REPRODUCCIÓN DE TRAFICO GIGABIT EN UN TESTBED DE EMULACIÓN DE INTERNET A TRAVÉS DE NETFPGA

JOSÉ DAVID ORTIZ CUADROS

UNIVERSIDAD AUTONOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍA DE SISTEMAS
GRUPO DE INVESTIGACIÓN EN TECNOLOGÍAS DE LA INVESTIGACIÓN TELEMATICA
BUCARAMANGA, 2013

REPRODUCCIÓN DE TRAFICO GIGABIT EN UN TESTBED DE EMULACIÓN DE INTERNET A TRAVÉS DE NETFPGA

JOSÉ DAVID ORTIZ CUADROS

Trabajo de grado presentado para optar el título de: ingeniero de sistemas

Director
PhD. Cesar Dario Guerrero Santander

UNIVERSIDAD AUTONOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍA DE SISTEMAS
GRUPO DE INVESTIGACIÓN EN TECNOLOGÍAS DE LA INVESTIGACIÓN TELEMATICA
BUCARAMANGA, 2013

Nota de aceptación:

_____

_____

_____

_____
Presidente del jurado

_____
Jurado

_____
Jurado

Bucaramanga, 11 de Junio del 2013

DEDICATORIA

Mi tesis se la dedico con mucho amor y cariño a Luz Mery Gómez Restrepo, quien es mi fuente de inspiración y mi motivo de seguir adelante, siempre ha estado a mi lado en todo momento creyendo en mí, motivándome y dándome su apoyo incondicional.

A mis padres, quienes siempre estuvieron dándome su apoyo y consejos, siempre alentándome y señalándome en todo momento el camino a seguir.

AGRADECIMIENTOS

Me gustaría agradecer sinceramente a mi director de Tesis, PhD. Cesar Darío Guerrero Santander por su esfuerzo y dedicación. Sus conocimientos, sus orientaciones, su paciencia y su motivación han sido fundamentales para mí. A su manera, ha sido capaz de ganarse mi lealtad y admiración, así como sentirme en deuda con el por todo lo recibido durante el periodo de tiempo que ha durado esta Tesis.

A Manuel Fernando Jaimes Mejía, quien siempre me estuvo colaborando de la manera más humilde y desinteresada.

Así mismo a todas las personas que siempre han estado allí apoyándome.

TABLA DE CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

RESUMEN

La reproducción de tráfico capturado en el internet es utilizada para evaluar la operación de aplicaciones, protocolos y dispositivos de comunicaciones por parte de administradores de red e investigadores alrededor del mundo. Existen diversos programas que generan tráfico sintético pero que, al operar a nivel de capa de aplicación, están sujetos a errores en los tiempos de envío de los paquetes generados por las interrupciones de procesos que continuamente realiza el sistema operativo.

En este proyecto se utiliza una alternativa de reproducción de tráfico que elude los errores de generación de paquetes a través del software de aplicación y delega esa tarea al hardware de red. Para ello se configuró un testbed de red basado en Linux que opera a 1Gbps en donde se inyecta tráfico a través de la plataforma NetFPGA a nivel de hardware. El tráfico inyectado es tomado de una traza almacenada en un archivo PCAP. La traza reproducida por la NetFPGA fue nuevamente capturada por tcpdump con el fin de determinar las diferencias entre lo reproducido y la versión original del tráfico. Este mismo ejercicio se realizó con la herramienta de software más conocida para reproducción de tráfico, denominada *tcpreplay*.

Se evidenció que la NetFPGA por su arquitectura y operación a nivel de hardware logra una variación mínima (del orden de nanosegundos) y constante entre la traza original y la reproducida demostrando la efectividad de la reproducción de tráfico a alta velocidad. Por su parte, tcpreply muestra variaciones mayores y distribuidas uniformemente a lo largo de varios milisegundos. Esto implica, para el caso de tcpreply, reducciones en el ancho de banda diferente a lo que sucede con NetFPGA.

Como resultado del proyecto, se logra demostrar que la generación del tráfico a través de herramientas de software no logra igualar las características de las trazas originales capturadas en el internet. El uso de la plataforma NetFPGA para reproducir tráfico directamente desde el hardware logra mantener las características y tiempo de envío de los paquetes establecidos en las trazas originales.

PALABRAS CLAVES: Generación de tráfico sintético, Testbed, NetFPGA.

INTRODUCCION

Para el desarrollo de este proyecto se utiliza la NetFPGA dado que proporciona un mecanismo de reproducción de tráfico real del internet en un ambiente controlado; se creó una infraestructura de red que permitió evaluar el comportamiento del generador de paquetes que opera en el hardware implementado por la NetFPGA y de la misma manera se evaluó el comportamiento de una aplicación que opera por software llamada Tcpreplay.

La emulación de redes en ambientes controlados está limitada por la capacidad de generación de paquetes por parte de herramientas que operan a nivel de software como *TcpReplay*[1] o *MGEN*[2]. Al generar paquetes desde el software en un computador de propósito general que cuente con una tarjeta de red de 1 Gbps, la máxima tasa de transmisión es del orden de los 350 Mbps[3]. Esta baja utilización del hardware obedece a que el sistema operativo debe atender diferentes tareas adicional a la de envío de paquetes. Es decir, debe realizar planificación de diferentes procesos en donde la generación de paquetes no tiene la más alta prioridad.

Con este proyecto se utilizó la infraestructura NetFPGA[4] para generar tráfico en la infraestructura de red creada (testbed) debido que la NetFPGA opera a una velocidad de 1 Gbps. Esta generación se hace desde el hardware mismo lo cual no interfiere con la operación del sistema operativo y permite generar tráfico a tasas reales.

---

[1] Turner, A., & Bing, M. (2005). Tcpreplay. *URL http://tcpreplay. synfin. net*.
[2] Adamson, B., & Gallavan, S. (1997). MGEN. Retrieved from http://cs.itd.nrl.navy.mil/work/mgen/index.php
[3] Zhou, H., Wang, Y., Wang, X., & Huai, X. (2006). *Difficulties in Estimating Available Bandwidth.* Paper presented at the IEEE International Conference on Communications.
[4] Watson, G., McKeown, N., & Casado, M. (2006). *NetFPGA: A tool for network research and education*.

El objetivo general del proyecto fue el de implementar la reproducción de tráfico sintético desde una traza mediante la aplicación de NetFPGA en un testbed de emulación de internet con el fin de viabilizar evaluaciones de protocolos con tráfico real de internet.

Derivado de este propósito general para la realización del proyecto se plantearon cuatro objetivos específicos:

1. Configurar una red emulada con disponibilidad de operación de 1Gbps que permita el flujo de tráfico de extremo a extremo simulando el comportamiento de internet.

2. Implementar el generador de paquetes de NetFPGA sobre la infraestructura creada con el fin de replicar el tráfico proveniente de trazas de internet.

3. Evaluar la operación del generador de paquetes utilizando una traza capturada en el internet con el fin de determinar la correspondencia con las características de la traza original.

4. Construir una práctica de laboratorio que permita utilizar la plataforma implementada para fines académicos e investigativos en torno a la estimación en internet.

1. ESTADO DEL ARTE

La generación de tráfico sintético en una red emulada se basa en dos aproximaciones. Una de ellas es generar el tráfico a través de software (la más común) y la otra a través del hardware.

Cuando se utiliza una herramienta para generar tráfico sintético a través de software, se puede evidenciar que pequeños cambios pueden generar algunos problemas de exactitud de los experimentos. Botta*et. al.* compara 4 herramientas para generar tráfico y muestra las dificultades que ellas tienen para la generación de tráfico con precisión[5]. Estas dificultades han sido también evidenciadas por estudios realizados por Zhou quien evidencia la limitación que existe en las máquinas comunes que al generar tráfico no se puede realizar a las tasas reales por limitaciones que se generan en el software. Dichos resultados pueden ser observados en la Tabla 1.

Tabla 1. Tasas de transmisión en equipos de propósito general según la tarjeta de red.

Operating System	CPU	Achievablethroughput (Mb/s)		
		NIC=10	NIC=100	NIC=1000
Linux 2,4,1	Intel P4 2.0 Ghz	8.8	77.2	323.0
Linux 2,4,1	AMD Athlon XP 2500+	8.9	78.3	340.2
FreeBSD 4,8	Intel P4 2.0 Ghz	8.8	72.5	299.2
Solaris 2,8	Sparc 400Mhz	8.6	73.4	310.0
Mac OS X 10,2	Power G4 1Ghz	8.6	68.6	256.7
Windows XP SP2	Intel P4 2.0 Ghz	8.8	70.6	280.2

Fuente: Zhou, Wang, Wang, & Huai, 2006, *Difficulties in Estimating Available Bandwidth.*

[5] Botta, A., Dainotti, A., Pescape, x, & A. (2010). Do you trust your software-based traffic generator? *Communications Magazine, IEEE, 48*(9), 158-165. doi: 10.1109/mcom.2010.5560600

La mayor parte de los emuladores de red (testbeds) generan tráfico por software a través de computadores de propósito general que utilizan software libre bajo Linux. Existen varios ejemplos en la literatura de tales testbed que se utilizan para evaluación de herramientas de medición en el internet. En la evaluación de ancho de banda disponible, es común encontrar testbeds que utilizan herramientas como *MGEN* para generar tráfico utilizando diferentes distribuciones de probabilidad como periódica, exponencial o por ráfagas[6]. Otra herramienta utilizada es D-ITG - *Distributed Internet TrafficGenerator*[7] tal como se muestra en el testbed utilizado por Angrisani para realizar una comparación entre técnicas de estimación de ancho de banda disponible[8]. *Iperf*[9] es otra utilidad comúnmente utilizada para generar tráfico TCP tal como lo hace Sommers para la generación de tráfico complementario en su testbed[10].Una herramienta orientada a replicar tráfico tomado de trazas del internet es TcpReplay[11]. Esta herramienta tiene la posibilidad de escalar tráfico para ajustarlo a la capacidad de la red en la cual se va a enviar el tráfico generado. Esta herramienta es utilizada por Shriram en su testbed de evaluación[12]. Existen otras herramientas software para generación de tráfico que han sido utilizadas por diferentes testbed no solamente en la evaluación de ancho de banda disponible sino en la evaluación de otras herramientas de medición de capacidad y *delay* o en la evaluación de protocolos de diferente índole.[13]

---

[6] Guerrero, C. D., & Labrador, M. A. (2010). On the applicability of available bandwidth estimation techniques and tools. *Computer Communications, 33*(1), 11-22.

