ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA

"ANÁLISIS DE DESEMPEÑO DEL EMULADOR DE CANAL NETEM SOBRE EL PROTOCOLO TCP"

AUTOR: LUIS EDISON PAREDES CAICEDO

SANGOLQUÍ – ECUADOR

2013

ESCUELA POLITÉCNICA DEL EJÉRCITO

INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

LUIS EDISON PAREDES CAICEDO

DECLARO QUE:

El proyecto de grado denominado "ANÁLISIS DE DESEMPEÑO DEL EMULADOR DE CANAL NETEM SOBRE EL PROTOCOLO TCP", ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie, de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, a 5 de septiembre de 2013

_____

Luis Edison Paredes Caicedo

ESCUELA POLITÉCNICA DEL EJÉRCITO

INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, LUIS EDISON PAREDES CAICEDO

Autorizo a la Escuela Politécnica del Ejército la publicación, en la biblioteca virtual de la Institución del trabajo “ANÁLISIS DE DESEMPEÑO DEL EMULADOR DE CANAL NETEM SOBRE EL PROTOCOLO TCP”, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, a 5 de septiembre de 2013

_____

Luis Edison Paredes Caicedo

ESCUELA POLITÉCNICA DEL EJÉRCITO

INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES

CERTIFICACIÓN

DR. GONZALO OLMEDO
ING. CARLOS ROMERO

C E R T I F I C A N

Que el trabajo titulado “ANÁLISIS DE DESEMPEÑO DEL EMULADOR DE CANAL NETEM SOBRE EL PROTOCOLO TCP”, realizado por Luis Edison Paredes Caicedo, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la ESPE, en el Reglamento de Estudiantes de la Escuela Politécnica del Ejército.

Debido a que se trata de un trabajo de investigación recomiendan su publicación.

Sangolquí, a 5 de septiembre de 2013

_____	_____
Dr. Gonzalo Olmedo	Ing. Carlos Romero
DIRECTOR	**CODIRECTOR**

RESUMEN

En el presente proyecto se realizó un análisis de desempeño del emulador de canal Netem bajo el protocolo TCP, mediante la implementación de escenarios de pruebas con la ayuda de ordenadores, en las cuales se realizaron varias inyecciones de trafico mediante D-ITG con cada uno de los parámetros individualmente que proporciona el emulador, así también una inyección de trafico representando un posible escenario real con todos los parámetros en conjunto, bajándose a esta experiencia se procede a realizar guías prácticas de laboratorio las mismas que ayudaran al uso correcto del emulador en futuras investigaciones; luego de este proceso se puede decir que NetEm es una herramienta muy útil para representar los factores que afectan a las transmisiones de redes inalámbricas porque proporciona las opciones estadísticas necesarias para emular la repuesta de una red real.

DEDICATORIA

Dedico este proyecto de tesis a Dios por su infinito amor en momentos de penumbra, a mis padres y abuelos quienes han velado por mi bienestar y educación siendo pilar fundamental en mi vida, a mi novia, compañera inseparable de cada jornada.

AGRADECIMIENTO

A mi director, codirector, profesores, familiares, amigos y demás personas que de forma directa o indirecta contribuyeron para la realización de este proyecto mi más sincero agradecimiento.

PRÓLOGO

En las diversas universidades y departamentos que trabajan en la investigación de redes y protocolos necesitan herramientas y entornos para verificar su funcionamiento en condiciones reales. En ocasiones, las pruebas son muy difíciles de realizar debido al gran número de máquinas y medios que se requieren.

Para facilitar las pruebas se han desarrollado emuladores, NetEm es un emulador de redes confiable para su uso y bajo costo de implementación,

Tomando como punto de partida experiencias de proyectos anteriores donde se utilizó el emulador de canal NetEm y con el firme propósito de ampliar esa información para contribuir a la sociedad de la investigación, se inicia la adquisición de conocimientos para una posterior realización de prácticas y experimentos que nos permitirán conocer el desempeño del emulador bajo el protocolo TCP, dicha experiencia se encuentra expresada en el presente proyecto dividido de la siguiente manera:

En el capítulo 1 se hace una introducción al problema, antecedentes, justificación de la importancia de usar un emulador y no un simulador, alcance de la tesis y así también los objetivos a cumplirse a lo largo del proyecto.

En el capítulo 2, se menciona conceptos básicos del protocolo TCP, definiciones de funcionamiento, parámetros, limitaciones y demás consideraciones de NetEm, al igual que los principales aspectos y arquitectura del inyector de trafico D-ITG.

En el capítulo 3, se detalla los elementos requeridos y pasos para la instalación de NetEm, los comandos de *tc* a ser utilizados para su configuración y también

como habilitar el puente de red entre las dos interfaces que posee el ordenador donde se ejecutara el emulador.

En el capítulo 4, con tablas de resumen se muestra la configuración que debe tener el inyector de trafico D-ITG, así también los valores y porcentajes que se incluyen en los comandos de configuración de cada uno de los escenarios de pruebas de los parámetros de NetEm, describiendo el procedimiento de implementación.

En el capítulo 5, se expone el análisis de resultados del desempeño de NetEm en los diferentes escenarios de pruebas implementados mediante las gráficas de Throughput, Retraso, Jitter y el Archivo de resultados de la inyección de tráfico.

En el capítulo 6, se elaboró guías de prácticas de laboratorio basadas en NetEm

En el capítulo 7, se encuentran las conclusiones y recomendaciones, resultado de la experiencia a lo largo del desarrollo del proyecto.

ÍNDICE DE CONTENIDO

CAPÍTULO III

CAPÍTULO IV

CAPÍTULO V

CAPÍTULO VI

CAPÍTULO VII

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

GLOSARIO

AVERAGE: Media Aritmética

BITRATE Tasa de transferencia

DARPA: Defense Advanced Research Projects

DITG: Distributed Internet Traffic Generator

i.d.d.: Independent and identically distributed random variables

IDT: Inter Departure Time

IP: Internet Protocol

JITTER: Variabilidad temporal durante el envío de señales digitales.

LAN: Local Area Network

NETEM: Network Emulator

NTP: Network Time Protocol

OWD: One Way Delay

PS: Packet Size

QDISC: Queuing Discipline

RTT: Round Trip Time

SO: Sistema Operativo

TC: Traffic Control

TCP: Transmission Control Protocol

TESTBED: Banco de pruebas es una plataforma para experimentación de proyectos de gran desarrollo

TSP: Team Software Process

UDP: User Datagram Protocol

WAN: Wide Area Network

WWW: World Wide Web

CAPÍTULO I

1 PRESENTACIÓN DEL PROYECTO

1.1 ANTECEDENTES

Las redes de computadoras surgen por la necesidad de compartir recursos de alto costo entre varias personas, en sus inicios, los recursos eran de un costo elevado. Las primeras redes que se crearon eran del tipo centralizado, es así que el desarrollo de la tecnología permitió tener redes que comunicaran computadoras en sitios distantes, lo que obligó a crear protocolos de comunicación.

En 1973, la Agencia de Investigaciones avanzadas de la Defensa de los Estados Unidos (DARPA) inició un programa para investigar las técnicas y las tecnologías para la interconexión de redes de diversos tipos, este proyecto se denominó *Internetting Project* y el sistema de redes que emergió de estas investigaciones fue conocido como la Internet. El sistema de protocolos desarrollados en el transcurso de esta investigación, se conocería como la suite del protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol*).

Con el uso del protocolo TCP, las aplicaciones pueden comunicarse en forma segura, gracias al sistema de acuse de recibo del protocolo TCP, independientemente de las capas inferiores, esto significa que los routers que funcionan en la capa de Internet, sólo tienen que enviar los datos en forma de datagramas, sin preocuparse del monitoreo de datos porque esta función la cumple la capa de transporte o más específicamente el protocolo TCP.

Durante una comunicación usando el protocolo TCP, las dos máquinas deben establecer una conexión. La máquina emisora, que solicita la conexión, se denomina cliente y la máquina receptora la definimos como servidor. De acuerdo se estructura un entorno Cliente-Servidor. Los medios que participan en el entorno se comunican de forma lineal, es decir, que la comunicación se realiza en ambas direcciones. Cuando se establece la comunicación con fines de investigación empleando protocolos como TCP no siempre se tiene disponible los medios físicos, en este caso se procede a utilizar un emulador, definiendo a este como un software que permite ejecutar programas de computadora en una plataforma sea esta arquitectura hardware o sistema operativo, diferente del cual fueron escritos originalmente.

1.2 JUSTIFICACIÓN E IMPORTANCIA

Un emulador trata de modelar de forma precisa un dispositivo o sistema de manera que este funcione como si estuviese siendo usado en el aparato original, a diferencia de un simulador que sólo trata de reproducir el comportamiento del programa [1].

Por lo expuesto, se hace necesario usar un emulador confiable que se acerque a la realidad, el cual pueda modificarse de acuerdo a las necesidades de los diferentes escenarios que se desea emular, por esta razón el candidato más idóneo luego de un proceso de desempeño es el emulador de canal NetEm que es open source.

Este proyecto define su importancia al documentar de forma fidedigna y adecuada del uso del emulador NetEm a través de guías prácticas, las mismas que permitirán realizar investigaciones futuras en base a los resultados obtenidos.

1.3 ALCANCE DEL PROYECTO

El proyecto consiste en realizar un análisis de desempeño del emulador NetEm, sobre una configuración de escenarios en redes inalámbricas mediante la inyección de tráfico sobre el protocolo TCP, con el fin de obtener mediciones y

comportamientos de cada uno los parámetros de demora, perdida, duplicación, corrupción y reordenamiento, como así también un análisis de la emulación de un posible escenario real. Los resultados obtenidos permitirán elaborar guías de laboratorio para el uso adecuado del emulador.

1.4 OBJETIVOS

1.4.1 Objetivo General

Analizar el desempeño del emulador de canal NetEm sobre el protocolo TCP en redes inalámbricas.

1.4.2 Objetivos Específicos

- Estudiar el desempeño de las diferentes funcionalidades del emulador NetEm.
- Definir el marco teórico y conceptual sobre TCP y NetEm.
- Instalar y configurar escenarios para realizar las emulaciones.
- Aplicar parámetros en emulaciones de posibles escenarios reales.
- Analizar datos generados en los diferentes escenarios y presentación de resultados (elaborar guías de prácticas de laboratorio)
- Documentar conclusiones y recomendaciones para trabajos futuros

CAPÍTULO II

2 MARCO CONCEPTUAL Y TEÓRICO - FUNDAMENTOS DE TCP Y NETEM

2.1 INTRODUCCIÓN

Aunque en sus comienzos Internet se entendía como una infraestructura de investigación, hoy en día se ha convertido en un entorno operativo y es más difícil de usar como plataforma de pruebas. Muchas universidades y departamentos de I+D que trabajan en redes y protocolos necesitan herramientas y entornos para verificar su funcionamiento en condiciones reales. En ocasiones, las pruebas son muy difíciles de realizar debido al gran número de máquinas y entornos que se requieren, existen varias soluciones posibles para este problema, una pasa por el uso de herramientas de simulación que permiten realizar medidas controladas y repetibles a bajo coste. Tienen dos inconvenientes: el primero es la carga computacional, especialmente cuando el escenario de red incluye un gran número de máquinas; el segundo se debe al hecho de que los simuladores simplifican el sistema a medir, alejándolo de la realidad, usando implementaciones específicas de los protocolos, y no permitiendo el uso de cualquier aplicación o sistema operativo (SO).

Una opción es utilizar hardware real, que incluye la pila de protocolos completa del SO, esta puede ser una buena solución en muchos casos, pero también se han desarrollado muchos *testbed* y emuladores para facilitar las pruebas y reducir el coste. Algunos son híbridos, combinando las ventajas de la simulación, la emulación y las pruebas con equipos reales.

2.2 CONCEPTOS BÁSICOS DE TCP

2.2.1 Importancia y Características TCP

TCP/IP es un protocolo de comunicaciones que se basa en software utilizado en redes. Aunque el nombre TCP/IP implica que el ámbito total del producto es la combinación de dos protocolos - protocolo de control de transmisión - (*transmission control protocol*) y protocolo Internet (*Internet protocol*), el término TCP/IP no es una entidad única que combina dos protocolos, sino un conjunto de programas de software más grande que proporciona servicios de red, como registro de entrada remoto, transferencia de archivos remota y correo electrónico. TCP/IP ofrece un método de transferir información de una máquina a otra. Un protocolo de comunicaciones debe manejar los errores en la transmisión, administrar el encaminamiento y entrega de los datos, así como controlar la transmisión real mediante el uso de señales de estado predeterminadas. TCP/IP se ocupa de todo lo anterior [7].

TCP/IP es una de las redes más comunes utilizadas para conectar computadoras con sistema UNIX, las utilidades de red TCP/IP forman parte de la versión 4, muchas facilidades de red como un sistema UUCP, el sistema de correo, RFS y NFS, pueden utilizar una red TCP/CP para comunicarse con otras máquinas.

Para que la red TCP/IP esté activa y funcionando es necesario:

- Obtener una dirección Internet.
- Instalar las utilidades Internet en el sistema
- Configurar la red para TCP/IP
- Configurar los guiones de arranque TCP/IP
- Identificar otras máquinas ante el sistema
- Comenzar a ejecutar TCP/IP.

Ya que dentro de un sistema TCP/IP los datos transmitidos se dividen en pequeños paquetes, éstos resaltan una serie de características.

La tarea de IP es llevar los datos a granel (los paquetes) de un sitio a otro, las computadoras que encuentran las vías para llevar los datos de una red a otra (denominadas enrutadores) utilizan IP para trasladar los datos, en resumen IP mueve los paquetes de datos a granel, mientras TCP se encarga del flujo y asegura que los datos estén correctos.

Las líneas de comunicación se pueden compartir entre varios usuarios, cualquier tipo de paquete puede transmitirse al mismo tiempo, y se ordenará y combinará cuando llegue a su destino. Compare esto con la manera en que se transmite una conversación telefónica., una vez que establece una conexión, se reservan algunos circuitos para usted, que no puede emplear en otra llamada, aun si deja esperando a su interlocutor por veinte minutos.

