Análisis a Nivel-Paquete en simuladores de red contemporáneos.

Gilberto Flores Lucio, Marcos Paredes Farrera, Emmanuel Jammeh, Martin Fleury, Martin J. Reed
Mohammed Ghanbari *IEEE Fellow*
Universidad de Essex
Wivenhoe Park, Colchester, United Kingdom
Email: {gflore,mpared}@essex.ac.uk

***Abstract*- Este articulo presenta un estudio comparativo de los simuladores híbridos de red mas conocidos: Modeler de OPNET y Ns-2, así como NCTUns, que incorpora las capas reales del protocolo TCP/IP en la simulación. Otros estudios en esta área se han confinado generalmente a solo un simulador. La motivación de este articulo es proporcionar una guía a los investigadores que emprenden simulaciones de la red a nivel-paquete. Las salidas del simulador fueron comparadas contra los valores de salida de la red de prueba. La comparación experimental consistió en desplegar tráfico de Constant-Bit-Rate (CBR) y una sesión de File Transfer Protocol (FTP), ambos en la red de prueba y los simuladores. Tráfico de CBR fue utilizado como un medio de calibración para los resultados, mientras FTP fue utilizado debido a su simplicidad de modelar, puesto que modelos más complejos podrían obstruir la comparación. Una herramienta de análisis genérica fue utilizada para examinar el comportamiento de la red en diferentes escenarios con distintos tráfico de fondo. Los mismos escenarios fueron recreados en los simuladores con la finalidad de inspeccionar su veracidad. Los resultados muestran que el Modeler de OPNET replica el tráfico con una sobresaliente exactitud, los módulos de ns-2 son mas usados para un análisis general, al momento de las pruebas el software de NCTUns estaba en una etapa temprana de desarrollo.**

***Index Terms*— Análisis a nivel-paquete, NCTUns, ns-2, OPNET Modeler, Simulador de Red.**

I. Introducción

Los simuladores de redes han madurado desde que aparecieron por primera vez como herramientas de desarrollo administración y predicción. Normalmente los simuladores son utilizados como herramientas de administración de red, para el cual análisis a nivel-paquete no es requerido usualmente. Los simuladores de red han sido utilizados en detalle para estudios de calidad de servicio. En el análisis a nivel-paquete, métricas de desempeño como la media (promedio aritmético) para el tamaño del paquete y el arribo del paquete y demás son requeridas. Nuestra investigación [1] incluye el análisis a nivel paquete de una transmisión de video donde el retardo y la variación del retardo (jitter) a nivel paquete afecta la calidad del video recibido [2]. El tamaño del paquete y el ancho de banda instantáneo del paquete, varia de acuerdo al tipo de salida de la imagen, por lo que afecta la respuesta del enrutador [3]. Sin embargo mas estudios se requieren para establecer líneas de investigación a investigadores para que puedan seleccionar un simulador mas adecuado a los análisis a nivel-paquete [4]. La referencia [5] reporta que existe una "falta de credibilidad" que los estudios basados en simuladores necesitan investigar. La facilidad que los simuladores proveen en la evaluación del medio ambiente en una red no se puede descartar. Existe un numero considerable de herramientas de simulación disponibles. Las características principales que generalmente las dividen son: precisión, rapidez, facilidad de uso y costo.

Este articulo se concentra en la precisión de la simulación a nivel paquete en comparación con las mediciones tomadas en una red real. Tres simuladores de red que puede realizar este tipo de análisis son Modeler de OPNET[1] [6], ns-2 del proyecto Virtual Internetwork Testbed VINT [7] y el simulador de red de la Universidad Nacional Chiao Tung NCTUns [8]. Los primeros dos son seleccionados por su amplio uso en las comunidades académicas, comerciales e industriales [9]. El ultimo el cual ha sido desarrollado recientemente, fue seleccionado debido a que NCTUns a diferencia de Modeler y ns-2 utiliza la capa real de protocolos en lugar de modelar las capas. Los autores de este articulo no tienen ninguna conexión con los desarrolladores de los simuladores además de que no tienen ningún interés comercial en ellos.

La manera ideal para probar la precisión de un simulador es medir tráfico en una red y compararlo con los resultados de los simuladores. Por lo tanto este articulo presenta resultados reales y resultados simulados (de las tres herramientas de simulación), en donde tráfico de CBR y de sesión de FTP son generados con la intención de probar el comportamiento de la red. El análisis considera principalmente la utilización de ancho de banda al nivel de paquete-por-paquete. El tráfico de CBR fue seleccionado debido a la simplicidad que presenta, y FTP debido a que es utilizado comúnmente y a su comportamiento dinámico. CBR en particular es un análisis benéfico ya que provee un patrón exacto donde pocos factores pueden influenciar el modelo. Por lo que CBR actúa como una forma de calibración. FTP fue seleccionado debido a que es suficientemente complejo para presentar una variedad de distintos comportamientos pero no tan complejo como otros protocolos a nivel aplicación (p.e. HTTP). Mientras que podríamos haber simulado escenarios más complicados esto podría haber sido contraproducente ya que las diferencias

[1] OPNET Modeler fue obtenido bajo el programa universitario de OPNET

entre la red y la simulación podrían ser menos aparentes.