[7] Botta, A., Dainotti, A., & Pescapé, A. (2012). A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*.

[8] Angrisani, L., D'Antonio, S., Esposito, M., & Vadursi, M. (2006). Techniques for Available Bandwidth Measurement in IP Networks: A Performance Comparison. *Computer Networks, 50*(3), 332--349.

[9] Tirumala, A., Qin, F., Dugan, J., Ferguson, J., & Gibbs, K. (2005). Iperf: The TCP/UDP bandwidth measurement tool: Version.

[10] Sommers, J., Barford, P., & Willinger, W. (2007). Laboratory-based calibration of available bandwidth estimation tools. *Microprocessors and Microsystems, 31*(4), 222-235.

[11] Turner, A., & Bing, M. (2005). Tcpreplay. *URL http://tcpreplay. synfin. net*.

[12] Shriram, A., Murray, M., Hyun, Y., Brownlee, N., Broido, A., Fomenkov, M., & Claffy, K. (2005). *Comparison of Public End to End Bandwidth Estimation Tools on High Speed Links.* Paper presented at the Proceedings of the 6th Passive and Active Measurements Workshop.

[13]Un listado de algunas de estas herramientas se puede ver en http://www.cs.columbia.edu/~hgs/internet/traffic-generator.html (Consultado en Septiembre 10 de 2012)

Una opción de generación de tráfico a través de hardware, es a través de equipos comerciales de muy alto costo. Tal es el caso de SmartBits 6000B[14] que es utilizado por Shriram *et. al*. para generación de tráfico *pseudo-aleatorio* en cuanto al tamaño de los paquetes generados a velocidad Gigabit[15].

Frente a la necesidad de contar con una herramienta capaz de reproducir tráfico del internet tal como lo hace *TcpReplay* pero a través de hardware para poder solucionar los problemas de generación a través de software[16], la Universidad de Stanford ha implementado sobre su plataforma NetFPGA un generador de tráfico que opera a velocidades máximas de 1 Gbps pero que lo realiza a la velocidad real del enlace[17].

[14] Spirent Corp.: Smart bits 6000B. http://www.spirent.com/
[15] Shriram, A., Murray, M., Hyun, Y., Brownlee, N., Broido, A., Fomenkov, M., & Claffy, K. (2005). *Comparison of Public End to End Bandwidth Estimation Tools on High Speed Links.* Paper presented at the Proceedings of the 6th Passive and Active Measurements Workshop.
[16] Zhou, H., Wang, Y., Wang, X., & Huai, X. (2006). *Difficulties in Estimating Available Bandwidth.* Paper presented at the IEEE International Conference on Communications.
[17] Covington, G. A., Gibb, G., Lockwood, J. W., & McKeown, N. (2009, 5-7 April 2009). *A Packet Generator on the NetFPGA Platform.* Paper presented at the Field Programmable Custom Computing Machines, 2009. FCCM '09. 17th IEEE Symposium on.

2. CONCEPTOS BASICOS DE EMULACION A TRAVES DE UN TESTBED

La generación de tráfico sintético a través de hardware en un testbed de emulación de red involucra las siguientes definiciones que soportan la comprensión del proyecto a desarrollar.

2.1 TESTBED[18]

Un banco de pruebas (también comúnmente usado como banco de pruebas en las publicaciones de investigación) es una plataforma para la experimentación de proyectos de desarrollo de gran tamaño. Bancos de pruebas para permitir pruebas rigurosas, transparentes y replicables de las teorías científicas, las herramientas informáticas y nuevas tecnologías.

El término se utiliza en muchas disciplinas para describir un entorno de desarrollo que están protegidos contra los peligros de las pruebas en un entorno real o de producción. Se trata de un método de ensayo de un módulo en particular (función, clase o de biblioteca) de una manera aislada.

18 Bhatt, D., Ghonami, A., & Ramanujan, R. (1987). *An instrumented testbed for real-time distributed systems development.*

2.2 NETFPGA[19]

La plataforma NetFPGA permite a los investigadores e instructores construir prototipos de trabajo de alta velocidad y sistemas de redes de aceleración de hardware. La plataforma se puede utilizar en el aula para enseñar a los estudiantes cómo construir conmutadores Ethernet e Internet Prototcol (IP) utilizando routers físicos en lugar de software. La plataforma puede ser utilizada por investigadores de servicios de prototipos avanzados para redes de próxima generación.

El NetFPGA es una plataforma abierta y disponible para los desarrolladores de todo el mundo. Modelos de referencia incluidos con el sistema que incluye un router IPv4, un interruptor de Ethernet y cuatro puertos NIC. Los investigadores han utilizado la plataforma para construir sistemas de red avanzada de procesamiento de flujo. Una sola tarjeta NetFPGA puede enrutar paquetes en cuatro subredes y múltiples tablas.

El sistema consta de una placa de desarrollo reprogramable. La placa de desarrollo en sí es una tarjeta PCI que puede ser instalado en cualquier computador con una ranura disponible. Alojados en la placa de la tarjeta NetFPGA está un FPGA programable por el usuario (con dos procesadores PowerPC), SRAM, DRAM, y cuatro puertos Ethernet de 1 Gbps.

La programación y administración de la placa de desarrollo se llevan a cabo por el equipo anfitrión a través del bus PCI. Esto permite a los usuarios desarrollar y desplegar proyectos de forma remota, es decir, que el acceso físico a la tarjeta no es necesario.

---

[19] Lockwood, J. W., McKeown, N., Watson, G., Gibb, G., Hartke, P., Naous, J., . . . Luo, J. (2007). *NetFPGA--an open platform for gigabit-rate network switching and routing*.

La tecnología Acent ofrece sistemas pre-ensamblados NetFPGA, este equipo es aprobado por la Universidad de Stanford. Estos equipos pre-construidos y probados por completo son basados en Linux y están disponibles en un cubo de escritorio compactas o de configuración estándar de 1U para montaje en rack de servidor. La tecnología Acent permite que racionalice los procesos de manera más rápida, por ende se obtiene una puesta en marcha más rápida.

2.3 TCPREPLAY [20]

Es una herramienta que permite reproducir archivos guardados por tcpdump y los reproduce a velocidades arbitrarias; Tcpreplay tiene como objetivo evaluar el rendimiento de un NIDS mediante la reproducción real del tráfico. Tcpreplay permite controlar la velocidad a la que se repite el tráfico, es decir, puede reproducir las trazas a diferentes velocidades a las que fueron registradas por medio de herramientas como tcpdump. La diferencia con programas que permiten generar tráfico artificial es que tcpreplay no utiliza la inspección de aplicación / protocolo que lleva a cabo un NIDS, es decir, al utilizar tráfico sintético no se reproducen las anomalías que suelen observarse en el mundo real y como aparecen en las redes de producción (rutas asimétricas, ráfagas de tráfico, la fragmentación, las retransmisiones, etc.), tcpreplay permite la réplica exacta del tráfico real tal como se puede observar en las redes reales.

[20] Turner, A., & Bing, M. (2005). Tcpreplay. *URL http://tcpreplay. synfin. net*.

2.4 TCPDUMP [21]

Es una herramienta en línea de comandos cuya utilidad principal es analizar el tráfico que circula por la red.

Permite al usuario capturar y mostrar a tiempo real los paquetes transmitidos y recibidos en la red a la cual el ordenador está conectado. Está escrito por Van Jacobson, Craig Leres, y Steven McCanne que trabajaban en ese momento en el Grupo de Investigación de Red del Laboratorio Lawrence Berkeley. Más tarde el programa fue ampliado por Andrew Tridgell.

Tcpdump funciona en la mayoría de los sistemas operativos UNIX: Linux, Solaris, BSD, Mac OS X, HP-UX y AIX entre otros. En esos sistemas, tcpdump hace uso de la biblioteca libpcap para capturar los paquetes que circulan por la red.

Existe una adaptación de tcpdump para los sistemas Windows que se llama WinDump y que hace uso de la biblioteca Winpcap.

En UNIX y otros sistemas operativos, es necesario tener los privilegios del root para utilizar tcpdump.

El usuario puede aplicar varios filtros para que sea más depurada la salida. Un filtro es una expresión que va detrás de las opciones y que permite seleccionar los paquetes que se están buscando. En ausencia de ésta, el tcpdump volcará todo el tráfico que vea el adaptador de red seleccionado.

[21] Jacobson, V., Leres, C., & McCanne, S. (1989). Tcpdump.

2.5 WIRESHARK[22]

Antes conocido como Ethereal, es un analizador de protocolos utilizado para realizar análisis y solucionar problemas en redes de comunicaciones, para desarrollo de software y protocolos, y como una herramienta didáctica para educación. Cuenta con todas las características estándar de un analizador de protocolos.

La funcionalidad que provee es similar a la de tcpdump, pero añade una interfaz gráfica y muchas opciones de organización y filtrado de información. Así, permite ver todo el tráfico que pasa a través de una red (usualmente una red Ethernet, aunque es compatible con algunas otras) estableciendo la configuración en modo promiscuo. También incluye una versión basada en texto llamada tshark.

Permite examinar datos de una red viva o de un archivo de captura salvado en disco. Se puede analizar la información capturada, a través de los detalles y sumarios por cada paquete. Wireshark incluye un completo lenguaje para filtrar lo que queremos ver y la habilidad de mostrar el flujo reconstruido de una sesión de TCP.

Wireshark es software libre, y se ejecuta sobre la mayoría de sistemas operativos.

[22] Orebaugh, A., Ramirez, G., & Burke, J. (2007). *Wireshark & Ethereal network protocol analyzer toolkit*: Syngress Media Inc.