Los datos no tienen que enviarse directamente entre dos computadoras. Cada paquete pasa de computadora en computadora hasta llegar a su destino. Éste, claro está, es el secreto de cómo se pueden enviar datos y mensajes entre dos computadoras aunque no estén conectadas directamente entre sí. Lo que realmente sorprende es que sólo se necesitan algunos segundos para enviar un archivo de buen tamaño de una máquina a otra, aunque estén separadas por miles de kilómetros y pese a que los datos tienen que pasar por múltiples computadoras. Una de las razones de la rapidez es que, cuando algo anda mal, sólo es necesario volver a transmitir un paquete, no todo el mensaje.

Los paquetes no necesitan seguir la misma trayectoria. La red puede llevar cada paquete de un lugar a otro y usar la conexión más idónea que esté disponible en ese instante. No todos los paquetes de los mensajes tienen que viajar, necesariamente, por la misma ruta, ni necesariamente tienen que llegar todos al mismo tiempo.

La flexibilidad del sistema lo hace muy confiable. Si un enlace se pierde, el sistema usa otro. Cuando usted envía un mensaje, el TCP divide los datos en

paquetes, ordena éstos en secuencia, agrega cierta información para control de errores y después los lanza hacia fuera, y los distribuye. En el otro extremo, el TCP recibe los paquetes, verifica si hay errores y los vuelve a combinar para convertirlos en los datos originales. De haber error en algún punto, el programa TCP destino envía un mensaje solicitando que se vuelvan a enviar determinados paquetes.

2.2.2 Transporte de información a través de TCP

Una red TCP/IP transfiere datos mediante el ensamblaje de bloques de datos en paquetes, cada paquete comienza con una cabecera que contiene información de control; tal como la dirección del destino, seguido de los datos. Cuando se envía un archivo por la red TCP/IP, su contenido se envía utilizando una serie de paquetes diferentes. IP un protocolo de la capa de red, permite a las aplicaciones ejecutarse transparentemente sobre redes interconectadas. Cuando se utiliza IP, no es necesario conocer que hardware se utiliza, por tanto ésta corre en una red de área local [13].

TCP un protocolo de la capa de transporte, asegura que los datos sean entregados, que lo que se recibe, sea lo que se pretendía enviar y que los paquetes que sean recibidos en el orden en que fueron enviados. TCP terminará una conexión si ocurre un error que haga la transmisión fiable imposible.

2.3 CONCEPTOS BÁSICOS DE TESTBED

2.3.1 Funcionamiento de NETEM

NetEm provee la funcionalidad de emulación de red para realizar pruebas de protocolos, emulando las propiedades de una red WAN. La actual versión puede emular demora, pérdidas, duplicación, corrupción y reordenamiento de paquetes [12].

NetEm se controla a través de la línea de comando, con el comando tc, que es parte del paquete de herramientas de iproute2.

NetEm es una mejora reciente a las funcionalidades de control de tráfico de Linux. Se construye utilizando las funcionalidades existentes en Linux de QoS y Servicios Diferenciados (DiffServ).

La motivación que hay detrás de NetEm es la de proveer una manera de reproducir grandes redes en un ambiente de laboratorio. Primeramente se utilizó para evaluar nuevas mejoras a TCP en Linux.

Nació del código de TBF y solo soportaba una demora constante, pero fue evolucionando con el tiempo y el esfuerzo de desarrolladores y usuarios [12].

2.3.2 Diseño del Programa y Especificación de Parámetros

2.3.2.1 Diseño

NetEm consiste de dos partes, un pequeño modulo del kernel de Linux para disciplina de colas y una herramienta, *tc*, de línea de comando para configurarlo. El modulo está integrado desde la versión 2.6.8 y 2.4.28 del kernel, y la herramienta es parte del paquete *iproute2* [2].

La comunicación entre la herramienta y el kernel es hecha a través de la interface de sockets Netlink. Los pedidos son codificados en el formato estándar de mensajes que luego el kernel decodifica.

La Figura 2.1 muestra la arquitectura básica de NetEm. Las disciplinas de cola existen entre la salida del protocolo y el dispositivo de red [5].

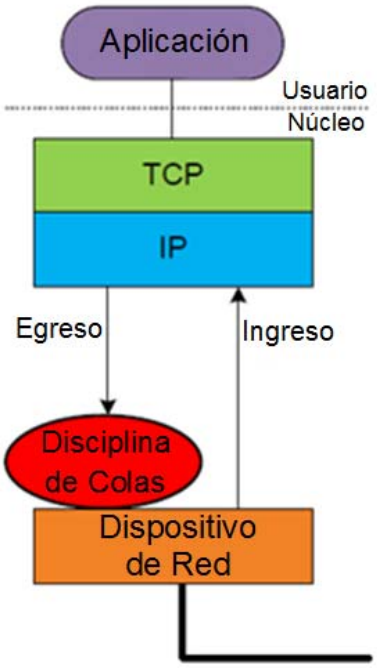

Figura 2.1. NetEm Básico

Una disciplina de cola es un objeto simple con dos interfaces claves. Una interface encola paquetes a ser enviados y la otra entrega paquetes al dispositivo de red. Las políticas a utilizar para la entrega de paquetes al dispositivo de red están en la configuración de la disciplina de cola. Se pueden configurar políticas complejas anidando distintas disciplinas de cola.

Internamente, NetEm es una disciplina de cola con clases con colas de paquetes. Una es una cola privada de espera tipo TFIFO y la otra es una disciplina de cola anidada, generalmente TFIFO. La interface de encolado toma paquetes, les asigna un *Timestamp* (tiempo de envío) y los pone en la cola de espera. Un reloj interno mueve los paquetes de la cola de espera a la disciplina de cola anidada. La interface de desencolado toma paquetes de la disciplina de cola anidada.

- **Parámetros**

  El usuario especifica los parámetros de emulación a través de la herramienta *tc*. Sin parámetros, NetEm se comporta como una cola TFIFO sin demora, perdida, duplicación o reordenamiento de paquetes.

Demora

Las redes de computadoras no muestran siempre la misma demora, ésta varía en base a otros flujos de tráfico que viajan por el mismo camino. Por esta razón, la demora se define por el valor promedio (μ), desviación estándar (σ) y correlación (ρ). Por defecto NetEm utiliza una distribución uniforme (μ±σ), pero es posible utilizar otra tabla de distribución generadas a partir de un modelo matemático o datos experimentales. Estas se pueden generar con herramientas que provee iproute2. Las distribuciones estándar a utilizar son uniforme, normal, pareto y paretonormal [2].

Pérdidas

La pérdida de paquete está implementada como la eliminación de un cierto porcentaje de paquetes al azar antes de ser encolados. Se define como un porcentaje de paquetes a perder y su correlación. Internamente, el código del núcleo de Linux no debe usar punto flotante, por lo que el porcentaje de pérdida se transfiere como una escala de 32 bits. Aunque en discusión, NetEm no tiene la capacidad de modificar los paquetes porque en el nivel de transporte paquetes corruptos sería equivalente a la pérdida de paquetes [2].

Duplicación

En redes con hardware confiable no duplican paquetes, pero con routers redundantes y hardware real la duplicación puede ocurrir. En NetEm la duplicación se realiza clonando un cierto porcentaje de paquetes al azar antes de ser encolados. Se define como un porcentaje de paquetes a duplicar y su correlación (la misma de pérdidas) [2].

Corrupción

Ruido aleatorio en los enlaces causa que algunos paquetes lleguen corruptos a su destino. NetEm puede generar un simple bit de error en el

paquete para corromperlo. Se define como un porcentaje de paquetes a corromper y su correlación [2].

Reordenamiento

Reordenamiento de paquete ocurre cuando paquetes atraviesan caminos con diferentes demoras. Algunos routers de alta velocidad usan múltiples buses y procesadores que internamente crean caminos alternativos. Los paquetes son reordenados cuando diferentes procesadores y buses tienen diferente demora [2].

NetEm tiene dos formas posible de reordenamiento. Una forma es definiendo un "*gap*" que actúa como un agente de seguridad aleatorio, este elige 1 de N paquetes y lo demora. Otra forma es definiendo un porcentaje de paquetes a demorar y su correlación. Un porcentaje de los paquetes serán enviados inmediatamente, los restantes sufrirán la demora configurada.

Limitación del ancho de banda

El *Token Bucket Filter* (TBF) se utiliza para limitar la cantidad de datos que pueden salir de la interfaz de red por segundo para la emulación de las limitaciones de ancho de banda WAN.

2.3.3 Control de Tasa

Por defecto NetEm usa una disciplina de cola TFIFO para la cola saliente, sin embargo se pueden utilizar otras, como se explicó en Disciplina de colas *qdisc* [5].

2.3.4 Limitaciones

NetEm trabaja en un flujo simple. Sin embargo, redes reales son complejas y la emulación no puede emular todos los casos posibles que se pueden

presentar. La granularidad del reloj de Linux afecta la naturaleza de tiempo real de NetEm, la elección del pseudo-generador de números aleatorios impacta en los resultados de la emulación, y los dispositivos de red no siempre están disponibles para ráfaga de paquetes. Los parámetros de NetEm no son suficientes para describir complejas redes con múltiples niveles de complejidad.

2.3.4.1 Relojes

Al no ser Linux un sistema de tiempo real, esto causa que NetEm sufra ciertas restricciones. Los relojes del kernel están limitados por el *tick* del sistema a 1000Hz o 1ms en Linux 2.6, o 100Hz o 10ms en Linux 2.4. Por ende, NetEm no puede emular demoras menores a 1ms. Pero este problema no lo sufre solamente NetEm, las disciplinas de control de tasa también lo sufren cuando corren sobre enlaces de alta velocidad. No es posible limitar una red de 10 Gbit a 100 Mbit con precisión sin utilizar relojes de alta resolución.[1]

2.3.4.2 Números al Azar

En el kernel de Linux existen varias fuentes de números pseudo-azarosos, pero ninguno es idóneo para una buena emulación. La segura función criptográfica de generación de números al azar *get_random_bytes ( )* no puede ser utilizada rígidamente ya que confía en los eventos del sistema para proveer entropía. La intensión de la misma es la de proveer claves criptográficas y se puede bloquear cuando hay poca entropía hasta que el pool de entropía se llene de nuevos eventos del sistema (búsquedas en disco, llegada de paquetes, movimientos del mouse, etc.)

El código de red tiene una simple función de números al azar de 32bit, *net_random* ( ), implementada como un generador lineal congruente (LCG). LCG no útiles para simulaciones porque producen patrones en la salida que pueden influenciar los resultados finales. Una mejor alternativa (que finalmente se

[1] NIST Net soluciona esta limitación programando uno de los alternativos relojes disponibles en la arquitectura de la PC para proveer relojes de alta resolución. Esto tiene impacto en la performance ya que la carga de interrupción es alta, y no portable a otras arquitecturas.

implementó) fue la utilización de generador *Tausworthe* combinado y máximamente equi-distribuido encontrado en la librería GNU *Scientific Library* 1.5.

2.3.4.3 Dispositivos de Red

Los dispositivos de red en Linux tienen un driver de transmisión en forma de anillo que tiene una referencia a los datos disponibles para que el hardware lo procese. Este anillo tiene un tamaño definido, limitado por la disponibilidad de los bloques de control de transmisión. Cuando la carga es alta, NetEm causara ráfaga de paquetes hacia el dispositivo cada 1ms. El anillo de transmisión debe ser lo suficientemente grande para manejar esta ráfaga de paquete o el dispositivo debe manejar el flujo correctamente [2].

2.4 INYECTOR DE TRÁFICO D-ITG

D-ITG es una plataforma capaz de producir el tráfico que se adhiere a la precisión de patrones definidos por el tiempo de salida entre paquetes (IDT) y el tamaño del paquete (PS) en procesos estocásticos. Tales procesos se implementan como un i.i.d. secuencia aleatoria variables. Una rica variedad de distribuciones de probabilidad está disponible: constante, uniforme, exponencial, Pareto, Cauchy, normal, De Poisson y gamma [14].

Además, D-ITG incorpora algunos modelos propuesto para emular fuentes de varios protocolos: TCP, UDP, ICMP, DNS, Telnet y VoIP, esto significa que el usuario simplemente elige uno de los protocolos soportados y la distribución de ambos IDT y PS será automáticamente establecido.

D-ITG puede realizar mediciones en sentido unidireccional (OWD) y bidireccional (RTT), evaluación de pérdida de paquetes, jitter y la medición del caudal. Para cada generación es posible establecer una semilla para las variables aleatorias que intervienen. Esta opción da la posibilidad de repetir muchas veces exactamente el mismo patrón de tráfico mediante el uso de la misma semilla. Además, D-ITG permite el ajuste de TOS (DS) y los campos de paquetes TTL.

D-ITG permite almacenar información tanto sobre el lado del receptor y el lado del remitente. Por lo tanto, es posible recuperar información sobre el patrón de tráfico generado. Además, D-ITG permite al emisor y al receptor delegar la operación de registro a un servidor de registro remoto. Esta opción es útil cuando el remitente o el receptor tienen una capacidad limitada de almacenamiento (por ejemplo, PDAs, PC de bolsillo, etc.) También, puede ser utilizado para analizar la información de registro "en la marcha", por ejemplo, en caso de que el remitente es instruido por una entidad controlador para adaptar la velocidad de transmisión en base a la congestión del canal y la capacidad del receptor.

Otra característica innovadora es que el remitente puede ser controlado a distancia mediante el uso de ITGApi. Esto significa que el remitente D-ITG se puede iniciar en modo demonio y esperar a que los comandos que indicamos que genere flujos de tráfico. D-ITG está actualmente disponible en Linux, Windows y la plataforma Linux Familiar.

2.4.1 Arquitectura de Software D-ITG

Plataforma D-ITG exhibe una arquitectura multi-componente distribuido. La Figura 2.2 muestra un resumen gráfico de la relación entre las principales bloques de la plataforma D-ITG. La comunicación entre el emisor y el receptor se realiza mediante el uso de un canal de señalización separado y gobernado por un protocolo para la configuración del experimento (Tráfico Especificación de protocolo - TSP).