Las pruebas iniciales utilizaron los perfiles de configuración que proveen los simuladores, sin embargo, diversas modificaciones manuales fueron realizadas en diferentes etapas de los experimentos para investigar posibles mejoras. Estas modificaciones no fueron mas que las que un usuario experimentado pudiera realizar en los simuladores. En dado caso en que estos ajustes no se realizaran, podríamos correr el riesgo de experimentar con los simuladores de una manera ingenua, mientras, de hecho, los autores en conjunto tienen 10 años de experiencia en los simuladores. Una herramienta de análisis genérica llamada `tcpflw` fue utilizada para analizar el comportamiento de la red real. `tcpflw` ha sido utilizada con éxito para analizar el desempeño de sesiones de transmisión de video [1][3] así como para proveer datos estadísticos para desarrollar un método de tarificación de servicios de Internet [10]. Los resultados mostrados aquí fueron comparados directamente con los resultados del simulador.

El objetivo de este articulo no es definir cual es el mejor simulador ya que existen muchas variaciones en los parámetros y diferentes posibles escenarios de red para determinar adecuadamente todos ellos en un solo articulo. En lugar de esto, este articulo demuestra como la adecuación de los simuladores puede validarse para el caso particular para el envío a nivel paquete en una red IP. Esperamos que este estudio actué como una guía hacia otros investigadores que deseen validar otros escenarios de red.

El resto del articulo esta organizado de la siguiente manera: Sección 2 presenta los simuladores de red, la red de prueba experimental y los experimentos utilizados para la prueba. Sección 3 presenta los resultados comparativos obtenidos y finalmente Sección 4 provee una discusión además de presentar conclusiones.

II. Descripción del trabajo experimental.

Los simuladores de red pueden dividirse en varios tipos (por protocolo, tecnología, o método de procesamiento), pero la categorización mas general es por el método de simular. Existen dos métodos típicos de simulación: 1) eventos discretos 2) simulación analítica [11][12]. El primero produce predicciones en la red a bajo nivel (paquete-por-paquete), lo cual los hace precisos pero lentos al momento de generar los resultados. El segundo, utiliza modelos matemáticos para producir los resultados a mayor velocidad en comparación con el primero, pero sacrifica precisión. La utilización más usual es combinar ambas metodologías para formar un simulador híbrido con el fin de proveer un desempeño aceptable en términos de velocidad pero manteniendo la precisión en áreas criticas. Los tres simuladores utilizados en los experimentos son simuladores híbridos.

A. Las herramientas de Simulación

Esta sección provee una descripción resumida de las capacidades de los simuladores.

OPNET Modeler. El Modeler es solo una de las muchas herramientas que están incluidas en el OPNET Technologies suite. La versión utilizada para los experimentos fue la versión 9.0A. La maquina del OPNET Modeler es un modelo de maquina de estado finito, en combinación con un modelo analítico. Modeler puede modelar protocolos, componentes y comportamientos de redes, con alrededor de 400 modelos de funciones de propósito especial [13]. La Interfase Grafica de Usuario (Graphical User Interface GUI) y el animador tiene una gran cantidad de documentación y casos de estudio disponible. Varios editores son proveídos con el fin de simplificar los diferentes niveles de modelaje que el operador de la red requiere. Ajustes en los parámetros del modelo pueden tener un gran efecto en la precisión de la simulación. A pesar de que el kernel de simulación no es “open source” el resto de los modelos pueden ser parametizados.

Ns-2 es la segunda versión de una herramienta de un simulador de redes desarrollado por el proyecto VINT [14]. Este es un simulador de redes basado en eventos el cual es ampliamente usado por la comunidad investigadores en redes. Incluye diversos modelos comunes del protocolo IP incluyendo las nuevas versiones, como Reliable Multicast y TCP Selective Acknowledgment [15]. Un animador de red llamado `nam` [16] reproduce una animación a nivel paquete y graficas especificas basadas de los protocolos con el fin de diseñar, detectar y corregir errores en los protocolos de la red. En adición, existen diferentes niveles de configuración en ns-2, esto debido a su naturaleza de ser “open source”, incluye igualmente la capacidad de crear aplicaciones y protocolos personalizados, de igual forma tiene la habilidad de cambiar parámetros en distintas capas. Ns-2 ha sido construido como un proyecto asistido por diferentes personas por lo que la confiabilidad de los módulos puede variar.

