una de las 10 ráfagas de bloques de paquetes, almacenando estos valores. Una vez ha hecho el cálculo 10 veces y dispone de 10 valores de BW, realiza la moda de los valores extraídos para obtener un valor de BW estable y consolidado, evitando calcular la media, la cual se ve afectada en mayor medida por algún cálculo del BW anómalo y falseando el valor real.

Una vez realizado este cálculo, el receptor le comunica al emisor el BW estimado, mediante un mensaje de confirmación selectiva (SACK), así como otros valores de los que ya dispone como la *Congestion Window* (CW) o el *Receiving Rate* (RR) en paquetes por segundo.

Para extraer el *Round Trip Time* (RTT), al enviar el cliente un mensaje de contestación (ACK) como confirmación, el servidor guarda el tiempo transcurrido entre el envío del mensaje de confirmación (SACK) y la recepción del ACK.

Este proceso de cálculo del ancho de banda máximo se realiza de manera *in-band* durante la transmisión de datos mediante los propios mensajes de datos.

B. Control de congestión

Tras el proceso de entrenamiento se descubre el ancho de banda máximo del que puede disponer ese flujo.

Con este valor extraído de la red, se establece un tanto por ciento de ese valor como la velocidad de envío (*Sending Rate* (SR)) inicial. Dependiendo de la agresividad deseada, se establece valor más o menos elevado.

El envío de los datos se realiza mediante ráfagas separadas por un tiempo determinado por el RTT o la mínima resolución temporal que pueda ofrecer el sistema operativo y el hardware sobre el que funcione el proceso. Este tiempo será calculado mediante Ec.2.

$$\mathrm{T_{burst}} = \max(50\mathrm{ms}, \mathrm{RTT}) \qquad \text{(Ec.2)}$$

Tras conocer la velocidad a la que se envían inicialmente los paquetes y determinada la separación entre ráfagas, se define el número de paquetes que se envían en cada ráfaga mediante Ec.3.

$$\#\mathrm{Packets\_per\_burst} = \mathrm{SR} * \mathrm{T_{burst}} \qquad \text{(Ec.3)}$$

En la *Fig.* 2 se muestra el proceso de envío de los datos. El receptor guarda la información que va recibiendo, a la vez que lista los paquetes perdidos, los cuales son pedidos en el siguiente mensaje de confirmación (SACK) de los datos recibidos.

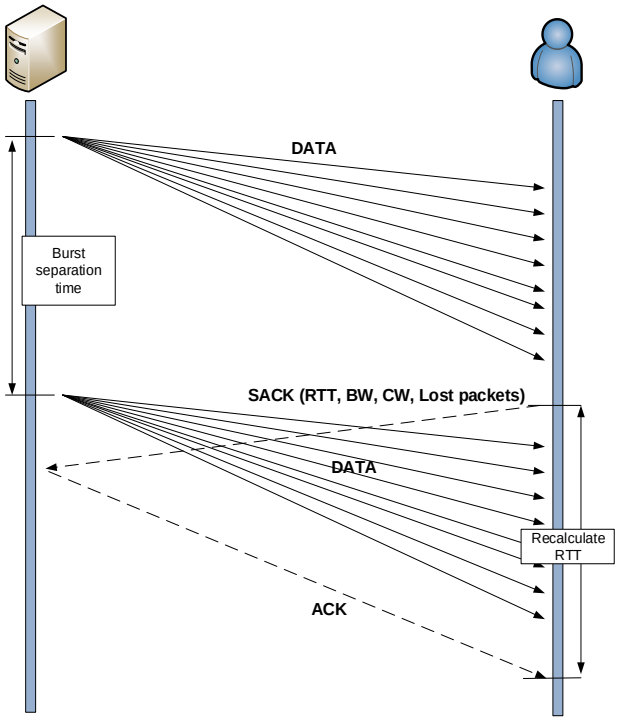


Fig. 2 Envío de datos

El mensaje de confirmación de datos por parte del receptor se envía de manera asíncrona respecto a la ráfaga del emisor al finalizar la ráfaga asociada.

Utilizando estos datos proporcionados por el receptor, el emisor irá modificando el valor la velocidad de envío de paquetes (*Sending Rate* (SR)).