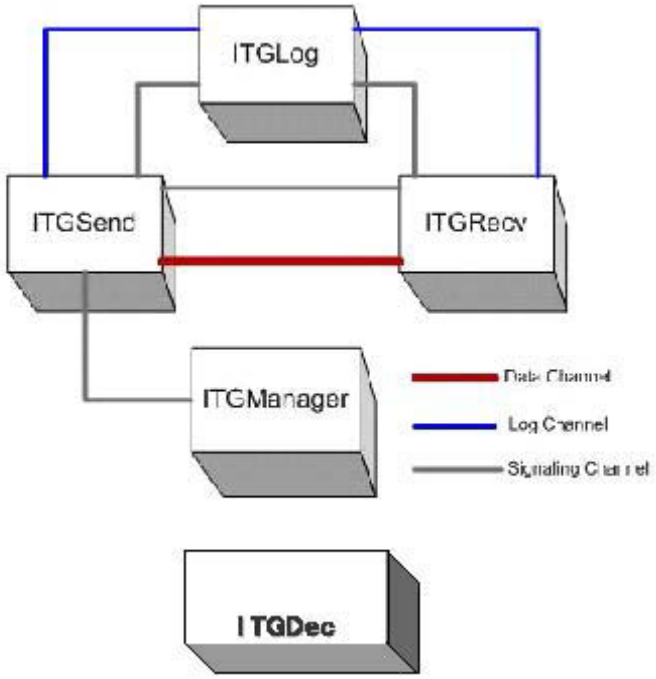

Figura 2.2. Arquitectura de Software D-ITG

ITGSend ITGSend es el componente emisor de la plataforma de generación de tráfico D-ITG. ITGSend puede funcionar en tres modos diferentes: (i) el modo de flujo único: ITGSend genera un solo flujo; un solo hilo es responsable de la generación de la corriente y la gestión del canal de señalización a través del protocolo TSP; (ii) el modo de múltiples flujos: ITGSend genera un conjunto de flujos. Funciona como aplicación multiproceso: uno de los temas implementa el protocolo TSP y conduce el proceso de generación, mientras que los otros generan los flujos de tráfico, (iii) el modo de demonio: ITGSend es controlado remotamente por ITGManager utilizando el ITGApi. Para recoger estadísticas sobre el ITGSend proceso de generación puede iniciar (de forma local o remota mediante la ITGLog servidor de registro) información detallada sobre los flujos generados [14].

ITGRecv ITGRecv siempre funciona como demonio concurrente, la escucha de nuevas conexiones de TSP. Cuando una petición de conexión llega TSP, ITGRecv genera un nuevo hilo que es responsable de la gestión de la comunicación con el remitente. Cada flujo único es recibido por un hilo separado. Como ITGSend, ITGRecv puede almacenar información de forma local o remota mediante el servidor de registro ITGLog [14].

ITGLog ITGLog es un "servidor de registro", que se ejecuta en un host diferente a ITGSend y ITGRecv, que recibe y almacena la información de registro de múltiples emisores y receptores. Las actividades de tala se manejan usando un protocolo de señalización. La información de registro puede ser enviado usando ya sea un canal fiable (TCP) o un canal no fiable (UDP) [14].

CAPÍTULO III

3 INSTALACIÓN Y CONFIGURACIÓN DE NETEM

3.1 INTRODUCCIÓN

Netem es una gran herramienta para la simulación de una conexión WAN, con toda la latencia, esperada, jitter, pérdida de paquetes, duplicado, y las limitaciones de ancho de banda. En este capítulo se revisaran instrucciones que guiaran a través de la creación de una máquina que se encuentra entre el servidor y el cliente que emula el comportamiento de una conexión WAN.

3.1.1 Cómo encontrar el sistema adecuado

Para encontrar un sistema adecuado debemos tener en cuenta algunos aspectos ente hardware y software, en el primer caso no se necesita de muchas prestaciones en el pc aunque mientras mejores características tenga la misma mejor rendimiento vamos a obtener, es necesario especificar que el pc debe contar con 2 tarjetas de red, en tanto al software existen muchas distribuciones de Linux, es necesario escoger una que se acople a las necesidades del presente proyecto. Una de las versiones de Linux mayormente utilizadas es Ubuntu ya que ha sido simplificada a fin de atraer a los usuarios que buscan un sistema operativo operacional inmediatamente, debido a esto y que NETEM ya está habilitado en el núcleo con una versión actual de iproute2

3.1.2 Comandos de utilización de NETEM

La herramienta ***tc*** se usa para configurar el control de tráfico en el núcleo Linux. Control de tráfico consiste en lo siguiente [3]:

- **Moldeado**

  Cuando el tráfico está conformado, su tasa de transmisión está bajo control. La conformación puede ser más que reducir el ancho de banda disponible - que también se utiliza para suavizar las ráfagas de tráfico de comportamiento de la red mejor.

- **Programación**

  Mediante la programación de la transmisión de paquetes, es posible mejorar la interactividad para el tráfico que necesita sin dejar de garantizar el ancho de banda para las transferencias a granel. Reordenamiento también se llama priorizar y ocurre sólo en salida.

- **Policiales**

  Cuando la formación se ocupa de la transmisión de tráfico, la policía se refiere al tráfico que llega. Policía así ocurre en el ingreso.

- **Caída**

  Tráfico superior a un ancho de banda ajustado también pueden ser dejados sin demora, tanto en entrada y en salida.

Qdisc

qdisc es la abreviatura “queuing discipline” (“disciplina de colas") y es elemental para el control de la comprensión del tráfico. Cada vez que el kernel tiene que enviar un paquete a una interfaz, se pone en cola a la qdisc configurada para la interfaz. Inmediatamente después, el kernel intenta obtener los paquetes a medida que sea posible a partir de la qdisc, para dárselos al controlador del adaptador de red.

3.2 INSTALACIÓN DEL SISTEMA

3.2.1 Elementos requeridos para la Aplicación

Se requiere tres computadoras, siendo una de ellas, la que realice la emulación de las diferentes condiciones de red necesarias, haciendo de puente entre las otras dos máquinas. De las restantes dos, una máquina actúa como cliente y otra como servidor. Estas dos últimas no tienen ninguna conexión directa entre sí, sino que se conectan a través de la maquina puente, que posee dos tarjetas de red, una para conectarse con la maquina cliente y otra para conectarse con la maquina servidor.

La máquina puente permite la conexión entre el cliente y el servidor de manera transparente, es decir, que ninguna conoce que existe una tercer maquina entre ellas.

3.2.2 Pasos para la instalación de NETEM

La instalación de NETEM se realiza en un simple paso, se necesita instalar el paquete iproute; el mismo que se lo realiza como root con el siguiente comando:

```
apt-get install iproute
```

Hay que tener en cuenta que para instalar un paquete de esta manera en Linux debemos tener una conexión activa a internet.

3.3 PUENTE DE LAS DOS INTERFACES DE RED

Se debe asegurar de desactivar la configuración de la red para las interfaces:

```
ifconfig eth0 0.0.0.0
ifconfig eth1 0.0.0.0
```

A continuación, se crea el puente y lo levantamos:

```
brctl addbr br0
brctl setfd br0 0
brctl addif br0 eth0
brctl addif br0 eth1
ifconfig br0 up
```

Hay que tener en cuenta de desactivar el retardo de reenvío ('*setfd*'). Esto hace al puente pasar el tráfico a través de él inmediatamente en vez del tiempo de retardo configurado.

3.4 CONFIGURACIÓN DE NETEM

Para la configuración de NETEM se debe aplicar comandos teniendo en cuenta el siguiente formato:

```
tc qdisc add dev <interfaz> root netem <parámetro> <valor>
```

Donde <parámetro> y <valor> están relacionados de la siguiente forma. <parámetro> puede ser:

- **delay** -> especifica en <valor> un retardo
- **loss** -> especifica en <valor> un porcentaje de pérdidas
- **corrupt** -> especifica en <valor> un porcentaje de paquetes dañados
- **duplicate** -> especifica en <valor> un porcentaje de paquetes duplicados.
- **reorder** -> especifica en <valor> un porcentaje de paquetes desordenados

Una vez definido un parámetro en una interfaz, se puede cambiar ese parámetro utilizando:

```
tc qdisc change dev <interfaz> root netem <parámetro><valor>
```

Para mostrar los parámetros que se ha definido se utiliza:

```
tc qdisc show
```

Para borrar la configuración de parámetros de una interfaz se utiliza:

```
tc qdisc del dev <interfaz> root
```

Para la configuración de ancho de banda se debe aplicar comandos para las 2 interfaces de red, teniendo en cuenta el siguiente formato:

```
tc qdisc add dev <interfaz> root handle 1:0 tbf rate
<valor> buffer <valor> limit <valor> latency <valor>
```

Ejemplo:

Supongamos que queremos emular una conexión WAN del PC emisor al PC receptor con 768kbps de bajada y 128kbps de subida. Asumiendo que el PC emisor está conectado a eth1, la interfaz eth1 recibe el tráfico entrante desde el PC receptor y eth0 envía el tráfico de salida. Como sabemos la una qdisc sólo funciona en el tráfico de salida, tenemos que limitar eth0 para nuestra velocidad de descarga de 768 kbps. Por el contrario, configuramos eth1 para nuestra velocidad de subida de 128 kbps.

```
tc qdisc add dev eth0 root handle 1:0 tbf rate 768kbit
burst 2048 latency 100ms
tc qdisc add dev eth1 root handle 2:0 tbf rate 128kbit
burst 2048 latency 100ms
```

CAPÍTULO IV

4 EJECUCIÓN Y APLICACIÓN DEL EMULADOR

En este capítulo se presenta los diferentes escenarios de pruebas con cada uno de los parámetros que ofrece NETEM, como así también una aplicación de todos los parámetro en conjunto de posibles escenarios reales y las respectivas herramientas que son necesarias para su implementación.

4.1 EJECUCIÓN DE NETEM

Para la ejecución de NETEM se implementa el escenario de pruebas que se muestra en la Figura 4.1, para la inyección de tráfico es necesario que tanto el PC emisor como el PC Receptor posean software de inyección de tráfico D-ITG cuya instalación y uso se detalla en el Anexo 1.

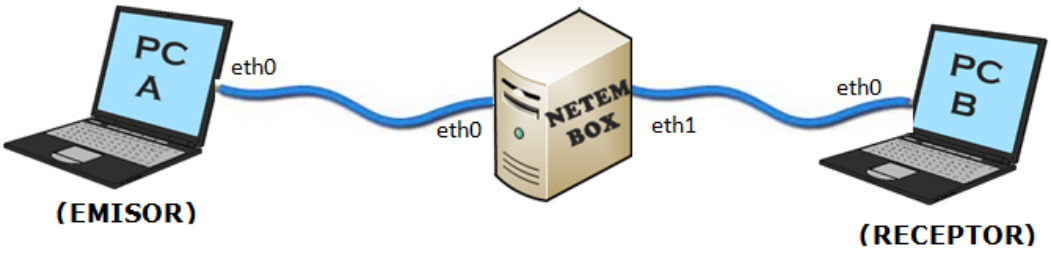

Figura 4.1. Escenario de Pruebas

- **PC A**

  `eth0 => ip estática (192.168.0.1)`

- **PC B**

  `eth0 => ip estática (192.168.0.4)`

- **NETEM BOX**

  `eth0 y eth1 => ip dinámicas`

4.1.1 Pruebas con parámetros de NETEM

La configuración de parámetros de inyección de tráfico se detalla en la Tabla 4.1 para *Define Flow* y Tabla 4.2 para *Settings* en el emisor. Los parámetros restantes se dejan con la configuración por defecto.

Tabla 4.1. Parámetros de Inyector de Trafico (*Define Flow*) – Emisor

Stream Option	
Meter	One-Way-Delay
Duration	60 (1minute)
Start Delay	0 (default)
Ramdon Seed	0 (default)
Headers Options	
Target host	192.168.0.4
Protocol	TCP
Destination Port	Default
Sourse Port	(Auto)

Tabla 4.2. Parámetros de Inyector de Tráfico (*Settings*) – Emisor

Sender Option	
Local Sender log	Local
Logging Server	-
Remote Receiver	Remote
Logging Server	192.168.0.4
Local Receiver Option	
Logging Type	None
Logging Server	-

En la Tabla 4.3 se muestra la configuración de parámetros de *Settings* en el receptor, no es necesario configurar *Define Flow* en el receptor ya que este hace la función de escuchar y no transmite.

Tabla 4.3. Parámetros de Inyector de Tráfico (*Settings*) – Receptor

Sender Option	
Local Sender log	None
Logging Server	-
Remote Receiver	None
Logging Server	-
Local Receiver Option	
Logging Type	Local
Logging Server	-

Para los parámetros que permite configurar NetEm se realiza una inyección de tráfico para cada uno de ellos, en la caja NetEm se configura los parámetros que presentamos a continuación:

a. Pruebas con parámetros de Demora

Para definir el valor del parámetro de demora en el emulador nos basamos en la tabla 4.4.

Tabla 4.4. Parámetros de Demora en caja NetEm

PARÁMETRO \ MEDIDA	%	ms
Demora	-	300
Perdida	-	-
Duplicación	-	-
Corrupción	-	-
Reordenamiento	-	-

En una ventana de terminal en modo de súper usuario se digita el comando que se muestra a continuación el mismo que va a dar como resultado una demora de 300 ms en la tarjeta de red eth1.

```
tc qdisc add dev eth1 root netem delay 300ms
```

b. Pruebas con parámetros de Pérdidas

De igual manera para el parámetro de pérdidas en el emulador NetEm se basa en la tabla 4.5 la que muestra el valor en porcentaje, para borrar la configuración del parámetro anterior se digita en el terminal el comando:

```
tc qdisc del dev eth1 root
```

Tabla 4.5. Parámetros de Perdida en caja NetEm

PARÁMETRO \ MEDIDA	%	ms
Demora	-	-
Perdida	50	-
Duplicación	-	-
Corrupción	-	-
Reordenamiento	-	-

Luego de borrar la configuración anterior se procede a digitar la nueva configuración en el terminal lo que permite tener pérdidas de hasta el 50% en la tarjeta de red eth1.

```
tc qdisc add dev eth1 root netem loss 50%
```

c. Pruebas con parámetros de Duplicación

Para el parámetro de duplicación se guiara en la tabla 4.6, la cual muestra el porcentaje que se duplicara el tráfico a su paso por el emulador.

Tabla 4.6. Parámetros de Duplicación en caja NetEm

PARÁMETRO \ MEDIDA	%	Ms
Demora	-	-
Perdida	-	-
Duplicación	10	-
Corrupción	-	-
Reordenamiento	-	-

Como se realizó en veces anteriores se digita el comando para borrar la configuración del emulador y posterior a esto se digita el siguiente comando para configurar la tarjeta de red eth1 con una duplicación del tráfico del 10%.

```
tc qdisc add dev eth1 root netem duplicate 10%
```

d. Pruebas con parámetros de Corrupción

El proceso de configuración de los parámetros en un poco repetitivo pero a su vez necesario, el porcentaje de corrupción se muestra en la tabla 4.7.