NCTUns fue creado por S.Y, Wang. NCTUns esta basado en el simulador de redes de Harvard [17]. El simulador tiene una arquitectura abierta la cual permite hacer modificaciones fácilmente en los módulos de los protocolos[18]. Esto nos permite ajustar los parámetros del protocolo en la red y las capas de transporte con el fin de obtener una mayor precisión en los resultados. El GUI incluye un modulo para trazar graficas de desempeño. Una de las características principales de esta herramienta es la metodología de “kernel-reentering simulation methodology”[19] la cual permite que un paquete entre y salga y reentre el kernel afectando el resultado en forma diferente en cada secuencia de la simulación. El kernel permite a los procesos trabajar primero como un host al inicio y como un enrutador en la siguiente y como enrutador en la siguiente. Adicionalmente el simulador utiliza las capas originales del protocolo TCP/IP así como aplicaciones reales. El simulador NCTUns funciona bajo el sistema operativo FreeBSD[20] versión 4.7 y el sistema X11 el cual usa las librerías de Qt[2]. NCTUns también es “open source” software.

Los siguientes parámetros del modelo fueron definidos para

[2] Qt es una multiplataforma basada en C++ que permite desarrollar aplicaciones que corren nativamente bajo Windows, Linux/Unix, Mac OS X [21].

igualar adecuadamente la configuración de la red real. Por supuesto los diseñadores de simuladores de red tienen la intención de proveer al usuario la facilidad de definir los parámetros de configuración, ya que variantes del protocolo pueden afectar considerablemente el desempeño. Este es el caso en particular cuando se hace un análisis a nivel-paquete. Un usuario no experimentado puede utilizar el simulador de red sin ajustar los parámetros del modelo a las características de la red bajo prueba. La operación por omisión de los simuladores difiere y consecuentemente es necesario realizar diferentes ajustes en los diferentes simuladores. El sistema operativo Linux fue utilizado en las maquinas de la red de prueba. Linux utiliza la versión *New Reno* del protocolo TCP y, donde fuera posible este parámetro se debe definir. (TCP con *New Reno*, mejora la capacidad de recuperación del "Fast Protocol" cuando múltiples paquetes se han perdido. *New Reno* también puede recuperar paquetes sin necesidad de retransmitir un tiempo fuera [22]). Otros dos parámetros fueron definidos: (1) *Window Scaling* (esto permite anunciar tamaños de ventana más grande que 65 KB [23]), y (2) la opción *Time-Stamp* [23], la cual reproduce la capacidad de recrear eco en nuestra red en ambas direcciones. En ns-2 la opción de *link latency* fue habilitada para rastrear *Round-Trip-Times* (RTT). El *tamaño máximo de segmento* fue definido a 1500, el cual es el tamaño máximo de una frame de Ethernet. El parámetro de *Windows Size* fue 100 para evitar valores del desplazamiento del tamaño de la ventana restringidos. Para permitir a otros que repitan los experimentos y verificar los resultados, los valores de los parámetros para los simuladores se muestran en la Tabla 1.

TABLA 1. PARÁMETROS USADOS EN LOS SIMULADORES

Simulador	*Parámetros*	Valor
OPNET Modeler	New Reno	Habilitado
	Window Scaling	Habilitado
	Time Stamp	Habilitado
ns-2	Link Latency	Habilitado
	Max Segment Size	1500
	Window Size	100
NCTUns	New Reno	Habilitado
	Fast Forwarding	Habilitado

B. La red de prueba.

Fig.1 muestra la red de prueba utilizada para estos experimentos. Esta conformada por cinco PC's, cada una con un procesador de 1.7 GHz, corriendo el sistema operativo Linux con tarjetas de red IntelPro 400. Dos ellas operan como un par Client-Server, y las otras dos como un par de generador de tráfico/sink de tráfico (Traffic G. Y Traffic S.), y la quinta como un enrutador (Router), en adición un hub de 10 Mbit/s es utilizado para conectar dos PC's al enrutador. Enlaces y de 10 y 100 Mbit/s fueron utilizados. Los sistemas Linux tuvieron una modificación (Universidad de Kansas Real-Time Linux KURT [24]) para reducir la granularidad de la temporización. Esto fue requerido para que la temporización de tráfico de CBR a alta velocidad se pudiera mantener con una razonable variación de retraso (jitter). Sin esta modificación se encontró que el desempeño decrecía drásticamente debido a problemas con la temporización.