Si al recibir un SACK, éste no indica que se han detectado pérdidas, se realiza el siguiente proceso (Ec.4, Ec.5, Ec.6), calculando el número de paquetes a incrementar (Inc_p) en la siguiente iteración (Ec. 4).

$$Inc_p = \max\left(\frac{1}{\mathrm{MTU}}, 10^{\log(\mathrm{BW}-(\mathrm{SR}*\mathrm{MTU}*8))-\mathrm{C})}\right) \qquad \text{(Ec.4)}$$

Donde el valor de C varía según la eficiencia de la comunicación (Ec.5). Considerando la eficiencia como el cociente entre el ancho de banda utilizado en la iteración anterior respecto al recientemente calculado.

$$C = \begin{cases} 7, & \frac{BW_{IteraciónAnterior}}{BW_{Actual}} < 0.8 \\ \left(\frac{BW_{IteraciónAnterior}}{BW_{Actual}} * 10\right) - 1, & \frac{BW_{IteraciónAnterior}}{BW_{Actual}} \geq 0.8 \end{cases} \qquad \text{(Ec.5)}$$

A diferencia con UDT, el mecanismo propuesto aplica un incremento dinámico en base a la eficiencia del enlace, siendo más agresivo cuando la eficiencia del envío es menor al 80%.

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

Posteriormente, se calcula el *Sending Rate* (paquetes/ráfaga) de la siguiente ráfaga (Ec.6).

$$\text{Sending Rate (SR)} = \frac{(T_{burst} * SR) + inc_p}{T_{burst}} \quad \text{(Ec.6)}$$

Si por el contrario se ha detectado algún paquete perdido, se aplica la siguiente fórmula (Ec.7):

$$\text{Sending Rate (SR)} = \frac{SR}{1 + 0.125 * \frac{(SR * MTU * 8)}{BW}} \quad \text{(Ec.7)}$$

El objetivo es lograr enviar el mayor número de paquetes en una ráfaga sin saturar el enlace con la mayor eficiencia posible. Esto es posible gracias a la estimación del ancho de banda.

En caso de producirse pérdidas, se realiza una reducción del número de paquetes a enviar en la siguiente ráfaga para evitar que se produzcan más pérdidas, la vez que se busca la máxima eficiencia en el envío.

En el siguiente apartado se muestra el comportamiento y los resultados de estos mecanismos en diferentes situaciones sobre redes LFN durante el envío de datos.

IV. Pruebas y Desarrollo

A partir del análisis y el diseño expuesto en los apartados anteriores, se realiza una primera implementación de los mecanismos presentados con tal de analizar el rendimiento de éstos en redes LFN.

A. *Estructura de implementación del protocolo*

Se debe diferenciar dentro del marco de la comunicación dos roles en lo referente a un envío *Peer to Peer*, emisor y receptor. Ambos nodos finales de la comunicación deben disponer de una instancia de control que gestione los flujos entrantes del protocolo, creando un proceso de recepción específico para cada flujo.

Así pues, una instancia gestiona la transmisión de un único flujo UDP, utilizándose este protocolo como "*Socket Transport*". Esta elección permite el desarrollo sobre una base sólida sin de mecanismos de control de congestión.

Partiendo de estas premisas, se implementan los mecanismos siguiendo una filosofía reactiva en recepción y secuencial en emisión, asumiendo esta última el rol de *master* en la comunicación.

A continuación, se plantea el *testbed* utilizado y las pruebas realizadas para mostrar el rendimiento de los mecanismos en redes LFN.

B. *Testbed*

El *testbed* desplegado para mostrar el comportamiento de los mecanismos LFN se presenta en el escenario de la Fig. 3.

Este consta de dos nodos extremos (receptor y emisor) interconectados a través de un nodo central el cual emula el comportamiento de una red WAN con características LFN. Las conexiones físicas entre dispositivos se realizan mediante pares trenzados CAT-5 a 100 Mbps Full Duplex y latencias de hasta 100ms.

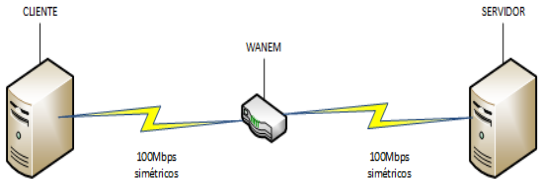


Fig. 3. Diagrama del escenario de pruebas

Para simular diferentes características de red, se utiliza el software *Wanem* [13] en el nodo central. Con tal de generar tráfico adicional entre los nodos finales se utiliza:

- *Iperf* [14] para generar tráfico *UDP*
- *File Transport Protocol* (FTP) para generar tráfico *TCP friendly*

Los objetivos que persiguen demostrar las pruebas a realizar sobre el *testbed* son:

(O1) Eficiencia. Máximo ancho de banda medio alcanzado (Mbps) con diferentes niveles de pérdidas producidas en el enlace y diferentes velocidades de enlace.