Tabla 4.7. Parámetros de Corrupción en caja NetEm.

PARÁMETRO \ MEDIDA	%	ms
Demora	-	-
Perdida	-	-
Duplicación	-	-
Corrupción	20	-
Reordenamiento	-	-

El siguiente comando permite cambiar la configuración anterior por la nueva del parámetro de corrupción con un porcentaje del 20% en la tarjeta de red eth1.

```
tc qdisc change dev eth1 root netem corrupt 20%
```

e. Pruebas con parámetros de Reordenamiento

En el parámetro de reordenamiento no se puede configurar y ejecutar independientemente, siempre tiene que configurarse con el parámetro de demora, los valores y porcentajes de estos dos parámetros se muestra en la tabla 4.8.

Tabla 4.8. Parámetros de Reordenamiento en caja NetEm.

PARÁMETRO \ MEDIDA	%	ms
Demora	-	200
Perdida	-	-
Duplicación	-	-
Corrupción	-	-
Reordenamiento	20	-

Como en otras ocasiones se procede a borrar la configuración anterior y se digita la nueva configuración para los parámetros de reordenamiento con un porcentaje del 20% y el parámetro de demora con un valor de 200 ms.

```
tc qdisc add dev eth1 root netem reorder 20% delay
200ms
```

Para verificar la configuración actual se puede digitar el siguiente comando, el mismo que indica los parámetros en uso.

```
tc qdisc show
```

4.2 APLICACIÓN DE PARÁMETROS DE UN POSIBLE ESCENARIO REAL EN EL EMULADOR

En esta sección se aplica una configuración con todos los parámetros al mismo tiempo que permite usar el emulador NetEm, en la Tabla 4.9 se tiene los porcentajes y valores muy cercanos a los de un posible escenario real que se configurará en la emulación.

Tabla 4.9. Parámetros de un posible escenario real en caja NetEm

PARÁMETRO \ MEDIDA	%	ms
Demora	-	100
Perdida	10	-
Duplicación	15	-
Corrupción	10	-
Reordenamiento	20	-

Para configurar el emulador se digita el comando que se muestra a continuación lo que provocara en la tarjeta de red eth1 una demora de 100ms, una pérdida de 10%, una duplicación del 15%, una corrupción del 10% y el reordenamiento del 20%, todos estos parámetros actuaran simultáneamente.

```
tc qdisc add dev eth1 root netem delay 100ms loss 10%
duplicate 15% corrupt 10% reorder 20%
```

CAPÍTULO V

5 ANÁLISIS DE RESULTADOS

En este capítulo se presenta los análisis del desempeño de cada uno de los parámetros del emulador NetEm como así también el respectivo análisis de una inyección de tráfico aplicando todos los parámetros en conjunto de un posible escenario real.

5.1 ANÁLISIS DE RESULTADOS Y DESEMPEÑO DE FUNCIONAMIENTO

Para poder evaluar los diferentes parámetros que nos ofrece el emulador de canal NetEm se utiliza las mediciones de caudal, retardo y jitter que nos proporciona el analizador del inyector de trafico D-ITG, los cuales a lo largo de este análisis están presentados gráficamente.

5.1.1 Análisis de Resultados y Desempeño del Parámetro de Demora en Netem

Las mediciones de caudal, retardo y jitter prácticamente dan como resultado gráficos muy similares, cada uno con sus respectivas unidades de medida, en el caso de Throughput que se muestra en la Figura 5.1 se tiene como resultado un promedio constante alrededor de los 4000 Kbit/s.

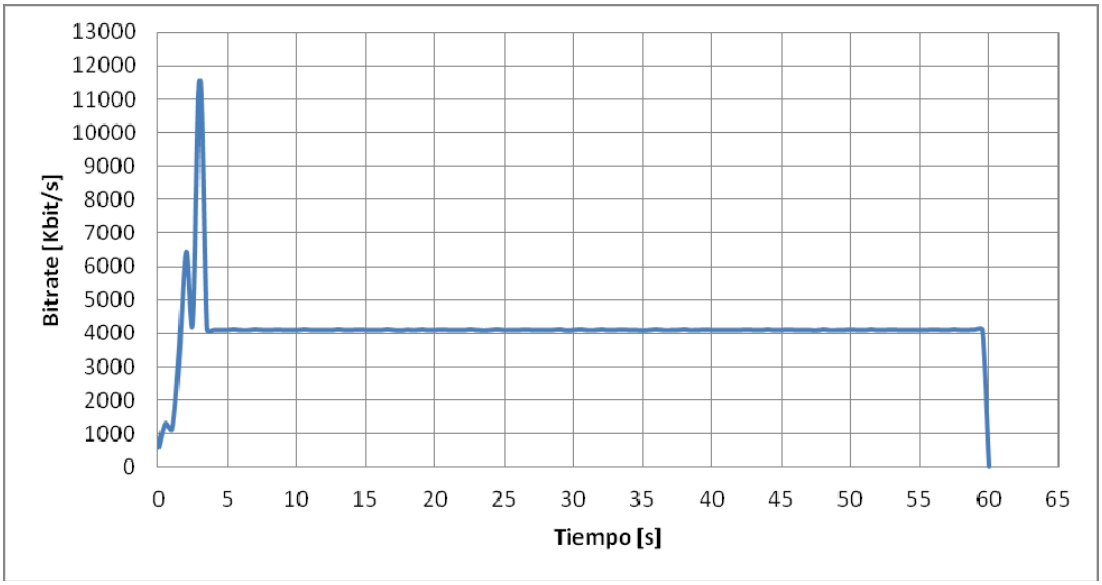

Figura 5.1. Throughput en función del tiempo del parámetro de Demora.

En este parámetro se configuro una demora de 300 ms, como promedio en la medición obtenemos la medida promedio de delay de 329 ms la cual es constante al transcurso de la mayoría de la inyección de tráfico, solamente con un incremento al inicio de la transmisión llegando a un máximo de 944 ms y un mínimo de 305 ms como se representa de mejor manera en la Figura 5.2 que nos muestra directamente la medición de este parámetro.

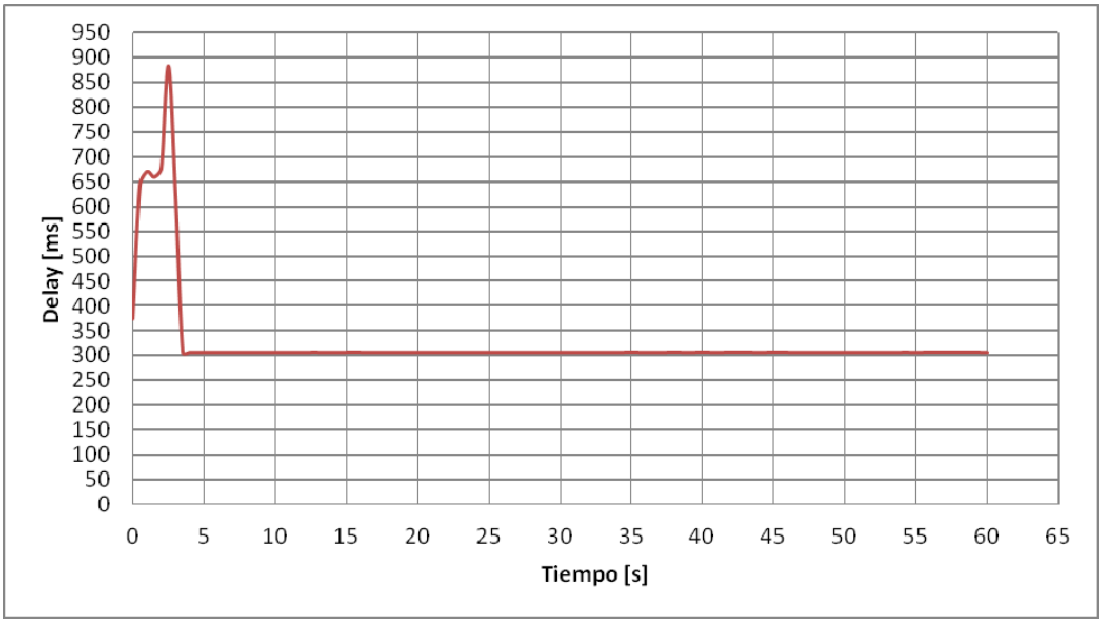

Figura 5.2. Retardo en función del tiempo del parámetro de Demora.

En el Figura 5.3 podemos apreciar que el Jitter es muy bajo y constante en la mayoría del tiempo con un valor de 1.32 ms.

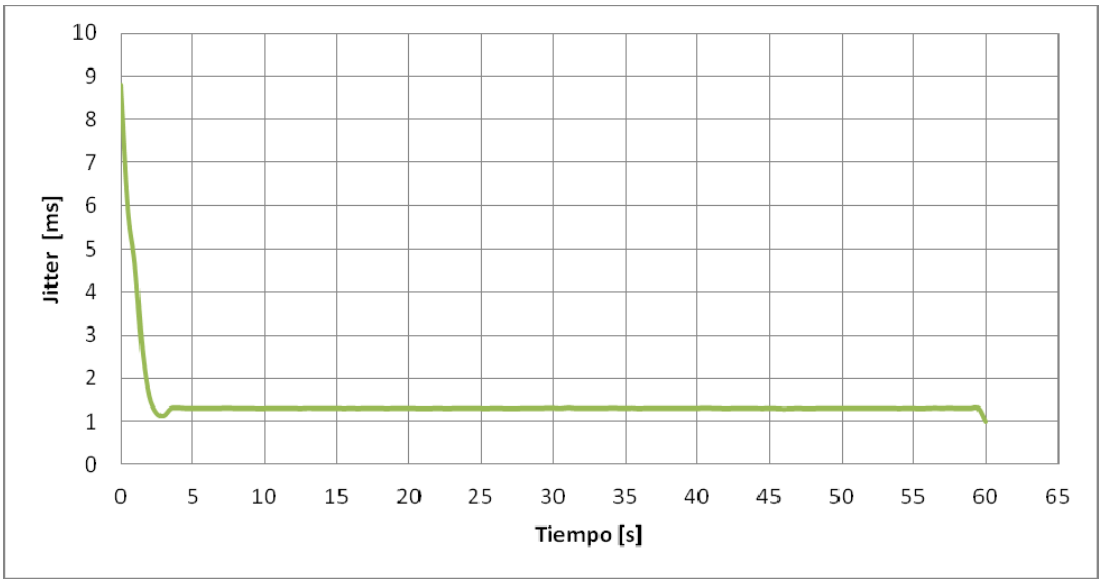

Figura 5.3. Jitter en función del tiempo del parámetro de Demora.

En la tabla 5.1 que se muestra a continuación se observa los valores mínimos, máximos y promedio de las mediciones, el emulador NetEm está cumpliendo con la configuración de demora, no se presentan pérdida de paquetes ni otros parámetros que puedan afectar a la transmisión.

Tabla 5.1. Archivo de resultados de inyección de tráfico del parámetro de Demora

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
---------------------------------------------------------
Flow number: 1
From 192.168.0.1:44529
To 192.168.0.4:8999
---------------------------------------------------------
Total time = 60.000382 s
Total packets = 60000
Minimum delay = 0.305448 s
Maximum delay = 0.944349 s
Average delay = 0.329059 s
Average jitter = 0.001327 s
Delay standard deviation = 0.096424 s
Bytes received = 30720000
Average bitrate = 4095.973922 Kbit/s
Average packet rate = 999.993633 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
---------------------------------------------------------
_________________________________________________________
**************** TOTAL RESULTS ******************