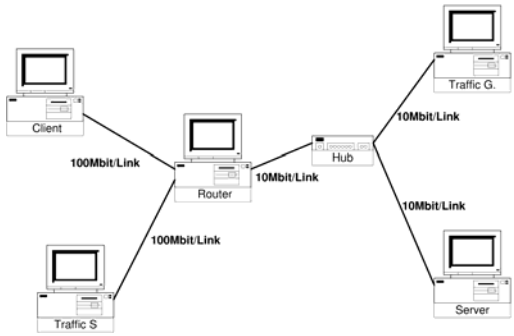

Fig. 1. Red de Prueba

C. Experimentos

Dos grupos de experimentos fueron llevados a cabo, el primero con el fin de conocer la precisión de los simuladores con tráfico de CBR y el segundo con una sesión de FTP para simular el efecto del tráfico de datos con "best-effort":

EL Tráfico de CBR se caracteriza por tener una utilización de ancho de banda fijo en la red y es típicamente usado por aplicaciones como video y audio; como ejemplo tenemos cuando reservación de ancho de banda por medio del protocolo "Resource Reservation Protocol (RSVP)" en la calidad de servicio IntServ [25]. Este tipo de aplicaciones normalmente requiere un estricto limite en "delay" y "jitter"en el envío de los paquetes de CBR. Por lo tanto, creemos que entender como se comporta el tráfico de CBR en la red puede ser de relevancia para aquellos que consideran transmitir video dentro de una red. Un patrón de tráfico CBR se genera al fijar el tamaño del paquetes así como el tiempo de arribo entre los paquetes (Inter Arrival Time IAT). Los generadores de tráfico *stg* y *rtg* los cuales vienen con el simulador NCTUns fueron utilizados en la red de prueba para crear tráfico con características de CBR. Este tráfico fue introducido después a los simuladores.

Sesión de FTP, FTP tiene la intención de compartir y transferir información entre dos computadoras. Bhushan [26], en el proyecto MAC del MIT, propuso el primer mecanismo de transferencia de archivos en 1971. Casi todos los servidores y clientes de FTP siguen el mas reciente RFC No. 959 [27], donde se explica: el modelo de utilización para FTP, el conjunto de comandos utilizados; y como el protocolo trabaja con los comandos de control y la conexión de datos.

D. Metodología

Para evaluar la precisión de los simuladores, datos de una red real fueron medidos. Dos grupos de experimentos fueron realizados con diferentes escenarios. Primero tráfico de CBR fue introducido a la red de prueba y segundo una sesión de FTP fue realizada de un Client a un Server para transferir una archivo de 10 MB. Bajo todos los escenarios, la técnica de medición consiste en monitorear el tráfico de IP utilizando `tcpdump` el cual es una herramienta utilizada ampliamente para la captura de paquetes por la comunidad de investigadores [28]. La información producida por cada paquete fue: medición del "timestamp", dirección IP fuente/destino y fuente del puerto, tamaño del paquete en Ethernet y tamaño del paquete de IP.

Posteriormente, los datos fueron analizados con el programa local `tcpflw`, el cual es capaz de clasificar el tráfico en flujos y después realiza 4 tipos de análisis: (1) Tiempo de arribo entre paquete (Packet Inter Arrival Time PIAT); (2) Tamaño del Paquete; (3) Utilización de ancho de banda por paquete; (4) Ancho de banda promedio.
`tcpflw` también puede exportar tráfico de la red hacia los simuladores.

En total, se llevaron a cabo 24 experimentos con 6 escenarios distintos, 16 experimentos fueron realizados para tráficos con CBR y 8 para la sesión de FTP. La Tabla 2 muestra los diferentes escenarios, algunos con tráfico cruzado y otros sin él. Cada escenario fue replicado en los tres simuladores y la red de prueba.

TABLA 2. ESCENARIOS DE TRÁFICO PARA CBR Y FTP.

Scenario	Client-Server	Generator- Sink
CBR1	cbr(2Mbit/s)	0
CBR2	cbr(2Mbit/s)	cbr(2Mbit/s)
CBR3	cbr(5Mbit/s)	0
CBR4	cbr(5Mbit/s)	cbr(6Mbit/s)
FTP1	FTP(10MB file)	0
FTP2	FTP(10MB file)	cbr(6Mbit/s)

III. RESULTADOS

A. Escenarios de CBR

Fig. 2. compara los resultados del escenario CBR1 obtenidos de los tres simuladores y de la red de prueba desde la perspectiva del enrutador. (Escenario CBR2 y CBR3 mostraron un comportamiento muy similar con el comportamiento de CBR1). Como se puede observar ns-2 emula mas acertadamente el comportamiento de CBR en comparación de OPNET y NCTUns, definimos que la razón es por el método de importar patrones de tráfico de CBR dentro del Modeler de OPNET. Existen 3 técnicas para importar ese tráfico en OPNET:

1. Importar medidas de tráfico como archivos de "trace" utilizando las herramientas de análisis de flujo.
2. Crear un modelo de procesos que importe archivos "trace".
3. Importar archivos "trace" como una fuente de tráfico en el fondo de OPNET.

Hemos utilizado la ultima, ya que en análisis de OPNET ACE no estaba disponible cuando estos experimentos fueron realizados y la generación de un modelo de procesos no fue requerido por ns-2. Modeler utiliza un archivo de "trace" como fuente para generar tráfico de fondo a una tarifa especifica durante esta parte del experimento, lo cual puede explicar el "comportamiento irregular" [29] presentado. NCTUns presenta un desplazamiento constante (0.25 Mbit/s) del nivel de CBR comparado con el nivel de la red de prueba, el cual probablemente sucede debido a los problemas de calibración en el programa de trazamiento dentro del simulador.

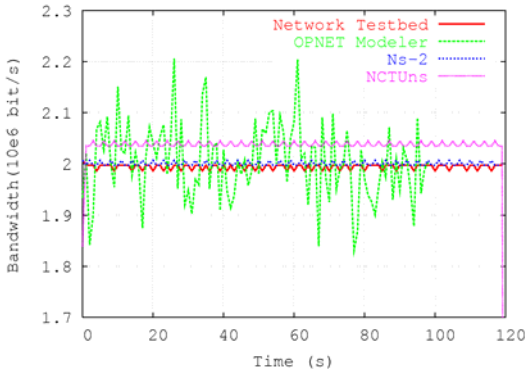

Fig. 2 CBR1 desde la perspectiva del enrutador.

Figs. 3-5 muestran los resultados del escenario CBR4, representando la combinación de dos flujos de tráfico de CBR, que efectivamente "inundan" el hub. Nuevamente, existieron discrepancias entre los resultados de la simulación y los resultados de la red de prueba.

Los valores aproximados de ns-2 tienen un desplazamiento de 0.5 Mbit/s encima del valor capturado en la red de prueba. Sin embargo, la salida incluye una pérdida repentina en ancho de banda entre 60 y 80 segundos como se puede observar por el Client (Fig. 3). Este experimento fue repetido varias veces para confirmar que esta anomalía vuelve a ocurrir. Una característica de los resultados de NCTUns para CBR4 (para el Server, Client y Router) es que el rango de tiempo en la salida es mas largo que en la red real. Este es un efecto común cuando los objetos de simulación no son eliminados dentro del programa y consecuentemente continúan generando datos de salida. La representación de tráfico de CBR en NCTUns fluctúa rápidamente y siempre se muestra desplazado debajo del valor de la red real, como puede verse en las Figs. 3-5; con valores de 0.5, 0.75 y 1.75 Mbit/s respectivamente. En el escenario CBR4, OPNET representó mas precisamente la respuesta de la red real en comparación de los demás simuladores. El ancho de banda fue desplazado aproximadamente 0.25 Mbit/s sobre el valor de la red.

En general, se encontró que en la red de prueba el administrador de paquetes del enrutador, manejo el tráfico de diferentes flujos inequitativamente mientras que ambos simuladores separaron los flujos equitativamente en el enrutador. Claramente, los simuladores asumen el comportamiento del administrador del enrutador mientras, en practica, un numero de diferentes algoritmos son utilizados en el "kernel" del enrutador. Esto muestra una debilidad en la simulación a nivel paquete, en que, sistemas actuales tienen un gran numero de variables de control para retransmitir paquetes y no todas estas pueden ser replicadas adecuadamente.

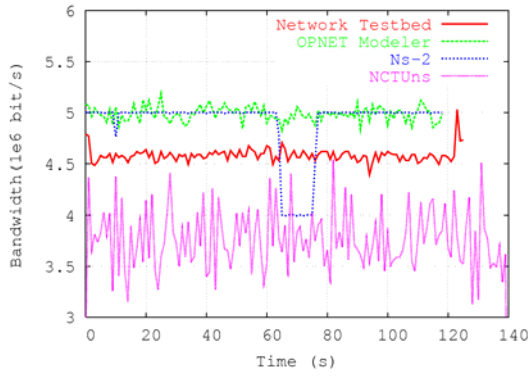

Fig. 3 CBR4: Desde la perspectiva del Client.

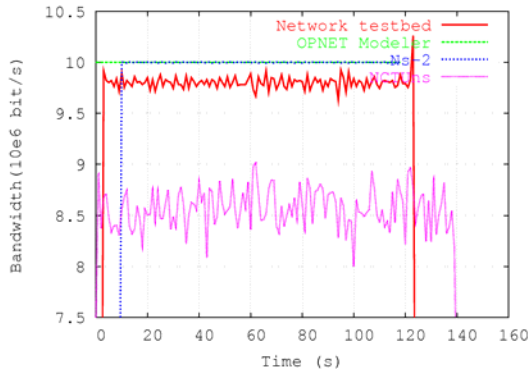

Fig. 4 CBR4: Desde la perspectiva del Enrutador.

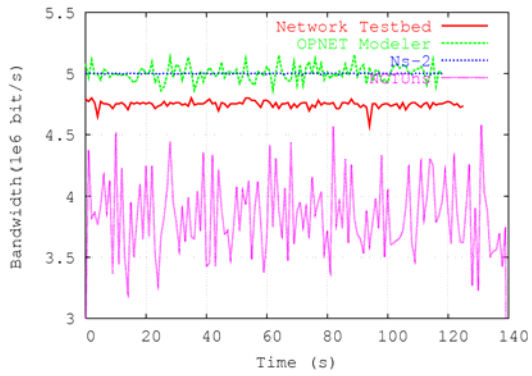

Fig. 5 CBR4: Desde la perspectiva del Server.

B. Escenario de FTP

Para los resultados obtenidos en el escenario FTP1 y Figs. 6 y 7 desde la perspectiva del enrutador, un archivo de 10MB fue transferido por medio del FTP del Server al Client. Tráfico que va en dirección reversa del Client al Server consistió de paquetes a nivel sesión y reconocimientos de transferencia de datos en TCP. En escenario FTP1, no existió tráfico de fondo, mientras que en el escenario FTP2, Figs. 8 y 9, tráfico de CBR de 6 Mbit/s fue enviado del Traffic G. al Traffic S. de la Fig 1. Los resultados graficados solo muestran el tráfico de la transferencia de archivos, estos no observan el tráfico preliminar para establecer la sesión de FTP así como el tráfico subsecuente para finalizar la conexión. La versión de FTP utilizado en la red de prueba fue vsFTPD versión 1.1.0[3]. De hecho vsFTPD tienen mas características de seguridad que otras versiones de FTP, sin embargo su comportamiento de tráfico parece ser el típico (i.e patrones de tráfico dinámico).

Debido a que el simulador ns-2 asume un ancho de banda casi constante la duración de la transferencia del archivo aparece en la salida del simulador como menor a la mostrada en la red de prueba. El ancho de banda consumido durante la transferencia de información esta alrededor de los valores de referencia de la red Figs. 6 y 8. Sin embargo en Figs. 7 y 9 la representación de ns-2 es a lo máximo de 1 Mbit/s de diferencia del valor de la red real. La duración de la transferencia del archivo de la salida del NCTUNs también fue reducida, esto es causado por el ancho de banda casi "plano" mostrado por el simulador. Los valores de ancho de banda varian de 0.8 Mbit/s a 1 Mbit/s separado del valor de la red real. En el escenario FTP2 no existieron datos a graficar del NCTUns debido a que simplemente el simulador se "congelaba" mientras modelaba este escenario.

El Modeler de OPNET mostró una muy precisa representación de los dos escenarios de FTP tanto en la duración de la transferencia del archivo, el nivel de ancho de banda consumido, así como las fluctuaciones en ancho de banda reportadas por la red real. El Modeler de OPNET es tan preciso en replicar el comportamiento de la red, que los resultados actúan como confirmación de los valores tomados por los instrumentos de la red.

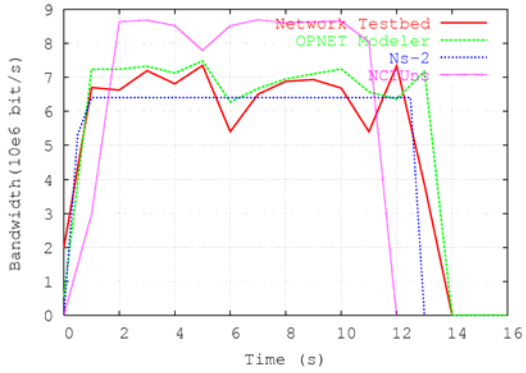

Fig. 6 FTP1: Desde la perspectiva del Enrutador (Desde el Server al Client).

[3] VsFTPD esta disponible en http://vsftpd.beasts.org/

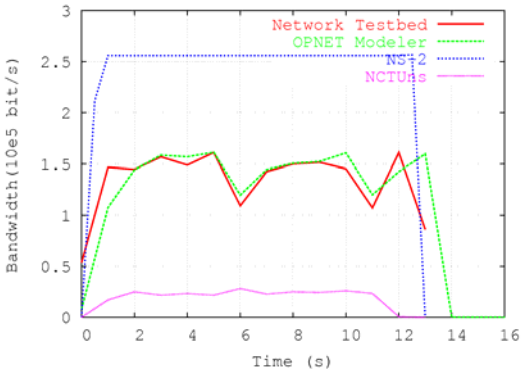

Fig. 7 FTP1: Desde la perspectiva del Enrutador (Desde el Client al Server)

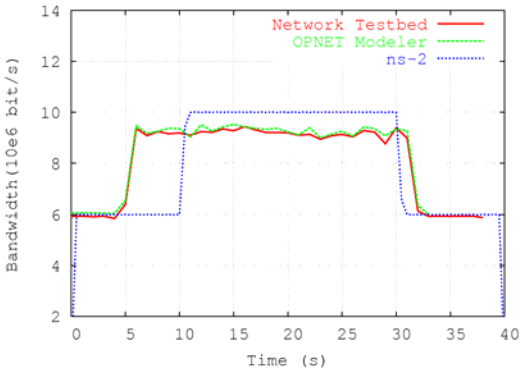

Fig. 8 FTP2: Desde la perspectiva del Enrutador (Desde el Server al Client).

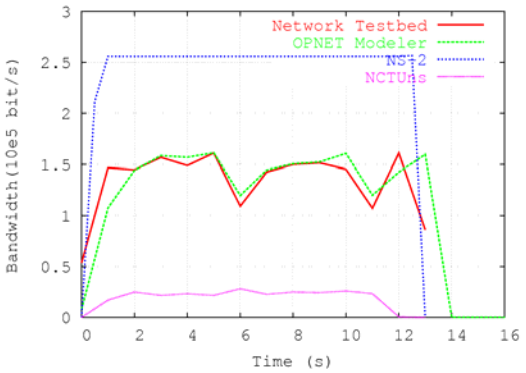

Fig. 9 FTP2: Desde la perspectiva del Enrutador (Desde el Client al Server)

IV. Conclusiones

En términos de la precisión en la estimación del ancho de banda para un flujo de tráfico simple del tipo CBR, ns-2 presento un mejor desempeño que el Modeler de OPNET y NCTUns en el escenario CBR4, el cual tiene mas de un flujo de tráfico ns-2 se comportó diferente a la red de prueba y en caso contrario Modeler presento resultados más precisos. NCTUns dio resultados más débiles en comparación con los otros dos simuladores pero se puede argumentar que este simulador esta aun en una etapa muy temprana de su ciclo de desarrollo. Hablando de los experimentos de FTP, el modelo de simulación de FTP del ns-2 solo indicaba un proceso de transferencia general en lugar de replicar el flujo de la red real. El Modeler de OPNET fue capaz de acercarse mas a los valores de la red de prueba, y en el caso de FTP2 las similitudes son sobresalientes. Una observación especifica se puede observar acerca de Modeler y ns-2 como resultado de estos experimentos. La precisión de la representación del tráfico de CBR fue muy similar. De otra manera FTP (a través de flujos de TCP y control de congestión) adapta su salida de acuerdo a las condiciones que prevalecen en la red, mientras que la respuesta del modulo de ns-2, no replicó este comportamiento, a diferencia del Modeler.

En términos del tiempo que se tomo para ejecutar la simulación, ambos ns-2 y el Modeler probaron ser más rápidos que NCTUns, requiriendo menos de un minuto para obtener los resultados. (No consideramos el tiempo de la simulación en ningún detalle mas que este). NCTUns provee la oportunidad [30] de emplear una aplicación real como lo es FTP en una red simulada, característica la cual de acuerdo a nuestro conocimiento no la proveen los otros simuladores. Ns-2 tiene una "suite" pequeña pero para redes de alta escala varias modificaciones y cuidado extra debe de considerarse para las alocución de la administración de la memoria y tiempo de CPU (técnicas abstractivas) [31]. El Modeler de OPNET tiene una "suite" grande (large software overhead) pero provee diversos módulos estadísticos a diferentes niveles [32]. NCTUns tiene también una "suite" pequeña.

Finalmente, no encontramos que la creación de este tipo de comparaciones de una manera precisa y justa entre diferentes herramientas de simulación de redes sea una tarea fácil a pesar de que nos centramos exclusivamente a dos tipos de tráfico relativamente simples. Recomendamos a aquellos investigadores, que realicen una evaluación futura para contemplar una metodología que evite tomar la decisión equivocada en referencia a un simulador.

References:

[1] E. Jammeh, M. Paredes-Farrera, and M. Ghanbari. Transporting real time transcoded video over the Internet using end-to-end control. In International PacketVideo Workshop, 2002.

[2] M. Ghanbari. Image Compression to Advanced Video Coding, IEE, London, UK, 2003.

[3] M. Paredes-Farrera, M. Fleury, M. Mbise, and M. Ghanbari, Best packetization scheme for Internet Video Communication, Electronic Letters, 40(23):1476-1478, 2004.

[4] J. Heidemann and K. Mills. Expanding confidence in network simulations. IEEE Network Magazine, 15(5): 58-63, 2001.

[5] K. Pawlikowski, H-D. J. Jeong, and J-S. R. Lee. On the credibility of simulation studies of telecommunication networks. IEEE Communications Magazine, 40(1): 132-139, 2002.