2.6 LIBPCAP [23]

Libpcap es una librería open source escrita en C la cual ofrece al programador una interfaz desde la captura en la capa de red y puede ser reconocida por la mayoría de sistemas operativos. A menudo existen problemas en el momento de ver un conjunto de datos complejos y comprender las relaciones de las distintas entidades. En lugar de leer a través del archivo, línea por línea, se hace menos complicado analizar los gráficos que visualizan los datos.

Existe un tipo de gráficos potente para visualizar las relaciones entre las entidades, son los gráficos vinculados o gráficos de red. Otro tipo de visualización se puede lograr con Treemaps, diferentes librerías de código abierto puede sacar este tipo de gráficos, en este caso se hace uso de Libpcap, pero las librerías requieren de entrada en un formato muy específico generalmente un lenguaje de descripción gráfica.

2.7 CASCADE PILOT [24]

Similar a Wireshark, es de ayuda para encontrar la solución de problemas y ayuda a los diagnósticos de redes de área local (LAN).

Es una integración a Wireshark, aumenta dramáticamente la eficiencia en la identificación y el diagnóstico de problemas de la red local.

Permite obtener análisis de manera más rápida a través de un rápido desglose en

[23] Desai, N. (2002). Increasing performance in high speed NIDS. *A look at Snort's Internals*.
[24] Sharmi, S., & Chauhan, M. S. (2012). *Virtual visualization of distributed network traffic using fractals.* Paper presented at the Radar, Communication and Computing (ICRCC), 2012 International Conference on.

captura de paquetes de gran tamaño utilizando la Tecnología Microflow que acelera el análisis de paquetes haciendo que sea hasta 1000 veces más rápido abrir y analizar las capturas de paquetes multi-gigabyte. Utiliza la tecnología de drill-down la cual es una potente utilidad para investigación ya que acelera el tiempo de resolución de problemas contando con una interfaz gráfica de usuario.

3. NETFPGA COMO PLATAFORMA DE REPRODUCCIÓN DE TRÁFICO

Debido a las variaciones que se pueden obtener en herramientas que operan a nivel de software se decidió implementar una herramienta que operara a nivel de hardware; las ventajas que posee esta herramienta y modo de operación se describirán a continuación, dando una idea más clara de su funcionamiento y porqué fue la opción escogida como solución para implementar en el proyecto.

3.1 NETFPGA 1G

Para el desarrollo del proyecto se eligió como sistema hardware la plataforma NetFPGA de 1 Gbps. Es una plataforma Open Source de bajo costo que integra una FPGA y cuatro interfaces Ethernet de 1 Gbps conectadas a ella.

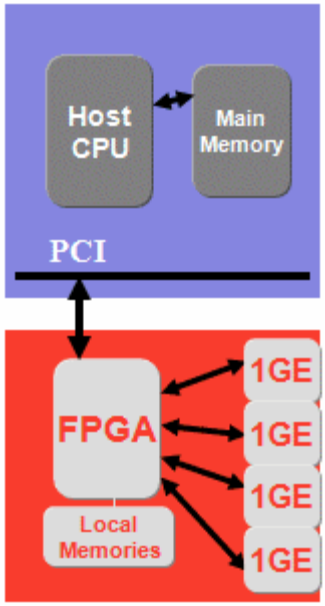
La plataforma NetFPGA está compuesta por la propia tarjeta y una serie de paquetes denominados NFPs (NetFPGA Packages)[25].

3.1.1 Tarjeta NetFPGA. La NetFPGA implementa todo el Datapath (ruta de datos) en el hardware. El sistema por la arquitectura y el hardware que posee es capaz de procesar paquetes al orden de 1 Gbps con una latencia de tan solo unos pocos ciclos de reloj. La tarjeta, mostrada en la

3.1.2
3.1.3
3.1.4 Figura 1, incluye todos los recursos, memoria e interfaces necesaria para un óptimo desempeño.

[25] Zhou, Z., Cong, L., Lu, G., Deng, B., & Li, X. (2010). HATS: high accuracy timestamping system based on NetFPGA *Advances in Computer Science and Information Technology* (pp. 183-195): Springer.

Figura 1 La NetFPGA se conecta a un PC mediante el PCI.

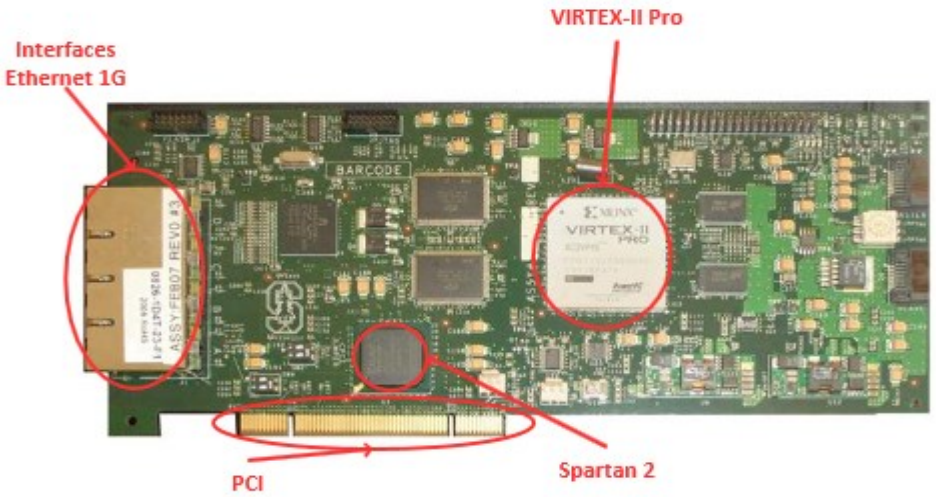


Los componentes integrados en la tarjeta NetFPGA tal como se ven en la Figura 2 son:

- Una Spartan 2, denominada CPCI (control de bus PCI). Se encarga de controlar la comunicación entre la Virtex II y el bus PCI[26].
- Una FPGA modelo Virtex-II Pro50 a la que se denomina CNET (chip de control de la NetFPGA). Mediante la programación de esta FPGA se consigue que la plataforma NetFPGA sea reconfigurable.
- Cuatro puertos Gigabit Ethernet.
- Memoria SRAM de 4,5 MBytes.
- Memoria DDR2 de 64 MBytes.

[26] Lockwood, J. W., McKeown, N., Watson, G., Gibb, G., Hartke, P., Naous, J., . . . Luo, J. (2007). *NetFPGA--an open platform for gigabit-rate network switching and routing*.

Figura 2 Componentes integrados de la NetFPGA.



3.1.5 NetFPGA Packages. Es un conjunto de ficheros con los códigos fuentes escritos en lenguajes hardware y software (VHDL, Verilog, Perl, etc.) necesarios para implementar alguna funcionalidad en la NetFPGA.

Un ejemplo de paquete NFP es el Packet Generator[27]. Éste contiene el conjunto de fuentes que permiten configurar la NetFPGA para que funcione como un generador de paquetes.

[27] Covington, G. A., Gibb, G., Lockwood, J. W., & McKeown, N. (2009, 5-7 April 2009). *A Packet Generator on the NetFPGA Platform.* Paper presented at the Field Programmable Custom Computing Machines, 2009. FCCM '09. 17th IEEE Symposium on.

4. METODOLOGÍA

En este capítulo se dará una breve explicación del software utilizado para implementar en la computadora que se instaló la NetFPGA; se explica la topología de red utilizada en el testbed implementado para las pruebas y se muestran los resultados obtenidos con sus respectivos análisis de los datos de las pruebas realizadas.

4.1 INSTALACIÓN Y PUESTA EN MARCHA DE LA PLATAFOMA NETFPGA

Durante el desarrollo del proyecto se tuvo problemas en cuanto a tiempo y esfuerzo en la instalación y puesta en marcha de la NetFPGA, en especial el correcto funcionamiento del módulo Packet Generator.

Con el fin de facilitar el uso de dicho módulo se ha incluido un apartado donde se describe el proceso de instalación, el cual se verá en el Anexo A.

4.1.1 Fedora y NetFPGA. El equipo necesario para instalar la NetFPGA es un computador que tenga una ranura PCI libre.

En cuanto al sistema operativo se eligió Fedora, el cual es posible descargar un live CD desde el portal web de la NetFPGA cuya distribución es Fedora Core 13 whit NetFPGA. Esta distribución proporciona todas las librerías y drivers desde la versión 2.2 de la NetFPGA.

Se procede a instalar el Java puesto que va a ser utilizado por el generador de paquetes, luego se programará mediante la terminal el CPCI, que como ya antes se había mencionado, es quien se encarga de controlar la comunicación del chip de control de la NetFPGA con el bus PCI.

Como medida de prevención se recomienda no realizar actualizaciones del sistema operativo puesto que los drivers de la tarjeta ya no funcionarán y no se verá instalada la tarjeta, lo cual hará imposible continuar con la configuración de la NetFPGA.

Luego de haber realizado los pasos descritos anteriormente se carga el módulo Packet Generator por medio de la terminal; este módulo es el que permite generar tráfico por medio de trazas capturadas en archivos PCAP. Al terminar los pasos descritos hasta el momento debe estar lista la instalación y configuración de la NetFPGA.

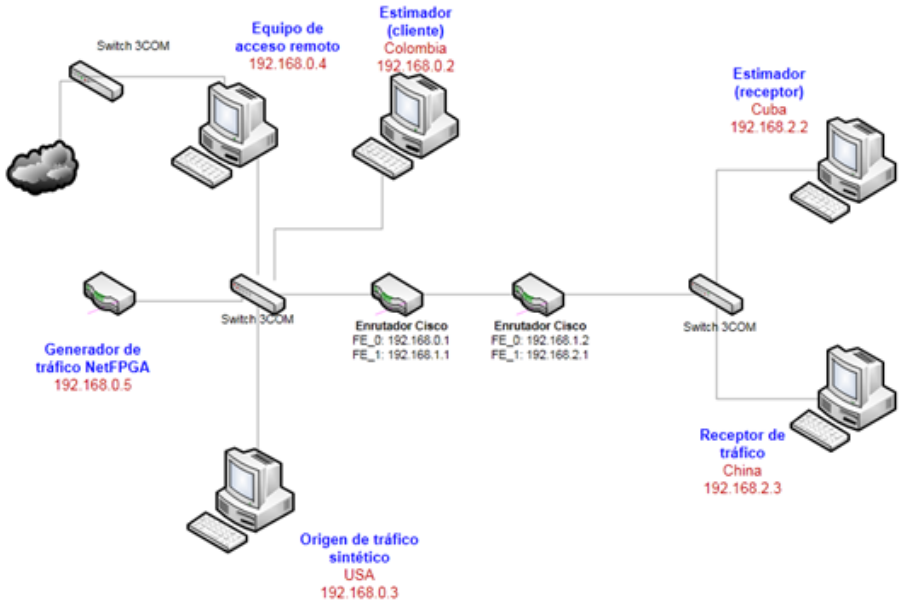
4.2 ARQUITECTURA DEL TESTBED

Se utilizaron 4 ordenadores los cuales trabajaron unos para realizar envío de paquetes y otros realizar monitoreo, todo esto configurado en una red privada. También en un extremo de la red está la NetFPGA, la cual se encargó del envío de paquetes; adicional se utiliza otro ordenador, al cual se le asignó una IP pública para acceder remotamente al testbed utilizando SSH; desde dicha computadora se podía comunicar con las otras computadoras que estaban en el testbed.

En la Figura 3 se pueden observar los 6 ordenadores en los cuales se implementaron las diferentes herramientas, evidenciando que los ordenadores de la izquierda, es decir, las que su red privada es la 192.168.0.0 son los

ordenadores clientes, es decir, los que generan el tráfico y de los cuales se envían los paquetes; adicional el ordenador con el cual se accedía remotamente al testbed y la NetFPGA que se encargaba de generar el tráfico sintético. A la derecha se encuentran dos máquinas con una red privada 192.168.2.0 que son los receptores del generador de tráfico y de los paquetes enviados.

Figura 3 Arquitectura del testbed.

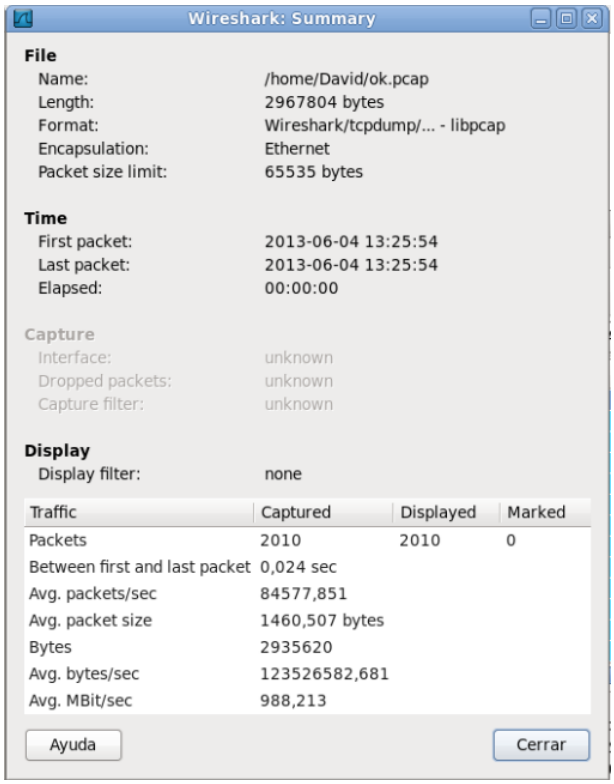


4.3 PROCESAMIENTO DE LA INFORMACIÓN OBTENIDA POR LA NETFPGA

Para la implementación del generador de paquetes se utilizó una traza en formato PCAP, dicha traza fue obtenida en internet y estaba a disposición tanto para descargar como para hacer uso libre de la misma; debido al extenso tamaño de la traza por medio de Wireshark se fragmentó obteniendo una nueva de 2010 paquetes. La nueva traza contó con operabilidad a un ancho de banda que

obedecía al orden de los 988,213 Mbps tal como se puede observar en el resumen dado por Wireshark en la Figura 4.

Figura 4 Ancho de banda traza original.

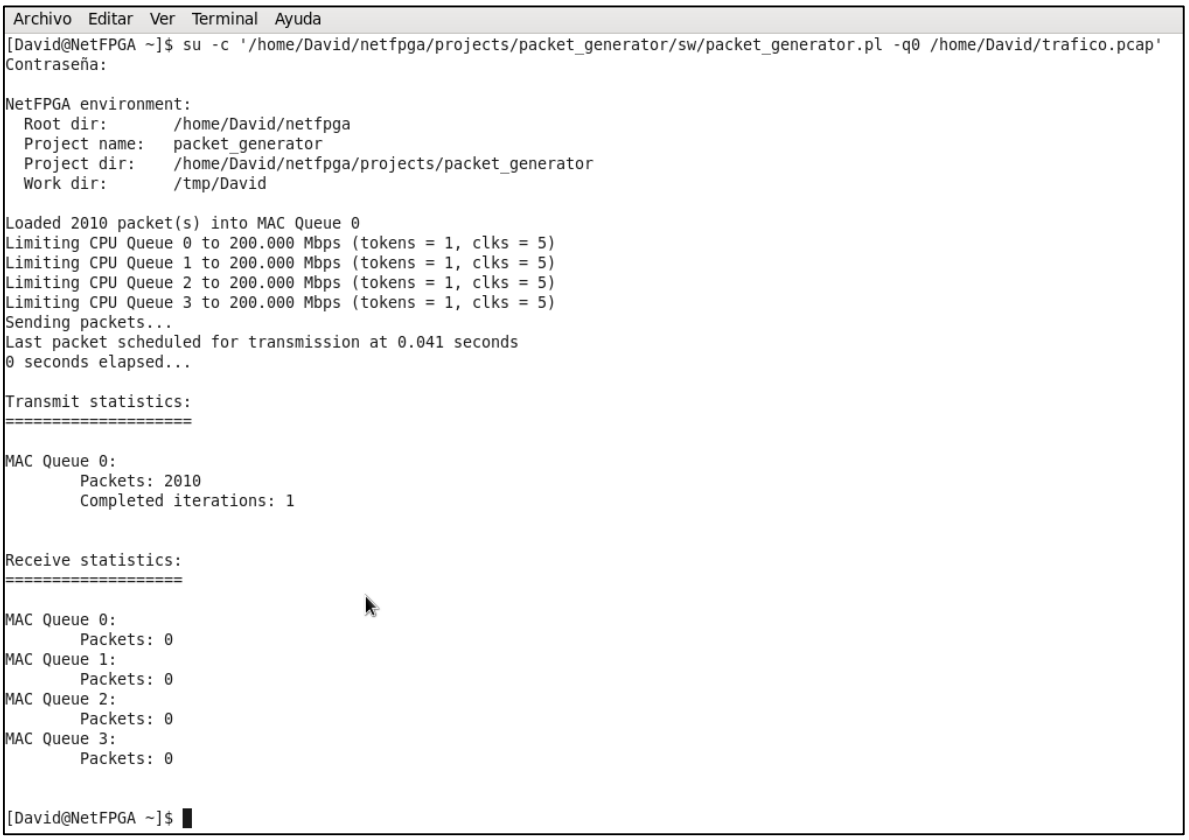


Para realizar la reproducción del tráfico se utilizó el módulo Packet Generator, el cual opera en la NetFPGA. Para la captura de esta traza generada se optó por usar tcpdump, el cual al finalizar la captura generaba como resultado un nuevo archivo PCAP el cual se evaluaría y compararía con la traza original para hacer la respectiva evaluación de la herramienta.

También se realizó una comparación entre funcionamiento de la NetFPGA contra la herramienta tcpreplay, el cual opera a nivel de software.

Para realizar el envío de los paquetes a través de la NetFPGA, se realizó utilizando los comandos mostrados en la Figura 5.

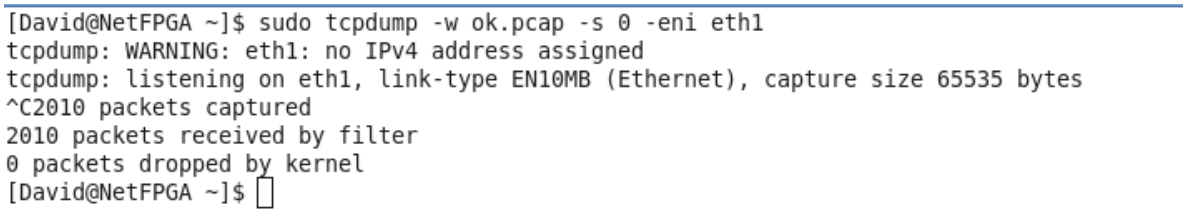
Figura 5 Forma de ejecutar el Packet Generator de la NetFPGA



Como ya antes se había mencionado, la captura del tráfico generado se realizó por medio de tcpdump como se ve en la

Figura 6, donde se utilizó el comando –n para no convertir direcciones a nombres, puesto que de otra manera se obtenían pérdida de paquetes causando pequeñas variaciones en el ancho de banda del archivo PCAP obtenido de la captura realizada por el generador de tráfico.

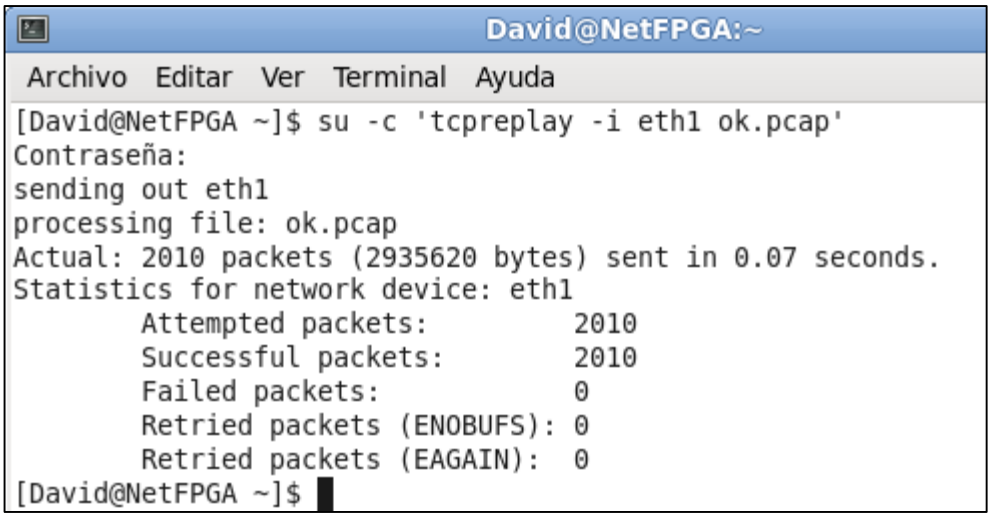
Figura 6 Comando para realizar captura de tráfico con tcpdump



Del mismo modo que se empleó tcpdump para la captura de tráfico realizada por la NetFPGA se implementó en la captura hecha por tcpreplay, puesto que de igual manera se obtenía una pérdida de paquetes.

El tráfico generado por tcpreplay se realizó directamente de la traza original y de igual manera se capturo con tcpdump como ya se mencionó anteriormente, el comando utilizado para generar el tráfico con tcpreplay se puede evidenciar en la Figura 7.

Figura 7 Comando utilizado para reproducir tráfico con tcpreplay



Tanto la traza original, como las trazas obtenidas al implementar los dos generadores de paquetes fueron analizadas con Cascade Pilot, esto con el fin de ver el comportamiento del ancho de banda durante el tiempo de la reproducción del tráfico, esto también se realizó con el fin de observar variaciones con respecto a la traza original para verificar que tan lejano se encontraba del comportamiento de la traza original.

Para el caso de la NetFPGA como los resultados se mantenían estables, donde los tiempos de reproducción y ancho de banda eran similares se optó por utilizar al azar una de las trazas para realizar los análisis; por otro lado para el caso de tcpreplay, se ejecutó en diversas ocasiones y se utilizó la traza capturada que obedecía al menor tiempo de ejecución y mayor ancho de banda obtenido de la nueva traza capturada.

5. RESULTADOS

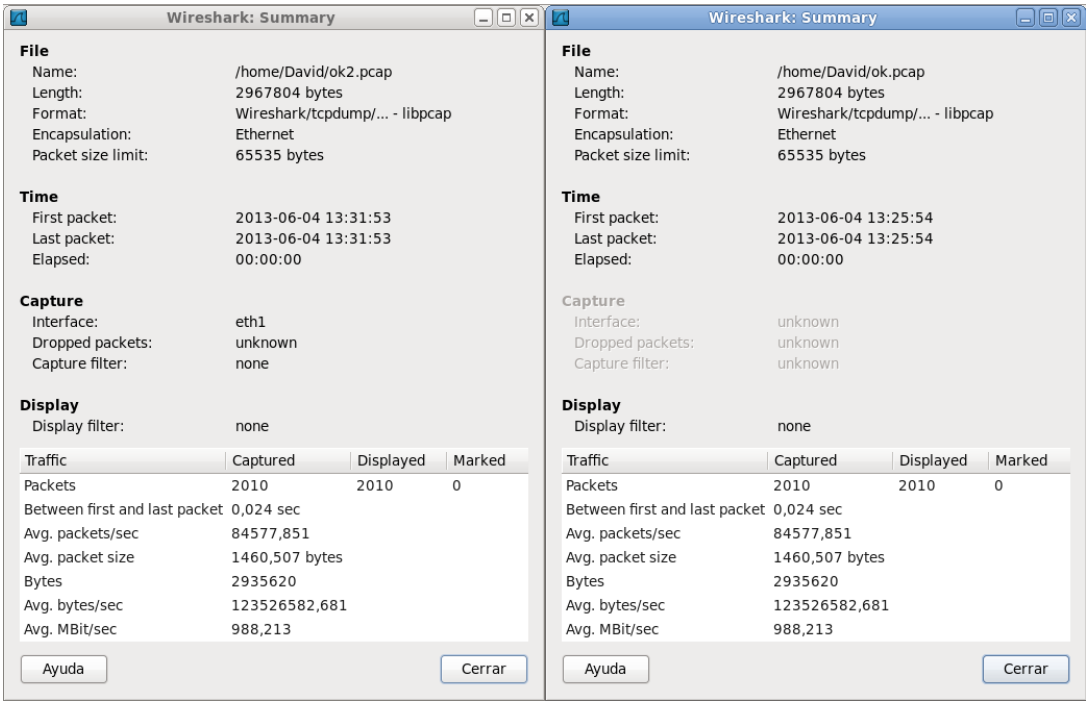
En este capítulo se presentan los resultados de los experimentos llevados a cabo en la plataforma de pruebas presentada en el capítulo anterior.

5.1 REPRODUCCIÓN DE TRÁFICO CON LA NETFPGA

Al realizar la captura obtenida por el Packet Generator se evidencia que el ancho de banda y el tiempo empleado para el envío total de los paquetes sigue siendo el mismo, esto se evidencia en la

donde se muestra el resumen de la traza original y la traza capturada por tcpdump luego de generarla por la NetFPGA; en dicha figura podemos observar a la izquierda los datos de la nueva traza y a la derecha los datos de la traza original, también cabe destacar que no se evidencia ningún tipo de cambio aparente entre la traza original a la traza capturada luego de ser reproducida.

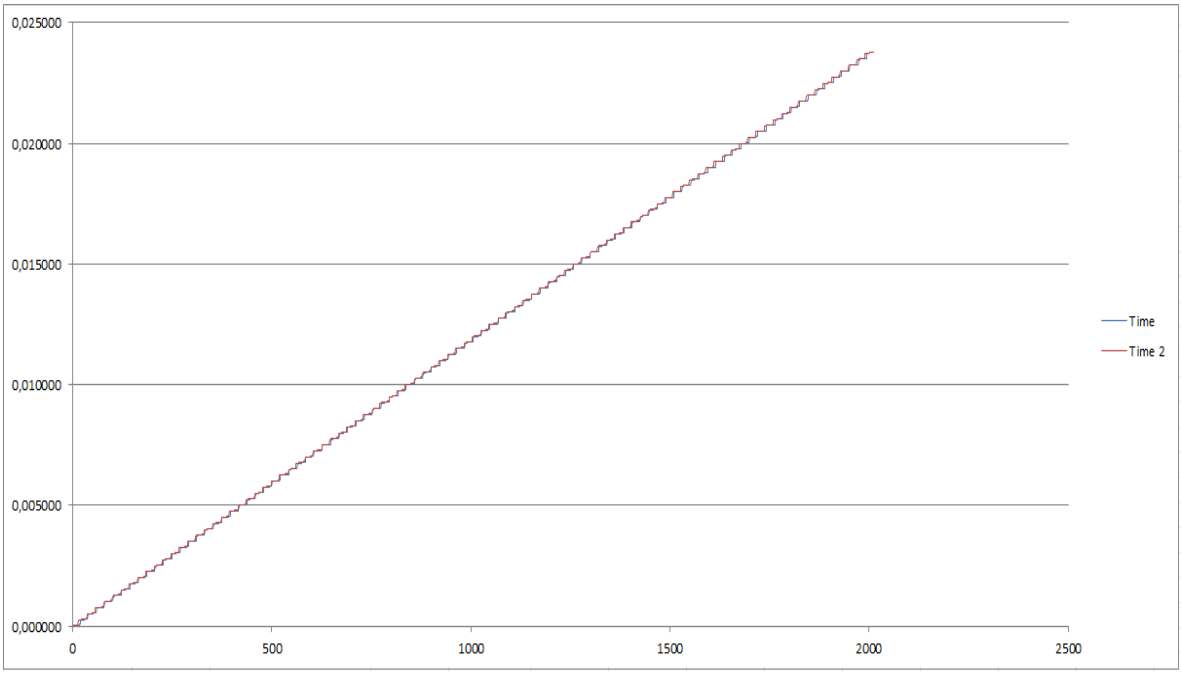
Figura 8 Resumen comparativo entre la traza original y la generada por la NetFPGA



Se extrajo del archivo PCAP la información para realizar una comparación más profunda en cuanto al tiempo de entrega de cada paquete evidenciándose una mínima variación en la llegada del paquete. Los análisis obtenidos de los datos extraídos se mostrarán más adelante. Cabe destacar que dichos cambios se deben a que tcpdump opera a nivel de software y no a nivel de hardware causando ciertos retardos en el timestamp a la llegada del paquete por parte del sistema operativo.

5.1.1 Error obtenido con la NetFPGA. Analizando el tiempo de envío de paquetes se observa que el tiempo de llegada de los mismos en las dos trazas de internet son similares tal como se aprecia en la Figura 9. Se evidencian cambios muy poco notorios, teniendo como margen de error aproximadamente 0,000038%, al hacer una evaluación más exhaustiva de los datos obtenidos se obtuvo una media de 0,00004 y una desviación estándar de 0,00006.

Figura 9 Tiempo vs Paquete utilizando NetFPGA



5.1.2 Comportamiento del ancho de banda NetFPGA. Como se puede observar en la Figura 10 y Figura 11, la variación referente al ancho de la traza original a la capturada luego de utilizar el Packet Generator de la NetFPGA se mantiene un tanto similar a la original. Como ya se había mencionado antes dicha variación se debe al sistema operativo ya que es quien realiza el marcado de los tiempos de llegada del paquete.

Figura 10 Ancho de banda traza original

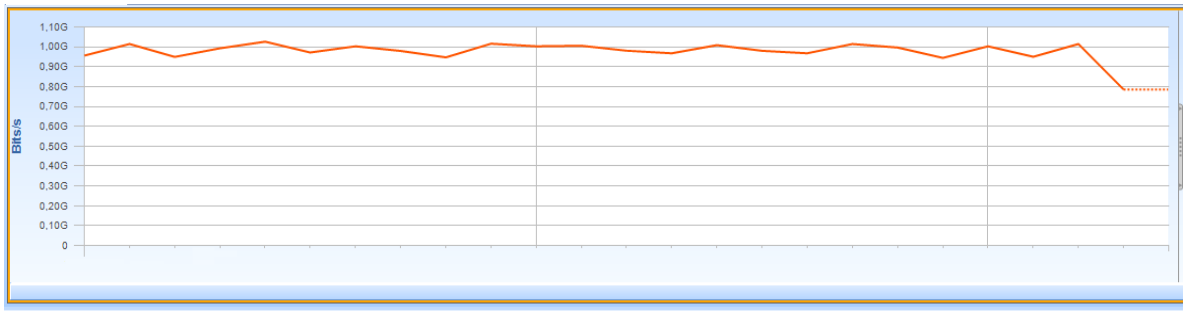
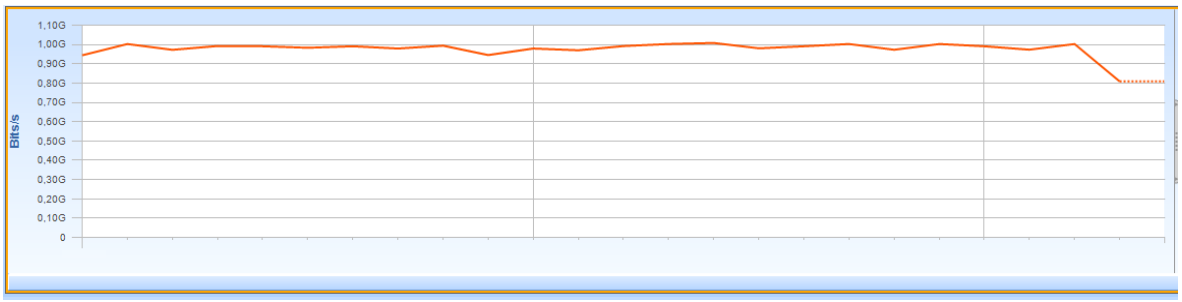


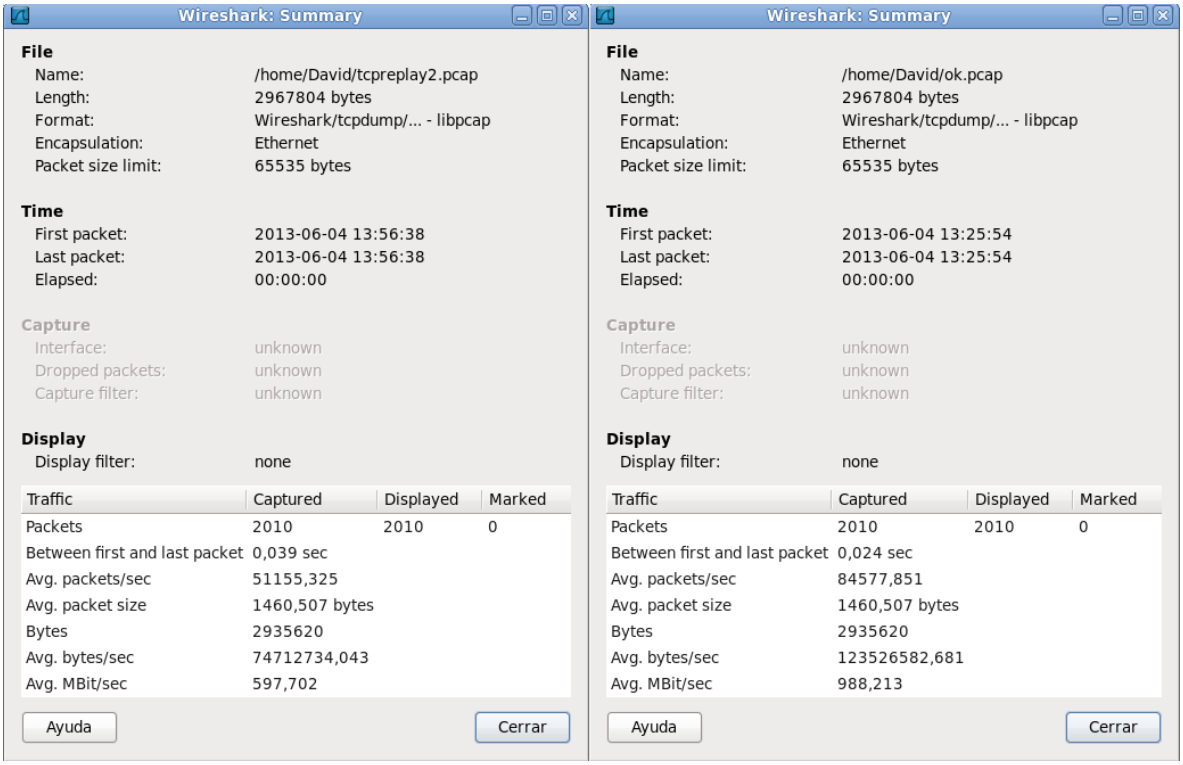
Figura 11 Ancho de banda traza generada por la NetFPGA



5.2 REPRODUCCIÓN DE TRÁFICO CON TCPREPLAY

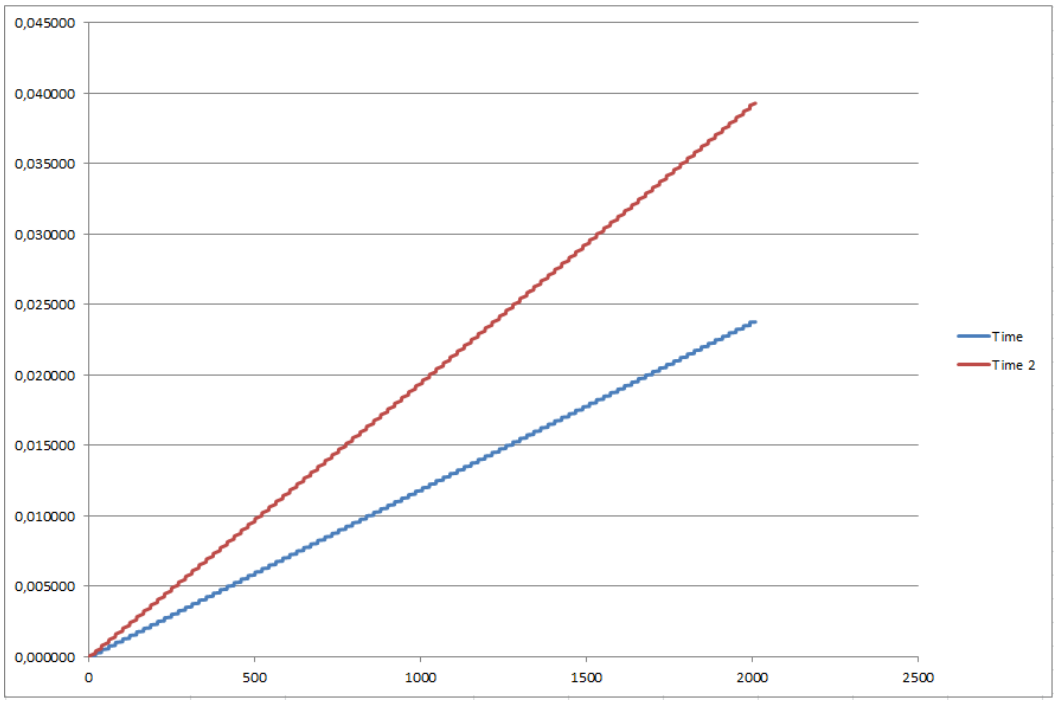
Al realizar la captura obtenida por el tcpreplay se observa que el ancho de banda y el tiempo empleado para el envío total de los paquetes cambia drásticamente. El tiempo aumenta y el ancho de banda se reduce en un 35,9%, estos cambios se pueden ver en la Figura 12 donde se muestra el resumen de la traza original y la traza capturada por tcpdump luego de reproducirla por tcpreplay. En dicha figura podemos observar a la izquierda los datos de la nueva traza y a la derecha los datos de la traza original en la cual se evidencian grandes cambios.

Figura 12 Resumen comparativo entre la traza original y la generada por la NetFPGA



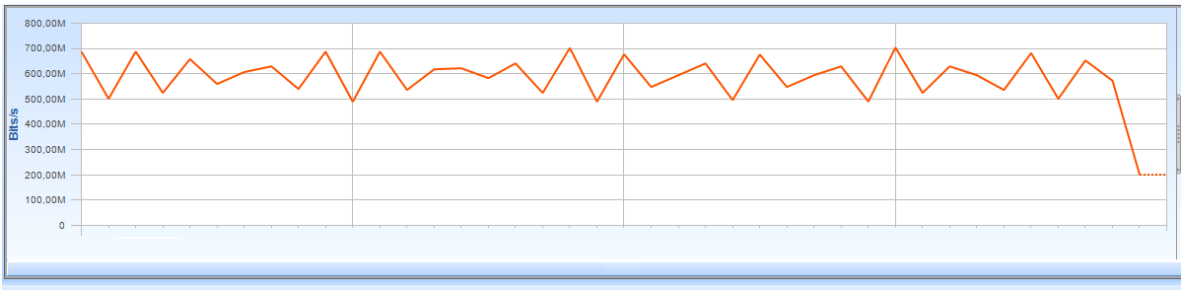
5.2.1 Error obtenido con TCPREPLAY. Analizando los resultados tiempo contra paquetes, podemos observar que el tiempo empleado en la llegada los paquetes en las dos trazas de internet son total mente distintos tal como se aprecia en la Figura 13. En dicha grafica se evidencian cambios muy notorios teniendo como margen de error aproximadamente 0,0022%. De la misma manera que se hizo con el Packet Generator de la NetFPGA se evaluaron los datos obtenidos con tcpreplay arrojando como resultado una media de 0,00769 y una desviación estándar de 0,00448.

Figura 13 Tiempo vs Paquete utilizando tcpreplay



5.2.2 Comportamiento del ancho de banda TCPREPLAY. Como se puede observar en la Figura 10 y la Figura 14, las variaciones referentes al ancho de la traza original a la capturada luego de utilizar tcpreplay varia de manera drástica. Esto se debe a retrasos ocasionados tanto en la ejecución del generador de paquetes como en la recepción de ellos ya que tanto timestamp como como tcpreplay se ejecutan a nivel de software.

Figura 14 Ancho de banda traza generada por tcpreplay



5.3 COMPARACIÓN ENTRE NetFPGA vs TCPREPLAY

En la Tabla 2Tabla 2 Datos de tiempo y ancho de banda de las trazas podemos observar el ancho de banda estimado de la captura de los paquetes y tiempo de reproducción obtenido de las dos herramientas utilizadas con respecto a la traza original.

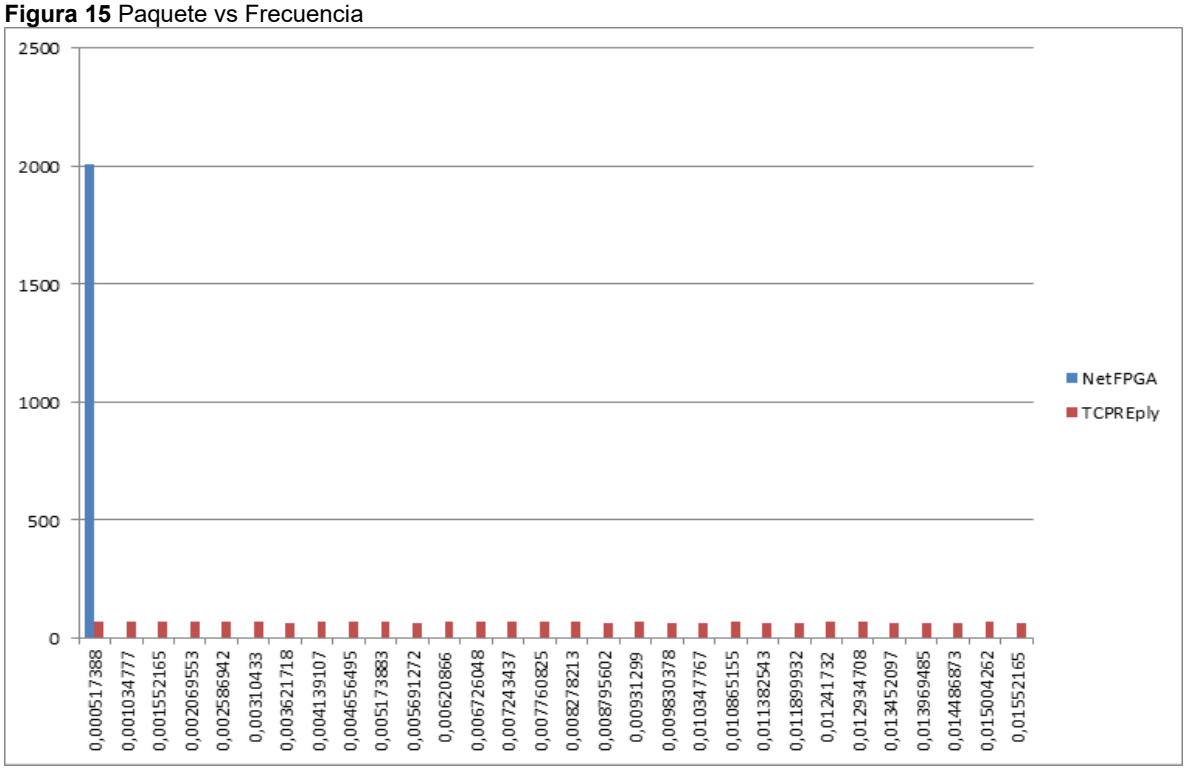
Tabla 2 Datos de tiempo y ancho de banda de las trazas

Parámetros	Traza original	NetFPGA	tcpreplay
Ancho de banda (Mbps)	988,213	988,213	597,702
Tiempo empleado	0,024 seg	0,024 seg	0,039 seg

Con los resultados obtenidos se puede comprobar que efectivamente el generar paquetes por medio de software no es la opción más viable. Los márgenes de error son demasiado grandes y a eso se suma la pérdida de ancho de banda. Para el caso de generación de tráfico por software, en escasas ocasiones se llega a una tasa de 700Mbps. El envío de paquetes en ningún momento llego a ser similar al de la traza original ni tampoco puede mantenerse en un rango estable; es constantemente variable y dichos cambios son altamente visibles e incluso el ancho de banda siempre es variable y poco estable. Lo anterior ocasiona que sea difícil tomar datos confiables para realizar análisis.

Para el caso de la transmisión de paquetes usando la NetFPGA, se observa un comportamiento muy similar a la traza original. Los datos siempre se mantenían estables y la traza generada nunca tomo más tiempo que el de la traza original.

Figura 15 Paquete vs Frecuencia



En la Figura 15 se puede observar que la NetFPGA se mantiene siempre estable, mientras que ocurre totalmente lo contrario con tcpreplay; como se puede apreciar siempre mantiene un margen de error desde que inicia la reproducción del tráfico hasta que finaliza el envío de paquetes, donde el error se mantiene siempre notorio y continuo.

6. CONCLUSIONES

La reproducción de tráfico sintético en una computadora convencional se ve afectada por los retardos que el sistema operativo genera cuando hace cambio entre procesos. Esos retardos se reflejan en diferencias entre los tiempos de envío de paquetes de la traza original y la traza reproducida por el software de generación de tráfico. Tal es el caso de *tcpreplay* que, como se muestra en este trabajo, presenta diferencias significativas en los tiempos de reproducción de tráfico que terminan afectando el ancho de banda de la traza original.

Por otra parte, la reproducción de tráfico sintético a través del hardware de la NetFPGA muestra un comportamiento muy diferente dado que no está sujeto a las afectaciones de cambio de procesos del sistema operativo. Las variaciones entre los tiempos de envío de los paquetes y los tiempos a los cuales se captura la traza reproducida, son del orden de nanosegundos y tienen una variabilidad mínima. Si se considera que estos retardos obedecen en parte a los tiempos de propagación de los paquetes mientras son capturados para análisis, se puede afirmar que la plataforma NetFPGA es una alternativa más precisa y con menor variabilidad que la opción de generación a través de software.

BIBLIOGRAFIA

ADAMSON, B., & GALLAVAN, S. (1997). MGEN. Retrieved from http://cs.itd.nrl.navy.mil/work/mgen/index.php

ANGRISANI, L., D'Antonio, S., ESPOSITO, M., & VADURSI, M. (2006). Techniques for Available Bandwidth Measurement in IP Networks: A Performance Comparison. *Computer Networks, 50*(3), 332--349.

BHATT, D., GHONAMI, A., & RAMANUJAN, R. (1987). *An instrumented testbed for real-time distributed systems development*.

BOTTA, A., DAINOTTI, A., PESCAPE, x, & A. (2010). Do you trust your software-based traffic generator? *Communications Magazine, IEEE, 48*(9), 158-165. doi: 10.1109/mcom.2010.5560600

BOTTA, A., DAINOTTI, A., & PESCAPÉ, A. (2012). A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*.

COVINGTON, G. A., GIBB, G., LOCKWOOD, J. W., & MCKEOWN, N. (2009, 5-7 April 2009). *A Packet Generator on the NetFPGA Platform.* Paper presented at the Field Programmable Custom Computing Machines, 2009. FCCM '09. 17th IEEE Symposium on.

DESAI, N. (2002). Increasing performance in high speed NIDS. *A look at Snort's Internals*.

GUERRERO, C. D., & LABRADOR, M. A. (2010). On the applicability of available bandwidth estimation techniques and tools. *Computer Communications, 33*(1), 11-22.

JACOBSON, V., LERES, C., & MCCANNE, S. (1989). Tcpdump.

LOCKWOOD, J. W., MCKEOWN, N., WATSON, G., GIBB, G., HARTKE, P., NAOUS, J., . . . LUO, J. (2007). *NetFPGA--an open platform for gigabit-rate network switching and routing*.

OREBAUGH, A., RAMIREZ, G., & BURKE, J. (2007). *Wireshark & Ethereal network protocol analyzer toolkit*: Syngress Media Inc.

SHARMI, S., & CHAUHAN, M. S. (2012). *Virtual visualization of distributed network traffic using fractals.* Paper presented at the Radar, Communication and Computing (ICRCC), 2012 International Conference on.

SHRIRAM, A., MURRAY, M., HYUN, Y., BROWNLEE, N., BROIDO, A., FOMENKOV, M., & CLAFFY, K. (2005). *Comparison of Public End to End Bandwidth Estimation Tools on High Speed Links.* Paper presented at the Proceedings of the 6th Passive and Active Measurements Workshop.

SOMMERS, J., BARFORD, P., & WILLINGER, W. (2007). Laboratory-based calibration of available bandwidth estimation tools. *Microprocessors and Microsystems, 31*(4), 222-235.

TIRUMALA, A., QIN, F., DUGAN, J., FERGUSON, J., & GIBBS, K. (2005). Iperf: The TCP/UDP bandwidth measurement tool: Version.

TURNER, A., & BING, M. (2005). Tcpreplay. *URL http://tcpreplay. synfin. net*.

WATSON, G., MCKEOWN, N., & CASADO, M. (2006). *NetFPGA: A tool for network research and education*.

ZHOU, H., WANG, Y., WANG, X., & HUAI, X. (2006). *Difficulties in Estimating Available Bandwidth.* Paper presented at the IEEE International Conference on Communications.

ZHOU, Z., CONG, L., LU, G., DENG, B., & LI, X. (2010). HATS: high accuracy timestamping system based on NetFPGA *Advances in Computer Science and Information Technology* (pp. 183-195): Springer.

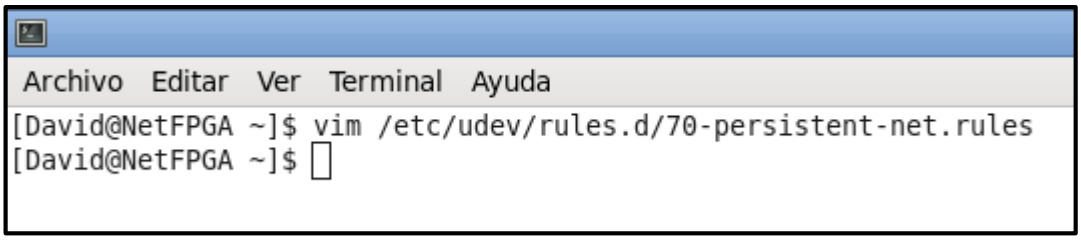
ANEXOS

ANEXO A. RECOMENDACIONES

Para el correcto funcionamiento del Packet Generator, es importante señalar que no se deben realizar actualizaciones al sistema operativo, puesto que al hacerlo se genera un conflicto con los drivers de la tarjeta NetFPGA haciendo que no pueda ser reconocida por el sistema operativo.

Se debe modificar el archivo correspondiente para asignarle Eth0 a la tarjeta de red incorporada a la tarjeta madre de la computadora, para así obtener un funcionamiento óptimo; dicho archivo se edita con los comandos que podemos observar en la Figura 16, lo anterior debido que al instalar el sistema operativo Fedora muchas veces no lista las tarjetas de red instaladas en el orden correcto.

Figura 16 Comando para configurar dispositivos de red



ANEXO B. PRÁCTICA DE LABORATORIO PARA LA INSTALACIÓN DEL PACKET GENERATOR EN LA NETFPGA

El objetivo de ésta práctica es instalar el generador de paquetes de la NetFPGA con el fin de reproducir trazas de internet en un ambiente controlado de red. Para ello, se requiere un sistema Linux, exactamente Fedora Core 13. También se requiere la tarjeta NetFPGA de 1GB instalada en el equipo a utilizar.

1. Abrir la terminal e iniciar sesión como root con el comando **"su"**.
2. Verificar si las tarjeta está instalada con el comando "**ifconfig**"", debe mostrar desde la nf2c0 hasta la nf2c3 tal como se ve en la Figura 17.

Figura 17 Interfaces de red

```
[root@NetFPGA David]# ifconfig
eth0      Link encap:Ethernet  HWaddr 20:CF:30:56:29:68
          inet addr:172.16.208.79  Bcast:172.16.208.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5566 errors:0 dropped:0 overruns:0 frame:0
          TX packets:83 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:760288 (742.4 KiB)  TX bytes:8877 (8.6 KiB)
          Interrupt:27 Base address:0x8000

eth1      Link encap:Ethernet  HWaddr 00:1B:21:76:BD:B1
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:26088 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10050 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:38101824 (36.3 MiB)  TX bytes:14678100 (13.9 MiB)
          Memory:dfe80000-dfea0000

eth2      Link encap:Ethernet  HWaddr 00:1B:21:76:BD:B0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Memory:dfee0000-dff00000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:500 (500.0 b)  TX bytes:500 (500.0 b)

nf2c0     Link encap:Ethernet  HWaddr 00:4E:46:32:43:00
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4228 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:1 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6176424 (5.8 MiB)  TX bytes:0 (0.0 b)
          Interrupt:20

nf2c1     Link encap:Ethernet  HWaddr 00:4E:46:32:43:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:20

nf2c3     Link encap:Ethernet  HWaddr 00:4E:46:32:43:03
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:20

[root@NetFPGA David]#
```

3. Instalar Java con el comando “**yum install java**”.

- Instalar el código del Kernel con **yum install kernel-devel:**

```
....
Dependencies Resolved
================================================================= Package
Arch Version Repository Size
=================================================================
Installing:
Kernel-devel          i686          2.6.18-92.1.22.el5       updates       4.8 M

Transaction Summary
=================================================================
.....

Total download size: 4.8 M
Is this ok [y/N]: y ←
Downloading Packages:
(1/1): kernel-devel-2.6.1 100% |=========================| 4.8 MB      00:00
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
   Installing: kernel-devel                ########################### [1/1]


Installed: kernel-devel.i686 0:2.6.18-92.1.22.el5
Complete!
```

4. Terminados los pasos anteriores se procede a instalar el repositorio que contiene el paquete base de NetFPGA.

- Usar el comando: **yum install netfpga-base**
- Se obtendrá algo como esto:

```
......
Dependencies Resolved
================================================================= Package
Arch          Version          Repository          Size
```

```
================================================================
Installing for dependencies:
.....
netfpga-base              i386            1.2.4-CentOS5       netfpga            3.1 M
netfpga-kernel            i386            1.2.4-CentOS5       netfpga            35 k
netfpga-utils             i386            1.2.4-CentOS5       netfpga            243 k
netfpga_lib               i386            1.1-2               netfpga            3.7 M


.....


Transaction Summary
================================================================
Install        20 Package(s)
Update         0 Package(s)
Remove         0 Package(s)
Total download size: 25 M
Is this ok [y/N]: y←
Downloading Packages:
(1/20): libgomp-4.1.2-42. 100% |=========================| 82 kB       00:00
(2/20): perl-Net-RawIP-0. 100% |=========================| 119 kB      00:01
.....


Dependency Installed: compat-libstdc++-296.i386 0:2.96-138 compat-libstdc++-33.i386
0:3.2.3-61   gcc.i386 0:4.1.2-42.el5 gcc-c++.i386 0:4.1.2-42.el5 glibc-devel.i386   0:2.5-
24.el5_2.2 .............perl-Net-Pcap.i386 0:0.16-1.el5.rf perl-Net-RawIP.i386 0:0.23-1.el5.rf  perl-
XML-Simple.noarch 0:2.18-1.el5.rf
Complete!
```

5. Ahora se procederá a compilar y cargar el driver.

- Se usará el siguiente comando “**make**” anteponiendo la ruta donde se encuentra instalada la carpeta de NetFPGA , en el caso de este proyecto se realizó de la siguiente manera:
  /usr/bin/netfpga/make

6. Se verificaran las interfaces de la NetFPGA con el comando “lsmod | grep nf2”.

- Obteniendo la salida: nf2 22156 0

7. Ahora se reprogramará la CPCI, se hará corriendo el siguiente comando: “**/usr/local/sbin/cpci_reprogram.pl –all**”

- Se obtendrá algo

```
Loading the CPCI Reprogrammer on NetFPGA 0
Loading the CPCI on NetFPGA 0
CPCI on NetFPGA 0 has been successfully reprogrammed
```

- Nota: cada vez que se reinicia la computadora debes ejecutar nuevamente el comando para reprogramar la CPCI.

8. Ahora se configurará el Packet Generator con el siguiente comando:

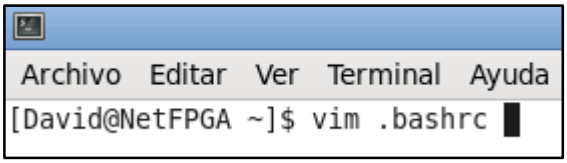
- nf_download /usr/local/netfpga/bitfiles/packet_generator.bit

```
Download completed - 2377668 bytes. (expected -1).
DONE went high - chip has been successfully programmed.
CPCI Information

----------

Version: 4 (rev 1)
Device (Virtex) Information

--------------------------
Project directory: packet_generator
Project name: Packet Generator
Project description: Packet Generator that replays PCAP files
```

9. Hay que tener en cuenta que al ejecutar el Packet Generator puede salir un error diciendo que no se puede cargar una librería de perl, el cual se debe que el nombre del proyecto cargado es diferente al que está en el archivo bashrc. El archivo bashrc se edita puesto que debe contener el nombre correcto del

proyecto, en este caso Packet Generator. Para editar el archivo se hace con el comando que se ve en la Figura 18.

Figura 18 Comando para editar el archivo bashrc



ANEXO C MANUAL DE USUARIO DE NETFPGA

Los puertos de la tarjeta están identificados como nf2c0 hasta nf2c3, donde se refiere al puerto 0, 1, 2, 3 respectivamente.

A continuación se explicará el uso de algunos comandos que pueden ser de utilidad para implementar y a su vez se explicarán algunas de las funciones de la NetFPGA.

Es importante saber que la NetFPGA permite no solo generar una traza de Internet tal cual como fue capturada, esta también permite capturar la traza si se conecta a ella misma, es decir, si conectamos el cable de red al puerto nf2c0 y al puerto nf2c1 de la tarjeta, podemos generar el tráfico y capturarlo en la misma NetFPGA, esto se hace con el siguiente comando:

- packet_generator.pl -q0 http.pcap -c1 nf2c1.pcap

Donde –q0 indica que el Puerto de salida es nf2c0 y –c1 el puerto que recibirá el trafico será nf2c1, dicho trafico será capturado y guardado como un archivo PCAP bajo el nombre nf2c1.pcap para este caso.

Cabe destacar que una de las funciones de gran utilidad, es que se puede utilizar una traza que posea un ancho de banda pequeño, pero, al generar trafico pueda hacerlo a una velocidad mayor que la de la traza original, haciendo que el ancho de banda de dicho archivo pueda llegar al orden de 1Gbps; esto se puede realizar haciendo uso del siguiente comando:

- packet_generator.pl –d0 0 -q0 http.pcap

Si se quiere capturar dicha traza, puede hacerse hacienda uso de Wireshark o como ya se mencionó anteriormente, la NetFPGA puede capturar el tráfico generando un nuevo archivo PCAP, pero para esto debe hacerse un ligero cambio en el comando, el cual debería quedar de la siguiente manera:

- packet_generator.pl –d0 0 -q0 http.pcap –d0 0 -c1 nf2c1.pcap

Al hacer uso del comando para generar trafico al orden de 1Gbps no puede usarse una traza que contenga más de 2044 paquetes, puesto que es lo máximo permitido por la NetFPGA por el tamaño de la memoria que esta posee, se puede hacer uso del comando –i, el cual sirve para especificar el número de iteraciones que se desea realice la NetFPGA, tal como se describe a continuación:

- packet_generator.pl –d0 0 -q0 http.pcap –i 10

Donde en este caso 10 será la cantidad de iteraciones deseadas.

Si se quiere limitar el ancho de banda al generar paquetes, se puede hacer mediante el uso del siguiente comando:

- packet_generator.pl -q0 http.pcap -r0 1000

Donde 1000 representaría 1000Kbps, ósea, 1Mbps y –r0 indica el puerto de salida, en este caso nf2c0, los valores aceptados esta dado en Kbps.

Nota: Siempre debe especificarse la ruta donde se encuentra el packet_generator.pl al igual que la ruta donde se encuentra el archivo PCAP.