_________________________________________________________
Number of flows = 1
Total time = 60.000382 s
Total packets = 60000
Minimum delay = 0.305448 s
Maximum delay = 0.944349 s
Average delay = 0.329059 s
Average jitter = 0.001327 s
Delay standard deviation = 0.096424 s
Bytes received = 30720000
Average bitrate = 4095.973922 Kbit/s
Average packet rate = 999.993633 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
---------------------------------------------------------
```

5.1.2 Análisis de Resultado y Desempeño del Parámetro de Perdida en Netem

La pérdida de paquetes fue configurado con un porcentaje del 50%, los mismos que se eliminan al azar, en la Figura 5.4 se observa que el Throughput es bajo y prácticamente en algunos lapsos de tiempo de la transmisión llega hasta cero, tiene un promedio de 15 Kbit/s lo que nos indica que el emulador realiza eficientemente la operación.

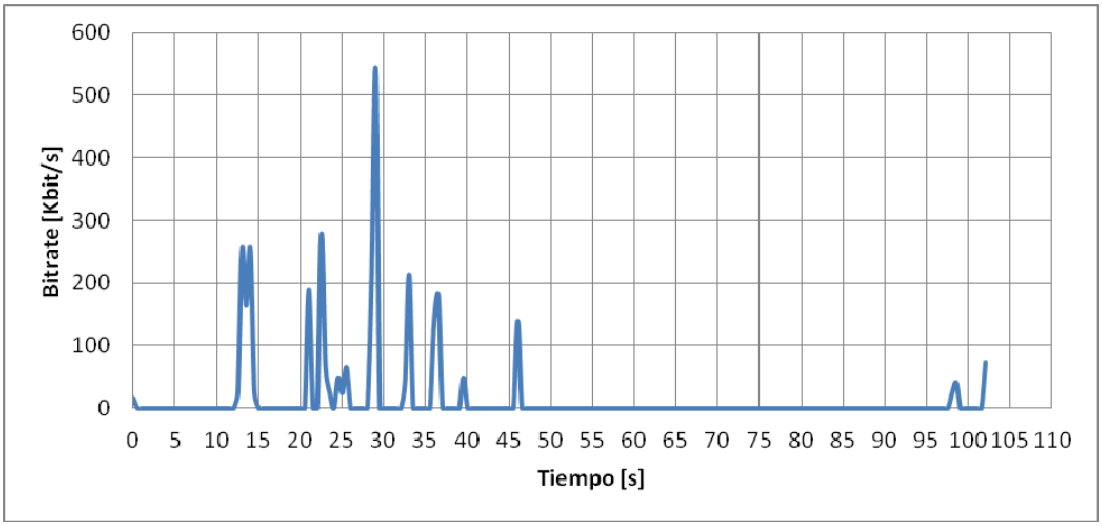

Figura 5.4. Throughput en función del tiempo del parámetro de Pérdida.

El tiempo que se configuro la inyección de tráfico es de 60 s pero la Figura 5.4, Figura 5.5 y Figura 5.6 se observa que se extiende hasta un tiempo de 102 s de transmisión, esto se debe a características propias del protocolo TCP que solicita retransmisión debido a que no recibe los paquetes por las pérdidas que genera el emulador NetEm.

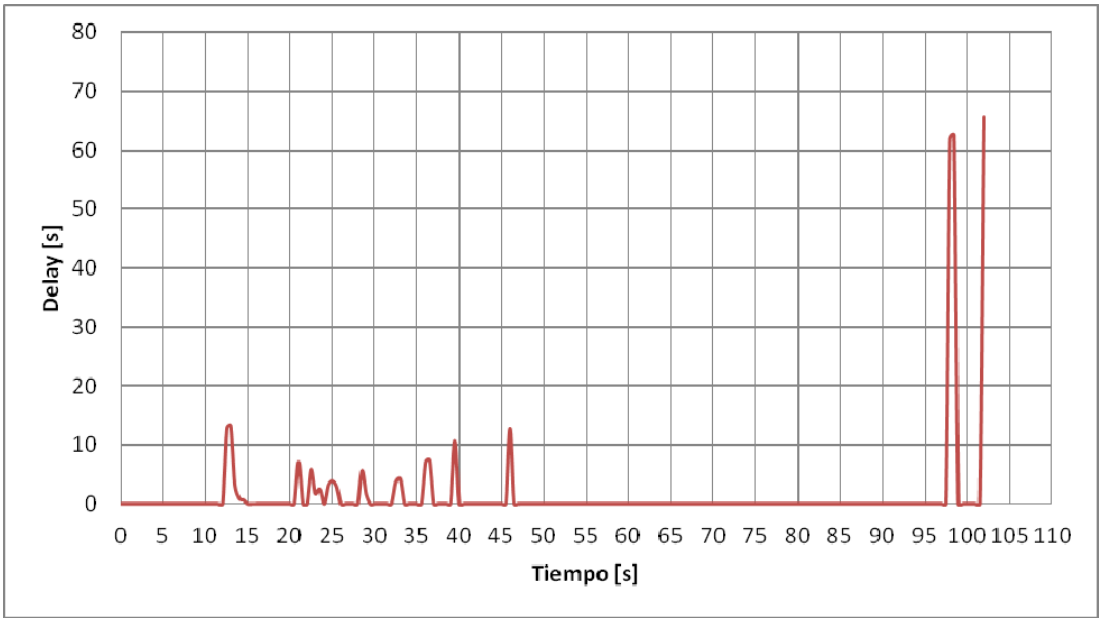

Figura 5.5. Retardo en función del tiempo del parámetro de Pérdida.

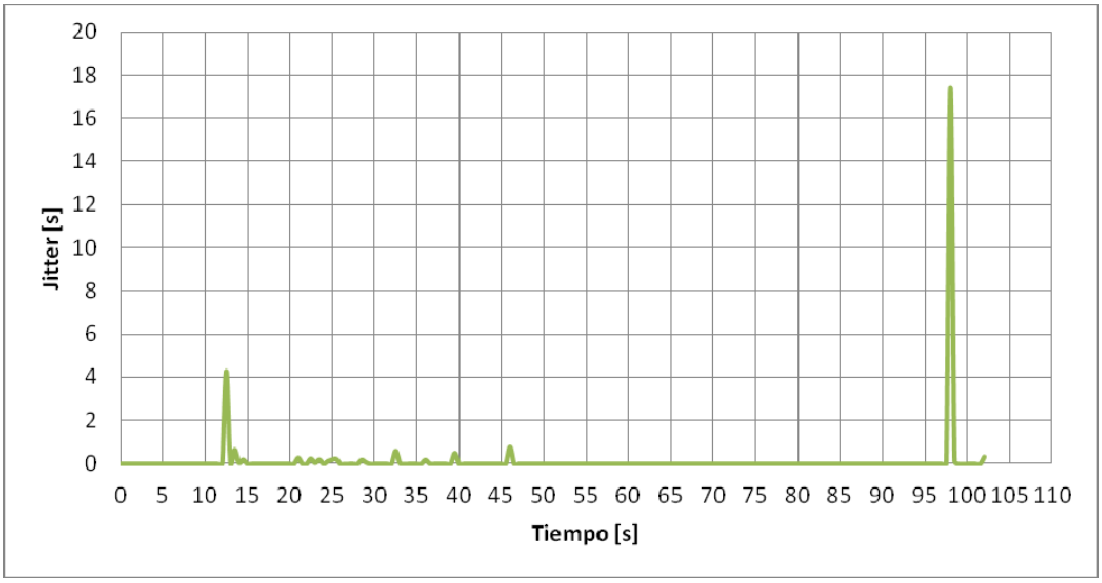

Figura 5.6. Jitter en función del tiempo del parámetro de Pérdida.

Con el archivo de resultados que se muestra en la Tabla 5.2 podemos confirmar que el emulador está funcionando correctamente, se observa que recibe 376 paquetes de 60000 paquetes que fueron enviados.

Tabla 5.2. Archivo de resultados de inyección de tráfico del parámetro de Perdida

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
-------------------------------------------------------
Flow number: 1
From 192.168.0.1:55479
To 192.168.0.4:8999
-------------------------------------------------------
Total time = 102.011316 s
Total packets = 376
Minimum delay = 0.014440 s
Maximum delay = 65.641595 s
Average delay = 8.006880 s
Average jitter = 0.369012 s
Delay standard deviation = 12.937408 s
Bytes received = 192512
Average bitrate = 15.097305 Kbit/s
Average packet rate = 3.685866 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
-------------------------------------------------------

************** TOTAL RESULTS ******************

Number of flows = 1
Total time = 102.011316 s
Total packets = 376
Minimum delay = 0.014440 s
Maximum delay = 65.641595 s
Average delay = 8.006880 s
Average jitter = 0.369012 s
Delay standard deviation = 12.937408 s
Bytes received = 192512
Average bitrate = 15.097305 Kbit/s
Average packet rate = 3.685866 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
-------------------------------------------------------
```

5.1.3 Análisis de Resultado y Desempeño del Parámetro de Duplicación en Netem

Para este parámetro se estableció una duplicación de paquetes del 10%, la misma que el emulador la realiza al azar antes de encolar los paquetes, en la Figura 5.7, se muestra un Throughput contante de 4.096 Mbps durante los 60 s de transmisión.

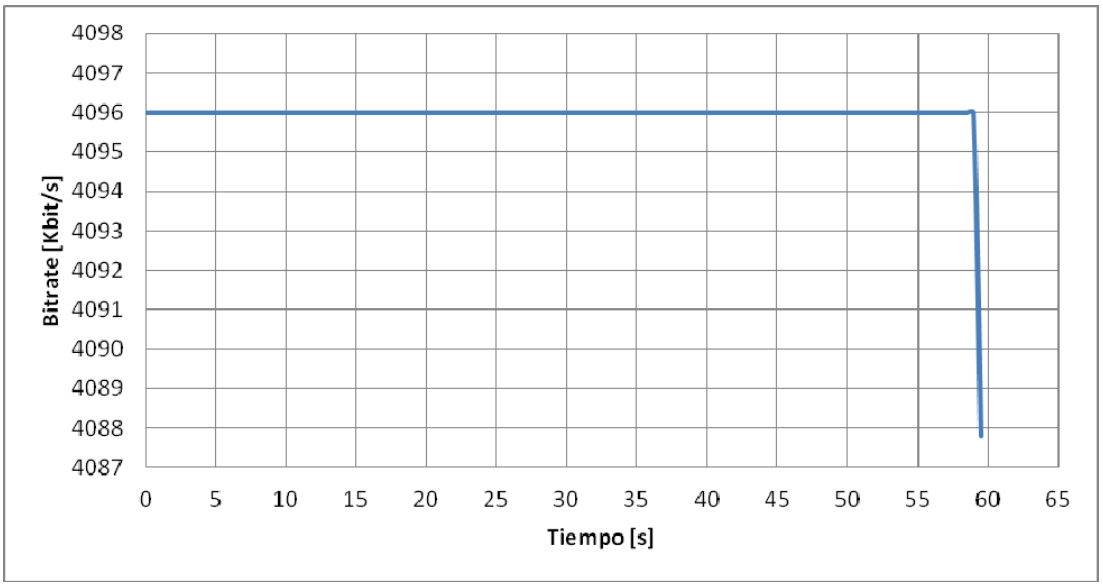

Figura 5.7. Throughput en función del tiempo del parámetro de Duplicación

Los valores de retardo y jitter que se muestran en la Figura 5.8 y Figura 5.9 son bajos y comparados conceptualmente con los valores del parámetro de perdida son inversos debido a que comparten el principio de correlación con diferente operación.

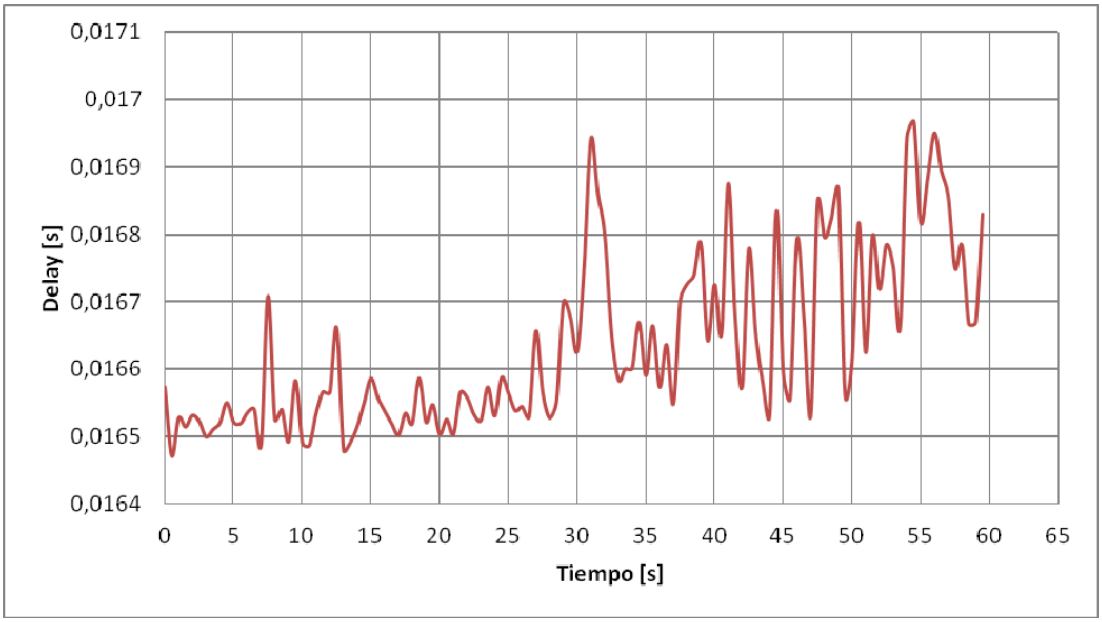

Figura 5.8. Retardo en función del tiempo del parámetro de Duplicación.

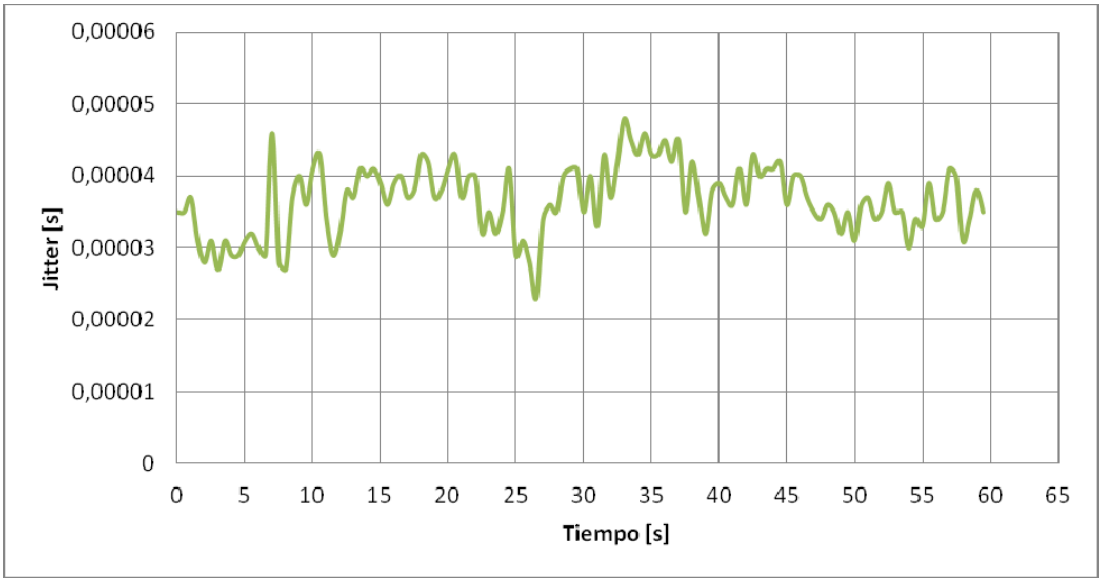

Figura 5.9. Jitter en función del tiempo del parámetro de Duplicación.

En la tabla 5.4 se muestra los promedios de las mediciones con sus máximos y mínimos.