[6] C. Zhu, O. W. W. Yang, J. Aweya, M. Oullette, and D. Y. Montuno. A comparison of active queue management algorithms using the OPNET Modeler. IEEE Communications Magazine, 40(6): 158-167, 2002.

[7] V. Paxson and S. Floyd. Why we don't know how to simulate the Internet. In Winter Simulation Conference, pages 1037-1044, 1997.

[8] S.Y. Wang, C. L. Chou, C. H. Huang, C. C. Hwang, Z. M. Yang, C. C. Chiou, and C. C. Lin. The design and implementation of the NCTUns 1.0 network simulator. Computer Networks, 42(2): 175-197, 2003.

[9] L. Breaslu, K. Estrin, D. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, X. Ya, and H. Yu. Advances in network simulation. IEEE Computer, 33(5): 59-67, 2000.

[10] M. Paredes-Farrera, G. Flores Lucio, and M. Ghanbari. Billing Internet services based on user throughput and service categorization. In 3rd International Congress of Electromechanics and Systems Engineering, (CIIES 2002), 2002.

[11] A. M. Law, W. D. Kelton, Simulation Modeling and Analysis, 2nd edition, McGraw-Hill, New York, 1991
[12] M. Ghanbari, C. J. Hughes, M. C. Sinclair, and J. P.Eade, Principles of Performance Engineering for Telecommunication and Information Systems, IEE, Stevenage, UK, 1997
[13] OPNET Users' Manual, OPNET Architecture, OV.415.http://forums.opnet.com.
[14] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, M. Haldar, P. Handley, A. Helmy, J. Heidemann, P. Huang, S. Kumar, S. McCanne, R. Rejajie, S. Punnet, K. Varadhan, X. Ya, H. Yu, and D. Zappala. Improving simulation for network research. Technical Report 99-702b, USC Computer Science Department, 1999.
[15] K. Fall. Network emulation in the VINT/ns simulator. In Fourth IEEE Symposium on Computers and Communications (ISCC'99), pages 59-67,1999.
[16] D. Estrin, M. Handley, J. Heidemann, S. McCanne, Y. Xu, and H. Yu. Network visualization with the VINT Network Animator Nam. Technical Report 99-703b, USC Computer Science Department, 1999.
[17] S. Y. Wang and H. T. Kung. A simple methodology for constructing extensible and high-fidelity TCP/IP network simulators. In IEEE INFOCOM'99, pages 1134-1143, 1999.
[18] S. Y. Wang, C. H. Huang, and C. C. Lin. NCTUns 1.0 Protocol Module Writer Manual.
[19] S. Y. Wang, A. J. Su, K. C. Liao, H. Y. Chen, and M. C. Yu. NCTUns 1.0 GUI User Manual. http://www.csie.nctu.edu.tw/~shieyuan/publications/UserManual.pdf
[20] [20] N. Jørgensen. Putting it all in the trunk: Incremental development in the FreeBSD open source http://www.ics.uci.edu/~wscacchi/Papers/New/ICSE02-Workshop-Scacchi-01.pdf.
[21] M .K. Dahlhemier. Programming with Qt. O'Reilly, London, 2002.
[22] The New Reno Modification to TCP's Fast Recovery Algorithm, Internet Engineering Task Force RFC 2582.
[23] TCP Extensions for High Performance, Internet Engineering Task Force RFC 1323. pages 9-11
[24] W. Dinkel, D. Niehaus, M. Frisbie, and J. Woltersdorf. KURT Linux Users Manual, 2002. http://www.ittc.ku.edu/kurt/papers/user-manual-DRAFT.pdf.
[25] R. Braden, D. Clark, and S. Shenker, Integrated Services in the Internet Architecture: An Overview, RFC 1633, Internet Engineering Task Force, 1994.
[26] A. Bhushan. A file transfer protocol, Internet Engineering Task Force RFC 114, 1971.
[27] J. Postel. File transfer protocol (ftp), Internet Engineering Task Force RFC 959, 1985.
[28] V. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics," Ph.D. dissertation, U.C. Berkeley, 1997.
[29] Organizing server assets: Methodology for organizing server assets, 2003. From the OPNET Methodology and Case Studies Website, part of the Technical Resources papers.
[30] S. Y. Wang, C. L. Chou, C. H. Huang, C. C. Hwang, Z. M. Yang, C. C. Chiou, and C. C. Lin. The design and implementation of the NCTUns 1.0 network simulator. Technical report, Department of Computer Science and Information Engineering in the National Chiao Tung University in Taiwan, 2002.
[31] The Network Simulator ns-2: Tips and Statistical Data for Running Large Simulations in NS; http://www.isi.edu/nsnam/ns/ns-largesim.html.
[32] W. G. Bragg. Which network design tool is right for you? IEEE IT Pro Magazine, 2(5): 23-31,2000.