(O2) Adaptabilidad. Detección de congestión y modificación del *Sending Rate* para minimizar las pérdidas y maximizar el ancho de banda útil utilizado (Mbps).

(O3) *Friendly Aggressiveness*. Agresividad frente a otros flujos TCP y UDP contemplando el estado de la red.

C. *Pruebas*

Sobre el Testbed (*Fig.* 3) se plantean 3 conjuntos de pruebas:

(P1) Transmisión de un único flujo ante las parametrizaciones descritas en la Tabla I con tal de demostrar la eficiencia del envío ante diferentes velocidades y niveles de pérdidas en el enlace (O1).

(P2) Transmisión de un único flujo compartiendo enlace con flujos agresivos intermitentes no adaptativos (generados con *Iperf*) para demostrar la adaptabilidad del control de congestión (O2).

(P3) Transmisión de un flujo compartiendo enlace con otros protocolos, con tal de demostrar el *Friendly Aggressiveness* frente a estos (O3):

a. UDP (*Iperf*)

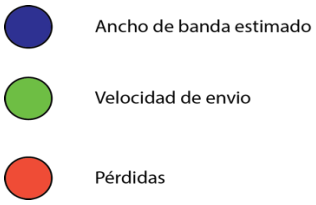
b. TCP (FTP)

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)
EDITORIAL UNIVERSITAT POLITÈCNICA DE VALÈNCIA

c. UDP (Mismo control de congestión)

En los siguientes subapartados se exponen, de manera ordenada, los resultados de las pruebas planteadas.

Durante todas las pruebas se envía un total de 1GB. A nivel de gráficas, el color azul indica el ancho de banda estimado, el color verde la velocidad de envío y el color rojo las pérdidas.



Prueba 1

Se plantean diferentes configuraciones (*Tabla I*) con dos objetivos (O1). El primer objetivo (O1.1) es comparar el ancho de banda utilizado respecto al real del enlace (pruebas $P1_1$ y $P1_2$). Y el segundo objetivo (O1.2) es observar la reacción ante la introducción de pérdidas (pruebas $P1_{2\text{-}8}$).

Tabla I
PARAMETRIZACIONES PRUEBA I

Prueba	Velocidad de enlace	Pérdidas
$P1_1$	50 Mbps	0%
$P1_2$	100 Mbps	0%
$P1_3$	100 Mbps	0.001%
$P1_4$	100 Mbps	0.01%
$P1_5$	100 Mbps	0.1%
$P1_6$	100 Mbps	1%
$P1_7$	100 Mbps	3%
$P1_8$	100 Mbps	5%

En la *Fig. 4* y *Fig. 5* se puede observar el resultado de ($P1_1$) y ($P1_2$). En $P1_1$ se aprecia como trabajando a 50Mbps, el ancho de banda estimado por el protocolo es de 49Mbps. En ($P1_2$), a 100Mbps, se estima una velocidad de 96Mbps.

En ambos casos se supera el 95% de utilización de enlace, confirmando parcialmente (O1). Cabe destacar que la velocidad de envío real del protocolo se sitúa en la mayoría de las situaciones por debajo de aquella estimada. Esto es debido al *step* usado al incrementar la velocidad de envío, el definido por el tamaño de los paquetes (*Jumboframe*) que se envían.

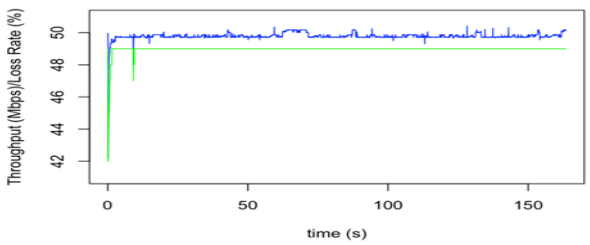


Fig. 4. Resultado de la prueba $P1_1$

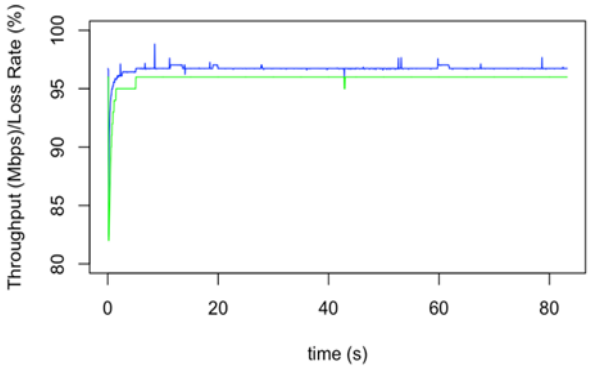


Fig. 5. Resultado de la prueba $P1_2$

Se puede observar como a mayor velocidad de enlace, se reduce la precisión de estimación. Dicha reducción es debida a diversos factores (tamaño de los paquetes, tiempos de ráfaga,…). Esto resalta la importancia del proceso de conocimiento del estado de la red para una parametrización eficiente de las variables dependientes del tipo de enlace.

En las siguientes pruebas se incorporan diferentes niveles de pérdidas. En la prueba ($P1_4$), por ejemplo, se incorpora un porcentaje de pérdidas moderado (0.01%), observándose una disminución de velocidad de transmisión (Fig. 6). La variación máxima de velocidad es de 12Mbps, la cual se recupera rápidamente. No obstante, la velocidad de transmisión media no se ve afectada por las pérdidas moderadas. Esta se aproxima a la velocidad de transmisión máxima estimada, 96 Mbps.

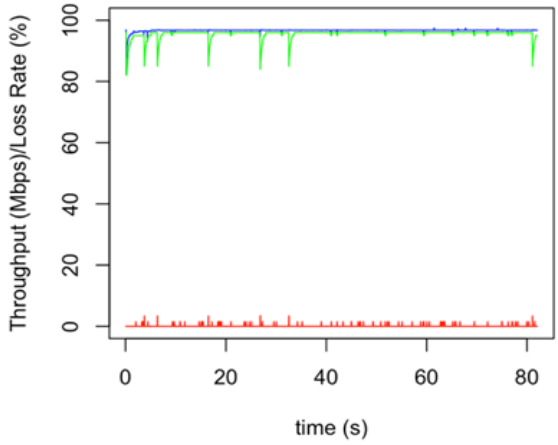


Fig. 6. Resultado de la prueba con pérdidas del 0,01% - $P1_4$

Finalmente, en la *Fig. 7*, se encuentra recogida la velocidad media de transmisión del protocolo ante los diferentes niveles de pérdidas aleatorios planteados en la Tabla I. Cabe destacar que la generación de pérdidas es arbitrária y generada por un software, por lo que, a pesar de modificar el protocolo su comportamiento para evitar pérdidas en el enlace, no se modifica el % de pérdidas producidas. El objetivo principal de la prueba es mostrar la resiliencia ante pérdidas y no el rendimiento ante éstas.

En dicha gráfica se puede observar como la zona de trabajo óptima del protocolo se encuentra en el rango 0%

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)
EDITORIAL UNIVERSITAT POLITÈCNICA DE VALÈNCIA

– 0,1%, donde la velocidad de transmisión no se ve apenas afectada por las pérdidas.

Hasta aproximadamente el 1% de pérdidas no llega a valores inferiores al 50% de *throughput* y en pérdidas de un 5% solamente se ha reducido hasta el 10% del *throughput* total.

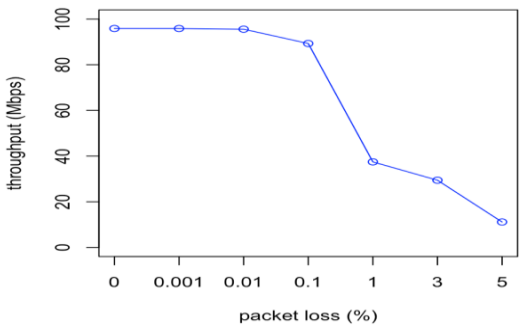


Fig. 7. Comparativa de la velocidad de transmisión media ante diferentes niveles de pérdidas (%)

De esta primera prueba se extrae la capacidad del protocolo para ceñirse al ancho de banda estimado en el proceso de estado de la red. Esto permite alcanzar el máximo ancho de banda disponible de manera rápida con valores de pérdidas aleatorias cercanas al 1%, confirmando el objetivo (O1).

Prueba 2

Mediante la prueba (*P2*), tal y como ha sido expuesto, se pretende verificar (O2). Con tal propósito, se plantea una prueba dividida en diversas etapas, dejando entre etapas un tiempo de recuperación.

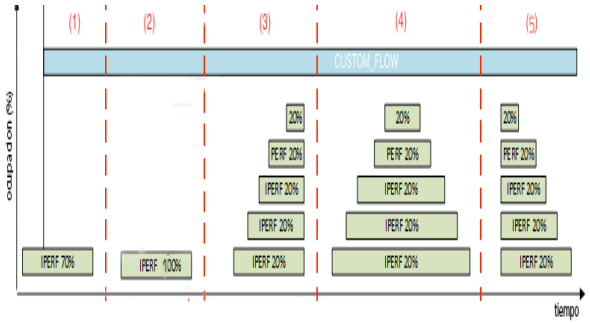


Fig. 8. Secuencia de ejecución dividida en tramos de P2

La secuencia de ejecución de flujos (*Fig. 8*) se describe a continuación:

(P2.1) Se inicia la prueba con una ocupación inicial de enlace del 70%. Se inicia un flujo poco después.

(P2.2) Se añade un tráfico interferente transmitiendo al 100%

(P2.3) Se introduce e incrementa de manera progresiva un flujo interferente en *steps* del 20% de ocupación hasta llegar al 100%

(P2.4) Se incrementa de manera progresiva un flujo interferente en *steps* del 20% de ocupación hasta llegar al 100% y se decrementa hasta llegar al 0%.

(P2.5) Se decrementa de manera progresiva el tráfico interferente de 100% a 0% de ocupación, en steps del 20%.

En la *Fig. 9* se pueden observar los resultados de (*P2*), identificando los tramos definidos anteriormente.

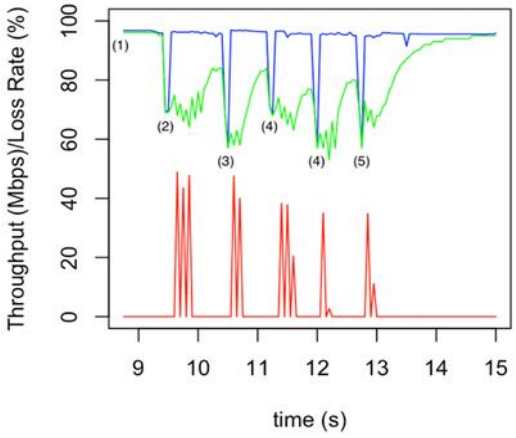


Fig. 9. Resultados de (P2)

Observando los resultados, la reactividad a las pérdidas se puede descomponer en tres fases (entre los segundos 10 y 12), observables en la *Fig. 10*.

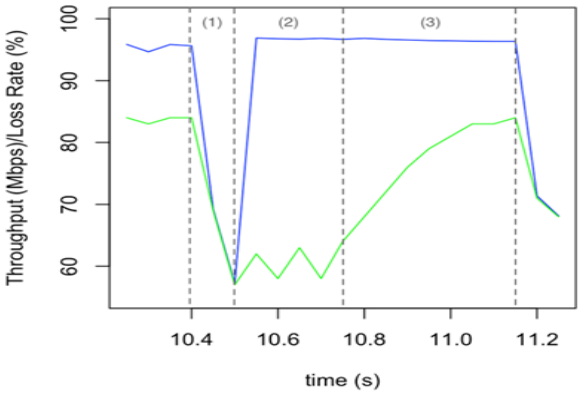


Fig. 10. Descomposición en fases de la reacción a pérdidas

En un primer lugar, se disminuye la velocidad de envío inicial hasta que la velocidad de envío agregada entre el flujo interferente y el flujo generado deja de provocar congestión (*Fig. 10* - (1)).

El control de congestión detecta nuevamente que el ancho de banda disponible es el máximo del canal. En ese punto (segundo 10,5), se entra en una segunda fase en la que se experimentan pérdidas generadas por el período de congestión anterior (*Fig. 10* - (2)). Por otro lado, en esa misma fase, dado que la estimación del enlace ya no detecta congestión, se incrementa la ventana de envío.

Estos dos aspectos, generan una variación en la velocidad de envío alrededor del valor en que se finaliza la primera fase de pérdidas. En ciertas situaciones,

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

cuando estas son considerablemente elevadas, la velocidad de envío decrementa proporcionalmente a estas. Finalmente, una vez se han tratado las pérdidas, se entra en recuperación (*Fig. 10* - (3)).

Se observa que los mecanismos reaccionan tanto a la congestión como a las pérdidas. La congestión provoca una disminución de la velocidad de envío, la cual se produce de manera continuada hasta descongestionar el enlace. Las pérdidas provocan disminuciones puntuales que previenen que el mecanismo entre en recuperación cuando el canal aún no está preparado. Éste también es capaz de recuperar la velocidad de envío en cuanto el canal lo permite en tiempos inferiores a medio segundo, demostrando la adaptabilidad del control de congestión.

Prueba 3

Con tal de demostrar (O3) se plantean 3 comparativas. Para ello, se muestra el comportamiento de un flujo dotado con los mecanismos diseñados al compartir el enlace con otros protocolos de transporte.

En un primer lugar (O3.1), se comprueba cómo se comporta ante un flujo agresivo como es un flujo UDP sin sistema de control de congestión. Los resultados se pueden observar a continuación.

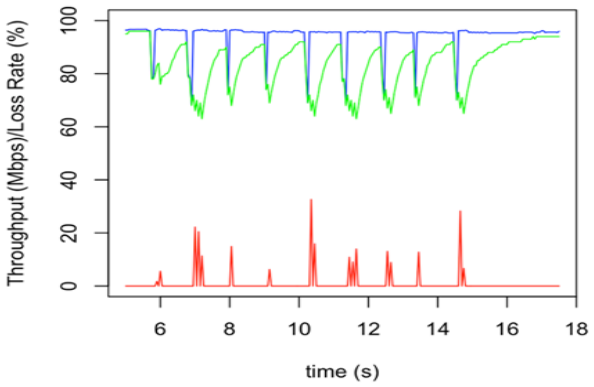


Fig. 11. Rendimiento en comparativa entre UDP

En la *Fig. 11* se puede observar el rendimiento de un flujo ante la presencia de un flujo UDP constante a 25Mbps. En estas circunstancias, el control de congestión no es capaz de ocupar todo el canal dado que el flujo con el que lo comparte es de elevada agresividad. Dicha situación genera pérdidas considerables que hacen que, aunque en promedio la velocidad de envío media sea de aproximadamente 75Mbps (el ancho de banda libre del canal), el gran número de retransmisiones a realizar implican una disminución significativa de la velocidad real de transmisión de datos. Esto es debido a que el flujo UDP se muestra invariable ante las pérdidas por saturación del enlace.

El comportamiento observado ante flujos UDP agresivos es coherente con el planteamiento del protocolo (obtener el mayor ancho de banda posible en cada momento, siendo agresivo al compartir el enlace).

Uno de los puntos a estudiar es una mayor estabilización ante flujos que no implementan control de congestión y que, por lo tanto, no varían su velocidad de envío a lo largo de la transmisión.

En contraste a dicha situación, se encuentran los resultados mostrados en la *Fig. 12* y *Fig. 13* representando el comportamiento del protocolo al compartir el enlace con un flujo *TCP friendly* (O3.2).

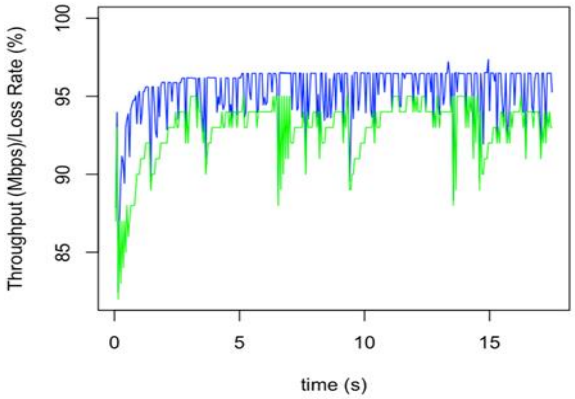


Fig. 12. Rendimiento MBTAP en comparativa TCP Friendly vs UDP-modified (TCP primero)

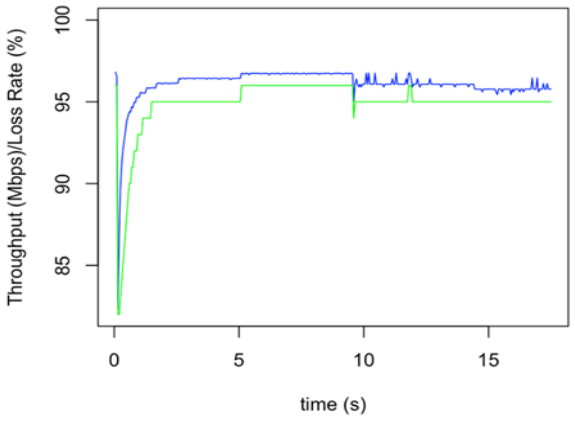


Fig. 13. Rendimiento en comparativa TCP Friendly vs UDP-modified (UDP-modified primero)

Tanto en la *Fig. 12*, se inicia un flujo *TCP Friendly* cuando el flujo de UDP modificado ya se encuentra en transmisión, como en la *Fig. 13*, se inicia un flujo UDP modificado cuando el *TCP Friendly* ya se encuentra en transmisión, se puede apreciar como la presencia de un flujo *TCP Friendly* en el canal no comporta una disminución de la velocidad de transmisión.

De los resultados mostrados, se puede concluir que el control de congestión se encuentra en el punto de agresividad planteado en su diseño. Suficientemente agresivo para utilizar la mayor parte del enlace en detrimento de otros flujos sin llegar al nivel de agresividad de UDP.

Por último, la Fig. 14 y la Fig. 15 muestran el comportamiento al transmitir simultáneamente dos flujos con el mismo mecanismo de control de congestión (O3.3). En ellas se observa cómo existe una contienda por el ancho de banda del canal (100Mbps), logrando una repartición del 50% para cada uno pero produciéndose pérdidas.

Ante dicha situación se podría esperar que ambos flujos se estabilizarán en velocidades de envío cercanas a 50Mbps. No obstante, la velocidad en la que el flujo se

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)

estabiliza depende de diversos factores, los cuales fluctúan durante la transmisión. Dado que el estado de la red que considera el mecanismo es independiente para cada uno de los flujos, el comportamiento de cada uno de estos es indeterminado para el otro flujo. Esto hace que ambos flujos compitan por el ancho de banda disponible sin tener en cuenta la presencia de otros flujos, con lo que se genera congestión dada la agresividad del protocolo. Así pues, la implementación actual provoca una transmisión poco estable al encontrar diversos flujos que utilizan el mismo mecanismo a través de un mismo enlace.

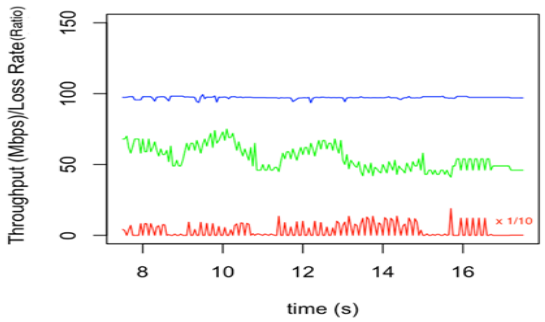


Fig. 14. Transmisión simultánea entre dos flujos con el control de congestión propuesto (A)

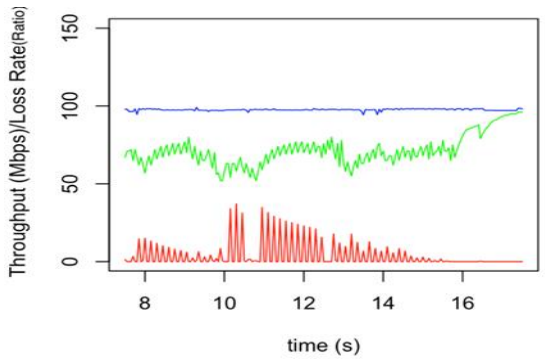


Fig. 15. Transmisión simultánea entre dos flujos con el control de congestión propuesto (B)

V. Conclusiones

El diseño de los protocolos de transporte base como son TCP y UDP no fue concebido para su uso en redes LFN. Esto es debido a que su alto ancho de banda y elevado retardo provoca ineficiencias para este tipo de comunicaciones.

Este artículo repasa el estado del arte de propuestas sobre TCP, UDP e incluso nuevos protocolos como SCTP que ofrecen soluciones eficientes, pero no definitivas.

En este sentido, se presentan dos mecanismos. El primero es el cálculo de estado de la red, el cual se realiza al inicio de la comunicación, así como durante la misma de manera *in-band* mediante el uso de los mensajes de datos. Este proceso se compone del envío de varias ráfagas de mensajes que permiten calcular el ancho de banda máximo del enlace, además de extraer otra información como el *Receiving Rate* (RR) y el *Round Trip Time* (RTT). Esto permite al segundo mecanismo, el control de congestión, iniciar la comunicación utilizando todo el ancho de banda disponible, así como adaptar el envío durante la comunicación.

Estos mecanismos combinados tienen como objetivo obtener una alta eficiencia y adaptabilidad sobre las redes LFN, así como un comportamiento agresivo frente a otros flujos.

Se ha demostrado mediante diferentes pruebas sobre un testbed de emulación WAN que estos mecanismos permiten obtener un alto ancho de banda, cercano al 96% de su capacidad total, de manera eficiente, incluso con valores de pérdidas aleatorias cercanas al 0,1%.

Además, en una de las pruebas se ha evaluado y verificado su capacidad de adaptación ante variaciones de ancho de banda disponible durante la comunicación, introduciendo diferentes flujos concurrentes en diferentes momentos de la transmisión.

Finalmente, con el fin de evaluar también el nivel de agresividad de estos mecanismos frente a otros flujos, se han realizado varias pruebas que prueban este comportamiento agresivo frente a estos (TCP, UDP y otro flujo que también implementa el mecanismo propuesto).

De cara al futuro, está previsto diseñar un protocolo de transporte que incluya estos dos mecanismos y realizar pruebas de funcionalidad integral de éste. Además, se deberían evaluar los mecanismos propuestos en entornos WAN reales y proponer alguna solución de mejora del *fairness* entre varias sesiones utilizando el mecanismo de congestión propuesto.

Referencias

[1] Jacobson, V., LBL, Braden, R., ISI, "TCP Extensions for Long-Delay Paths" October 1988

[2] Jacobson, V., Braden, B., Borman, D., Satyanarayanan, M., Kistler, J.J., Mummert, L.B. and Ebling, M.R., 1992. RFC 1323: TCP extensions for high performance.

[3] Bullot, H.; Les Cottrell, R., "Evaluation of Advanced TCP Stacks on Fast Long-Distance Production Networks" Journal of Grid Computing, 2003, Volume 1, Number 4, Page 345

[4] Hanson, R. H., Lespagnol, A., Mazraani, T. Y., Milburn, B. J., White, J. B., & Dabir, S. C. (1997). "Traffic management and congestion control for packet-based networks." U.S. Patent No. 5,633,861. Washington, DC: U.S. Patent and Trademark Office.

[5] Iren, S., Amer, P. D., & Conrad, P. T. (1999). The transport layer: tutorial and survey. ACM Computing Surveys (CSUR), 31(4), 360-404.

[6] Bashir, K., "Comparative study on advance TCP stacks and their performance analysis," 8th International Multitopic Conference, 2004. Proceedings of INMIC 2004., 2004, pp. 256-263.

[7] Kelly, T., "Scalable TCP: improving performance in highspeed wide area networks". SIGCOMM Comput. Commun. Rev. 33, 2 (April 2003), 83-91.

[8] Leith, D.; Shorten, R., "H-TCP: TCP for high-speed and long-distance networks"

[9] Nagamalai, D., Jae-Kwang Lee, (2004) "Performance of SCTP over high speed wide area networks," Cybernetics and Intelligent Systems. IEEE Conference on , vol.2, no., pp.890,895, 20046.

[10] Gu, Y., Grossman., R. L., (2007) "UDT: UDP-based data transfer for high-speed wide area networks." Computer Networks 51, no. 7: 1777-1799.

[11] Eckart, B., He, X., & Wu, Q. (2008, April). Performance adaptive UDP for high-speed bulk data transfer over

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)
EDITORIAL UNIVERSITAT POLITÈCNICA DE VALÈNCIA

dedicated links. In Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on (pp. 1-10). IEEE.

[12] Gu, Y., R. L. Grossman. (2007) "UDT: UDP-based data transfer for high-speed wide area networks." Computer Networks 51, no. 7: 1777-1799.

[13] WANem, The Wire Area Network emulator. Version 3.0 "wanem.sourceforge.net" [Fecha de consulta: 27 abril 2017]

[14] iPerf3 - The ultimate speed test tool for TCP, UDP and SCTP https://iperf.fr [Fecha consulta: 27 abril de 2017]

This work is licensed under a Creative Commons 4.0 International License (CC BY-NC-ND 4.0)