Tabla 5.3. Archivo de resultados de inyección de tráfico del parámetro de Duplicación

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
-------------------------------------------------------
Flow number: 1
From 192.168.0.1:55483
To 192.168.0.4:8999
-------------------------------------------------------
Total time = 59.999005 s
Total packets = 59999
Minimum delay = 0.016269 s
Maximum delay = 0.018660 s
Average delay = 0.016634 s
Average jitter = 0.000037 s
Delay standard deviation = 0.000216 s
Bytes received = 30719488
Average bitrate = 4095.999659 Kbit/s
Average packet rate = 999.999917 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
-------------------------------------------------------
_______________________________________________________
**************** TOTAL RESULTS ******************

_______________________________________________________
Number of flows = 1
Total time = 59.999005 s
Total packets = 59999
Minimum delay = 0.016269 s
Maximum delay = 0.018660 s
Average delay = 0.016634 s
Average jitter = 0.000037 s
Delay standard deviation = 0.000216 s
Bytes received = 30719488
Average bitrate = 4095.999659 Kbit/s
Average packet rate = 999.999917 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
-------------------------------------------------------
```

5.1.4 Análisis de Resultado y Desempeño del Parámetro de Corrupción en Netem

La corrupción en transferencia de datos se da por muchos factores como interrupción de la transmisión o condiciones ambientales entre otras, para recrear una situación similar se configuro una corrupción en el emulador con una porcentaje del 20%, al llegar los datos con errores son descartados, por tal motivo se puede apreciar en la Figura 5.10 que la tasa de transmisión disminuye teniendo como promedio de 727 Kbit/s con variantes muy notorias al transcurso de los 60 s que se realiza la inyección de tráfico a través del emulador.

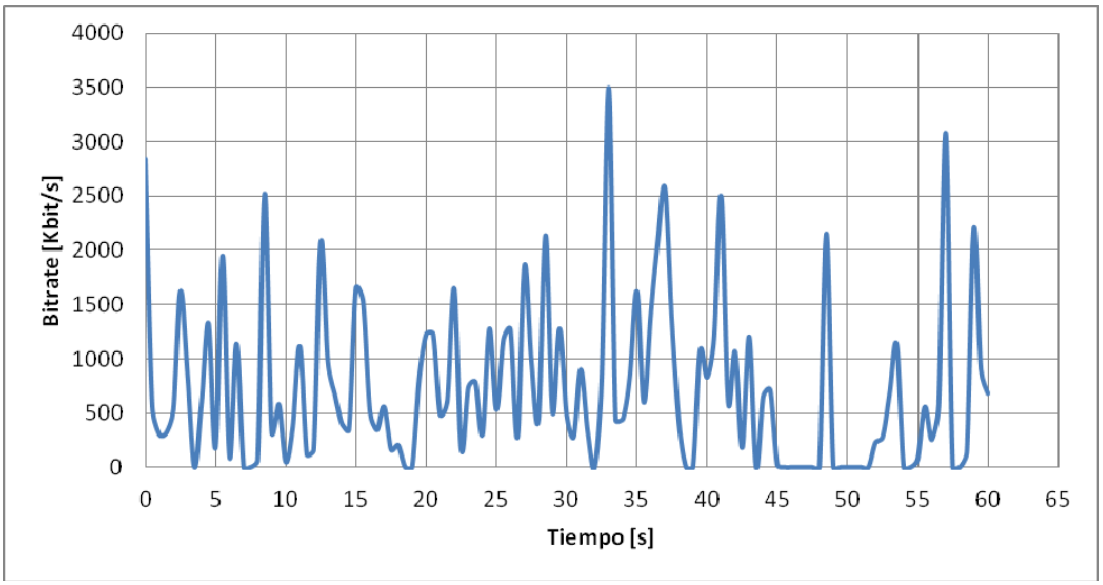

Figura 5.10. Throughput en función del tiempo del parámetro de Corrupción.

La Figura 5.11 muestra las mediciones de delay, como así también en la Figura 5.12 representa las mediciones del jitter al transcurso de la transmisión, en el que observamos el comportamiento del parámetro de corrupción que genera el emulador, el promedio de jitter es de 0.0112 s, el mismo que compagina con la configuración que se realizó en caja NetEm.

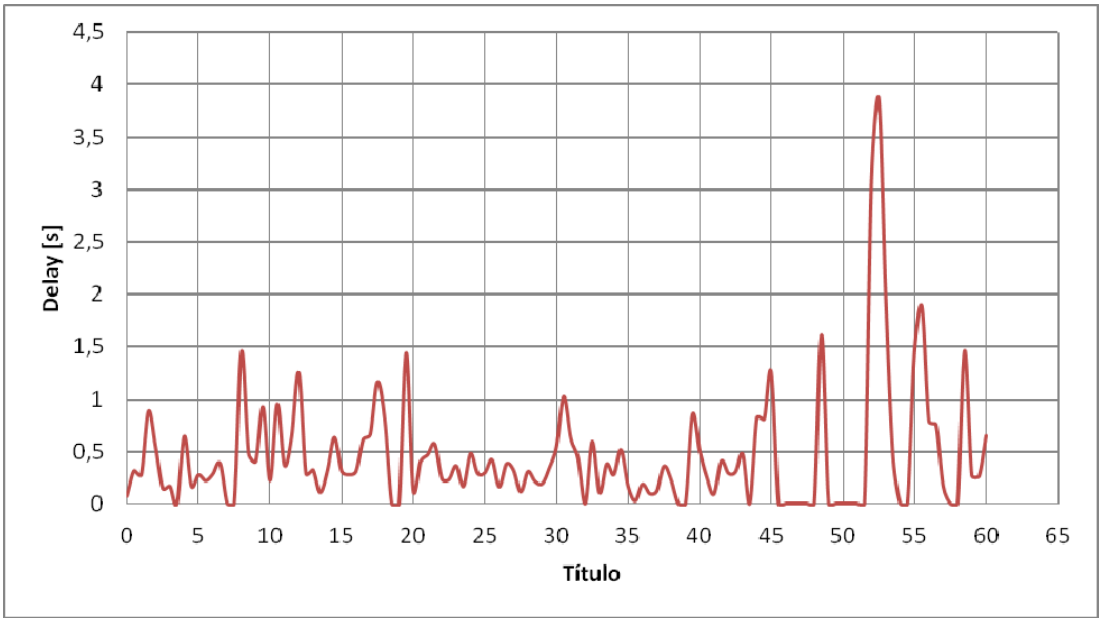

Figura 5.11. Retardo en función del tiempo del parámetro de Corrupción.

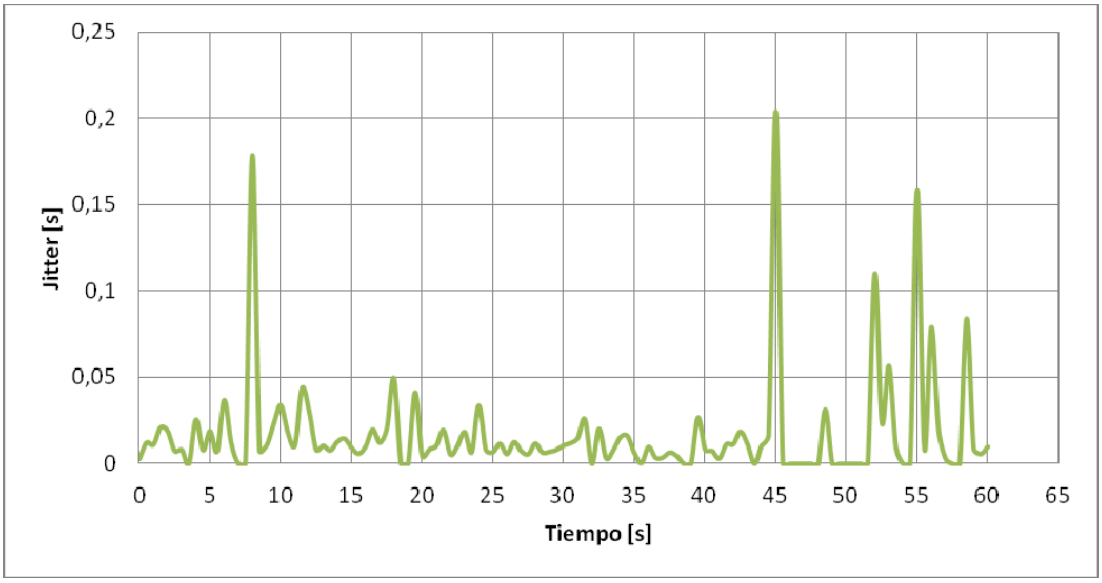

Figura 5.12. Jitter en función del tiempo del parámetro de Corrupción.

Debido a que los paquetes con errores son descartados en el protocolo TCP, el archivo de resultados muestra que existe perdida de paquetes y una tasa de transmisión baja de 177.55 paquetes por segundo, entre otros valores de medición presentados en la Tabla 5.4.

Tabla 5.4. Archivo de resultados de inyección de tráfico del parámetro de Corrupción

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
------------------------------------------------------------
Flow number: 1
From 192.168.0.1:58718
To 192.168.0.4:8999
------------------------------------------------------------
Total time = 60.394203 s
Total packets = 10723
Minimum delay = 0.015938 s
Maximum delay = 4.519819 s
Average delay = 0.400834 s
Average jitter = 0.011199 s
Delay standard deviation = 0.636632 s
Bytes received = 5490176
Average bitrate = 727.245428 Kbit/s
Average packet rate = 177.550153 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
------------------------------------------------------------

****************** TOTAL RESULTS ******************

Number of flows = 1
Total time = 60.394203 s
Total packets = 10723
Minimum delay = 0.015938 s
Maximum delay = 4.519819 s
Average delay = 0.400834 s
Average jitter = 0.011199 s
Delay standard deviation = 0.636632 s
Bytes received = 5490176
Average bitrate = 727.245428 Kbit/s
Average packet rate = 177.550153 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
------------------------------------------------------------
```

5.1.5 Análisis de Resultado y Desempeño del Parámetro de Reordenamiento en Netem

Se provoca un reordenamiento del 20% en la transmisión con una demora de 200 ms, al inicio de la inyección de tráfico se observa en la Figura 5.13 un aumento de la tasa de transmisión hasta 14 Mbps y luego disminuye hasta estabilizar en aproximadamente 4 Mbps, esto se debe al principio que utiliza NetEm para reordenar los paquetes.

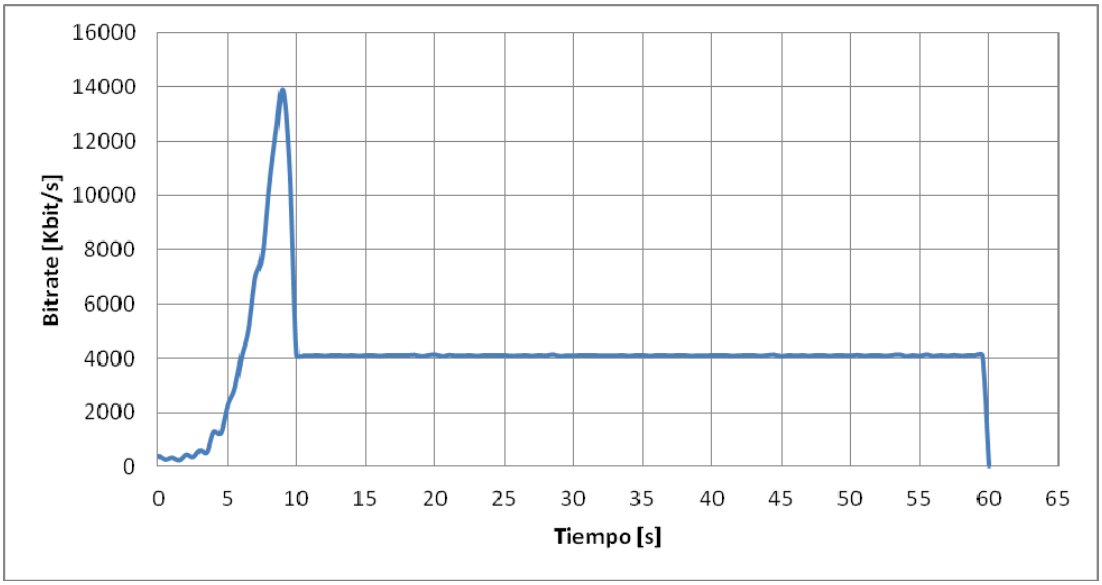

Figura 5.13. Throughput en función del tiempo del parámetro de Reordenamiento.

En la Figura 5.14 nos presenta el retardo al transcurso de la transmisión, teniendo como demora máxima de 1.14 s acompañado de variantes hasta estabilizar en 216 ms en el resto de la inyección, lo cual es muy aproximado a lo configurado en el emulador.

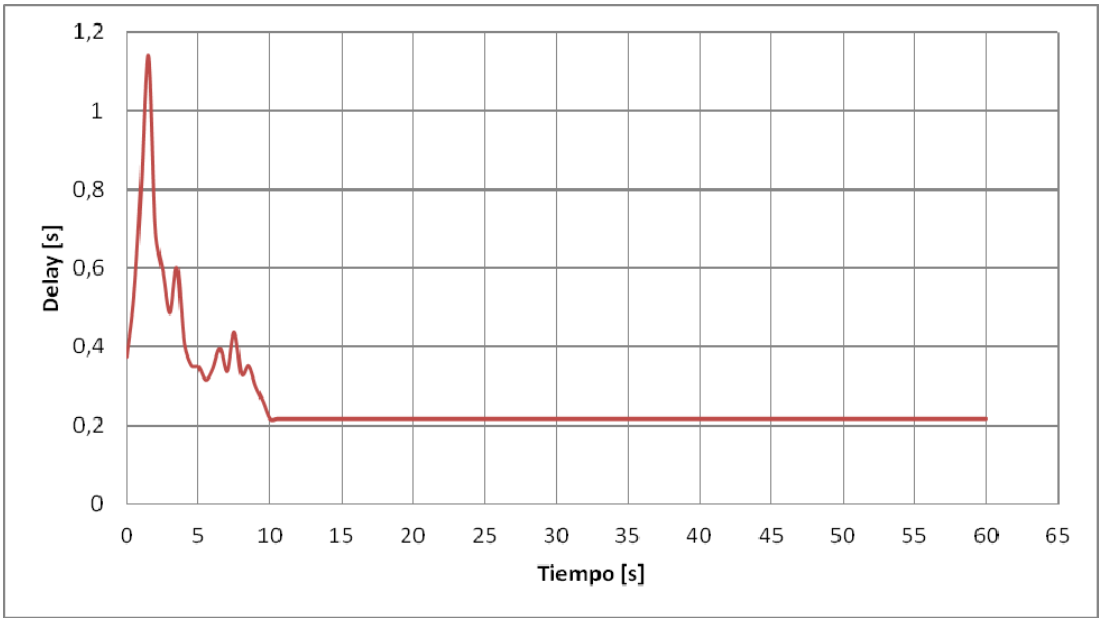

Figura 5.14. Retardo en función del tiempo del parámetro de Reordenamiento.

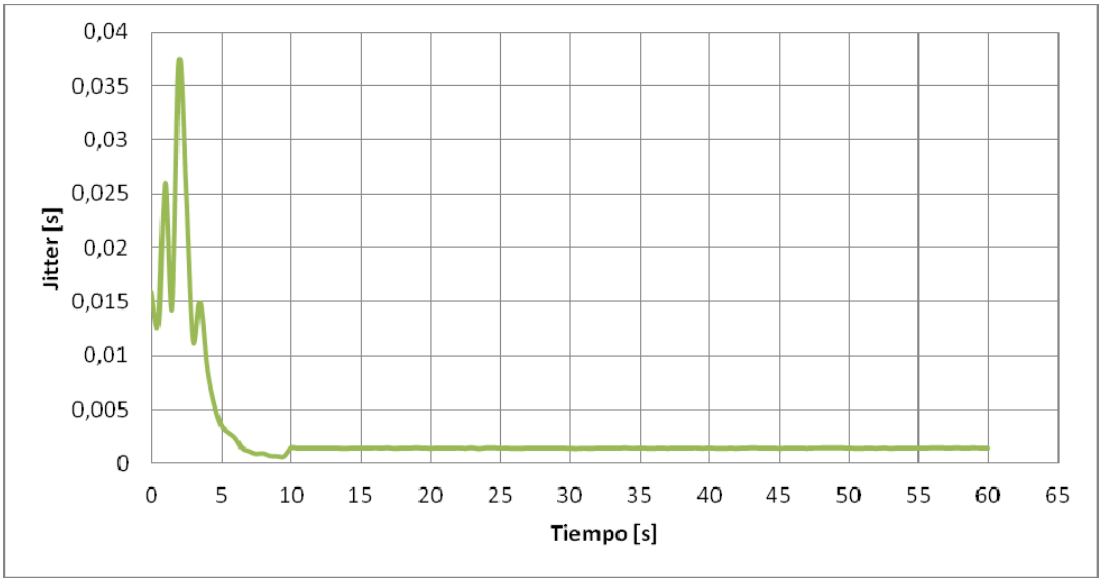

Figura 5.15. Jitter en función del tiempo del parámetro de Reordenamiento.

La transmisión no tiene presencia de ruido más que el que siempre existe debido a los factores naturales como de cables, entre otros, esto se puede apreciar en la Figura 5.15 que nos muestra el grafico de Jitter, tampoco existe perdida de información, los valores promedio y demás valores de mediciones se presenta en la Tabla 5.5.

Tabla 5.5. Archivo de resultados de inyección de tráfico del parámetro de Reordenamiento

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
-------------------------------------------------------
Flow number: 1
From 192.168.0.1:50712
To 192.168.0.4:8999
-------------------------------------------------------
Total time = 60.000294 s
Total packets = 60000
Minimum delay = 0.212221 s
Maximum delay = 1.142141 s
Average delay = 0.238541 s
Average jitter = 0.001499 s
Delay standard deviation = 0.069036 s
Bytes received = 30720000
Average bitrate = 4095.979930 Kbit/s
Average packet rate = 999.995100 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
-------------------------------------------------------

************** TOTAL RESULTS ******************

Number of flows = 1
Total time = 60.000294 s
Total packets = 60000
Minimum delay = 0.212221 s
Maximum delay = 1.142141 s
Average delay = 0.238541 s
Average jitter = 0.001499 s
Delay standard deviation = 0.069036 s
Bytes received = 30720000
Average bitrate = 4095.979930 Kbit/s
Average packet rate = 999.995100 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
-------------------------------------------------------
```

5.1.6 Análisis de Resultado y Desempeño de los Parámetros en un posible escenario real en Netem

Para evaluar el desempeño de Netem en un posible escenario real bajo el protocolo TCP se aplica simultáneamente todos los parámetros que nos proporciona el emulador, una vez realizado la inyección de tráfico con D-ITG, en la Figura 5.16 se muestra un Throughput relativamente bajo y muy cambiante sin sobrepasar los 900 Kbit/s, en algunos instantes llega hasta ser nulo, con 200 Kbit/s de promedio en toda la inyección.

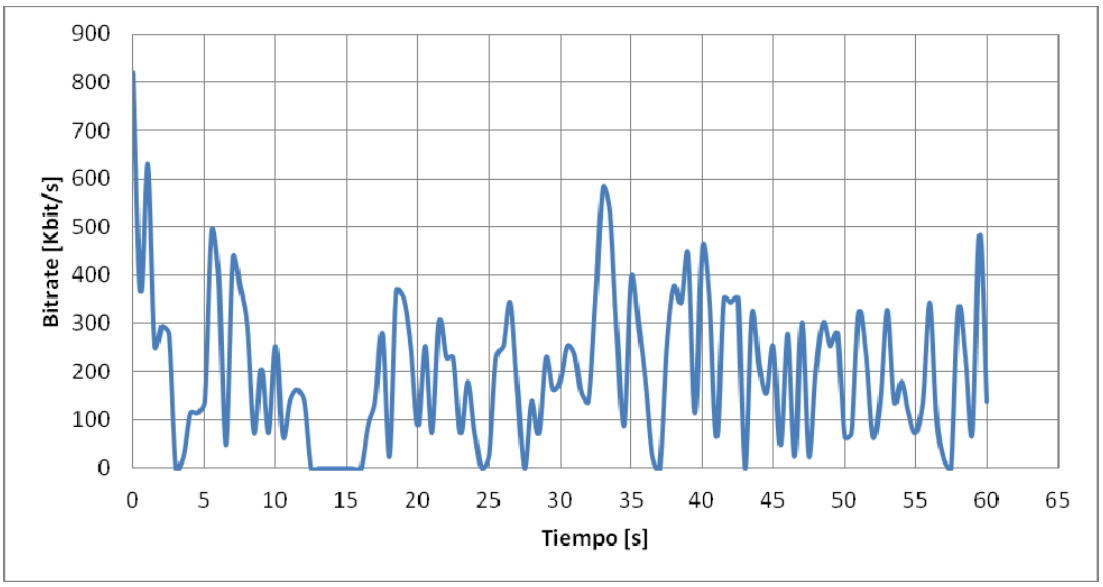

Figura 5.16. Throughput en función del tiempo de un posible escenario real.

El retraso que presenta la Figura 5.17 no solamente está relacionado por la configuración del parámetro de Demora, ya que otros parámetros como el de pérdidas también influyen, el desempeño general de la transmisión nos muestra una retardo promedio de 1.92 s, con un pico máximo de 7.73 s y mínimo de 0.22 s con lo que se puede afirmar que los parámetros configurados en la caja NetEm están desempeñando de manera eficiente.

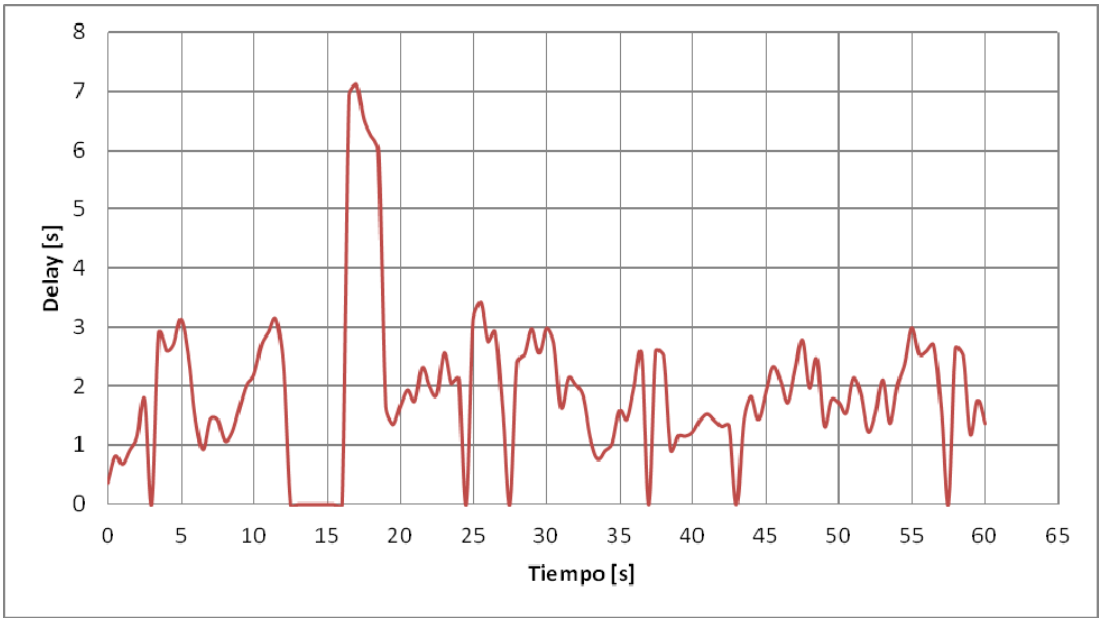

Figura 5.17. Retardo en función del tiempo de un posible escenario real.

Con un porcentaje de corrupción de 10% y considerando la influencia de los demás parámetros configurados en el emulador se observa en la Figura 5.18 un jitter promedio de 0.04 s.

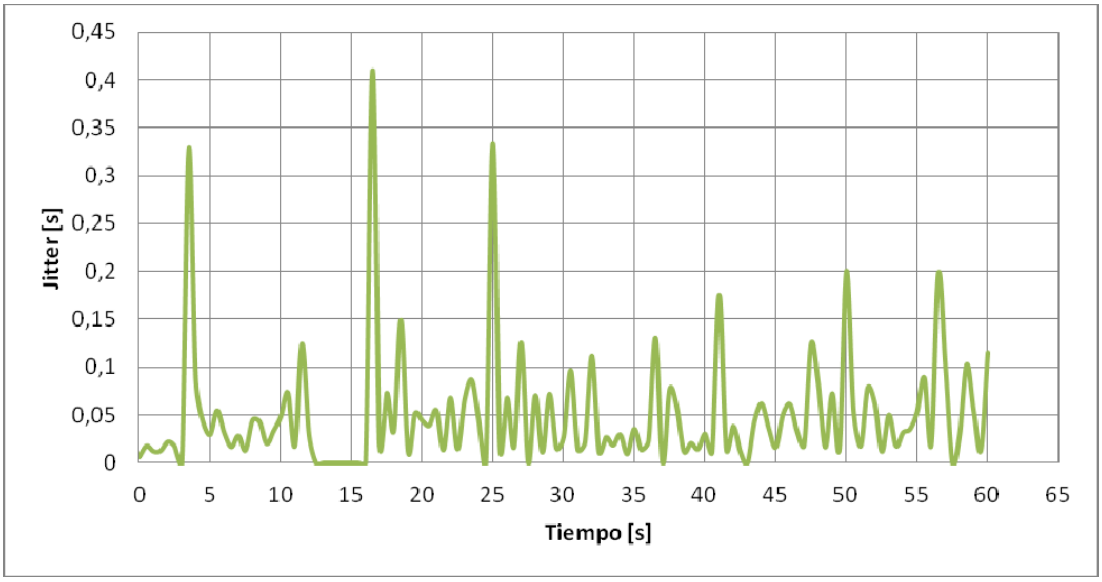

Figura 5.18. Jitter en función del tiempo de un posible escenario real.

Tabla 5.6. Archivo de resultados de inyección de tráfico de un posible escenario real.

```
ITGDec version 2.8.0-rc1 (r457:458)
Compile-time options:
-------------------------------------------------------
Flow number: 1
From 192.168.0.1:33336
To 192.168.0.4:8999
-------------------------------------------------------
Total time = 60.937482 s
Total packets = 2987
Minimum delay = 0.220986 s
Maximum delay = 7.726704 s
Average delay = 1.923996 s
Average jitter = 0.040376 s
Delay standard deviation = 1.200314 s
Bytes received = 1529344
Average bitrate = 200.775477 Kbit/s
Average packet rate = 49.017450 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0.000000 pkt
-------------------------------------------------------

************** TOTAL RESULTS ******************

Number of flows = 1
Total time = 60.937482 s
Total packets = 2987
Minimum delay = 0.220986 s
Maximum delay = 7.726704 s
Average delay = 1.923996 s
Average jitter = 0.040376 s
Delay standard deviation = 1.200314 s
Bytes received = 1529344
Average bitrate = 200.775477 Kbit/s
Average packet rate = 49.017450 pkt/s
Packets dropped = 0 (0.00 %)
Average loss-burst size = 0 pkt
Error lines = 0
-------------------------------------------------------
```

CAPÍTULO VI

6 GUÍAS PRÁCTICAS DE LABORATORIO

6.1 INSTALACIÓN Y USO DE NETEM

6.1.1 Fecha Límite de Entrega

(Practica NETEM 1): dd / mm / aaaa

Los artículos recibidos con posterioridad a esta fecha no serán evaluados.

6.1.2 Documentación a Entregar

Memoria escrita por grupo entregada en formato electrónico, archivo tipo pdf, la cual debe ser enviada por correo electrónico a xxxxxx@espe.edu.ec. El asunto (subject) del e-mail debe tener el siguiente formato:

- Practica NETEM1.
- Apellidos de los integrantes.

Se recomienda emplear procesadores de texto afines a la investigación científica tecnológica como LATEX. Debe ser presentado en un formato tipo reporte o memoria, en el cual se recomienda los siguientes ítems, pudiendo cambiar de acuerdo al criterio del grupo de trabajo.

- Abstract o resumen
- Introducción
- Materiales y métodos

- Desarrollo
- Conclusiones y recomendaciones
- Bibliografía

La memoria o reporte no debe ser inferior a 3 páginas ni superior a 10, en caso de no cumplir con este requisito, el trabajo no será evaluado.

6.1.3 Objetivos

- Familiarizar al estudiante con el emulador de canal NETEM
- Realizar la instalación y configuración del emulador de canal NETEM
- Determinar las funcionalidades del emulador.

6.1.4 Introducción

A menudo muchos protocolos y aplicaciones funcionan mal cuando se exponen a las redes de la vida real con el retraso y la pérdida de paquetes. A menudo, es costoso y difícil de reproducir el comportamiento de la redes en un entorno controlado. Hay herramientas disponibles para las pruebas, pero son costosas soluciones de hardware, software propio, o proyectos de investigación limitados.

NetEm es una mejora reciente de las instalaciones de control de tráfico de Linux que permite añadir retardo, pérdida de paquetes y otros escenarios. NetEm se construye a partir de la Calidad existente de servicio (QoS) y servicios diferenciados (DiffServ) instalados en el kernel de Linux.

6.1.5 Materiales

- 2 Computadoras con sistema operativo Linux, distribución Ubuntu, instalado inyector de tráfico DITG (ver anexo de instalación)

- 1 Computadoras con sistema operativo Linux, distribución Ubuntu con 2 tarjetas de red.

- 2 Cables cruzados de 1m.

6.1.6 Procedimiento

6.1.6.1 Instalación NETEM BOX y Puente de Red

- En NETEM BOX establecer una conexión directa a internet.

- Abrir un terminal que se encuentra dentro de la pestaña de aplicaciones.

- Para instalar las utilidades de NETEM; Digitamos los siguientes comandos:

```
sudo apt-get update
sudo apt-get install iproute
```

- Para instalar utilidades de puente ejecutamos:

```
sudo apt-get install bidge-utils
```

6.1.6.2 Configuración del Puente de Red

Debemos aseguras de desactivar la configuración de la red para las interfaces:

```
ifconfig eth0 0.0.0.0
ifconfig eth1 0.0.0.0
```

A continuación, se crea el puente y levanta:

```
brctl addbr br0
brctl setfd br0 0
```

```
brctl addif br0 eth0
brctl addif br0 eth1
ifconfig br0 up
```

NOTA: Si se reinicia el computador se perderá la configuración de puente de red y se tendrá que configurar nuevamente.

6.1.6.3 Conexión de Equipos

A continuación realizamos la conexión de las computadoras como se muestra en el siguiente gráfico.

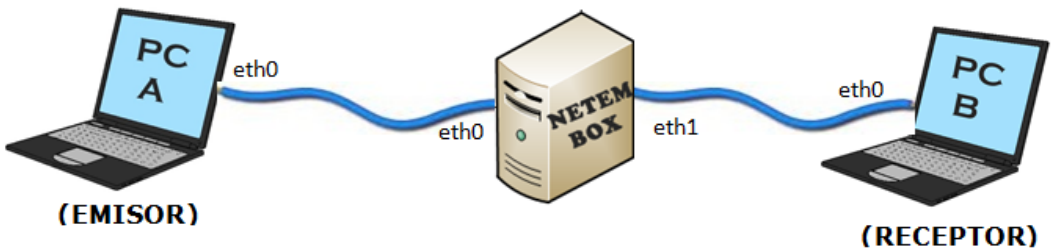

Figura 6.1. Escenario de Pruebas – Conexión equipos

- **PC A**

```
eth0 => ip estática (192.168.0.1)
```

- **PC B**

```
eth0 => ip estática (192.168.0.4)
```

- **NETEM BOX**

```
eth0 y eth1 => ip dinámicas
```

NOTA: En PC A y PC B deben tener instalado el inyector de tráfico que este caso es D-ITG (referir anexo D-ITG).

6.1.6.4 Configuración D-ITG

La configuración de parámetros de inyección de tráfico se detalla en la Tabla 6.1 para Define Flow y Tabla 6.2 para Settings en el emisor. Los parámetros restantes los dejamos con la configuración por defecto.

Tabla 6.1. Parámetros de Inyector de Trafico (Define Flow) – Emisor

Stream Option	
Meter	One-Way-Delay
Duration	60 (1minute)
Start Delay	0 (default)
Ramdon Seed	0 (default)
Headers Options	
Target host	192.168.0.4
Protocol	TCP
Destination Port	Default
Sourse Port	(Auto)

Tabla 6.2. Parámetros de Inyector de Tráfico (Settings) – Emisor

Sender Option	
Local Sender log	Local
Logging Server	-
Remote Receiver	Remote
Logging Server	192.168.0.4
Local Receiver Option	
Logging Type	None
Logging Server	-

En la Tabla 6.3 se muestra la configuración de parámetros de Settings en el receptor, no es necesario configurar Define Flow en el receptor ya que este hace la función de escuchar y no transmite.

Tabla 6.3. Parámetros de Inyector de Tráfico (Settings) – Receptor

Sender Option	
Local Sender log	None
Logging Server	-
Remote Receiver	None
Logging Server	-
Local Receiver Option	
Logging Type	Local
Logging Server	-

6.1.6.5 Configuración NETEM BOX

Tabla 6.4. Valores de parámetros de NetEm

PARÁMETRO \ MEDIDA	%	ms
Demora	-	
Perdida		
Duplicación		
Corrupción		
Reordenamiento		

(Valores proporcionados por encargado a dirigir la práctica)

Para la configuración de NETEM debemos aplicar comandos en un terminal teniendo en cuenta el siguiente formato:

```
tc qdisc add dev <interfaz> root netem <parámetro>
<valor>
```

Donde <interfaz> sustituimos por eth1 para todos los casos, <parámetro> sustituimos de acuerdo al parámetro a configurar y <valor> sustituimos en función de la Tabla 6.4.

Para cada escenario que me muestra a continuación digitamos los comandos que se indican y realizamos una inyección de tráfico desde PC A hacia PC B teniendo en cuenta las configuraciones de D-ITG de las Tablas 6.1, 6.2 y 6.3

6.1.7 Demora

```
tc qdisc add dev eth1 root netem delay <valor>ms
```

6.1.8 Perdida

```
tc qdisc del dev eth1
tc qdisc add dev eth1 root netem loss <valor>%
tc qdisc show
```

6.1.9 Duplicación

```
tc qdisc del dev eth1
tc qdisc add dev eth1 root netem duplicate <valor>%
tc qdisc show
```

6.1.10 Corrupción

```
tc qdisc del dev eth1
tc qdisc add dev eth1 root netem corrupt <valor>%
tc qdisc show
```

6.1.11 Reordenamiento

```
tc qdisc del dev eth1
tc qdisc add dev eth1 root netem reorder <valor>% delay
<valor>ms
tc qdisc show
```

Una vez finalizada la práctica obtendremos 5 resultados de mediciones con el Analyzer de D-ITG para su posterior análisis.

6.2 USO DE NETEM CON ESCENARIO REAL

6.2.1 Fecha límite de entrega

(Practica NETEM _2): dd / mm / aaaa

Los artículos recibidos con posterioridad a esta fecha no serán evaluados.

6.2.2 Documentación a entregar

Memoria escrita por grupo entregada en formato electrónico, archivo tipo pdf, la cual debe ser enviada por correo electrónico a xxxxxx@espe.edu.ec. El asunto (subject) del e-mail debe tener el siguiente formato:

- Practica NETEM_2.
- Apellidos de los integrantes.

Se recomienda emplear procesadores de texto afines a la investigación científica tecnológica como LATEX. Debe ser presentado en un formato tipo reporte o memoria, en el cual se recomienda los siguientes ítems, pudiendo cambiar de acuerdo al criterio del grupo de trabajo.

- Abstract o resumen
- Introducción
- Materiales y métodos
- Desarrollo
- Conclusiones y recomendaciones
- Bibliografía

La memoria o reporte no debe ser inferior a 3 páginas ni superior a 10, en caso de no cumplir con este requisito, el trabajo no será evaluado.

6.2.3 Objetivos

- Analizar el desempeño del emulador de canal NETEM
- Evaluar el comportamiento de una red en un escenario controlado.

6.2.4 Introducción

A menudo muchos protocolos y aplicaciones funcionan mal cuando se exponen a las redes de la vida real con el retraso y la pérdida de paquetes. A menudo, es costoso y difícil de reproducir el comportamiento de la redes en un entorno controlado. Hay herramientas disponibles para las pruebas, pero son costosas soluciones de hardware, software propio, o proyectos de investigación limitados.

NetEm es una mejora reciente de las instalaciones de control de tráfico de Linux que permite añadir retardo, pérdida de paquetes y otros escenarios. NetEm se construye a partir de la Calidad existente de servicio (QoS) y servicios diferenciados (DiffServ) instalados en el kernel de Linux.

6.2.5 Materiales

- 2 Computadoras con sistema operativo Linux, distribución Ubuntu, instalado inyector de tráfico DITG (ver anexo de instalación)

- 1 Computadoras con sistema operativo Linux, distribución Ubuntu con 2 tarjetas de red.

- 2 Cables cruzados de 1m.

6.2.6 Procedimiento

Para cada escenario que se muestra a continuación digitamos los comandos que se indican y se realiza una inyección de tráfico desde PC A hacia PC B teniendo en cuenta las configuraciones de D-ITG como se muestra en el Anexo 1.

6.2.6.1 Configuración NETEM BOX escenario 1

Tabla 6.5. Valores de parámetros de NetEm Iny1

PARÁMETRO \ MEDIDA	%	ms
Demora	-	
Perdida		
Duplicación		
Corrupción		
Reordenamiento		

(Valores proporcionados por encargado a dirigir la practica)

En la sentencia de comandos que se muestra a continuación sustituimos <valor> en función de la Tabla 6.5.

```
tc qdisc add dev eth1 root netem delay <valor>ms loss <valor>% duplicate <valor>% corrupt <valor>% reorder <valor>%
tc qdisc show
```

6.3 CONFIGURACIÓN NETEM BOX ESCENARIO 2

Tabla 6.6. Valores de parámetros de NetEm Iny2

PARÁMETRO \ MEDIDA	%	ms
Demora	-	
Perdida		
Duplicación		
Corrupción		
Reordenamiento		

(Valores proporcionados por encargado a dirigir la practica)

En la sentencia de comandos que se muestra a continuación sustituimos <valor> en función de la Tabla 6.6.

tc qdisc del dev eth1

tc qdisc add dev eth1 root netem delay <valor>ms loss <valor>% duplicate <valor>% corrupt <valor>% reorder <valor>%
tc qdisc show

Una vez finalizada la práctica se obtendrá los resultados de mediciones con el Analyzer de D-ITG para su posterior análisis.

CAPÍTULO VII

7 CONCLUSIONES Y RECOMENDACIONES

7.1 CONCLUSIONES

- NetEm es una herramienta muy útil para representar los factores que afectan a las transmisiones de redes inalámbricas bajo el protocolo TCP. Proporciona las opciones estadísticas necesarios para emular la respuesta de la red de escenarios reales.

- Con el inyector de tráfico D-ITG se generó el tráfico unidireccional necesario en el lado del emisor, el mismo que atravesó el emulador de canal NetEm, en el cual se configuro cada parámetro con valores y porcentajes altos que alteran el trafico inyectado, al llegar al receptor se pudo conseguir los valores y archivos de mediciones por medio del analyzer que es una parte de DITG, para posteriormente generar las gráficas de Bitrate, Delay y Jitter para su respectivo análisis.

- Las guías prácticas de laboratorio se desarrollaron basadas en la experiencia de investigación e implementación de los diferentes escenarios de pruebas que se realizó con NetEm durante el presente proyecto de tesis.

- Para la implementación del escenario de pruebas donde se ejecutó NetEm no requirió incurrir en gastos mayores, ya que se utilizó software libre y ordenadores existentes, solamente se adquirió una tarjeta de red que se instaló en el ordenador que actuó como caja NetEm para poder realizar el puente de red.

- En el presente proyecto de tesis se realizó un análisis de cada uno de los diferentes parámetros que permite configurar NetEm como así también utilizando todos los parámetros al mismo tiempo, para poder comprender su funcionamiento, no se efectuó emulaciones de casos específicos de comportamiento de alguna red.

7.2 RECOMENDACIONES

- Para poder realizar una correcta inyección de tráfico con D-ITG es necesario que los relojes de los ordenadores se encuentren sincronizados, la sincronización se puede realizar de dos maneras, igualando los relojes de los sistemas operativos Linux en configuraciones o con una conexión a internet mediante un comando de sincronización en un terminal hacia un servidor NTP.

- Es recomendable después de cada configuración que se realice en la caja NetEm digitar el comando *tc qdisc show* para verificar los parámetros introducidos y confirmar a que interfaz de red está afectando.

- Se recomienda para trabajos futuros realizar alguna interfaz gráfica con la ayuda de JAVA, con escenarios pre configurado, que permitan un ambiente más amigable para el usuario.

REFERENCIAS BIBLIOGRÁFICAS

[1] api_user_11797_luis_alejandro013. (s.f.). *Simulador y Emulador*. Recuperado el 15 de Mayo de 2012, de Scribd Inc: http://es.scribd.com/doc/7131728/Simulador-y-Emulador

[10] Lara C., R. (2010). *GUIAS PRACTICAS DE LABORATORIO - DITG.* Sangolquí: ESCUELA POLITECNICA DEL EJERCITO.

[11] Quirós, P. d. (2007). Herramientas de diagnóstico: netcat, netem.

[12] Saldaña, J. M., Murillo, J., Fernández-Navajas, J., Ruiz-Mas, J., Viruete, E. A., & Aznar, J. I. (2010). *EMULACIÓN DE ESCENARIOS DE RED MEDIANTE UN TESTBED.* Zaragoza: Universidad de Zaragoza.

[13] wikipedia. (s.f.). *Transmission Control Protocol.* Recuperado el 23 de Mayo de 2012, de wikipedia.org: http://es.wikipedia.org/wiki/Protocolo_de_Control_de_Transmisi%C3%B3n

[2] CodeAlias. (s.f.). *Netem, tc and qdisc loss and delay emulation examples*. Recuperado el 5 de Junio de 2012, de CodeAlias: http://www.codealias.info/technotes/network_delay_and_packet_loss_emulation_using_netem

[3] die. (s.f.). *tc(8) - Linux man page*. Recuperado el 16 de Enero de 2013, de die.net: http://linux.die.net/man/8/tc

[4] Distler, T. (10 de Junio de 2011). *Netem WAN Emulation: How to Setup a Netem Box*. Obtenido de tdistler.com: http://tdistler.com/2011/06/10/netem-wan-emulation-how-to-setup-a-netem-box#more-973

[5] Hemminger, S. (2005). *Network Emulation with NetEm.* Open Source Development Lab.

[6] Keller, A. (20 de Julio de 2006). *Manual tc Packet Filtering and netem.* Obtenido de http://tcn.hypert.net/tcmanual.pdf

[7] Kioskea. (11 de Abril de 2012). *Protocolo TCP*. Obtenido de kioskea.net: http://es.kioskea.net/contents/281-protocolo-tcp

[8] kioskea. (Abril de 2013). *Elegir una distribución Linux*. Obtenido de Kioskea.net: http://es.kioskea.net/faq/266-elegir-una-distribucion-linux

[9] Kriger, J. (2009). *"Un nuevo protocolo de transporte SCTP-RR".* Universidad de Buenos Aires, Departamento de Computacion, Buenos Aires.

[14] Avallone, S., Guadagno, S., Emma, D., Pescape, A., & Ventre, G. (s.f.). D-ITG Distributed Internet Traffic Generator. Napoli: Universita di Napoli Federico II.

FECHA DE ENTREGA

El proyecto fue entregado al Departamento de Eléctrica y Electrónica y reposa en la Escuela Politécnica del Ejército desde:

Sangolquí, a 5 de septiembre de 2013

ELABORADO POR:

_____

Luis Edison Paredes Caicedo

1803521598

AUTORIDAD:

_____

Ing. Darío Duque

DIRECTOR DE LA CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES