



UNIVERSIDAD DE CHILE  
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS  
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

APLICACIÓN DE NETWORK CALCULUS PARA ANÁLISIS  
DE REDES DE DATOS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

FELIPE ANDRES TORO REYES

PROFESOR GUÍA:  
SR. ALBERTO CASTRO ROJAS

MIEMBROS DE LA COMISIÓN:  
SR. CLAUDIO ESTEVEZ MONTERO  
SR. JUAN PEREZ RETAMALES

SANTIAGO DE CHILE

MAYO 2012

RESUMEN DE LA MEMORIA  
PARA OPTAR AL TÍTULO DE  
INGENIERO CIVIL ELECTRICISTA  
POR: FELIPE TORO REYES  
FECHA: 10/05/2012  
PROF. GUÍA: SR. ALBERTO CASTRO R.

## APLICACIÓN DE NETWORK CALCULUS PARA ANÁLISIS DE REDES DE DATOS

En las redes de datos cada día se desarrollan nuevos servicios y topologías más complejas, con más usuarios y restricciones diferenciadas para los diversos servicios. De esta forma, la estimación de parámetros de red, como por ejemplo, *delay*, *backlog*, tasas máximas de transferencia de datos, factores de ocupación, etc. son de interés para el dimensionamiento de los nodos a utilizar en las redes.

Los flujos de tráfico que atraviesan las redes de computadores están sujetos a restricciones impuestos por los componentes del sistema. Estas restricciones pueden ser expresadas y analizadas con *Network Calculus*, lo que permite entender las propiedades de los distintos servicios entregados por las redes convergentes, para planificar y dimensionar los parámetros de las redes.

El objetivo de este trabajo es aplicar *Network Calculus* en redes de datos y comparar la estimación de parámetros con mediciones reales y cálculos algebraicos. Se utilizan herramientas Open Source para la implementación de escenarios de red que simulan el servicio de video *streaming*. De los resultados que se obtienen de las mediciones de *delay* y pérdidas de paquetes del tráfico generado, se comparan con las estimaciones que resulta de aplicar *Network Calculus* a las curvas de servicio y llegada.

*Network Calculus* resulta ser una herramienta que permite hacer estimaciones del peor caso, es por esto que el *delay* que se obtiene al ser aplicado *Network Calculus* es superior al *delay* medido. No se obtienen pérdidas de paquetes para ver si las estimaciones de *backlog*, que entrega el *Network Calculus*, son las correctas. Esto se debe principalmente a la gran capacidad de transmisión que tienen los *routers*.

Como posibles futuros trabajos tenemos: realizar escenarios de pruebas más complejos; aplicar *Network Calculus* a otros tipos de tráfico; utilizar diversas políticas de encolamiento en los *routers* y una mejora en la estimación en la curva de servicio de los *routers*.

# Agradecimientos

Quiero agradecer a Alberto Castro por permitirme realizar este tema como trabajo de título y además por toda la ayuda que me brindó durante estos meses.

A toda mi familia y a mi mujer Katta, por el apoyo y comprensión durante todos los años de mi vida, y que siempre me motivan a ser una mejor persona.

Agradezco también a la gente del trabajo, en especial a Marcelo Villa por su buena disponibilidad que me entregó en el trabajo, siempre dispuesto a darme flexibilidad en el horario y sus buenos consejos.

# Índice

Introducción.....	1
1.1 Motivación .....	1
1.2 Objetivos.....	1
1.2.1 Objetivo General .....	1
1.2.2 Objetivos Específicos.....	2
1.3 Hipótesis de Trabajo y Metodología.....	2
1.4 Alcances.....	2
1.5 Estructura del Documento .....	3
Antecedentes.....	4
2.1 Modelos de Flujo de Datos .....	4
2.2 <i>Backlog</i> y <i>Delay</i> .....	5
2.3 Fundamento Matemático .....	6
2.3.1 Funciones crecientes <i>wide-sense</i> .....	6
2.3.2 Funciones cóncavas y convexas.....	7
2.3.3 Funciones <i>Star-Shaped</i> .....	8
2.4 Algebra Min-Plus.....	8
2.4.1 Funciones sub-aditivas .....	9
2.4.2 Cierre sub-aditivo.....	9
2.5 Curvas de Llegada .....	11
2.6 Curvas de Servicio .....	12
2.7 Modelo de servicios integrados para <i>routers</i> de Internet.....	13
2.8 Concatenación.....	13
2.9 Límites de Rendimiento.....	15
2.9.1 Teorema Límite de <i>Backlog</i> .....	15

2.9.2 Teorema Límite de <i>Delay</i> .....	15
2.9.3 Teorema Límite de Salida .....	16
2.10 <i>Streaming</i> de video .....	16
2.11 Calidad de servicio <i>QoS</i> en redes de datos .....	17
2.12 Aportes de este Trabajo .....	18
Implementación .....	19
3.1 Objetivos.....	19
3.2 Herramientas de Software Utilizadas .....	21
3.2.1 <i>Flow-tools</i> .....	21
3.2.2 <i>Graphical Network Simulator (GNS3)</i> .....	21
3.2.3 Ubuntu.....	23
3.2.4 Matlab.....	23
3.2.5 <i>VLC Media Player</i> .....	24
3.2.6 VirtualBox.....	24
3.3 Herramienta de Hardware.....	24
3.4 Escenarios de Prueba .....	25
3.4.1 Escenario 1: Medición Tráfico de Entrada <i>router 3</i> .....	25
3.4.2 Escenario 2: Medición de Tráfico de Salida <i>Router 3</i> .....	30
3.4.3 Escenario 3: Medición de <i>Delay</i> en <i>Router 3</i> .....	31
3.4.4 Escenario 4: Medición de <i>Delay</i> Concatenación de Nodos .....	32
Resultados y Discusiones .....	34
4.1 Mediciones Realizadas.....	34
4.1.1 Medición Tráfico de Entrada <i>Router 3</i> .....	34
4.1.2 Medición Tráfico de Salida <i>Router 3</i> .....	42
4.1.3 Medición de <i>Delay</i> .....	43
4.1.4 Medición de <i>Delay</i> concatenación de nodos .....	44

Conclusiones.....	50
5.1 Herramientas.....	50
5.2 Aplicación de <i>Network Calculus</i> .....	50
5.3 Objetivos.....	51
5.4 Proyecciones del trabajo .....	51
Referencias .....	52
6.1 Bibliografía.....	52
6.2 Acrónimos.....	54
Anexos.....	55
7.1 Gráficos de velocidades en los routers.....	55
7.2 Instalación y Configuración Software .....	60
7.2.1 Instalación y Configuración GNS3 .....	60
7.2.2 Instalación y Configuración VirtualBox .....	60
7.2.3 Instalación y Configuración Flow-tools.....	61
7.2.4 Instalación y Configuración CyNC.....	62
7.3 Descripción DVD .....	63

# Índice de Figuras

Figura 1: Ejemplo de funciones de entrada y salida.....	4
Figura 2: Backlog y Delay.....	5
Figura 3: Funciones crecientes de buen comportamiento.....	7
Figura 4: Curva de Llegada.....	11
Figura 5: Curva de Servicio.....	12
Figura 6: Funciones $\lambda_5$ y $\delta_2$ .....	12
Figura 7: Convolución entre $\lambda_5$ y $\delta_2$ .....	13
Figura 8: Concatenación de nodos.....	14
Figura 9: Desviaciones vertical y horizontal entre f y g.....	16
Figura 10: Arquitectura de un sistema streaming.....	17
Figura 11: Delay nodal en el router A.....	19
Figura 12: Interfaz gráfica de GNS3.....	23
Figura 13: Escenario de Prueba 1.....	25
Figura 14: Suavizado streaming de video.....	27
Figura 15: Router 2.....	28
Figura 16: Router 3.....	28
Figura 17: Escenario de Prueba 3.....	31
Figura 18: Ping a Interfaz Ethernet 1/1.....	32
Figura 19: Escenario 4.....	32
Figura 20: Delay en el router 3.....	40
Figura 21: Backlog en el router 3.....	41
Figura 22: Curva de servicio del escenario 4.....	46
Figura 23: Delay en el router 3.....	48
Figura 24: Backlog en el router 3.....	49

# Índice de Tablas

Tabla 1: Direcciones IP Routers.....	26
Tabla 2: Direcciones IP Computadores.....	26
Tabla 3: Características clip de video.....	26
Tabla 4: Mediciones del tráfico de entrada al router 3.....	30
Tabla 5: Descripción de campos.....	30
Tabla 6: Configuración Velocidades Interfaces Routers.....	34
Tabla 7: Tráfico de entrada para las 6 configuraciones.....	35
Tabla 8: Velocidades promedio de entrada y salida router 3.....	37
Tabla 9: Diferencia entre velocidad medida y configurada.....	37
Tabla 10: Polinomio aproximación del tráfico de entrada router 3.....	37
Tabla 11: Curvas de Servicio.....	39
Tabla 12: Curvas de Llegada.....	40
Tabla 13: Delay estimado por Network Calculus.....	41
Tabla 14: Backlog estimado por Network Calculus.....	42
Tabla 15: Tráfico de entrada y salida para los 6 escenarios de velocidades en los routers.....	42
Tabla 16: Pérdidas de paquetes.....	43
Tabla 17: Delay promedio y máximo medido para los 6 escenarios de velocidades.....	44
Tabla 18: Delay promedio y máximo.....	44
Tabla 19: Parámetros de flow-capture.....	62



# Índice de Gráficos

Gráfico 1: Funciones f y g.....	10
Gráfico 2: Convolución entre f y g.....	11
Gráfico 3: Curvas de servicio de nodo 1 y 2.....	14
Gráfico 4: Convolución entre curvas de servicio.....	15
Gráfico 5: Tráfico entrada interfaz Ethernet 1/0 router 3.....	34
Gráfico 6: Velocidad de entrada a la interfaz Ethernet 1/0 del router 3.....	35
Gráfico 7: Tráfico acumulado en la interfaz Ethernet 1/0 del router 3.....	36
Gráfico 8: Tráfico acumulativo en función del tiempo.....	36
Gráfico 9: Curva de servicio router 3.....	38
Gráfico 10: Curva de Llegada del router 3.....	39
Gráfico 11: Delay medido con comando ping.....	43
Gráfico 12: Tráfico entrada router 3 escenario 4.....	45
Gráfico 13: Velocidad de entrada a la interfaz Ethernet 1/0 del router 3.....	45
Gráfico 14: Tráfico acumulado en la interfaz Ethernet 1/0 del router 3.....	46
Gráfico 15: Curva de llegada del router 3.....	47
Gráfico 16: Tráfico entrada router 3 para escenario 1.....	55
Gráfico 17: Velocidad de entrada del router 3 para escenario 1.....	55
Gráfico 18: Tráfico acumulado del router 3 para escenario 1.....	56
Gráfico 19: Tráfico entrada router 3 para escenario 2.....	56
Gráfico 20: Velocidad de entrada del router 3 para escenario 2.....	56
Gráfico 21: Tráfico acumulado del router 3 para escenario 2.....	57
Gráfico 22: Tráfico entrada router 3 para escenario 3.....	57
Gráfico 23: Velocidad de entrada del router 3 para escenario 3.....	57
Gráfico 24: Tráfico acumulado del router 3 para escenario 3.....	58
Gráfico 25: Tráfico entrada router 3 para escenario 4.....	58
Gráfico 26: Velocidad de entrada del router 3 para escenario 4.....	58
Gráfico 27: Tráfico acumulado del router 3 para escenario 4.....	59
Gráfico 28: Tráfico entrada router 3 para escenario 5.....	59
Gráfico 29: Velocidad de entrada del router 3 para escenario 5.....	59
Gráfico 30: Tráfico acumulado del router 3 para escenario 5.....	60

# 1

## Introducción

### 1.1 Motivación

En las redes de datos cada día se desarrollan nuevos servicios y topologías más complejas, con más usuarios y restricciones diferenciadas por servicio. De esta forma, la estimación de parámetros de red, como por ejemplo, *delay*, *backlog*, tasas máximas de transferencia de datos, factores de ocupación, etc. son relevantes para el dimensionamiento de los nodos a utilizar en las redes.

A medida que el tráfico pasa a través de la red está sujeto a restricciones impuestas por los componentes del sistema. Estas restricciones pueden ser expresadas y analizadas con *Network Calculus*, lo que permite entender las propiedades de los distintos servicios entregados por las redes convergentes, para planificar y dimensionar parámetros de red.

Las principales diferencias con las teorías tradicionales de redes de datos son la utilización de operaciones algebraicas, en vez de teoría de colas o estadísticas de red, donde las operaciones son la adición que se convierte en el cálculo del mínimo y la multiplicación que se convierte en adición.

A través del uso de curvas de llegada y curvas de servicio que se basan en parámetros tales como tasas de llegada y tasas de servicio, se puede cuantificar las características del flujo a medida que viaja de un nodo a otro a través de la red.

### 1.2 Objetivos

#### 1.2.1 Objetivo General

El objetivo general de este trabajo de título es aplicar *Network Calculus* en redes de datos y comparar su aplicación en la estimación de parámetros.

De la comparación de los distintos modelos, se entregan recomendaciones de uso, y validación, en distintos escenarios de redes. Se utilizan distintas técnicas y herramientas de simulación de redes de datos, para modelamiento y generación de los escenarios de comparación.

## 1.2.2 Objetivos Específicos

Los objetivos específicos para el desarrollo de este trabajo son los siguientes:

- Revisión de *Network Calculus*, de sus características principales.
- Utilización de la *toolbox* CyNC de Matlab para operar con algebra de *Network Calculus*.
- Uso de software de simulación de red de datos para generar los distintos escenarios de comparación. Siendo los componentes de la red, los nodos y links.
- Comparar los métodos tradicionales de manejo de QoS (*Quality of Service*) con *Network Calculus*.
- Recomendar para qué escenarios *Network Calculus*, permite optimizar los recursos de la red.

## 1.3 Hipótesis de Trabajo y Metodología

Como hipótesis se plantea que utilizando herramientas de código libre es posible desarrollar escenarios de pruebas y aplicar *Network Calculus* en los tráficos que se generan para realizar estimaciones de parámetros de *delay* y *backlog*.

La metodología de trabajo comprende las etapas de revisión de *Network Calculus* y algunas de las aplicaciones en el tráfico de datos. Además se describen los parámetros de tráfico a evaluar con *Network Calculus*.

Después se comparan las diferentes herramientas a ocupar en el desarrollo de los escenarios de prueba, definiendo qué características son de utilidad para el proyecto y cómo se relacionan entre sí para cumplir con los objetivos.

Se continúa con el diseño y construcción de los escenarios de prueba. Con los ensayos realizados en los diferentes escenarios, se analizan los resultados obtenidos de las mediciones y se comparan con las estimaciones de parámetros al aplicar *Network Calculus*.

## 1.4 Alcances

En el presente trabajo se aplica *Network Calculus* al tráfico que se genera al realizar un *streaming* de video que se implementa en una red simulada en base a herramientas de código libre. El uso de *Network Calculus* incluye aplicaciones para problemas de ingeniería de red, tales como: calidad de servicio, reserva de recursos, planificación de capacidades, etc.

## 1.5 Estructura del Documento

El documento se divide en siete capítulos que se describen a continuación:

El capítulo de Introducción entrega un marco general del trabajo realizado, se describen la motivación, los objetivos, la hipótesis de trabajo, la metodología empleada y los alcances de este trabajo de título.

El capítulo de Antecedentes muestra un resumen de información y conceptos sobre *Network Calculus*. El capítulo termina con una descripción de los escenarios de prueba a simular con el aporte del mismo al tema.

El capítulo de Implementación describe detalladamente los escenarios de pruebas y las herramientas utilizadas para su creación. Se explican además todas las partes que componen los escenarios de pruebas, lo que permite comprender adecuadamente los resultados mostrados en el capítulo siguiente.

En el capítulo de Resultados y Discusiones se muestran los datos obtenidos gracias a los escenarios de prueba y los resultados al aplicar *Network Calculus* al tráfico de datos que se produce por el *streaming* de video. Las estimaciones de *Network Calculus* para los parámetros de *delay* y *backlog* se comparan con los valores medidos por las herramientas de monitoreo.

En el capítulo de Conclusiones se analiza el cumplimiento de los objetivos planteados para este trabajo, junto con evaluar las herramientas utilizadas. Además se mencionan las posibles proyecciones de este trabajo.

En el capítulo de Referencias se listan los distintos textos, documentos y recursos multimedia que aportan en el desarrollo del presente trabajo.

Por último, están los Anexos que comprenden los distintos datos que complementan la información presentada en los capítulos anteriores.

# 2

## Antecedentes

En este capítulo se describen temas referentes al *Network Calculus* abarcando conceptos como curvas de servicio, curvas de llegada, *backlog* y *delay*. Además se desarrolla el algebra *mis-plus* que es la base del *Network Calculus*, enfocándose principalmente en la convolución y una serie de funciones de interés para este trabajo. Se describen también aspectos relativos al *streaming* de video y a la calidad de servicio en redes de datos.

### 2.1 Modelos de Flujo de Datos

Los flujos de datos conviene describirlos por medio de funciones acumulativas  $R(t)$ , que se definen como el número de bits del flujo en un intervalo de tiempo  $[0, t]$ . Por convención, se considera  $R(0) = 0$  si no se indica lo contrario [1]. Se consideran los siguientes tipos de modelos:

- Tiempo discreto:  $t \in \mathbb{N} = \{0, 1, 2, 3, \dots\}$ .
- Fluido:  $t \in \mathbb{R}^+ = [0, +\infty)$  y con  $R(t)$  una función continua.
- General, Tiempo continuo:  $t \in \mathbb{R}^+$  y  $R(t)$  es una función continua por la izquierda o derecha.

En la Figura 1 se muestran ejemplos de funciones de entrada a un sistema  $S$ , que puede ser visto como un caja negra, y funciones de salida. Por convención, el signo \* se utiliza para distinguir la función de salida del sistema.  $R_1$  y  $R_1^*$  muestra una función continua o de tiempo continuo.  $R_2$  y  $R_2^*$  muestra continuidad en el tiempo con discontinuidad en los tiempos de llegada de paquetes.  $R_3$  y  $R_3^*$  muestra un modelo de tiempo discreto; el sistema se observa en el tiempo  $0, 1, 2, \dots$ .

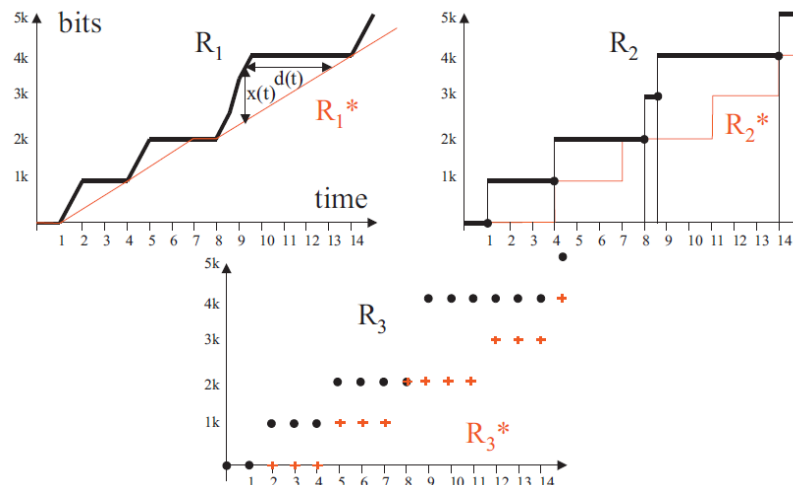


Figura 1: Ejemplo de funciones de entrada y salida.

## 2.2 Backlog y Delay

Un sistema  $S$ , que puede ser visto como una caja negra, recibe datos de entrada descrito por su función acumulativa  $R(t)$  y entrega los datos después de un retraso variable. Se llama  $R^*(t)$  a la función de salida, es decir, la función acumulativa de la salida del sistema  $S$ . El sistema  $S$  puede ser, por ejemplo, un buffer simple que trabaja a una tasa constante, un nodo de comunicación complejo o incluso una red completa [1]. De las funciones de entrada y salida se derivan dos parámetros de interés para este trabajo: *delay* y *backlog*.

El *backlog* en el tiempo  $t$  es  $R(t) - R^*(t)$  y representa la cantidad de bits que están dentro del sistema; si el sistema es un *buffer* único, es el largo de la cola. Si el sistema es más complejo, el *backlog* es el número de bits en tránsito, asumiendo que se puede observar la entrada y salida simultáneamente.

El retardo virtual en el tiempo  $t$  es  $d(t) = \inf \{ \tau \geq 0 : R(t) \leq R^*(t + \tau) \}$  significa el retraso que puede ser experimentado por un bit llegando al tiempo  $t$  si todos los bits recibidos previos son servidos antes.

En la Figura 2 se muestra el resultado que se obtiene de utilizar la función *rtcploth* de la *toolbox* de Matlab [4]. El *backlog* es mostrado como la desviación vertical entre las funciones de entrada  $R_1(t)$  y salida  $R_1^*(t)$ ; el *delay* es la desviación horizontal.

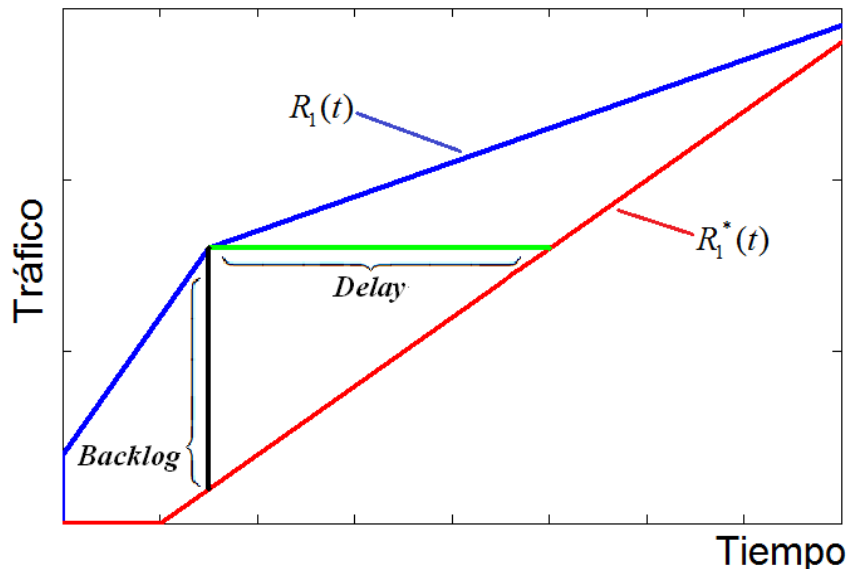


Figura 2: Backlog y Delay.

## 2.3 Fundamento Matemático

Un conjunto de funciones importantes del algebra *min-plus* [1] [2] es:

- Funciones crecientes *wide-sense*.
- Funciones sub-aditivas.
- Funciones convexas y cóncavas.

### 2.3.1 Funciones crecientes *wide-sense*

Una función  $f$  es creciente de buen comportamiento si y sólo si  $f(s) \leq f(t)$  para todo  $s \leq t$ . Se denota por  $F$  al conjunto de secuencias crecientes de buen comportamiento o funciones tal que  $f(t) = 0$  para  $t < 0$ . Las siguientes funciones son de buen comportamiento y de interés para el desarrollo del trabajo:

- Función *peak rate*  $\lambda_R$

$$\lambda_R(t) = \begin{cases} Rt & \text{si } t > 0 \\ 0 & \text{otro caso} \end{cases} \quad (2.1)$$

para algún  $R \geq 0$  (la tasa).

- Función *burst-delay*  $\delta_T$

$$\delta_T = \begin{cases} +\infty & \text{si } t > T \\ 0 & \text{otro caso} \end{cases} \quad (2.2)$$

para algún  $T \geq 0$  (el retraso).

- Función *rate-latency*  $\beta_{R,T}$

$$\beta_{R,T} = R[t-T]^+ = \begin{cases} R(t-T) & \text{si } t > T \\ 0 & \text{otro caso} \end{cases} \quad (2.3)$$

para algún  $R > 0$  (la tasa) y  $T > 0$  (el retraso).

- Función afín  $\gamma_{r,b}$

$$\gamma_{r,b}(t) = \begin{cases} rt + b & \text{si } t > 0 \\ 0 & \text{otro caso} \end{cases} \quad (2.4)$$

para algún  $r \geq 0$  (la tasa) y  $b \geq 0$  (la rafaga).

- Función *step*  $v_T$

$$v_T(t) = 1_{\{t > T\}} = \begin{cases} 1 & \text{si } t > T \\ 0 & \text{otro caso} \end{cases} \quad (2.5)$$

para algún  $T > 0$ .

- Función *staircase*

$$v_{T,\tau}(t) = \begin{cases} \left\lceil \frac{t+\tau}{T} \right\rceil & \text{si } t > 0 \\ 0 & \text{en otro caso} \end{cases} \quad (2.6)$$

De la combinación de todas estas funciones básicas, representadas en la Figura 3 con la ayuda *Matlab*, se pueden obtener funciones más generales, lineales por trozos.

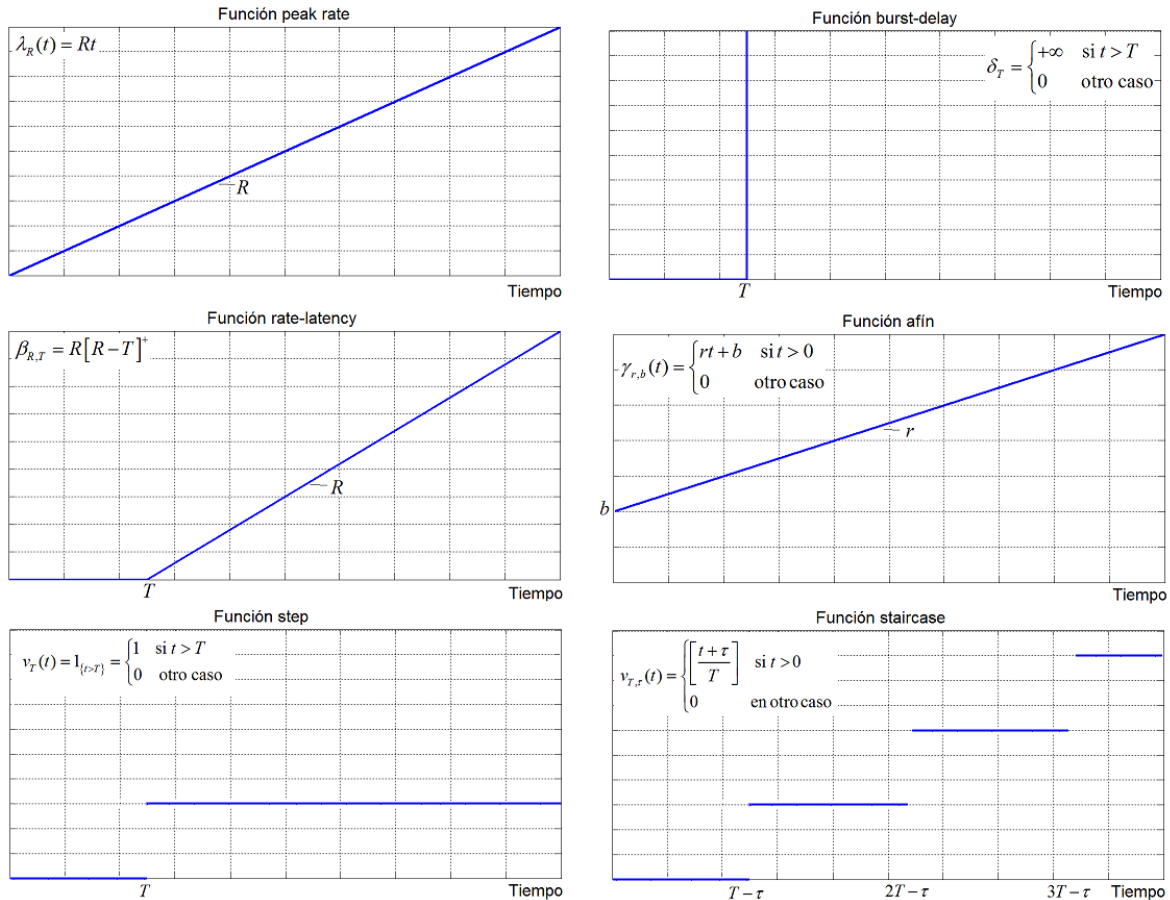


Figura 3: Funciones crecientes de buen comportamiento.

### 2.3.2 Funciones cóncavas y convexas

Una importante clase de funciones en el cálculo *min-plus* son las funciones cóncavas y convexas, que se definen a continuación. Sea  $u$  un real tal que  $0 \leq u \leq 1$ , se tiene

- Subconjunto  $S \subseteq R^n$  es convexo si y solo si  $ux + (1-u)y \in S$  para todo  $x, y \in S$
- Función  $f$  de un subconjunto  $D \subseteq R^n$  a  $R$  es convexa si y solo si  $f(ux + (1-u)y) \leq uf(x) + (1-u)f(y)$  para todo  $x, y \in D$
- Función  $f$  de un subconjunto  $D \subseteq R^n$  a  $R$  es cóncava si y solo si  $-f$  es convexa.



### 2.3.3 Funciones *Star-Shaped*

Chang [15] introduce funciones *start-shaped*, las cuales se definen como sigue:

Una función  $f \in F$  es *star-shaped* si y solo si  $f(t)/t$  es de buen comportamiento decreciente para todo  $t > 0$ . Los siguientes dos teoremas se utilizan por el *Network Calculus*:

**Teorema 1:** Sea  $f$  y  $g$  dos funciones *star-shaped*. Entonces  $h = f \wedge g$  es también *star-shaped* [1], en donde  $\wedge$  es el mínimo.

**Teorema 2:** Funciones cóncavas son funciones *star-shaped* [1].

### 2.4 Algebra Min-Plus

En la siguiente sección se resume una parte del algebra *min-plus*. En [1] se puede encontrar un detalle de todos los teoremas y sus demostraciones.

En *Network Calculus* se aplica teoría de sistemas para el problema de análisis de tráfico de flujo dentro de redes. Cuando la teoría de sistemas se usa para analizar circuitos electrónicos, el algebra convencional  $(R, +, \times)$  puede ser usada. Sin embargo, cuando se analiza el flujo de redes, se adopta el algebra *min-plus*. Con el algebra *min-plus*, la adición se reemplaza por el *min* y el producto por el *plus*. En el algebra convencional, la señal de salida  $y(t) \in R$  de un circuito es la convolución de una señal de entrada  $x(t) \in R$  y la respuesta al impulso del circuito  $y(t) \in R$ . Así, la convolución  $y * x$ , es dada por:

$$(x * y)(t) = \int_{0 \leq s \leq t} x(s)y(t-s)d(s) \quad (2.7)$$

En el algebra *min-plus*, el operador convolución es:

$$(f \otimes g)(t) = \min_{0 \leq s \leq t} [f(s) + g(t-s)] \quad (2.8)$$

La operación dual de la convolución *min-plus* es la deconvolución *min-plus*, la cual se define como sigue:

$$(f \oslash g)(t) = \sup_{u \geq 0} [f(t+u) + g(u)] \quad (2.9)$$

Donde  $f$  y  $g$  son no-negativas, funciones crecientes de *buen comportamiento*. El algebra *min-plus* es la *dioid*:  $(\{R \cup \infty\}, \oplus, \otimes)$ , la cual es una estructura algebraica conmutativa con las siguientes propiedades:

- Conmutatividad

$$a \oplus b = b \oplus a$$

$$a \otimes b = b \otimes a$$

- Asociatividad

$$(a \oplus b) \oplus c = a \oplus (b \oplus a)$$

$$(a \otimes b) \otimes c = a \otimes (b \otimes a)$$

- Distributividad

$$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$$

Operaciones con matrices también se pueden realizar en el algebra *min-plus*. Así, para la operación  $P \oplus Q$ , se tiene:

$$\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \oplus \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \begin{pmatrix} \min[p_1, q_1] \\ \min[p_2, q_2] \end{pmatrix} \quad (2.10)$$

Para la operación  $P \otimes Q$ , se tiene:

$$\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \otimes \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} = \begin{pmatrix} \min[p_1 + m_{11}, p_1 + m_{12}] \\ \min[p_2 + m_{21}, p_2 + m_{22}] \end{pmatrix} \quad (2.11)$$

### 2.4.1 Funciones sub-aditivas

La sub-aditividad es una propiedad importante en *Network Calculus* [1][2]. Una función o secuencia de  $f$  es sub-aditiva si y solo si  $\forall s \leq t$  y  $t \geq 0$ :

$$f(t+s) \leq f(t) + f(s) \quad (2.12)$$

Esta definición es equivalente a imponer que  $f \leq f \otimes f$ . Si  $f(0) = 0$ , es igual a atribuir que  $f \otimes f = f$ .

### 2.4.2 Cierre sub-aditivo

Dada una función  $f \in F$ , si  $f(0) = 0$ , entonces  $f \geq f \otimes f \geq 0$ . Al repetir esta operación, se consigue una secuencia de funciones que son cada vez más pequeñas y converge a una función limitadora, que es la mayor función menor a  $f$  y cero en  $t=0$ . Se le llama cierre sub-aditivo de  $f$ . Un definición más formal es la siguiente:

**Definición cierre sub-aditivo:** Sea  $f$  una función o secuencia de  $F$ . Se denota por  $f^{(n)}$  a la función que se obtiene al repetir  $(n-1)$  convoluciones de  $f$  a sí misma. Por convención,  $f^{(0)} = \delta_0$ , por lo tanto  $f^{(1)} = f$ ,  $f^{(2)} = f \otimes f$ , etc. Entonces el cierre sub-aditivo de  $f$ , denotado por  $\bar{f}$ , se define como

$$\bar{f} = \delta_0 \wedge f \wedge (f \otimes f) \wedge (f \otimes f \otimes f) \wedge f \dots = \inf_{n \geq 0} \{f^{(n)}\} \quad (2.13)$$

Una aplicación del álgebra *min-plus* es en funciones o secuencias. Para el siguiente ejemplo se tienen las funciones  $f$  y  $g$  definidas como:

$$f(t) = \begin{cases} 5(t-8)+12 & \text{si } 8 < t \\ 1.5t & \text{si } 0 \leq t \leq 8 \\ 0 & \text{si } t < 0 \end{cases} \quad (2.14)$$

$$g(t) = \begin{cases} 7.5(t-11)+20.5 & \text{si } 11 < t \\ 3(t-5)+2.5 & \text{si } 5 < t \leq 11 \\ 0.5t & \text{si } 0 \leq t \leq 5 \\ 0 & \text{si } t < 0 \end{cases} \quad (2.15)$$

Funciones que se definen en *Matlab* con la *toolbox* CyNC [4]. En el Gráfico 1 se puede ver las funciones  $f$  y  $g$ . La función  $f$  está compuesta por dos segmentos de línea, el primer trozo tiene una pendiente  $p_2 = 1.5[\text{kbyte}/\text{seg}]$  hasta el instante  $T_2 = 8\text{seg.}$  para luego cambiar a una pendiente  $p_4 = 5[\text{kbyte}/\text{seg}]$ . La función  $g$  está formada por 3 segmentos de rectas, el primero tiene una pendiente  $p_1 = 0.5[\text{kbyte}/\text{seg}]$  hasta el instante  $T_1 = 5\text{seg.}$ ; luego la pendiente cambia a  $p_3 = 3[\text{kbyte}/\text{seg}]$  hasta el instante  $T_1 + T_3 = 5 + 6 = 11\text{seg.}$ ; el tercer trozo tiene una pendiente  $p_5 = 7.5[\text{kbyte}/\text{seg}]$  hasta el instante  $T_4 = 30\text{seg.}$

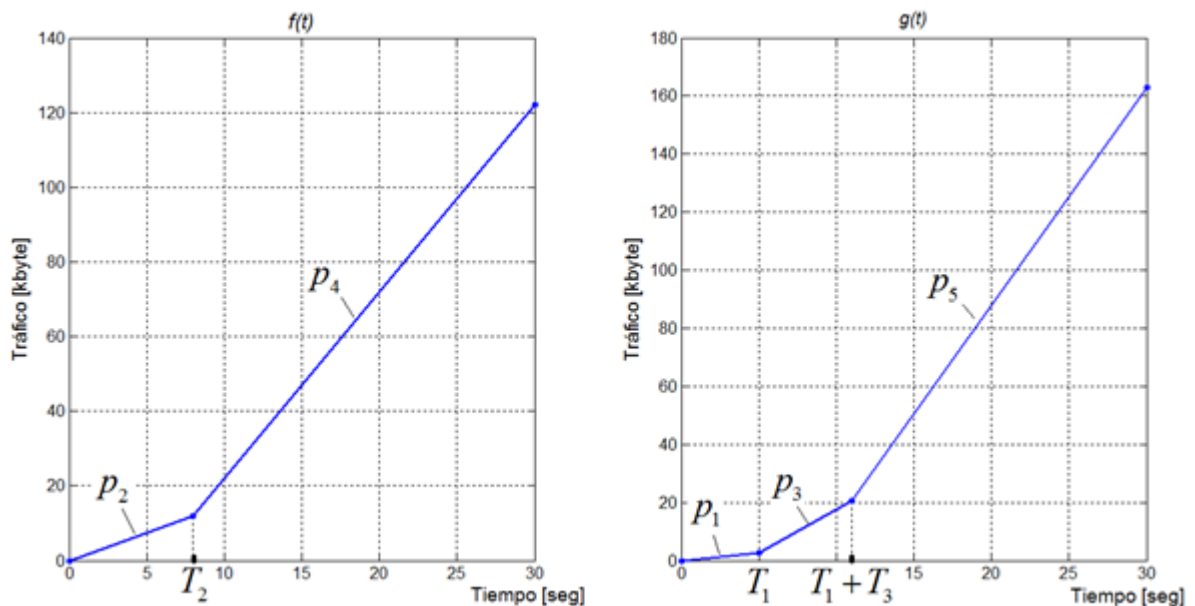


Gráfico 1: Funciones  $f$  y  $g$ .

En el Gráfico 2 se muestra el resultado de aplicar la convolución entre las funciones  $f$  y  $g$ . Esta nueva función está compuesta de 4 trozos de rectas en orden ascendente de acuerdo a la pendiente de estas.

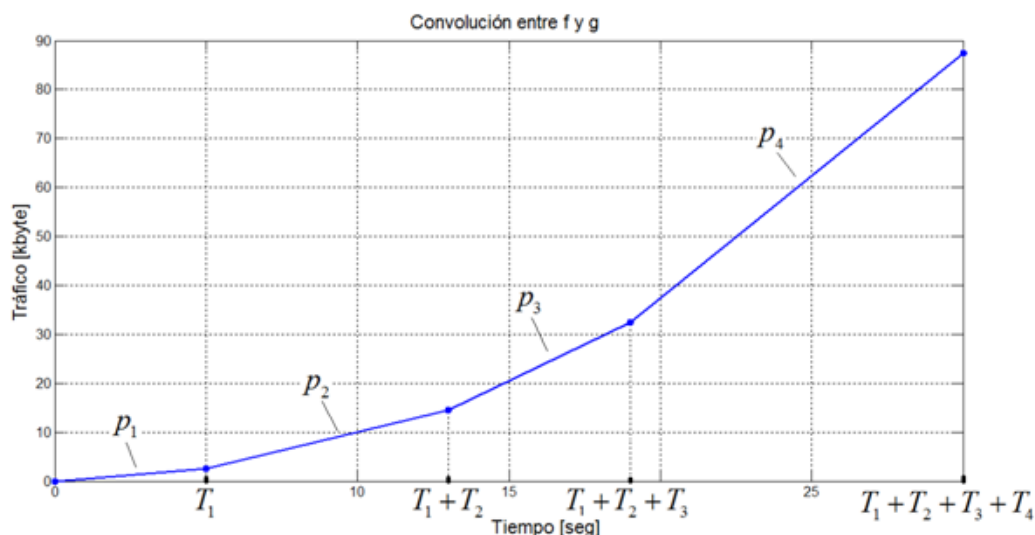


Gráfico 2: Convolución entre  $f$  y  $g$ .

## 2.5 Curvas de Llegada

Con los servicios integrados en redes (por ejemplo ATM o servicios integrados de internet) se necesita limitar el tráfico enviado por las fuentes, esto se hace utilizando el concepto de curvas de llegada [1].

Dada una función  $\alpha$  creciente de *buen comportamiento* definida para  $t \geq 0$ , se dice que un flujo  $R(t)$  está limitado por  $\alpha$  sí y sólo sí para todo  $s \leq t$ :

$$R(t) - R(s) \leq \alpha(t - s) \quad (2.16)$$

Es equivalente a imponer que  $R = R \otimes \alpha$ , es decir:

$$R = \min_{0 \leq s \leq t} (R(s) + \alpha(t - s)) \quad (2.17)$$

Se dice que  $R(t)$  tiene  $\alpha(t)$  como curva de llegada, o también que  $R(t)$  está  $\alpha$ -suavizada. En la Figura 4 se muestra esta definición en donde se puede apreciar que el tráfico  $R(t)$  en todo momento está restringido por la curva  $\alpha(t)$ . Para  $t=0$  se tiene que  $s=0$ , lo que implica  $0 \leq \alpha(0) * 0$  según 2.16, por lo que no es necesario que  $\alpha(0) = 0$ .

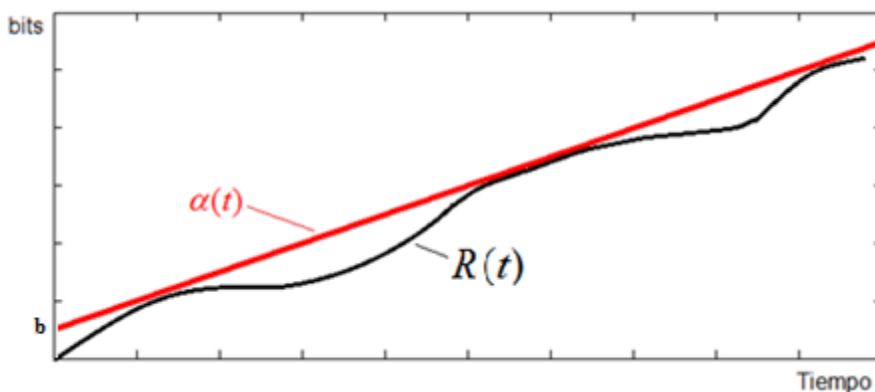


Figura 4: Curva de Llegada.

## 2.6 Curvas de Servicio

Con el objetivo de proporcionar reservación, los nodos de red necesitan ofrecer garantías a los flujos. Esto es hecho por los programadores de paquetes [16]. Un sistema  $S$  ofrece una curva de servicio  $\beta$  si y solo si  $\beta$  es de buen comportamiento creciente,  $\beta(0) = 0$  y  $R^* \geq R \otimes \beta$ , que es equivalente a

$$R^*(t) \geq \inf_{s \leq t} (R(s) + \beta(t-s)) \quad (2.18)$$

En la Figura 5 se observa como el tráfico de salida  $R^*(t)$  tiene una tasa de servicio mínima garantizada que se representa mediante la curva de servicio  $\beta$ .

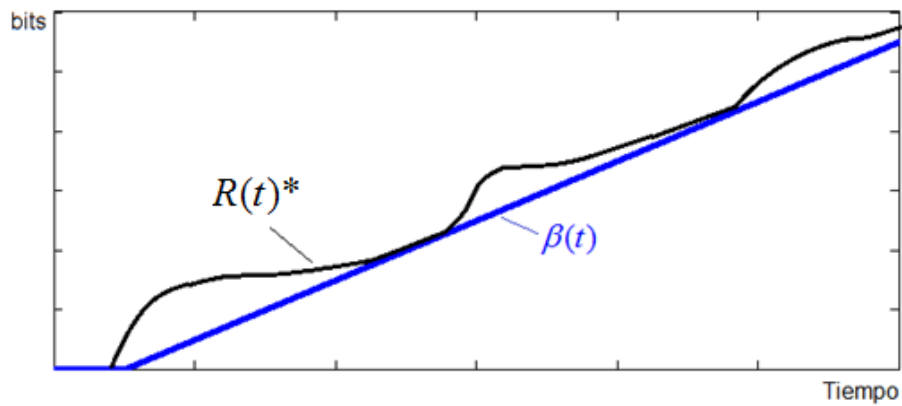


Figura 5: Curva de Servicio.

A modo ilustrativo, veamos el siguiente caso en que se considera un enlace con una tasa de bits constante (CBR), el cual garantiza que un flujo recibe una tasa de servicio  $R = 5 \text{ bit / seg}$ , que puede ser representada utilizando una función *peak rate*  $\lambda_5$ . Se supone también que el nodo introduce para el enlace CBR un *delay* máximo de  $2 \text{ seg}$ . Se utiliza la función *burst-delay*  $\delta_2$  para representar el *delay* del nodo. En la Figura 6 se muestran estas dos funciones  $\lambda_5$  y  $\delta_2$ .

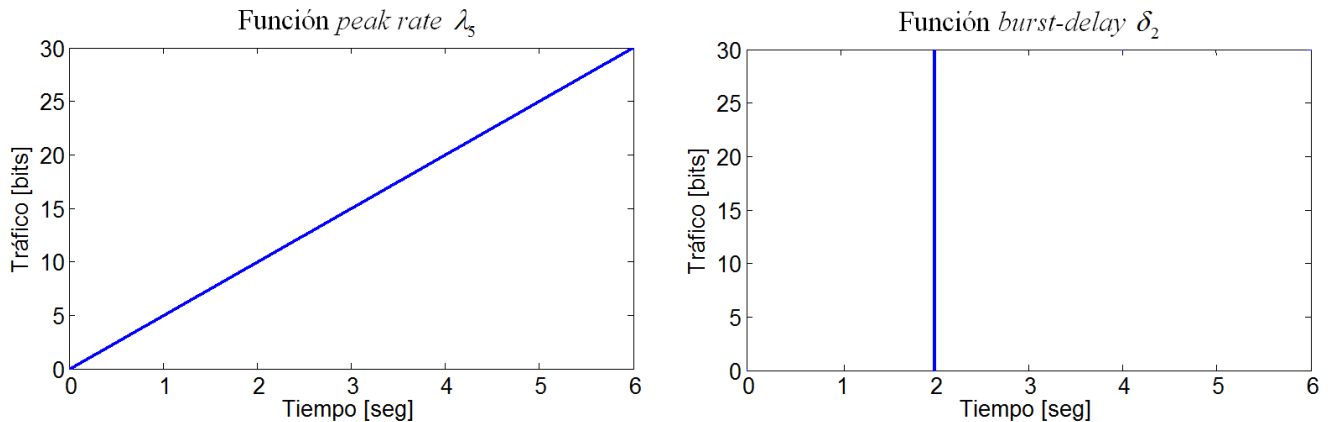


Figura 6: Funciones  $\lambda_5$  y  $\delta_2$ .

La curva de servicio general está dada por la convolución de las dos curvas de servicio:  $\beta = \lambda_5 \otimes \delta_2$ . En la Figura 7 se muestra la curva de servicio resultante, que es equivalente a la función *rate-latency*  $\beta_{5,2}$ .

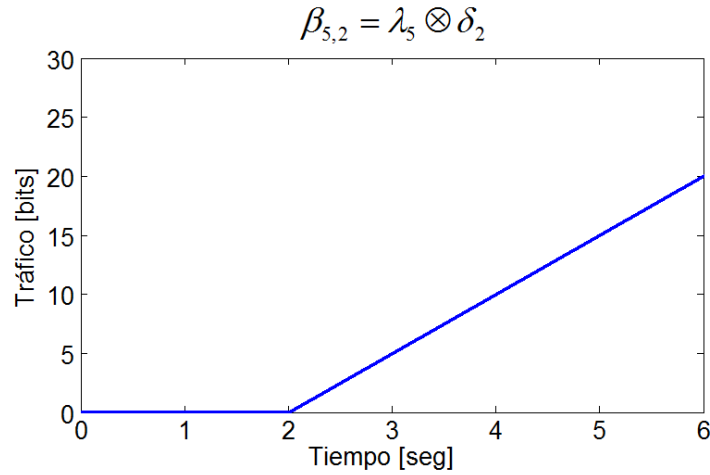


Figura 7: Convolución entre  $\lambda_5$  y  $\delta_2$ .

## 2.7 Modelo de servicios integrados para *routers* de Internet

La fase de reservación asume que todos los *routers* pueden exportar sus características utilizando un modelo simple. El modelo se basa en la idea que un *router* implementa una aproximación práctica de un nodo GPS (*Generalized Processor Sharing*), tal como un PGPS (*Packet Generalized Processor Sharing*), o de forma más general un nodo GR (*Guaranteed Rate*). La curva de servicio que ofrece a un flujo un *router* que implementa GR es una función *rate-latency*, con tasa  $R$  y latencia  $T$  que relacionan de la forma:

$$T = \frac{C}{R} + D \quad (2.19)$$

En donde  $C$  es el tamaño de paquete máximo para el flujo y  $D=L/r$ , donde  $L$  es el tamaño máximo de paquetes en el *router* de todos los flujos que lo atraviesan, y  $r$  es la tasa total de los paquetes repartidos. Este es el modelo que se define para un nodo de Internet [17].

## 2.8 Concatenación

En *Network Calculus* hay un teorema que permite obtener una curva de servicio para un flujo que atraviesa una serie de nodos en una red.

**Teorema Concatenación de nodos** [1]. Se tiene un flujo que atraviesa los sistemas  $S_1$  y  $S_2$  en secuencia.  $S_1$  ofrece una curva de servicio  $\beta_1$  y  $S_2$  a  $\beta_2$ . Luego la concatenación de los dos sistemas ofrece una curva de servicio  $\beta_1 \otimes \beta_2$  para el flujo.

En el siguiente ejemplo se ve el resultado de aplicar el teorema de concatenación de nodos. En la Figura 8 se tiene una red compuesta por dos nodos ofreciendo cada uno una curva de servicio *rate-latency*, una fuente generadora de tráfico y un receptor. El nodo uno introduce un *delay* al tráfico  $R_1$  resultando un nuevo tráfico  $R_2$ . Luego este flujo atraviesa el nodo 2 que produce otro *delay* provocando un tráfico  $R_3$

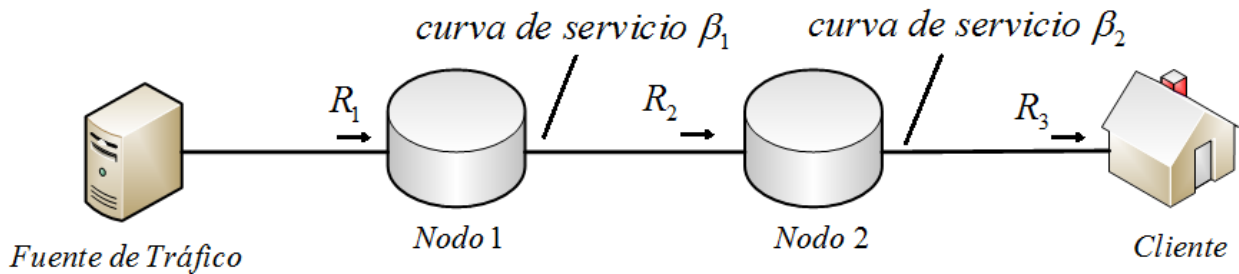


Figura 8: Concatenación de nodos.

El nodo uno tiene una curva de servicio  $\beta_1(t) = \beta_{2,4}(t) = 2[t-4]^+$  y el nodo dos  $\beta_2(t) = \beta_{3,2}(t) = 5[t-2]^+$  como se muestra en el Gráfico 3.

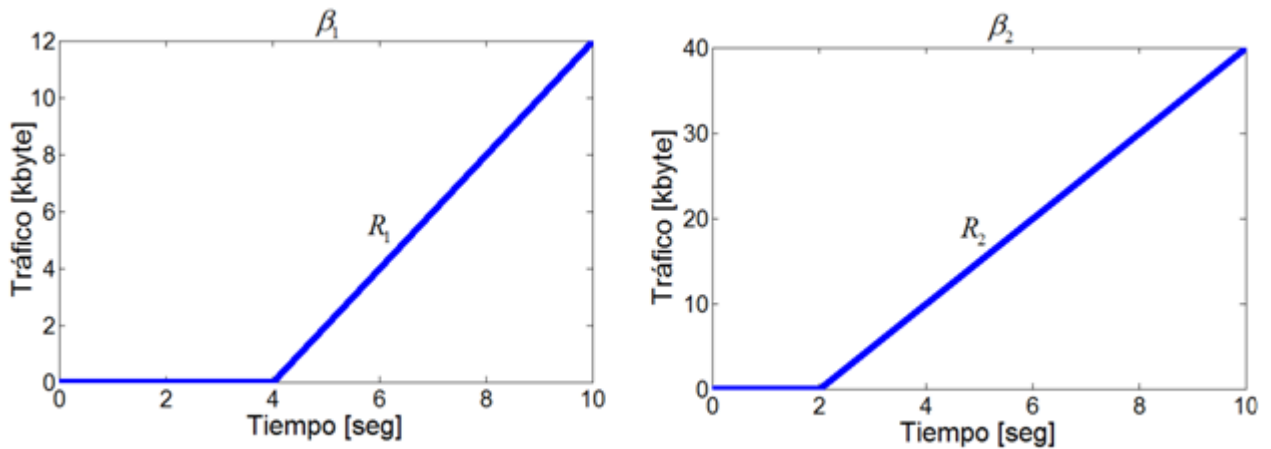


Gráfico 3: Curvas de servicio de nodo 1 y 2.

En el Gráfico 4 se muestra el resultado de la convolución entre estas dos curvas de servicio. Como se puede ver, la concatenación adiciona una latencia igual a la suma de latencias de las curvas de servicio y toma como tasa de servicio la pendiente de menor magnitud.

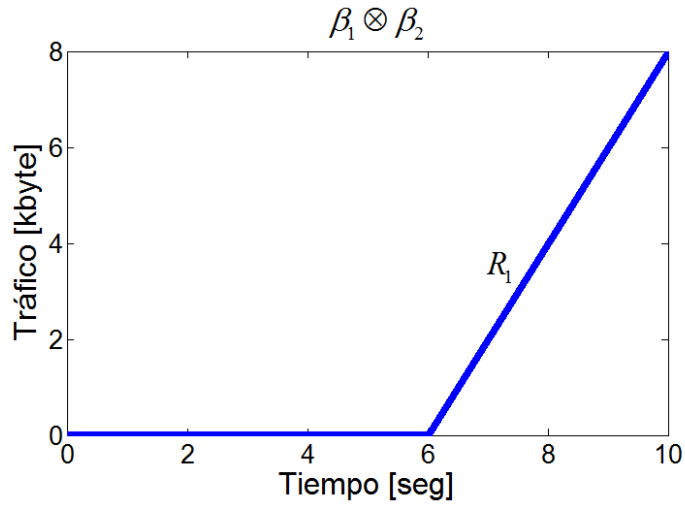


Gráfico 4: Convolución entre curvas de servicio.

## 2.9 Límites de Rendimiento

Un resultado simple al aplicar *Network Calculus*, pero de importancia para este trabajo, son los límites que se obtienen para un sistema con garantía de servicio. Estos límites se muestran en los siguientes teoremas [1].

### 2.9.1 Teorema Límite de *Backlog*

Se asume un flujo, restringido por la curva de llegada  $\alpha$ , que atraviesa un sistema que ofrece una curva de servicio  $\beta$ . El *backlog*  $R(t) - R^*(t)$  para  $t$  satisface:

$$R(t) - R^*(t) \leq \sup_{s \geq 0} \{ \alpha(s) - \beta(s) \} \quad (2.20)$$

### 2.9.2 Teorema Límite de *Delay*

Se asume un flujo, restringido por la curva de llegada  $\alpha$ , que atraviesa un sistema que ofrece una curva de servicio  $\beta$ . El *delay*  $d(t)$  para todo  $t$  satisface:  $d(t) \leq h(\alpha, \beta)$ . En donde  $h$  es la desviación horizontal definida como:

$$h(f, g) = \sup_{t \geq 0} \{ \inf \{ d \geq 0 \text{ tal que } f(t) \leq g(t + d) \} \} \quad (2.21)$$

En la Figura 9 se muestra las desviaciones vertical y horizontal entre las funciones  $f$  y  $g$ .



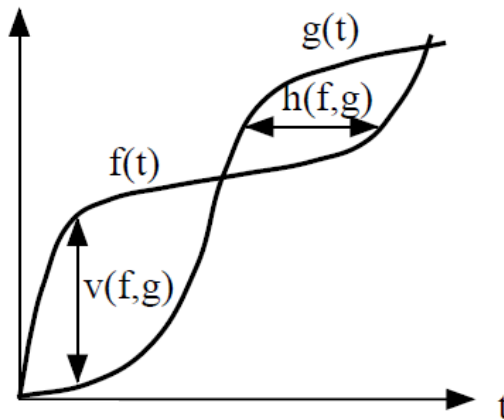


Figura 9: Desviaciones vertical y horizontal entre  $f$  y  $g$ .

### 2.9.3 Teorema Límite de Salida

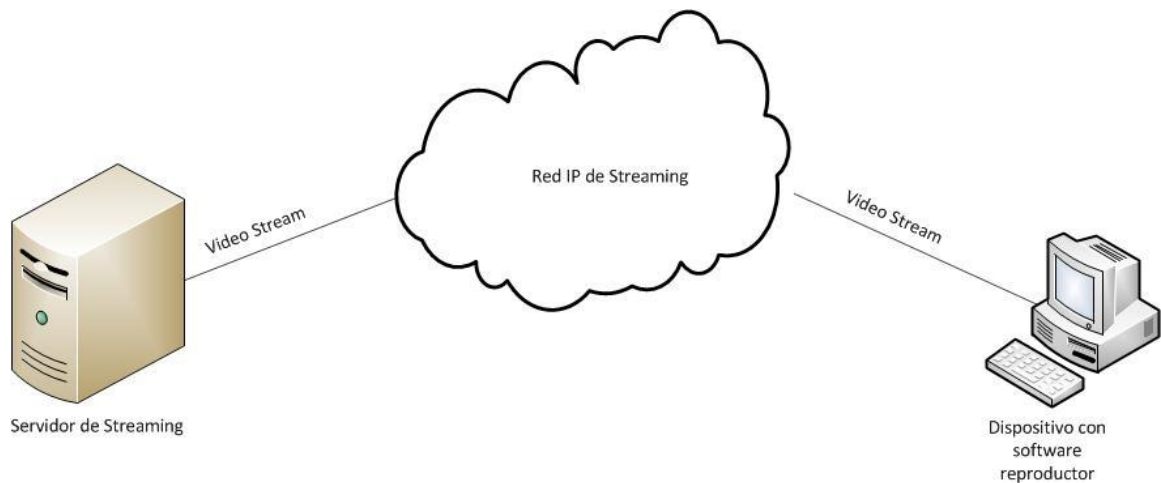
Se asume un flujo, restringido por la curva de llegada  $\alpha$ , que atraviesa un sistema que ofrece una curva de servicio  $\beta$ . El flujo de salida está restringido por la curva de llegada  $\alpha^* = \alpha \oslash \beta$ .

### 2.10 Streaming de video

El *streaming* es una técnica de distribución de video que no necesita la descarga completa del archivo hacia el computador para poder visualizarlo. Esto quiere decir que el video se reproduce a medida que se va recibiendo y luego se descarta. Antes de esta tecnología, la reproducción de contenido multimedia en Internet necesitaba de la descarga completa del archivo al computador.

Como los archivos de video son generalmente de gran tamaño, la descarga y acceso son procesos lentos. Sin embargo, mediante el *streaming* un archivo se puede descargar y reproducir al mismo tiempo, de forma tal que el tiempo de espera se reduce.

En la Figura 10 se muestra una arquitectura de un sistema *streaming* típico. Un elemento fundamental es el servidor de *streaming*, el cual tiene la función de entregar el video cuando se requiere. Otro elemento importante es el software reproductor que recibe el tráfico de video entrante desde la red IP y reproduce la imagen en el computador del usuario. Otro elemento del sistema es la red de transporte entre el servidor y el dispositivo de visualización.



**Figura 10: Arquitectura de un sistema *streaming***

## 2.11 Calidad de servicio *QoS* en redes de datos

La calidad de servicio corresponde a la medida de rendimiento para una red de transmisión que refleja su calidad de transmisión y disponibilidad de servicio [18].

Existen parámetros de calidad de servicio en una red IP que se encuentran estandarizados: retardos, pérdidas de paquetes, ancho de banda disponible y *jitter*. Si un flujo de datos IP cumple con determinados valores de estos parámetros, el flujo recibe la calidad de servicio configurada.

Debido a que las redes se pueden congestionar, es necesario establecer mecanismos de calidad de servicio ya que no es posible sobredimensionar todos los enlaces de una red. Hay algunas formas de proporcionar calidad de servicio:

- Limitar el tráfico inyectado por los usuarios en la red.
- Controlar la cantidad de usuarios que acceden a la red.
- Políticas de encolamiento en los *routers* intermedios.

Un *router* permite implementar mecanismos de calidad de servicio en una red. Los mecanismos de QoS de un *router* son: clasificación, marcado, manejo de colas, administración de tráfico, políticas de tráfico, eliminación de paquetes y eficiencia de enlace (compresión de datos reduciendo el tamaño de los *frames* a ser transmitidos). Estos mecanismos se utilizan dentro de la administración de redes donde se necesita garantizar la entrega de los datos que dependen del tipo de servicio que se entrega.

## 2.12 Aportes de este Trabajo

En la teoría de colas tradicional los tráficos que arriban a los nodos puede ser expresados como procesos estocásticos. En *Network Calculus*, un flujo de llegada es caracterizado por funciones, llamadas curvas de llegada, las cuales definen un límite superior sobre el tráfico acumulado. Por otro lado, los recursos de la red también pueden ser expresados por funciones, las curvas de servicio.

El aporte de este trabajo de título es aplicar *Network Calculus* al tráfico que se produce al realizar un *streaming* de video en una red simulada, en base a código libre, y derivar métricas de QoS, tales como *delay* y *backlog*.

# 3

## Implementación

### 3.1 Objetivos

El objetivo principal del trabajo es medir el tráfico que se produce al realizar un *streaming* de video en una red. Para lograr esto, se crean distintos escenarios de prueba y se monitorea el tráfico en distintos puntos de la red.

Las métricas que se utilizan para comparar la estimación de parámetros al aplicar *Network Calculus* a las curvas de servicio y llegada son: el **delay** y la **pérdida de paquetes**.

Una definición del *delay*, según [10], [11] y [12], es el tiempo que transcurre entre que la primera parte (el primer bit) o un objeto (un paquete) pasa por un punto de observación y el tiempo en que la última parte (el último bit) u objeto (un paquete de respuesta) pasa por el punto de observación. Mientras un paquete va desde un nodo (*host o router*) hasta el nodo subsiguiente (*host o router*) a lo largo de su recorrido, el paquete sufre diferentes tipos de *delay* en cada nodo. Estos *delay* son: *delay* de procesamiento, *delay* de cola, *delay* de transmisión y *delay* de propagación [14].

Se grafican los *delays* en el contexto de red de la Figura 11. Como parte de su ruta terminal a terminal entre la fuente y destino, un paquete se envía desde el nodo anterior a través del router A al router B. El objetivo es categorizar el *delay* nodal en el router A.

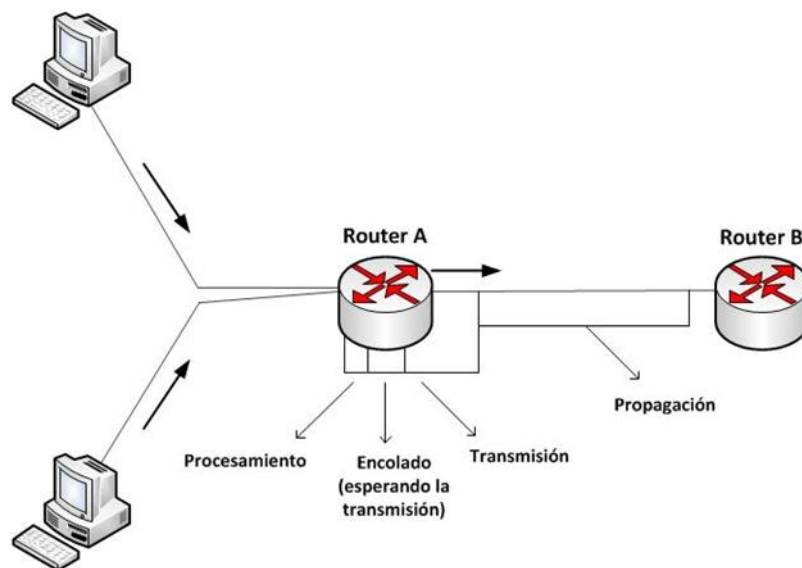


Figura 11: *Delay* nodal en el router A.

- *Delay* de procesamiento: es el tiempo requerido para examinar la cabecera del paquete y determinar hacia donde debe dirigirse. Los retardos de procesamiento en *routers* de elevada velocidad están en el orden de los microsegundos o menos. Después del procesamiento nodal, el *router* dirige el paquete a la cola que precede al enlace hacia el *router* B.
- *Delay* de cola: es el tiempo que el paquete espera en la cola para ser transmitido en el enlace. El *delay* de un paquete específico va a depender del número de paquetes que hayan llegado anteriormente y que están encolados y esperando la transmisión sobre el enlace.
- *Delay* de transmisión: Es la cantidad de tiempo que se requiere para transmitir todos los bits del paquete en el enlace. Por ejemplo, si la longitud del paquete es  $L$  bits, y la tasa de transmisión del enlace desde el *router* A al *router* B es  $R$  bits/seg, el *delay* de transmisión es  $L/R$ .
- *Delay* de propagación: Es el tiempo necesario para propagarse desde el comienzo del enlace hasta el *router* B. El bit se propaga a la velocidad de propagación del enlace. Esta velocidad depende del medio del enlace (fibra óptica, cable de cobre de par trenzado, etc.) y está en el rango de la velocidad de la luz o un poco menor. El *delay* de propagación es igual a la distancia entre los dos *routers* dividida por la velocidad de propagación del enlace.

El *delay* que se va a medir en las pruebas realizar corresponde a la suma de *delay* de procesamiento, *delay* de cola y *delay* de transmisión. Como las pruebas se realizan en una red en la cual se emulan los *routers*, la distancia entre estos no se considera como influyente en el *delay* de propagación.

Para medir el *delay* se realiza una monitorización activa, la cual consiste en probar directamente las propiedades de la red generando el tráfico necesario para realizar la medida. Esto permite utilizar métodos de análisis mucho más directos, pero también presenta el problema de que el tráfico introducido puede tener un impacto negativo en las prestaciones recibidas por otros tipos de tráfico.

Para las simulaciones se utilizan paquetes pequeños de prueba y se envían en ciertos intervalos de tiempo (ping), lo cual puede tener las siguientes desventajas:

- El tráfico extra puede ser despreciable, pero la calidad de servicio que se obtienen desde el paquete de prueba no es igual a las experimentadas por el usuario que recibe el *streaming* de video.
- Puede ser catalogado como tráfico hostil o intento de ataque. Por ejemplo, algunos *routers* rechazan tráfico ICMP o limitan su tasa, por si se trata de un intento de *spoofing*.

La otra métrica a medir es la pérdida de paquetes, que es medida como un porcentaje de los paquetes transmitidos. Son los paquetes que siendo enviados nunca llegan a su destino.

Los motivos que originan una pérdida de paquetes son múltiples, siendo la congestión la causa principal. Concretamente, el límite de capacidad de los buffers en los dispositivos de red es el punto donde se registra el problema. A esto se suma, la retransmisión realizada por las aplicaciones cuando detectan que los paquetes no llegan.

Una alternativa para evitar esta pérdida de paquetes es disponer de una velocidad de salida de datos mayor que la de entrada. Esta opción es en la mayoría de los casos inviable por el costo que supone. Siendo así, las medidas a tomar son más propias de la naturaleza de las comunicaciones, proponiendo utilizar UDP en vez de TCP, o incluso observando que algunas aplicaciones son tolerantes con un cierto porcentaje de pérdida de paquetes.

Debido a que una de las restricciones del *Network Calculus* es que la velocidad de salida de un *router* es mayor a la velocidad de entrada, no se provocan pérdidas de paquetes ya que no se tiene congestión. Lo que se hace para causar pérdidas de paquetes es realizar cambios en las tablas de ruteo activas en el *router*.

Todos los dispositivos de la red se implementan con software de código libre lo que implica un ahorro de costos y una posterior repetición de las pruebas.

## **3.2 Herramientas de Software Utilizadas**

Para el desarrollo de los diferentes escenarios de prueba se ocupan varias herramientas de código abierto, que se describen a continuación.

### **3.2.1 *Flow-tools***

*Flow-tools* [20] es una librería y colección de programas que se utilizan para recoger, enviar y generar reportes a partir de datos *NetFlow*, protocolo de red desarrollado por *Cisco Systems* para recopilar información de tráfico IP. Las herramientas se pueden usar junto a un servidor único o distribuido por múltiples servidores para despliegues grandes. La librería *flow-tools* proporciona una API para el desarrollo de aplicaciones personalizadas para las versiones de exportación de *Netflow* 1, 5, 6 y 14.

### **3.2.2 *Graphical Network Simulator (GNS3)***

Es un simulador de red gráfico [19] que permite la simulación de redes complejas, sin la necesidad de tener dispositivos de hardware o software adicionales a la máquina en la que está instalado esta herramienta. Esto agrega un gran valor, pues esta herramienta dispone de interfaces de hardware genéricas y específicas de *Cisco*. El usuario puede extender una red real

conectándola a la topología virtual. Para proporcionar simulaciones completas y precisas, GNS3 utiliza los siguientes componentes:

1. *Dynamips*: Es una herramienta de software libre y descargable, escrito por Christopher Fillot. Es un emulador de *router Cisco*, que ejecuta las imágenes IOS estándar, emula series 1700, 2600, 3600, 3700, 7200. Para ejecutar *Dynamips* primero se deben instalar las bibliotecas que se utilizan para acceder fácilmente a las capas de red de bajo nivel *libpcap* o *WinPcap* dependiendo de la plataforma, Linux o Windows respectivamente. *Dynamips* se encuentra en desarrollo activo y hay algunos complementos escritos para él, los más populares son *Dynagen* y *Pemu*.

2. *Dynagen*: Es el generador de las configuraciones de las redes a simular. Está escrito en *Python* el cual se utiliza como lenguaje de programación, lo que ahorra un tiempo considerable en el desarrollo del programa, *Dynagen* fue desarrollado en *python* a través de *PyQt* que es un conjunto de herramientas para crear aplicaciones de la interfaz gráfica de usuario (GUI), confeccionada con la poderosa librería *Qt* que es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario. Utiliza el lenguaje de programación *C++*, aunque *PyQt* es una fusión entre *python* y la biblioteca *Qt*. GNS3 integra la consola de administración de *Dynagen* que permite a los usuarios listar los dispositivos, suspender y recargar instancias, determinar y administrar los valores de idle-PC, realizar capturas, y mucho más.

3. *Pemu*: Es el emulador de servidor de seguridad Cisco Pix *firewalls* (incluyendo el encapsulador) basada en *Qemu* que es un emulador y virtualizador genérico de la CPU.

GNS3 realiza soportes a tecnologías como *Frame Relay*, ATM, PIX firewalls, Ethernet, FastEthernet, Giga Ethernet e inalámbrica, VLAN, NAT, PAT, Protocolos UDP, DHCP, RIP, TCP, OSPF, FTP, entre otras.

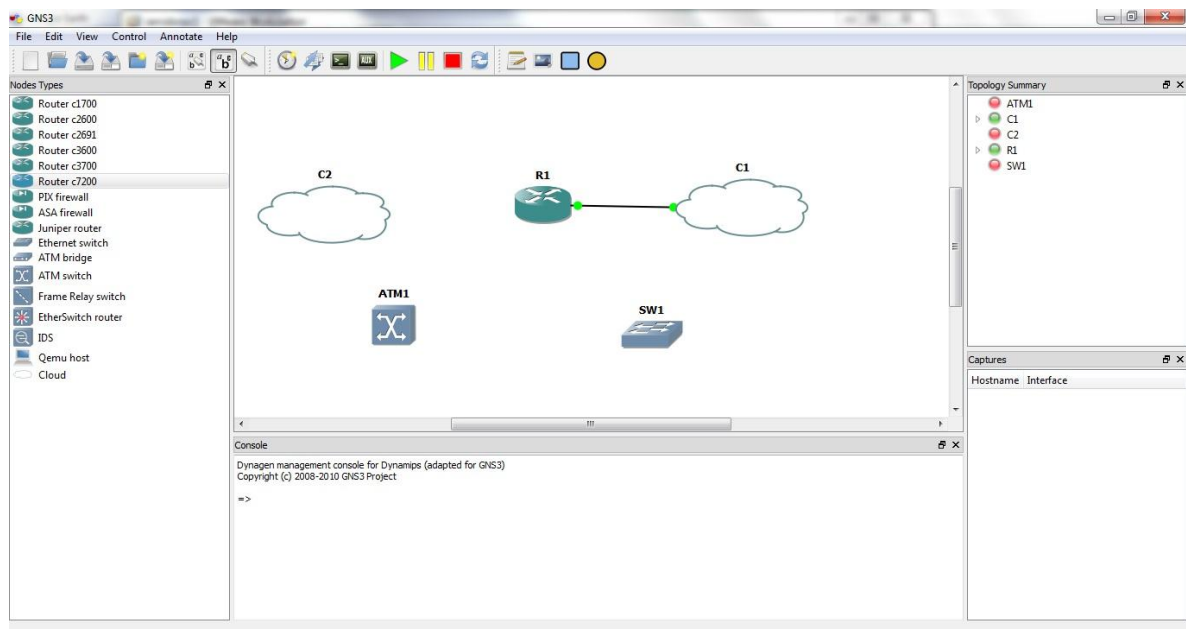
Se puede utilizar en múltiples sistemas operativos, incluyendo Windows, Linux y MacOSX.

Ventajas:

- Diseño de alta calidad y simulación de topologías de red complejas.
- Pone en marcha la construcción de redes de computadores, sin la necesidad de tener dispositivos de hardware o software adicionales a la máquina en la que está instalado.
- Se pueden crear una o varias redes, simulando una configuración real. Esto agrega un gran valor, pues esta herramienta dispone de interfaces de hardware genéricas y específicas de Cisco.
- Tras la creación de las redes, es necesario únicamente seguir los pasos sistemáticos que se realizaron en la herramienta, para poder echar a andar la red real, en la que intervienen dispositivos de hardware real.
- Permite la captura de paquetes utilizando *Wireshark* (capturador/analizador de paquetes de red).

Desventajas: El simulador debe instalarse en equipos de alta capacidad de memoria y procesamiento ya que utiliza el consumo de CPU como el dispositivo real. Otro de sus inconvenientes es que GNS3 únicamente simula dispositivos Cisco y un router Juniper genérico.

En la Figura 12 se muestra la interfaz gráfica de GNS3.



**Figura 12: Interfaz gráfica de GNS3.**

### 3.2.3 Ubuntu

Ubuntu es una distribución GNU/Linux que ofrece un sistema operativo predominantemente enfocado a ordenadores de escritorio aunque también proporciona soporte para servidores [21]. Se basa en Debian GNU/Linux y concentra su objetivo en la facilidad de uso, la libertad de uso, los lanzamientos regulares (cada 6 meses) y la facilidad en la instalación.

La versión estable más reciente de Ubuntu es la 10.10, nombre clave "Maverick Meerkat" liberada el 10 de octubre del 2010. La última versión LTS (soporte extendido de 3 años para escritorio y 5 para servidor) es Ubuntu 10.04, versión "Lucid Lynx" liberada el 29 de abril de 2010.

### 3.2.4 Matlab

Matlab contiene una serie de funciones de computación científica. Se puede integrar el código de Matlab con otros lenguajes y aplicaciones, y distribuir los algoritmos y aplicaciones que se desarrolló usando Matlab [22].

#### Características principales

- Lenguaje de alto nivel para cálculo técnico
- Entorno de desarrollo para la gestión de código, archivos y datos
- Herramientas interactivas para exploración, diseño y resolución de problemas iterativos
- Funciones matemáticas para álgebra lineal, estadística, análisis de Fourier, filtraje, optimización e integración numérica
- Funciones gráficas bidimensionales y tridimensionales para visualización de datos



- Herramientas para crear interfaces gráficas de usuario personalizadas
- Funciones para integrar los algoritmos basados en MATLAB con aplicaciones y lenguajes externos, tales como C/C++, FORTRAN, Java, COM y Microsoft Excel.

Se utiliza la *toolbox Cyclic Network Calculus* (CyNC) que es una herramienta *open source* para Matlab que permite realizar operaciones del algebra *min-plus*, como convolución y deconvolución; realizar cálculos de desviaciones horizontales y verticales; además permite crear todas las funciones que se definen en 2.3.1

### 3.2.5 VLC Media Player

*VLC media player* [23] es un reproductor multimedia que forma parte del proyecto *VideoLan* y tiene la característica de ser un software libre, el cual se distribuye bajo licencia GPL. Este software soporta una gran variedad de códecs tanto para video como para audio, así como diferentes tipos de archivos; además soporta formatos DVD, VCD y diversos protocolos de *streaming*. Por otra parte, este software se puede utilizar como servidor de medios *unicast* o *multicast*, tanto en IPv4 como en IPv6. Ocupa la biblioteca códec *libavcodec* del proyecto *FFmpeg* para manipular los muchos formatos que soporta, y utiliza la biblioteca de descryptación DVD *libdvdcss* para poder reproducir los DVDs cifrados.

### 3.2.6 VirtualBox

VirtualBox es un potente software de virtualización para arquitecturas x86/amd64 desarrollado por Oracle Corporation como parte de su familia de productos de virtualización. Por medio de esta aplicación es posible instalar sistemas operativos adicionales, dentro de otro sistema operativo anfitrión, cada uno con su propio ambiente virtual [24].

Entre los sistemas operativos soportados se encuentran GNU/Linux, Mac OS X, OS/2 Warp, Microsoft Windows, y Solaris/OpenSolaris, y dentro de ellos es posible virtualizar los sistemas operativos FreeBSD, GNU/Linux, OpenBSD, OS/2 Warp, Windows, Solaris, MS-DOS y muchos otros.

En cuanto a la emulación de hardware, los discos duros de los sistemas invitados son almacenados en los sistemas anfitriones como archivos individuales en un contenedor llamado Virtual Disk Image, incompatible con los demás software de virtualización.

## 3.3 Herramienta de Hardware

A continuación se describen las herramientas de Hardware facilitadas para el desarrollo de las pruebas.

- Notebook Inspiron N5110
- Procesador : Intel® Core i7-2670Q 2.2 GHz
- Memoria RAM : 6 GB DDR

- Disco Duro : 500GB Ultra DMA 7200 rpm
- Puertos : 3 puertos USB 2.0, 1 Paralelo, 1RJ-45 y 1 VGA
- Tarjeta de Red : Conexión integrada de red Broadcom NetLink.

### 3.4 Escenarios de Prueba

A continuación se describen los escenarios que se utilizan para realizar las pruebas. El objetivo general es obtener mediciones del tráfico, el cual se genera al transmitir un *streaming* de video. Las pérdidas de paquetes y *delay* que se obtienen de estas mediciones se comparan con las estimaciones de la aplicación de *Network Calculus*.

#### 3.4.1 Escenario 1: Medición Tráfico de Entrada *router 3*

El primer escenario tiene como objetivo medir el tráfico de entrada en una interfaz de un *router* para obtener la curva de llegada del tráfico entrante al *router*. La red se compone de tres *routers*, un servidor de video y un cliente; el diseño se muestra en la Figura 13.

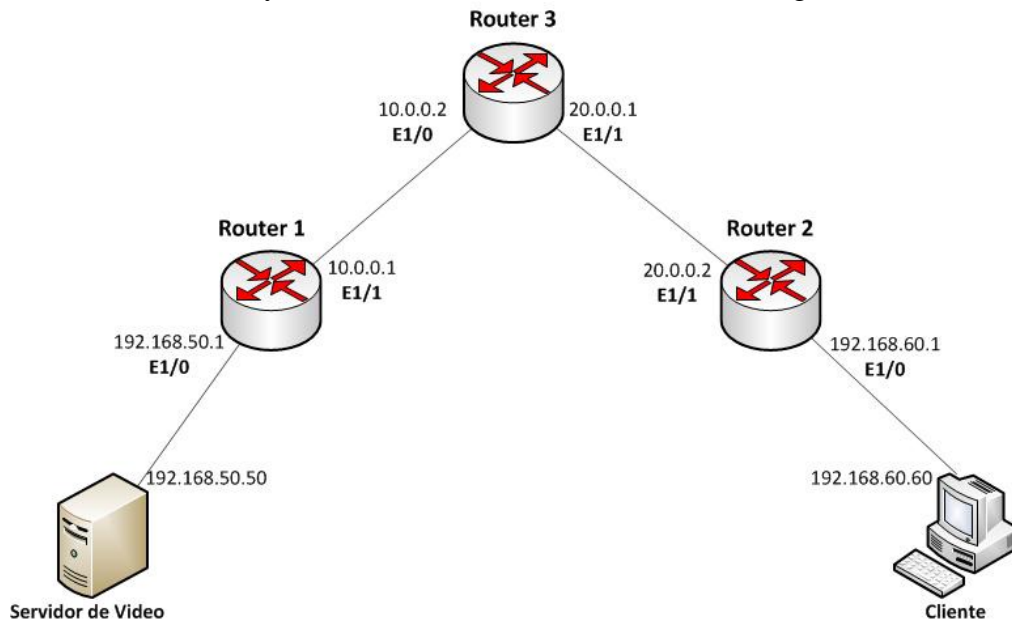


Figura 13: Escenario de Prueba 1.

Para el correcto funcionamiento de la red se necesita configurar el direccionamiento IP de todos los *routers* para garantizar que la transmisión del *streaming* de video sea recibida por el cliente. Se realizan simulaciones en donde el protocolo de enrutamiento que se configura en los tres *routers* es *Enhanced Interior Gateway Routing Protocol* (EIGRP) y pruebas con configuración de rutas estáticas; la política de encolamiento es la *FIRST IN FIRST OUT* (FIFO). Los detalles de la configuración se detallan en los Anexos. El direccionamiento IP de los elementos de red se muestran en las siguientes Tablas 1 y 2:

**Tabla 1: Direcciones IP Routers.**

Router	Interfaz E 1/0	Interfaz E 1/1
Router 1	192.168.50.1	10.0.0.1
Router 2	192.168.60.1	20.0.0.2
Router 3	10.0.0.2	20.0.0.1

**Tabla 2: Direcciones IP Computadores.**

Dispositivo	Dirección tarjeta de red
Servidor de Video	192.168.50.50
Cliente	192.168.60.60

A continuación se describe cada elemento del diseño del escenario 1:

### 3.4.1.1 Servidor de Video

Para simular un servidor de video se utiliza la aplicación *VirtualBox*. El sistema operativo que se instala en esta máquina es Ubuntu 10.10. Para recoger y generar los reportes de las mediciones de tráfico se utiliza *flow-tools*. La aplicación que se utiliza para transmitir el *streaming* de video es *VLC media player* y el archivo de video a emitir tiene las siguientes características, Tabla 3:

**Tabla 3: Características clip de video**

Nombre archivo	Dexter.S06E05.avi
Duración	0:51:28
Tamaño	547MB
Ancho fotograma	624
Alto fotograma	352
Velocidad de fotograma	23 fotogramas/segundo
Velocidad de datos	1361 Kbps

En el servidor de video se ubica además el colector, que es el host que recibe los datos enviados por la interfaz Ethernet 1/0 del *router 3*. El programa *flow-capture*, que es parte de los programas que incluye la herramienta *flow-tools*, se encarga de escuchar un puerto específico para capturar los datos y escribir registros del tráfico en el disco. *Flow-capture* debe saber dónde escribir los archivos, cuán a menudo comenzar un nuevo archivo y ver de quién acepta información de tráfico. El comando que se ejecuta en el servidor es el siguiente:

```
flow-capture -p /var/run/flow-capture.pid -n 287 -w /var/db/flows/router1 -S 5 192.168.50.50/0/5678
```

El argumento *-p* dice a *flow-capture* donde almacenar el proceso PID. La ubicación que se utiliza es */var/run/flow-capture.pid*.

La opción *-n* dice a *flow-capture* cuántas veces debería crear un archivo de *log* en un periodo de 24 horas. El número 287 indica a *flow-capture* crear un nuevo archivo *log* cada 5 minutos (un día contiene 288 períodos de 5 minutos).

Con la opción `-w` se dice donde *flow-capture* escribe los archivos. El directorio que se utiliza es `/var/db/flows/router1`.

La opción `-S 5` dice a *flow-capture* como almacenar los mensajes en *syslog*, indica cuántos flujos ha procesado, cuántos paquetes ha recibido, y cuántos flujos aproximadamente ha descartado. El número 5 indica los minutos entre mensajes *log*.

El último argumento `192.168.50.50/0/5678` especifica la configuración de red de *flow-capture*. La primera dirección `192.168.50.50` es la dirección IP donde se ubica el colector, servidor de video en el escenario 1. La segunda dirección es la del sensor que se configura para enviar datos a este colector. Al dejar en 0 esta dirección *flow-capture* acepta flujo de cualquier dirección. Finalmente, el número 5678 es el puerto UDP en que *flow-capture* escucha.

### 3.4.1.2 Router 1

Para emular el *router 1* se utiliza GNS3. El modelo que se utiliza es un *router* Cisco 2600 y en el escenario se utilizan dos interfaces Ethernet 10Mb/s. Se configura de tal forma que se limita el tráfico de salida de la interfaz Ethernet 1/1, lo cual se logra al aplicar suavizado de tráfico (*traffic shaping*). Este suavizado impone un límite al tráfico de video, con lo cual se obtiene la curva de llegada del tráfico entrante al *router 3*. Se realizan pruebas para diferentes velocidades de esta interfaz. En la Figura 14 se muestra lo anteriormente descrito.

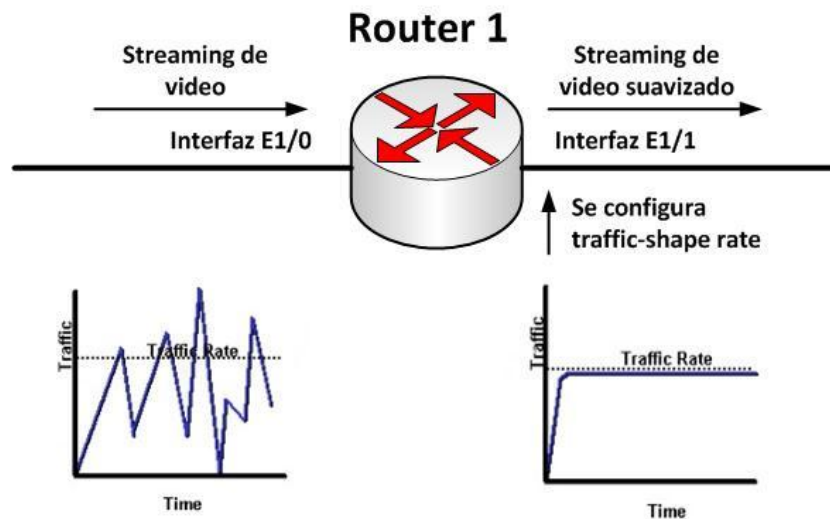
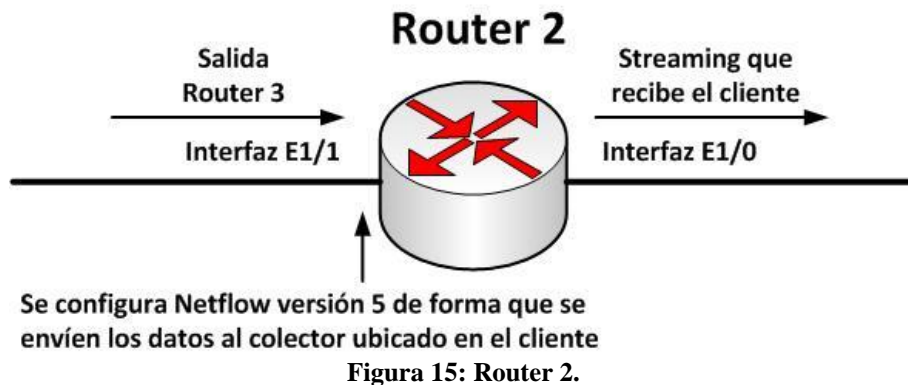


Figura 14: Suavizado *streaming* de video.

### 3.4.1.3 Router 2

Para emular el *router 2* se utiliza GNS3. El modelo que se utiliza es un *router* Cisco 2600 y en el escenario se utilizan dos interfaces Ethernet 10Mb/s. Se configura en la interfaz Ethernet 1/1 *Netflow* versión 5 de forma tal que se envíen los datos de tráfico hacia el colector ubicado en el cliente. Se tiene que configurar en esta interfaz ya que *Netflow* versión 5 envía datos del tráfico entrante a la interfaz, y en este diseño el tráfico que entra al *router 2* es el tráfico que sale de la interfaz Ethernet 1/1 del *router 3*, si se considera que no hay pérdidas entre el enlace del *router 3* y 2.

En la siguiente Figura 15 se muestra la descripción previa:



#### 3.4.1.4 Router 3

Para emular el *router* se utiliza GNS3. El modelo que se utiliza es un *router* Cisco 2600 y en el escenario se utilizan dos interfaces Ethernet 10Mb/s. Se configura de forma tal que se limita el tráfico de salida de la interfaz Ethernet 1/1, que se logra al aplicar suavizado de tráfico (*traffic shaping*). Este suavizado impone un límite al tráfico con lo cual podemos obtener la tasa de servicio del *router* 3. Se realizan pruebas para diferentes velocidades en esta interfaz. Se configura también en la interfaz Ethernet 1/0 *Netflow* versión 5 de forma tal que se envíen los datos de tráfico que ingresan a esta interfaz hacia el colector ubicado en el servidor de video. En la siguiente Figura 16 se muestra lo descrito:

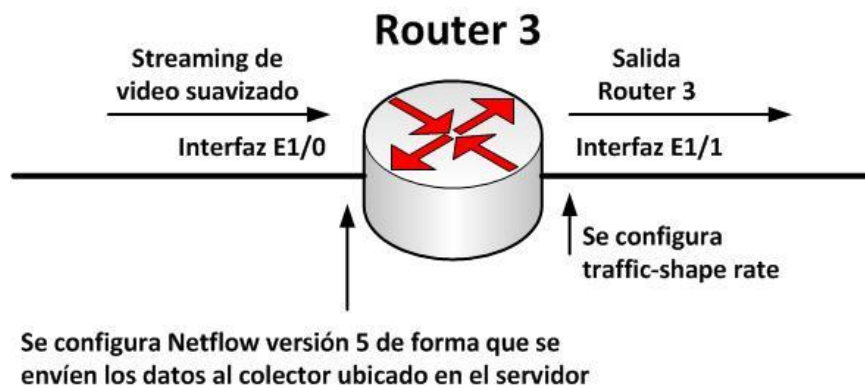


Figura 16: Router 3.

#### 3.4.1.5 Cliente

Para simular el Cliente se utiliza la aplicación *VirtualBox*. El sistema operativo que se instala en esta máquina es Ubuntu 10.10. Para recoger y generar los reportes de las mediciones de tráfico se utiliza *flow-tools*. La aplicación que se utiliza para recibir el *streaming* de video es *VLC media player*. En el cliente se ubica, al igual que en el servidor de video, el colector que es el host que recibe los datos enviados en este caso por la interfaz Ethernet 1/1 del *router* 2. El comando que se utiliza para recoger la información es básicamente el mismo que se utiliza en el servidor y se cambian las direcciones de las interfaces de red y el puerto a escuchar al 9996:

```
flow-capture -p /var/run/flow-capture.pid -n 287 -w /var/db/flows/router1 -S 5 192.168.60.60/0/9996
```

### 3.4.1.6 Streaming de Video

Una vez que se realiza la configuración y se verifica la conectividad entre el servidor y el cliente se continua con la transmisión del video. Para ello se realiza un *streaming* UDP en el servidor de video. Los pasos para lograr esto se describen a continuación:

- 1- Lo primero es abrir el reproductor multimedia VLC en el Servidor, se selecciona la opción *medio* y después *emitir*. Se añade el archivo a emitir y se presiona en *emitir*. Luego en *Fuente* se comprueba que la fuente coincida con el archivo de entrada y se pulsa en *siguiente*. En *Destinos* se seleccionan los métodos de emisión a utilizar, en este caso se realiza un *streaming* UDP y se añade la dirección del cliente 192.168.60.60 y el puerto 1234 que se utiliza en el *streaming*. Se habilita la opción de *transcodificar* y elige la WMV+WMA (ASF) para presionar *siguiente*. En *Opciones* no se realiza ningún cambio, sólo se presiona *emitir*. La cadena de salida de emisión generada es la siguiente:

```
:sout=#transcode{vcodec=WMV2,vb=1000,fps=30,scale=1,acodec=wma2,ab=128,channels=2,samplerate=44100}:duplicate{dst=udp{dst=192.168.60.60:1234},dst=display} :no-sout-rtp-sap :no-sout-standard-sap :sout-keep
```

- 2- En el lado del Cliente, para ver el *streaming*, se abre el reproductor multimedia VLC, se elige la opción *medio* y luego se selecciona *avanzado*. En *abrir medio* se elige la opción *red* y se introduce la URL *udp://@* para luego pulsar *emitir* y se comienza a reproducir el *streaming* de video. Se aprecia a simple vista un deterioro en la calidad de la imagen y un cierto desfase producto principalmente de las limitaciones de tráfico que se hacen en las interfaces de los *routers* 1 y 3.

### 3.4.1.7 Mediciones del tráfico de entrada al router 3

Con la configuración de *flow-tools* que se realiza en el colector ubicado en el servidor se crean automáticamente carpetas en el directorio */var/db/flows/router1/*, las cuales se ordenan por año, mes y día. Dentro de cada carpeta día se generan archivos cada 5 minutos, los cuales tienen la siguiente estructura de nombre:

```
ft-v05.2012-03-22133501-0400
```

ft-v05 corresponde a *flow-tools* y a la versión 5 de *Netflow*. Luego viene la fecha y hora en la cual se crea el archivo. El último número (0400) corresponde al *Coordinated Universal Time* (UTC) que es el tiempo de la zona horaria de referencia respecto a la cual se calculan todas las otras zonas del mundo.

Una vez que se termina de transmitir el video entre el servidor y el cliente, se dirige al directorio del día que se realiza la prueba y se selecciona los archivos que se crean por el *streaming* de video y se copian en un nuevo directorio. El comando que se ejecuta en Ubuntu para mostrar todo el tráfico que entra a la interfaz Ethernet 1/0 del *router* 3 es el siguiente:

# flow-cat \* | flow-print -f 5

Comando que genera una serie de datos los cuales se pueden exportar a un archivo de texto para su posterior manipulación en Microsoft Excel y Matlab. En la siguiente Tabla 4 se muestran los datos medidos en un intervalo de 9 minutos con el fin de mostrar que es lo que entrega el comando descrito anteriormente.

**Tabla 4: Mediciones del tráfico de entrada al router 3.**

Start	End	Sif	SrcIPAddress	SrcP	Dif	DstIPAddress	DstP	P	Fl	Pkts	Octets
0228.23:12:14.751	0228.23:13:14.739	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1271	1708224
0228.23:13:14.779	0228.23:14:15.747	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1290	1733760
0228.23:14:15.807	0228.23:15:16.747	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1289	1732416
0228.23:15:16.783	0228.23:16:17.707	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1288	1731072
0228.23:16:17.767	0228.23:17:18.755	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1291	1735104
0228.23:17:18.815	0228.23:18:19.723	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1288	1731072
0228.23:18:19.771	0228.23:19:20.715	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1289	1732416
0228.23:19:20.767	0228.23:20:21.755	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1290	1733760
0228.23:20:21.775	0228.23:21:22.719	3	192.168.50.50	49737	4	192.168.60.60	1234	17	0	1289	1732416

A continuación en la Tabla 5 se describe cada campo:

**Tabla 5: Descripción de campos.**

Campo	Descripción
Start	Fecha y hora exacta en la cual comienza el muestreo de un minuto.
End	Fecha y hora exacta en la cual termina el muestreo de un minuto.
Sif	Número de interfaz de la fuente .
SrcIPAddress	Dirección IP de la fuente.
SrcP	Puerto de la fuente.
Dif	Número de interfaz del destino.
DstIPAddress	Dirección IP de la fuente.
DstP	Puerto del destino.
P	Protocolo del tráfico.
Pkts	Cantidad de paquetes en el intervalo de muestreo.
Octets	Cantidad de bytes en el intervalo de muestreo.

### 3.4.2 Escenario 2: Medición de Tráfico de Salida Router 3

El siguiente escenario tiene como objetivo medir el tráfico de salida del *router 3* para compararlo con el tráfico entrante a la interfaz Ethernet 1/0 del *router 3* y así calcular las pérdidas de paquetes. El escenario es el mismo que el anterior pero las mediciones se realizan ahora en la interfaz Ethernet 1/1 del *router 2*. Como *Netflow* versión 5 exporta datos del tráfico entrante a la interfaz, las medidas se tienen que realizar en la interfaz 1/1 *del router 2* y no directamente en la interfaz de salida del *router 3*.

Como la velocidad de la interfaz de salida del *router* 3 es mayor a la de entrada y además no se tiene otros tráficos más que el *streaming* de video no se generan pérdidas de paquetes. Lo que se realiza para provocar pérdidas es realizar cambios en las tablas de ruteo en pleno funcionamiento de los *routers*.

Se realizan pruebas para distintas velocidades del tráfico de entrada y salida del *router* 3. Además las pruebas se realizan primero sin realizar ninguna modificación en funcionamiento de los *routers*, pero luego para las mismas velocidades se realizan cambios en los *routers* para generar pérdidas, en las tablas de rutas.

### 3.4.3 Escenario 3: Medición de *Delay* en *Router* 3

El escenario 3 tiene como objetivo medir el *delay* que se produce en el *router* 3. Para realizar esta medición se agrega a la red ya implementada un nuevo computador que actúa como un perturbador en la red para inyectar tráfico en el *router* 3. En la Figura 17 se muestra este escenario.

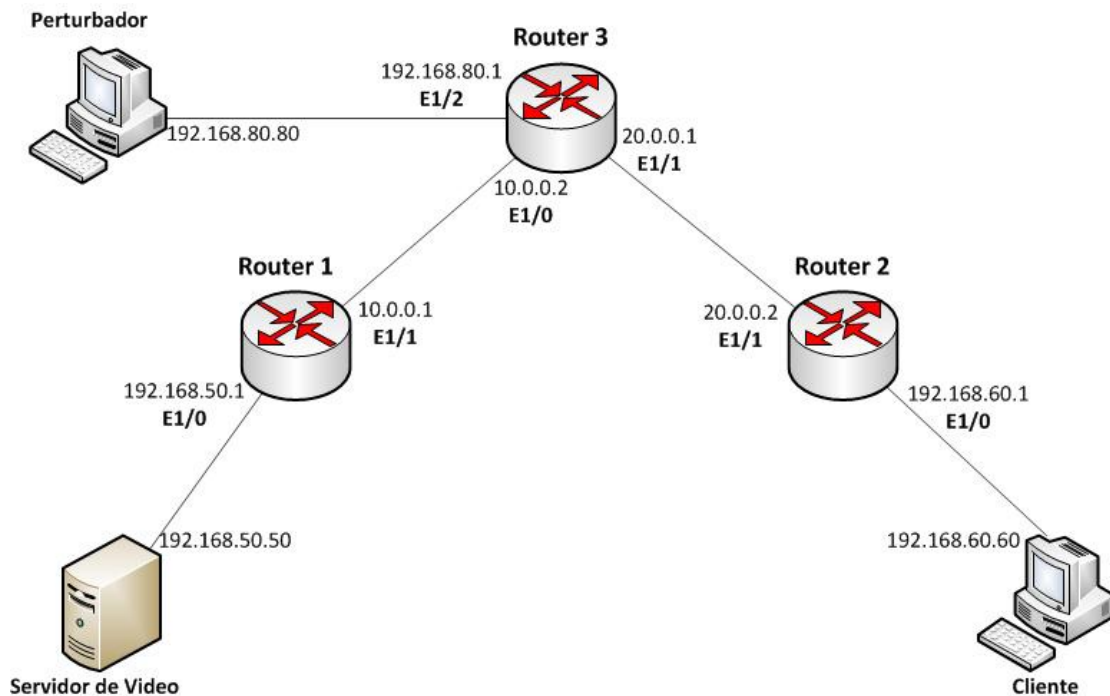


Figura 17: Escenario de Prueba 3.

A continuación se describe el dispositivo que se agrega a la red.

#### 3.4.3.1 Perturbador

Para simular el Perturbador se utiliza la aplicación *VirtualBox*. El sistema operativo que se instala en esta máquina es Ubuntu 10.10. La función de este computador es inyectar tráfico en el *router* 3, para lo cual se realiza un ping hacia la interfaz Ethernet 1/1 del *router* 3 como se muestra en la Figura 18.



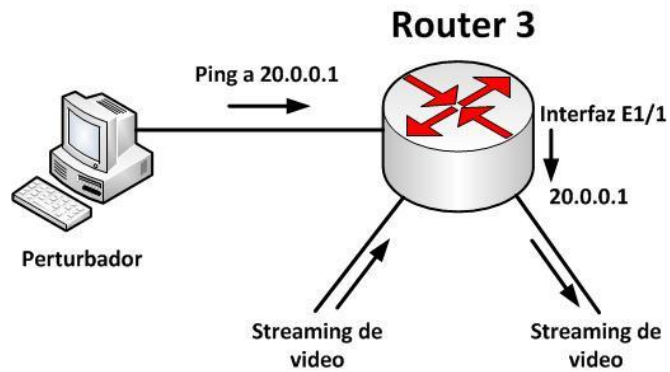


Figura 18: Ping a Interfaz Ethernet 1/1.

En esta prueba se utilizan distintas velocidades de la interfaz Ethernet 1/1 del *router 1* para tener varias medidas de *delay* en función de esta velocidad.

### 3.4.4 Escenario 4: Medición de *Delay* Concatenación de Nodos

El siguiente escenario tiene como objetivo medir el *delay* que se produce al agregar un cuarto *router* a la red. En la Figura 19 se muestra el escenario con los cuatro *routers*.

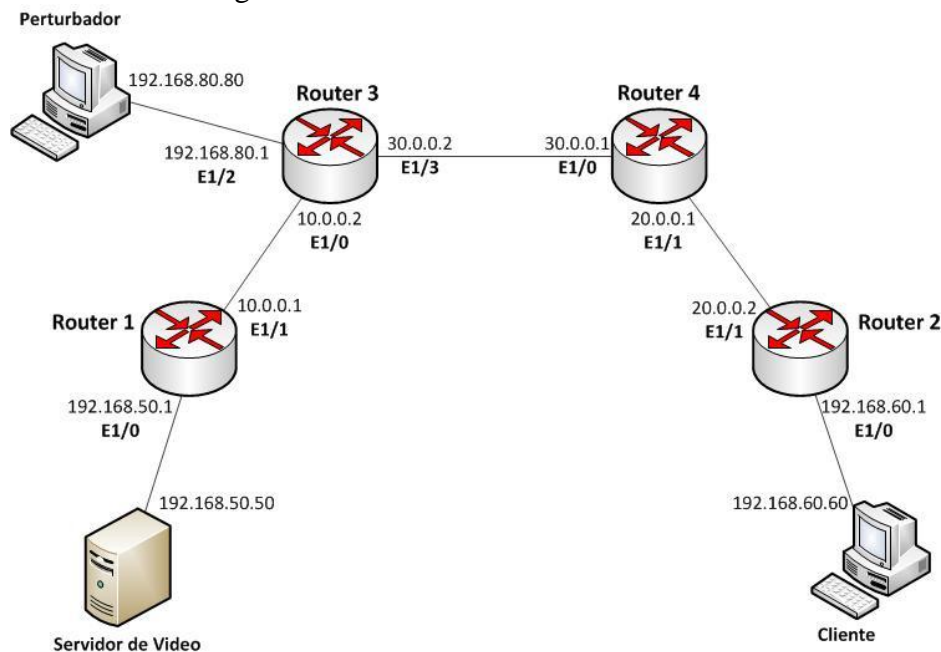


Figura 19: Escenario 4.

#### 3.4.4.1 Router 4

Para emular el *router 4* se utiliza GNS3. El modelo que se utiliza es un *router* Cisco 2600 y en el escenario se utilizan dos interfaces Ethernet 10Mb/s. Se configura de forma tal que se limita el tráfico de salida de la interfaz Ethernet 1/1, que se logra al aplicar un suavizado de tráfico (*traffic shaping*). Con esto se puede obtener la tasa de servicio del *router 4*.

Con el computador Perturbador se realizan una serie de pings hacia la interfaz Ethernet 1/1 del *router* 4. Se realizan pruebas para una velocidad de 224.6 [kbits/seg] de la interfaz Ethernet 1/1 del *router* 1, la misma velocidad para la interfaz Ethernet 1/1 del *router* 4 y una velocidad fija de 244.14 [kbits/seg] en la interfaz Ethernet 1/1 del *router* 3.

Con las mediciones de tráfico que se realizan se hacen estimaciones del *backlog* y de *delay* al aplicar *Network Calculus*.

# 4

## Resultados y Discusiones

### 4.1 Mediciones Realizadas

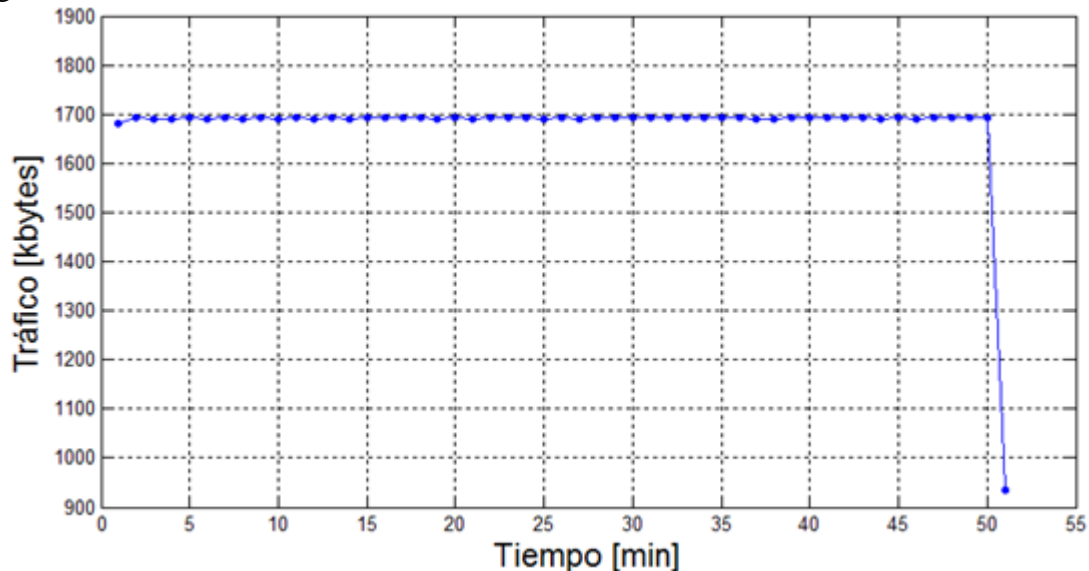
Se realizan mediciones en los diferentes escenarios para distintas velocidades de la interfaz Ethernet 1/1 del *router* 1 (que corresponde a la velocidad de entrada de la interfaz Ethernet 1/0 del *router* 3) y velocidades de la interfaz Ethernet 1/1 del *router* 3. En la siguiente Tabla 6 se resumen las configuraciones de velocidades que se utilizan en los *routers* 1 y 3.

**Tabla 6: Configuración Velocidades Interfaces Routers.**

Escenario	Velocidad Tráfico Salida Configurada en Intefaz E1/1 Router 1 [kbits/seg]	Velocidad Tráfico Salida Configurada en Intefaz E1/1 Router 3 [kbits/seg]
1	146,48437	292,96875
2	195,3125	292,96875
3	273,4375	292,96875
4	146,484375	244,140625
5	195,3125	244,140625
6	224,609375	244,140625

#### 4.1.1 Medición Tráfico de Entrada Router 3

Se realizan mediciones del tráfico de entrada en la interfaz Ethernet 1/0 del *router* 3. Debido a la configuración de *Netflow* en esta interfaz, el muestreo del tráfico es cada un minuto. En el Gráfico 5 se muestra el tráfico que ingresa a la interfaz Ethernet 1/0 del *router* 3 para la Configuración 6 de velocidades de las interfaces del escenario 1.



**Gráfico 5: Tráfico entrada interfaz Ethernet 1/0 router 3.**

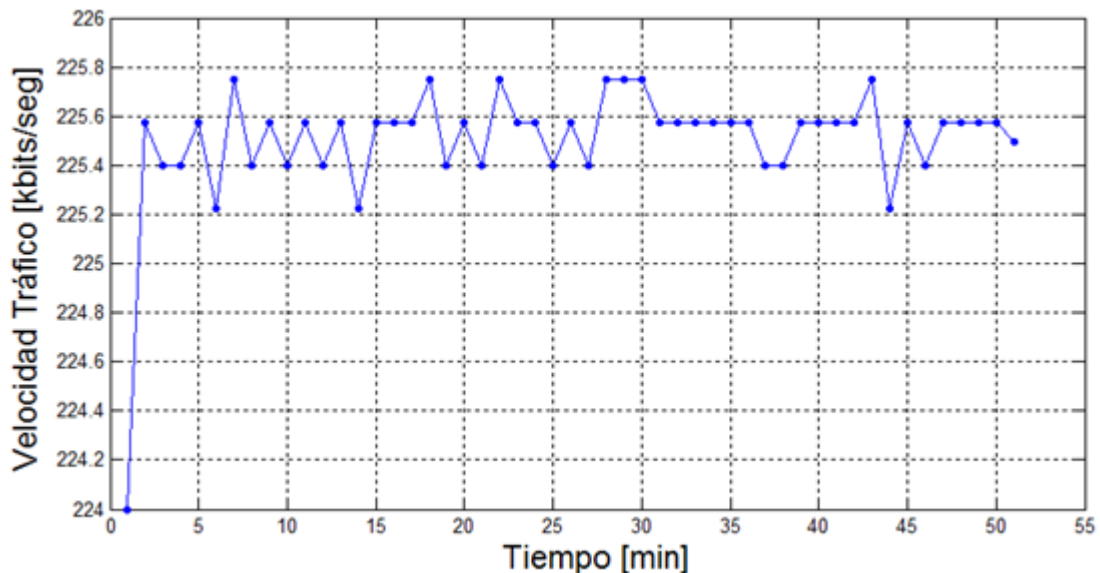
Del la Gráfico 5 se puede notar que la velocidad del tráfico es prácticamente constante en los 51 minutos 28 segundos que dura el *streaming* de video. El último punto del gráfico muestra la cantidad de kbytes que ingresan a la interfaz en el intervalo de un minuto, pero debido a que el video en este intervalo dura 28 segundos el tráfico en este intervalo es de 935 [kbytes].

En la Tabla 7 se muestra el tráfico de entrada total a la interfaz Ethernet 1/0 del *router* 3 para los 6 escenarios de velocidades en los *routers*.

**Tabla 7: Tráfico de entrada para las 6 configuraciones.**

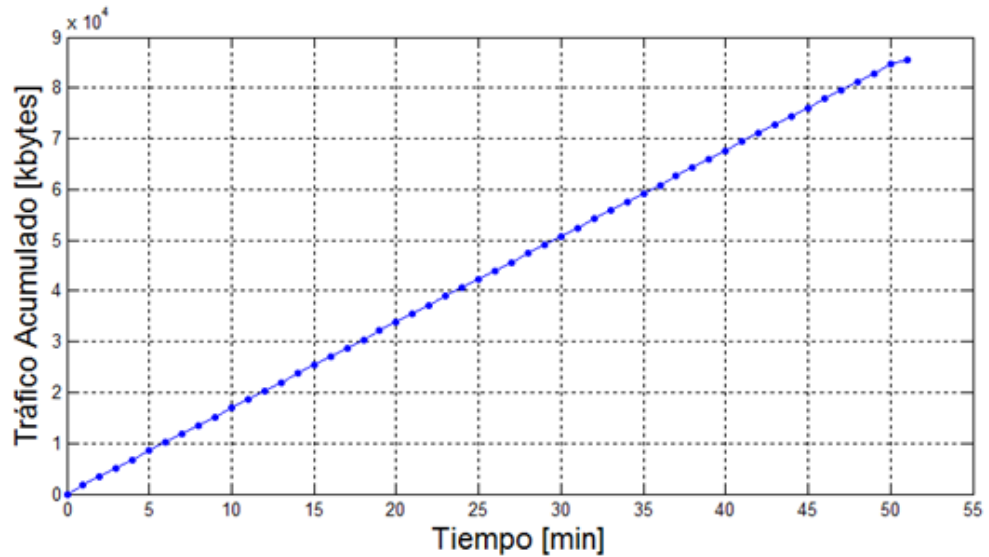
Escenario	Tráfico Entrada [Mbytes]
1	54,50079346
2	72,63354492
3	101,6777344
4	54,48669434
5	72,64379883
6	83,49627686

En el Gráfico 6 se muestra la velocidad promedio del tráfico de entrada a la interfaz Ethernet 1/0 del *router* 3 en función del tiempo. La velocidad máxima es de 225.75 [kbites/seg] y la velocidad promedio es de 225.50484 [kbites/seg], levemente superior a los 224,609375 [kbites/seg] de la Configuración 6. Esto se debe a que el *streaming* de video se realiza a una velocidad constante.



**Gráfico 6: Velocidad de entrada a la interfaz Ethernet 1/0 del router 3.**

En el siguiente Gráfico 7 se muestra el tráfico acumulado en función del tiempo.



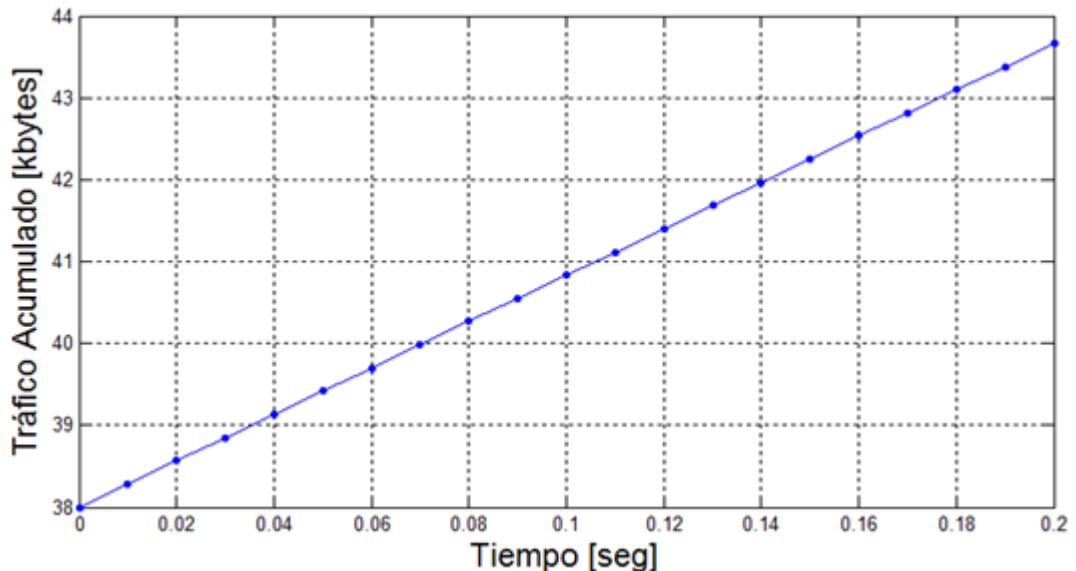
**Gráfico 7: Tráfico acumulado en la interfaz Ethernet 1/0 del router 3.**

Para llevar el gráfico anterior a una escala de tiempo de segundos se realiza una interpolación de grado cinco con un *basic fitting* de Matlab y el polinomio que se obtiene es:

$$\text{Tráfico}(t) = -8e-005*t^5 + 0.0092*t^4 - 0.37*t^3 + 6.2*t^2 + 1.7e+003*t + 38 \quad (3.1)$$

Con  $t$  en segundos y la unidad del tráfico en kbytes. Para  $t = 0$  la aproximación del tráfico es de valor 38 [kbytes] y dado que escala es de  $10^4$  [kbytes], la interpolación del tráfico acumulado es aproximadamente 0 para  $t = 0$ .

En el siguiente Gráfico 8 se muestra el tráfico acumulado en un intervalo de 0.2 segundos.



**Gráfico 8: Tráfico acumulativo en función del tiempo**

En los Anexos se muestran los gráficos anteriores para los 6 escenarios de distintas velocidades. En la Tabla 8 se muestra la velocidad promedio de entrada y salida del *router 3* que se mide para las diferentes configuraciones de velocidades de los *routers*.

**Tabla 8: Velocidades promedio de entrada y salida router 3.**

Configuración	Velocidad Tráfico Entrada Medida en Intefaz E1/0 Router 3 [kbits/seg]	Velocidad Tráfico Salida Medida en Intefaz E1/1 Router 3 [kbits/seg]
1	146,997496	147,0354575
2	196,1100058	196,1257497
3	274,64082	274,6267825
4	147,0637255	147,0784314
5	196,0230392	196,0573529
6	225,5316399	225,4296713

En la Tabla 9 se muestra la diferencia porcentual entre la velocidad medida y la velocidad configurada en la interfaz Ethernet 1/0 del *router* 3. Se observa que esta diferencia no es superior al 0.5 [%] en las seis configuraciones que se utilizan, lo que significa que nuestro colector implementado con *flow-tools* cumple con su objetivo.

**Tabla 9: Diferencia entre velocidad medida y configurada.**

Configuración	Diferencia entre velocidad medida y configurada [%]
1	0,350294045
2	0,408322953
3	0,440071301
4	0,395503268
5	0,363796078
6	0,398677883

En la Tabla 10 se muestra el polinomio de interpolación de grado 5 el cual aproxima el tráfico de entrada a la interfaz Ethernet 1/0 del *router* 3 para las 6 configuraciones de velocidad de los *routers*, donde  $y$  representa el tráfico en kbytes y  $x$  el tiempo en segundos.

**Tabla 10: Polinomio aproximación del tráfico de entrada router 3.**

Escenario	Polinomio interpolación grado 5
1	$y = - 4.8e-005*x^{\{5\}} + 0.0055*x^{\{4\}} - 0.22*x^{\{3\}} + 3.8*x^{\{2\}} + 1.1e+003*x + 20$
2	$y = - 6.7e-005*x^{\{5\}} + 0.0077*x^{\{4\}} - 0.31*x^{\{3\}} + 5.1*x^{\{2\}} + 1.4e+003*x + 36$
3	$y = - 9.7e-005*x^{\{5\}} + 0.011*x^{\{4\}} - 0.44*x^{\{3\}} + 7.3*x^{\{2\}} + 2e+003*x + 55$
4	$y = - 5e-005*x^{\{5\}} + 0.0058*x^{\{4\}} - 0.23*x^{\{3\}} + 3.9*x^{\{2\}} + 1.1e+003*x + 23$
5	$y = - 7e-005*x^{\{5\}} + 0.008*x^{\{4\}} - 0.32*x^{\{3\}} + 5.4*x^{\{2\}} + 1.4e+003*x + 30$
6	$y = - 8e-005*x^{\{5\}} + 0.0092*x^{\{4\}} - 0.37*x^{\{3\}} + 6.2*x^{\{2\}} + 1.7e+003*x + 38$

#### 4.1.1.1 Obtención Curva de Servicio

Para obtener la curva de servicio del *router* 3 utilizamos el modelo de servicio integrados de internet [1]. La curva de servicio que ofrece a un flujo un *router* que implementa una tasa garantizada es una función *rate-latency*, con tasa  $R$  y latencia  $T$ , que se relacionan de la forma:

$$T = \frac{C}{R} + D \quad (3.2)$$

Donde  $C$  es el tamaño de paquete máximo para el flujo y  $D=L/r$ , donde  $L$  es el tamaño máximo de paquetes en el *router* de todos los flujos que lo atraviesan, y  $r$  es la tasa total de los paquetes repartidos.

Para obtener el tamaño máximo de paquetes del tráfico en medición se utiliza *flow-tools* por medio del siguiente comando:

```
flow-cat * /flow-report -v TYPE=packet-size -v SORT=+key
```

El cual entrega información detallada del tráfico y entre otros datos entrega el tamaño de todos los paquetes que se miden.

```
root@servidortesis-VirtualBox:/home/servidortesis/Escritorio/Traficos/Medida600# flow-cat * /
flow-report -v TYPE=packet-size -v SORT=+key
packet size/flow flows octets  packets duration
1344      87  150240384 111786 5285940
```

Como se observa, el tamaño de los paquetes es constante y de un tamaño de 1344 [bytes]. El valor de  $R$  que utilizaremos es la velocidad que se configura en la interfaz Ethernet 1/1 del *router* 3. Como tenemos sólo un flujo que atraviesa un *router* la expresión (3.2) queda:

$$T = \frac{C}{R} + D = \frac{C}{R} + \frac{C}{R} = 2 * \frac{C}{R} = 2 * \frac{1344 * 8 [bit]}{250000 [bits / seg]} = 0.086016 [seg] = 86 [ms]$$

La curva de servicio para el *router* 3 es por lo tanto:

$$\beta_{R,T} = R[t - T]^+ = 244.14 [kbit / seg] * [t - 0.086016 [seg]]^+$$

En el Gráfico 9 se muestra la curva de servicio para las 6 configuraciones de velocidad de los *routers*.

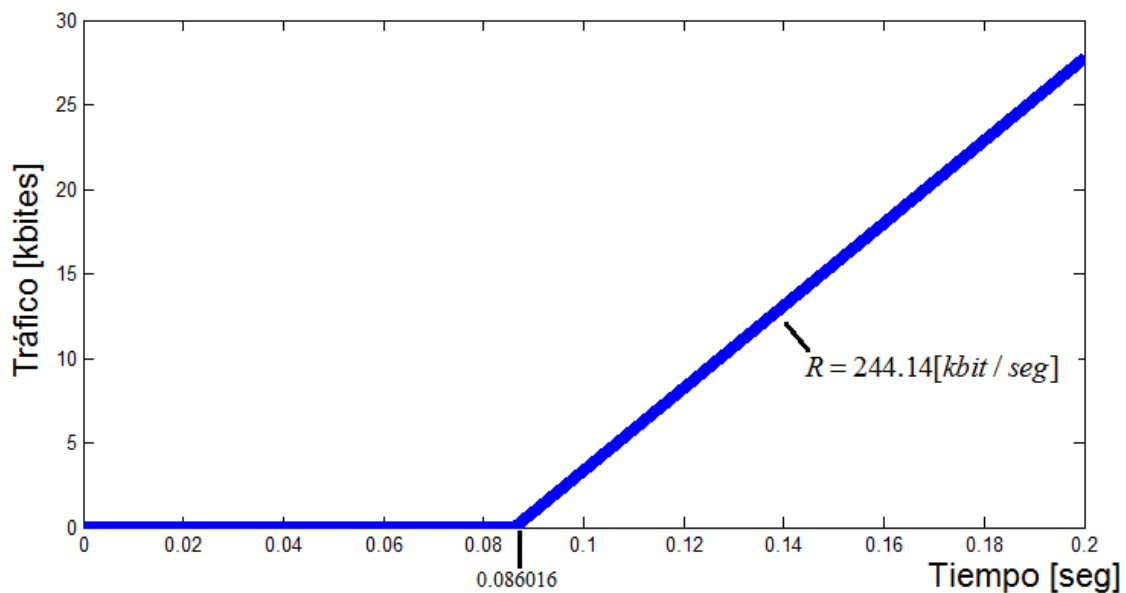


Gráfico 9: Curva de servicio *router* 3.

En la Tabla 11 se muestran las diferentes curvas de servicios para los 6 escenarios de distintas velocidades en los *routers*.

**Tabla 11: Curvas de Servicio.**

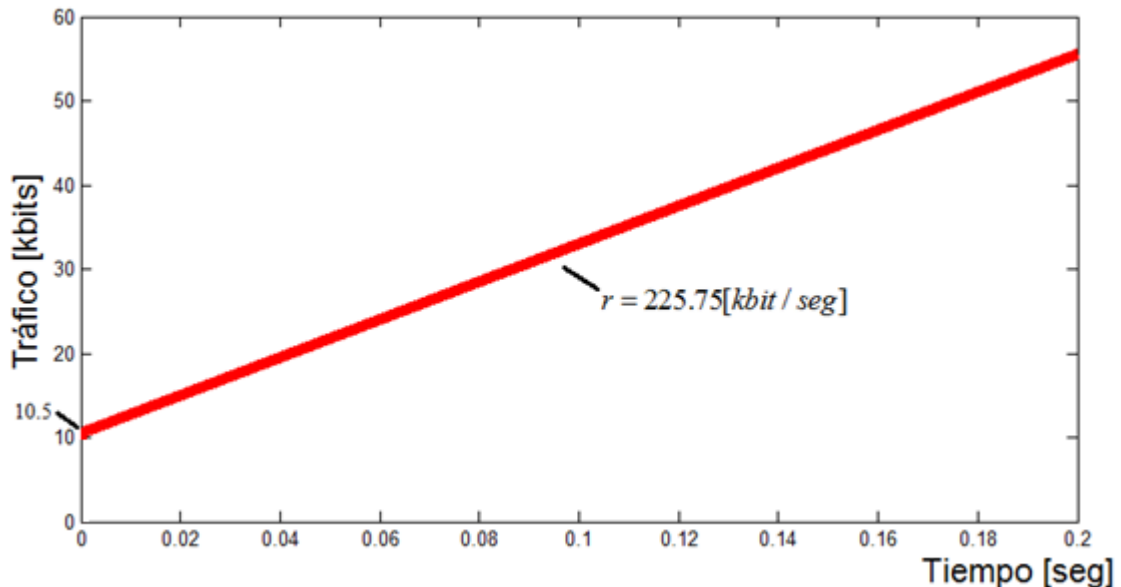
Escenario	Curva de Servicio
1	$292,96875 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$
2	$292,96875 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$
3	$292,96875 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$
4	$244.14 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$
5	$244.14 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$
6	$244.14 \text{ [kbit/seg]} * (t - 0.086016 \text{ [seg]})$

#### 4.1.1.2 Obtención Curva de Llegada

La curva de llegada es una curva afín de la siguiente forma:

$$\gamma_{r,b}(t) = \begin{cases} rt + b & \text{si } t > 0 \\ 0 & \text{otro caso} \end{cases}$$

En donde la tasa que utilizamos  $r$  es la velocidad de entrada máxima que se mide en la interfaz Ethernet 1/0 del *router* 3 y es de 225.575 [kbites/seg] para la Configuración 6. El valor del parámetro  $b$  se obtiene a partir del tamaño máximo que atraviesa la interfaz y es de 1344 [bytes]= 10.5 [kbits]. En el Gráfico 10 se muestra la curva de llegada del *router* 3.



**Gráfico 10: Curva de Llegada del router 3.**

En la Tabla 12 se muestran las curvas de llegada para los 6 escenarios de distintas velocidades en los *routers*.

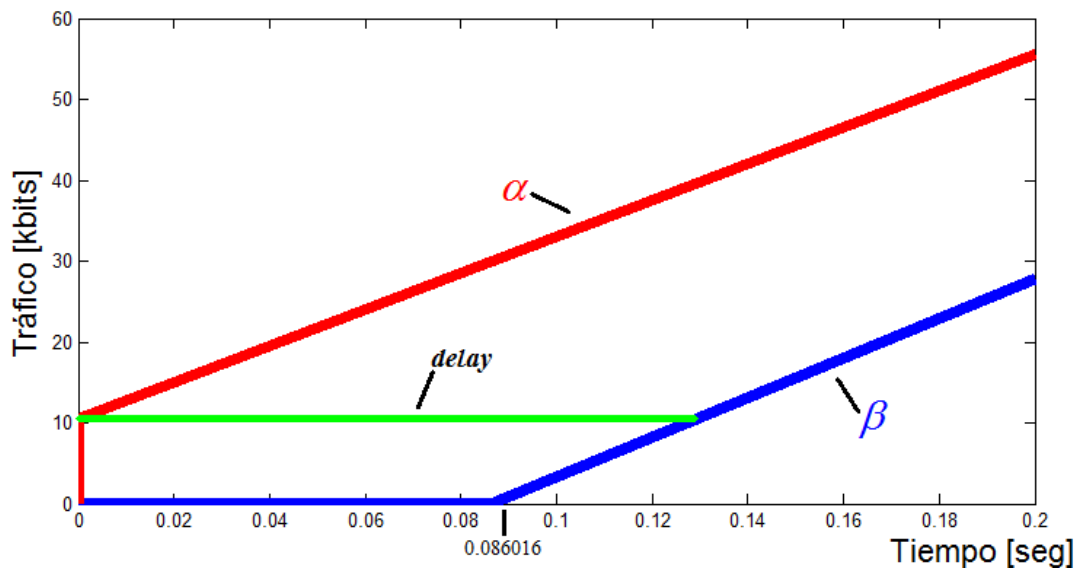


**Tabla 12: Curvas de Llegada.**

Escenario	Curva de Llegada
1	147,03 [kbit/seg]*t+10,5 [kbit]
2	196,12 [kbit/seg]*t+10,5 [kbit]
3	274,60 [kbit/seg]*t+10,5 [kbit]
4	147,35 [kbit/seg]*t+10,5 [kbit]
5	196,13 [kbit/seg]*t+10,5 [kbit]
6	225,75 [kbit/seg]*t+10,5 [kbit]

#### 4.1.1.3 Estimación Delay

Con *Network Calculus* se puede hacer una estimación del *delay* que se produce en el *router 3*. Este *delay* es la desviación horizontal entre las curvas de llegada y servicio. En la Figura 20 se muestra esta desviación.



**Figura 20: Delay en el router 3.**

El valor del *delay* es el tiempo en cual la curva de servicio es de 10.5 [kbit], que se calcula como sigue:

$$\beta_{R,T} = 10.5 \text{ [kbit]}$$

$$244.14 \text{ [kbit / seg]} * [t^* - 0.086016 \text{ [seg]}]^+ = 10.5 \text{ [kbit]}$$

$$t^* = 0,1290241101007619 \text{ [seg]}$$

Es decir, al aplicar *Network Calculus*, el *delay* que se produce en el *router 3* es de 0,12902 [seg] o 129.02 [mseg].

Otra forma de calcular el *delay* es utilizar la *toolbox CyNC* de Matlab la cual tiene dentro de sus funciones la *rtcplot*, la que aparte de graficar el *delay* entre dos funciones calcula el valor de este:

```
>> d = rtcplot(falfa, fbeta)
```

```
d =
```

```
0.129024110100762
```

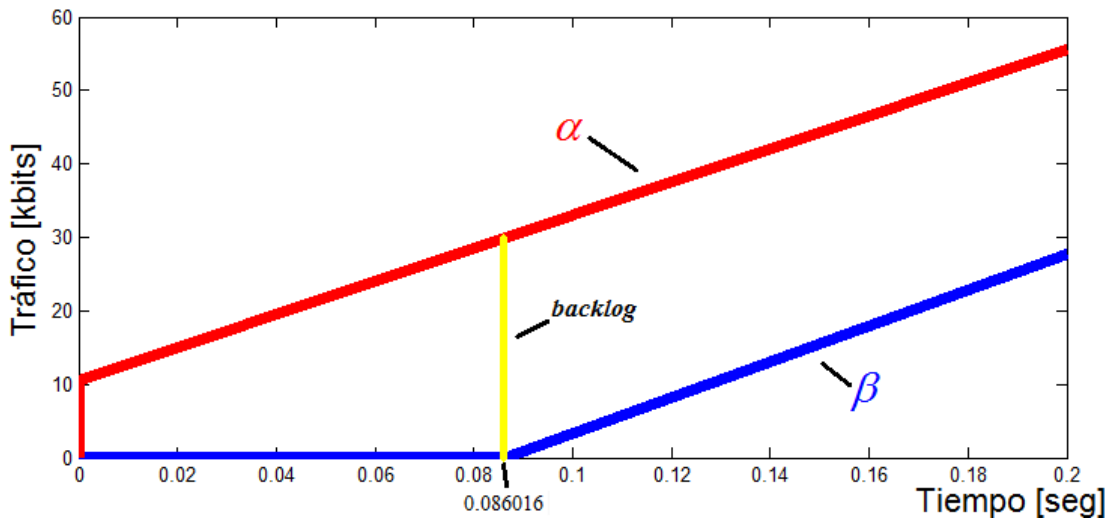
En la Tabla 13 se resume el *delay* estimado por *Network Calculus* para los 6 escenarios de velocidades de las interfaces en los *routers*.

**Tabla 13: Delay estimado por Network Calculus.**

Escenario	Delay [ms]
1	121,856
2	121,856
3	121,856
4	129,02
5	129,02
6	129,02

#### 4.1.1.4 Estimación *Backlog*

Se puede hacer una estimación del *Backlog* que se produce en el *router 3* con *Network Calculus*. Este *Backlog* es la desviación vertical entre las curvas de llegada y servicio. En la Figura 21 se muestra esta desviación.



**Figura 21: Backlog en el router 3.**

El valor del *backlog* es el tráfico al evaluar la curva de llegada en el tiempo 0.086016[seg].

$$\gamma_{r,b}(0.086016) = 225.75 [kbit / seg] * 0.086016 + 10.5 [kbit]$$

$$\gamma_{r,b}(0.086016) = 29,918112 [kbit]$$

Es decir, al utilizar *Network Calculus*, el *backlog* que se produce en el *router 3* es de 29,918112 [kbit].

Otra forma de calcular el *backlog* es utilizar la *toolbox CyNC* de Matlab la cual tiene dentro de sus funciones la *rtcplotv*, la que aparte de graficar el *backlog* entre dos funciones calcula el valor de este:

```
>> v = rtcplotv(falfa, fbeta)
v =
29.918111999999997
```

En la Tabla 14 se resume el *backlog* estimado por *Network Calculus* para los 6 escenarios de velocidades de las interfaces en los *routers*.

**Tabla 14: Backlog estimado por Network Calculus.**

Escenario	Velocidad Máxima Tráfico Entrada [kbits/seg]	Backlog [kbits]
1	147,35	23,1744576
2	197,3382353	27,47424565
3	275,1	34,1630016
4	147,35	23,1744576
5	196,525	27,4042944
6	225,75	29,918112

#### 4.1.2 Medición Tráfico de Salida Router 3

Se realizan mediciones del tráfico de salida del *router 3* para las seis configuraciones de velocidades de las interfaces de los *routers*. En la Tabla 15 se muestra el tráfico total de salida de la interfaz Ethernet 1/1 del *router 3* y el tráfico total de entrada en la interfaz Ethernet 1/0 del *router 3* para los distintos escenarios de velocidades en los *routers*.

**Tabla 15: Tráfico de entrada y salida para los 6 escenarios de velocidades en los routers**

Escenario	Tráfico Entrada [Mbytes]	Tráfico Salida [Mbytes]
1	54,50079346	54,50079346
2	72,63354492	72,63354492
3	101,6777344	101,6777344
4	54,48669434	54,48669434
5	72,64379883	72,64379883
6	83,49627686	83,49627686

Se puede observar que el tráfico de entrada es exactamente igual al tráfico de salida en el *router 3*. Debido a que la velocidad de salida que se configura en la interfaz Ethernet 1/1 de salida del *router 3* es siempre mayor a la velocidad de entrada, y además hay sólo un tráfico que atraviesa el *router 3* que no se está generando pérdidas de paquetes.

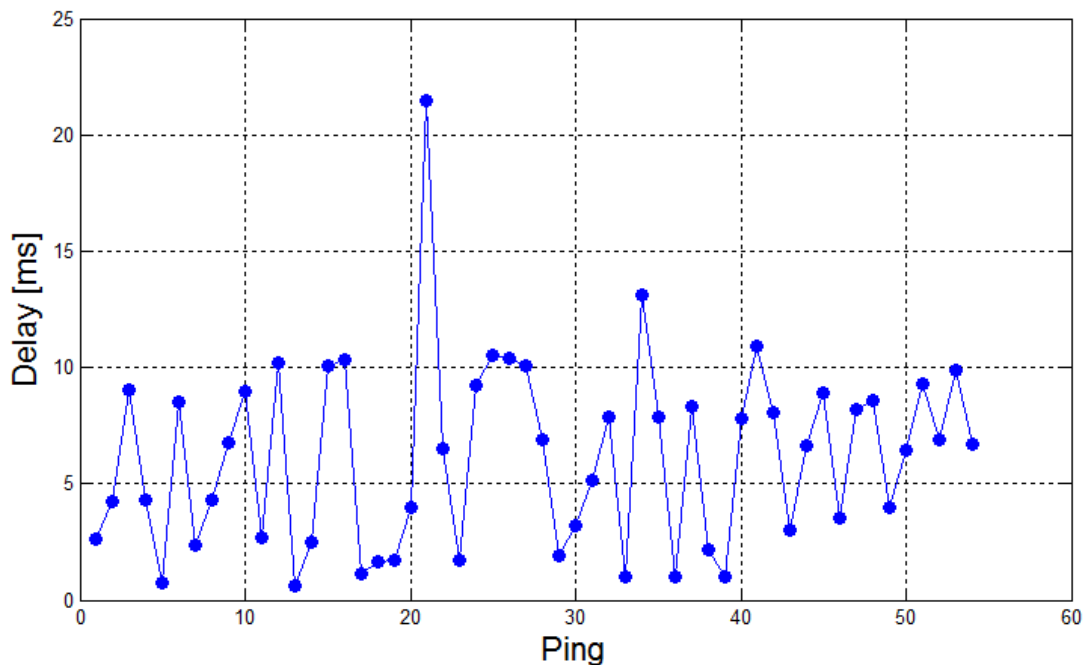
Para generar las pérdidas de paquetes se realizan cambios en las tablas de ruteo en pleno funcionamiento de los *routers*. En la Tabla 16 se muestra el tráfico total de salida de la interfaz Ethernet 1/1 del *router 3*, el tráfico total de entrada en la interfaz Ethernet 1/0 del *router 3* y la pérdida de paquetes para los distintos escenarios de velocidades en los *routers*.

**Tabla 16: Pérdidas de paquetes**

Escenario	Tráfico Entrada [Mbytes]	Tráfico Salida [Mbytes]	Pérdida Paquetes [%]
1	54,53283691	51,97064209	4,698444037
2	72,6361084	69,31768799	4,568554791
3	101,6482544	97,36212158	4,21663199
4	54,49438477	52,77941895	3,147050522
5	72,62457275	68,78320313	5,289352465
6	81,87103271	79,83563232	2,486105675

### 4.1.3 Medición de *Delay*

En el Gráfico 11 se muestra el *delay* que se produce en el *router* 3 al realizar una serie de pings entre el perturbador y la interfaz Ethernet 1/1 del *router* 3 cuando la velocidad de salida que se configura en el *router* 3 es de 244.14 [kbts/seg] y la velocidad de entrada de 224.6 [kbts/seg].



**Gráfico 11: Delay medido con comando ping**

El *delay* máximo que se produce es de 21.5 [ms] y el *delay* promedio de 6.207[ms] para las 6 velocidades en los *routers*. A continuación en la Tabla 17 se muestran los *delay* promedios y máximos para las seis configuraciones de velocidad que se utilizan. Se agrega además a la tabla una columna que indica el *delay* estimado por *Network Calculus*.

**Tabla 17: Delay promedio y máximo medido para los 6 escenarios de velocidades.**

Escenario	Delay promedio medido [ms]	Delay máximo medido [ms]	Delay estimado por NC [ms]
1	5,64943662	16,7	121,856
2	5,718150685	14,5	121,856
3	5,687492308	18,9	121,856
4	5,822526316	10,5	129,02
5	5,775537037	10,8	129,02
6	6,207111111	21,5	129,02

Se observa que para los 6 escenarios siempre el valor de *delay* estimado por *Network Calculus* es superior al *delay* máximo medido.

#### 4.1.4 Medición de *Delay* concatenación de nodos

Se realizan 10 mediciones en las cuales se envía una serie de pings desde el perturbador hacia la interfaz Ethernet 1/1 del *router* 4. En la siguiente Tabla 18 se muestra el *delay* promedio y máximo para el escenario 6 de la velocidad en los *routers*.

**Tabla 18: Delay promedio y máximo.**

Medida	Delay promedio [ms]	Delay máximo [ms]
1	15,51912281	26,5
2	15,21782609	27,8
3	15,62807018	28,4
4	15,65840336	29,2
5	15,72101852	27,8
6	15,80269565	27,7
7	15,97344538	27,7
8	15,90625000	27,8
9	15,51166667	28,8
10	15,88704348	27,8
Promedio	15,68255421	27,95

No hay grandes variaciones en el *delay* promedio en las 10 mediciones que se realizan y tampoco para los *delay* máximos que se registran.

A continuación en el Gráfico 12 se muestra el tráfico que ingresa a la interfaz Ethernet 1/0 del *router* 3.



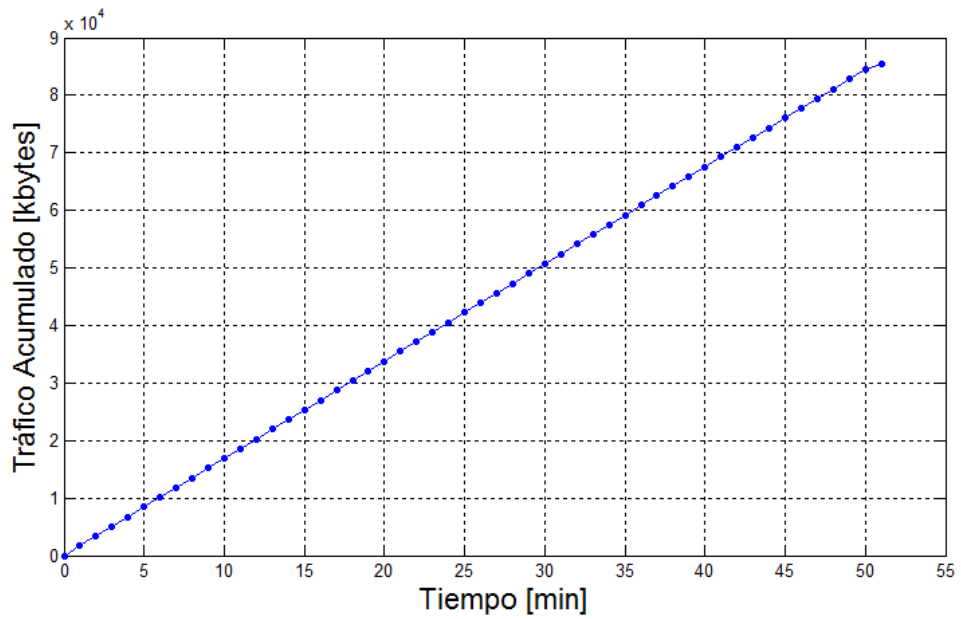


Gráfico 14: Tráfico acumulado en la interfaz Ethernet 1/0 del router 3.

#### 4.1.4.1 Obtención Curva de Servicio

La curva de servicio de los dos *routers* es la que resulta de la convolución entre la curva de servicio  $\beta_{router\ 3}$  del *router* 3 con la curva de servicio  $\beta_{router\ 4}$  del *router* 4. En la Figura 22 se muestra la curva de servicio para los *routers* 3 y 4; además la curva de servicio que se obtiene de la convolución entre estas.

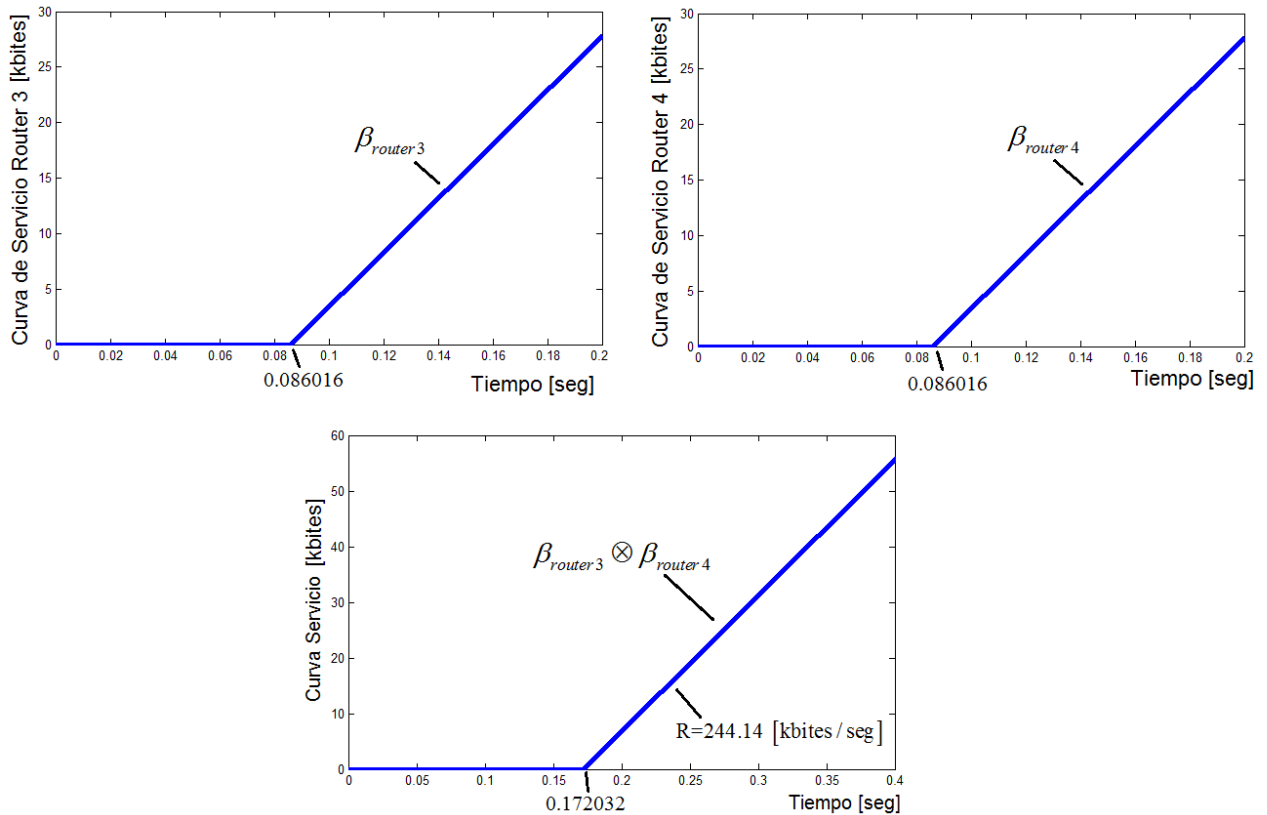


Figura 22: Curva de servicio del escenario 4.

#### 4.1.4.2 Obtención Curva de Llegada

La curva de llegada es una curva afín de la siguiente forma:

$$\gamma_{r,b}(t) = \begin{cases} rt + b & \text{si } t > 0 \\ 0 & \text{otro caso} \end{cases}$$

En donde la tasa que utilizamos  $r$  es la velocidad de entrada máxima que se mide en la interfaz Ethernet 1/0 del *router* 3 y es de 226.1 [kbites/seg]. El valor del parámetro  $b$  se obtiene a partir del tamaño máximo que atraviesa la interfaz y es de 1344 [bytes]= 10.5 [kbites]. En el Gráfico 15 se muestra la curva de llegada del *router* 3.

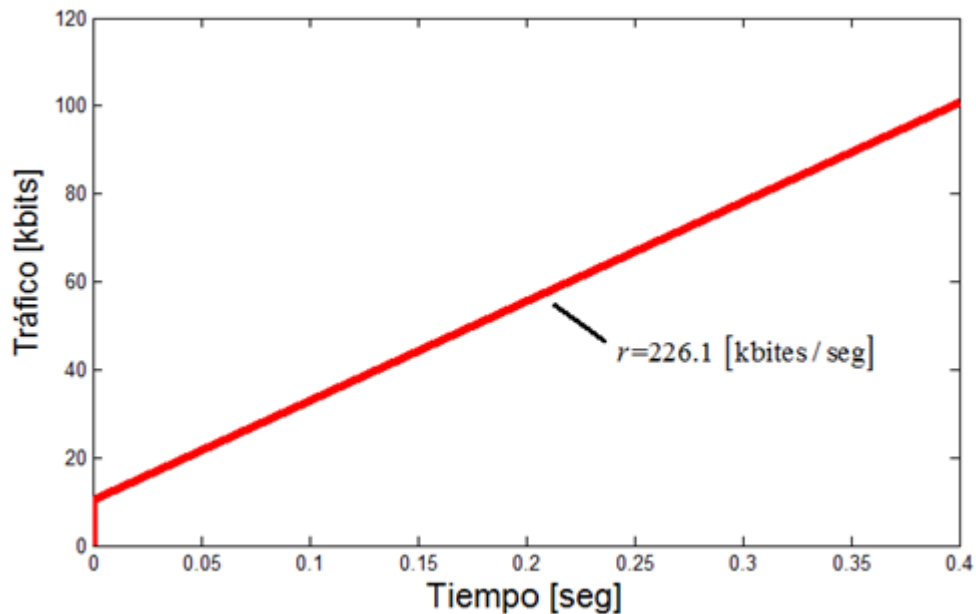


Gráfico 15: Curva de llegada del router 3.

#### 4.1.1.3 Estimación Delay

Con *Network Calculus* se puede hacer una estimación del *delay* que se produce al agregar el *router* 4. Este *delay* es la desviación horizontal entre las curvas de llegada y servicio. En la Figura 23 se muestra esta desviación.



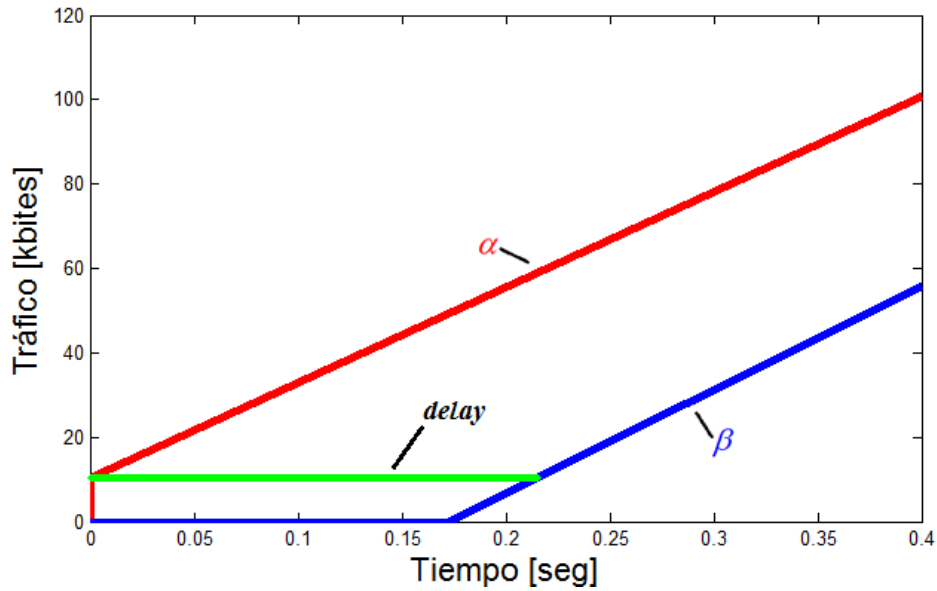


Figura 23: Delay en el router 3.

El valor del *delay* es el tiempo en cual la curva de servicio es de 10.5 [kbit], se calcula como sigue:

$$\beta_{R,T} = 10.5 [kbit]$$

$$244.14 [kbit / seg] * [t^* - 0.172032 [seg]]^+ = 10.5 [kbit]$$

$$t^* = 0,2150401101007619 [seg]$$

Es decir, al aplicar *Network Calculus*, el *delay* que se produce en el *router 3* es de 0,21504011[seg] o 215.04011[mseg].

Al utilizar la función *rtcploth* de la *toolbox CyNC* de Matlab el *delay* entre dos funciones es de:

```
>> d = rtcploth(falga, fconcat)
d =
0.215040110100762
```

#### 4.1.1.4 Estimación *Backlog*

Se puede hacer un estimación del *Backlog* que se produce en el *router 3* con *Network Calculus*. Este *Backlog* es la desviación vertical entre las curvas de llegada y servicio. En la Figura 24 muestra esta desviación.

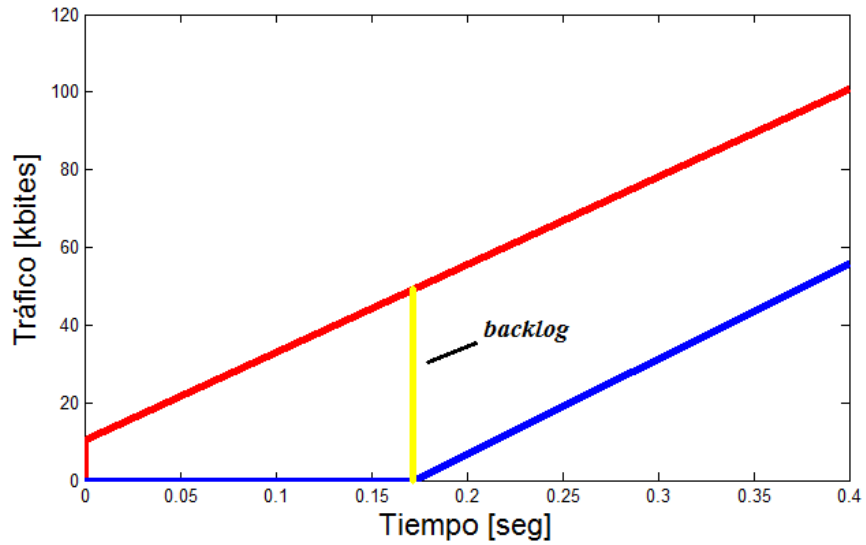


Figura 24: Backlog en el router 3.

El valor del *backlog* es el tráfico al evaluar la curva de llegada en el tiempo 0.172032 [seg].

$$\gamma_{r,b}(0.172032) = 226.1 [kbit / seg] * 0.172032 + 10.5 [kbit]$$

$$\gamma_{r,b}(0.172032) = 49.3964352 [kbit]$$

Es decir, al utilizar *Network Calculus*, el *backlog* que se produce en el *router 3* es de 49.3964352 [kbit].

Al utilizar la función *rtcplotv* de la *toolbox CyNC* de Matlab el *backlog* entre dos funciones es de:

```
>> v = rtcplotv(falga, fconcat)
v =
49.396435199999999
```

# 5

## Conclusiones

### 5.1 Herramientas

Las herramientas de código libre que se utilizan en el desarrollo de esta tesis cumplen con los requisitos que se necesitan para lograr el adecuado funcionamiento de los escenarios desarrollados.

GNS3 resulta ser una herramienta muy útil para la emulación de *routers* Cisco. Es fácil de utilizar y permite ahorrar los costos que implica adquirir equipos reales. Además se puede utilizar para realizar entrenamiento en configuraciones de variados elementos de red. El único inconveniente de esta herramienta es que utiliza un computador con gran cantidad de recursos, ya que al emular los *routers*, el uso de CPU y memoria se exigen al máximo.

*Flow-tools* permite realizar el monitoreo de los tráficos de manera muy precisa, ya que al comparar las velocidades de los flujos medidos con los configurados en los *routers*, resultan ser muy similares. La única desventaja de esta herramienta está es su difícil instalación y configuración, dado que se deben crear permisos adecuados, además de desbloquear los *firewalls* de los equipos en donde se instala.

La *toolbox* CyNC de Matlab demuestra ser una herramienta precisa para realizar las operaciones de *Network Calculus*. Los resultados que se obtienen al realizar cálculos de convolución, *delay* y *backlog* son los mismos que se obtienen al realizarlos de forma directa. En los tráficos que se generan no es difícil realizar cálculos de forma manual, pero de ser aplicados a tráficos con curvas de llegada y servicio más complejas, éstos no serán sencillos de realizar.

### 5.2 Aplicación de *Network Calculus*

*Network Calculus* resulta ser una herramienta que permite hacer estimaciones de parámetros tales como *backlog* y *delay*. Con la caracterización de los elementos de red por medio de curvas de llegada y servicio, se realiza un análisis determinístico de peor caso de estos parámetros.

Las estimaciones de *delay* que se hacen, resultan ser siempre superior al *delay* medido. Esto se debe principalmente a que *Network Calculus* es una herramienta que entrega la estimación en el peor caso; y además a una posible sobre estimación del parámetro de latencia de la curva de servicio del router.

En relación al parámetro *backlog*, a pesar de realizar las modificaciones en el tamaño de las colas en las interfaces de los *routers*, no se obtienen pérdidas de paquetes para ver si las estimaciones de *backlog*, que entrega el *Network Calculus*, son las correctas. Esto se debe principalmente a la gran capacidad de transmisión que tiene el *router*.

### 5.3 Objetivos

Se cumple con el objetivo general de aplicar *Network Calculus* al tráfico que se genera en una red simulada, lo que permite hacer estimaciones de *delay* y *backlog*. *Network Calculus* resulta ser una herramienta útil en la estimación de *delay* en el peor caso.

La *toolbox* CyNC resulta ser una herramienta simple en su uso y de gran ayuda para los cálculos de algebra *min-plus*, incluye funciones que permiten crear una gran variedad de curvas de servicio y llegada.

El software de simulación que se utiliza, satisface los requerimientos para poder implementar los diversos escenarios. Se simula una gran variedad de elementos de red, siendo la emulación de los *routers* de gran utilidad ya que reduce los costos en la implementación de los escenarios.

### 5.4 Proyecciones del trabajo

En cuanto a las proyecciones de este trabajo, existe la posibilidad de realizar escenarios de pruebas más complejos, siempre y cuando se cuente con equipos de mayor rendimiento a los utilizados, y así poder emular los dispositivos de red de mejor manera posible.

Además está la alternativa de aplicar *Network Calculus* a otros tipos de tráfico (no necesariamente *streaming* de video) que atraviesan de forma simultánea a los *routers*. Para esto se pueden utilizar generadores de tráfico o implementar máquinas virtuales simulando fuentes generadoras.

Realizar una mejor estimación de la curva de servicio de los *routers*. Puesto que es posible que se haya realizado un sobredimensionamiento de la latencia que se produce en el *router*. Además, se puede implementar políticas de encolamiento distintas a la FIFO, lo que implicaría una forma en la curva de servicio distinta a la que se utiliza.

El simulador GNS3 sólo simula dispositivos Cisco, lo cual limita en cierto modo su uso. Sería interesante poder integrar nuevos dispositivos, incluso de otros fabricantes. Los escenarios presentados pueden resultar sencillos, pero dejan abiertas las puertas a nuevos desarrollos que permitan el estudio de conceptos avanzados, como es el enrutamiento dinámico y la seguridad de red.

# 6

## Referencias

### 6.1 Bibliografía

- [1] Le Boudec J., Thiran P., "NETWORK CALCULUS, A Theory of Deterministic Queuing Systems for the Internet", Online Version of the Book Springer Verlag, 2004.
- [2] Holt A., "Network Performance Analysis, Using the J Programming Language", Springer, 2008.
- [3] Frances F., Fraboul C., Grieu J., "Using Network Calculus to optimize the AFDX network", 3rd European Congress ERTS Embedded real-time software, 25-27 Jan 2006.
- [4] Schioler H., Schwefel H., Hansen M., "CyNC - a MATLAB/SimuLink Toolbox for Network Calculus", 2nd International ICST Conference on Performance Evaluation Methodologies and Tools, 2007.
- [5] González M., "Modelamiento de tráfico en redes de datos", Universidad de Chile, 2007.
- [6] Bredel M., Bozakov Z., Jiang Y., "Analyzing Router Performance Using Network Calculus with External Measurements", 2010.
- [7] Michael L., "Network Flow Analysis", No Starch Press, 2010.
- [8] Fahimeh J., Jantsch A., and Lu Z., "Worst-Case Delay Analysis of Variable Bit-Rate Flows in Network-on-Chip with Aggregate Scheduling", 2011.
- [9] Fidlera M., Sanderb V., Klimalab W., "Traffic shaping in aggregate-based networks: implementation and analysis", 2004.
- [10] Almes G., Kalidindi S., Zekauskas M., "A One-way Delay Metric for IPPM", RFC 2679, September 1999.
- [11] Almes G., Kalidindi S., Zekauskas M., "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.
- [12] Paxson V., Almes G., Mahdavi J., M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [13] Almes G., Kalidindi S., Zekauskas M., "A One-way Packet Loss Metric for IPPM", RFC 2680, September 1999.
- [14] Kurose J., Ross K., "Computer Networking", Addison-Wesley, 2004.

- [15] Chang C., "Performance Guarantees in Communication Networks", Springer, 2000.
- [16] Keshav, "Computer Networking: An Engineering Approach2, Prentice Hall, 1996.
- [17] Stiliadis D., Varma A., "Rate latency servers: a general model for analysis of traffic scheduling algorithms", IEEE Infocom, 1996.
- [18] Crawley E., Nair R., Rajagopalan B., Sandick H., "A Framework for QoS-based Routing in the Internet", RFC 2386, 1998.
- [19] GNS3, <http://www.gns3.net/>.
- [20] Flow-tools, <http://code.google.com/p/flow-tools/>.
- [21] Ubuntu, <http://www.ubuntu.com/>.
- [22] Matlab, <http://www.mathworks.com/products/matlab/>
- [23] VideoLan, <http://www.videolan.org/>
- [24] VirtualBox, <https://www.virtualbox.org/>

## 6.2 Acrónimos

API: Application Programming Interface  
ASF: Advanced Streaming Format  
ATM: Asynchronous Transfer Mode  
CBR: Constant Bit Rate  
CyNC: Cyclic Network Calculus  
DHCP: Dynamic Host Configuration Protocol  
EIGRP: Enhanced Interior Gateway Routing Protocol  
FIFO: FIRST IN FIRST OUT  
FTP: File Transfer Protocol  
GNS3: Graphical Network Simulator 3  
GPS: Generalized Processor Sharing  
GR: Guaranteed Rate  
GUI: Graphical User Interface  
ICMP: Internet Control Message Protocol  
IP: Internet Protocol  
NAT: Network Address Translation  
OSPF: Open Shortest Path First  
PAT: Port Address Translation  
PGPS: Packet Generalized Processor Sharing  
PID: Process Identifier  
PIX: Private Internet eXchange  
QoS: Quality of Service  
TCP: Transmission Control Protocol  
UDP: User Datagram Protocol  
UTC: Universal Time Coordinated  
URL: Uniform Resource Locator  
VLAN: Virtual Local Area Network  
VLC: Video Lan Client  
WMA: Windows Media Audio  
WMV: Windows Media Video

# 7

## Anexos

### 7.1 Gráficos de velocidades en los routers

A continuación se muestran los gráficos para los 5 escenarios de velocidades en los routers.

Gráficos para escenario 1:

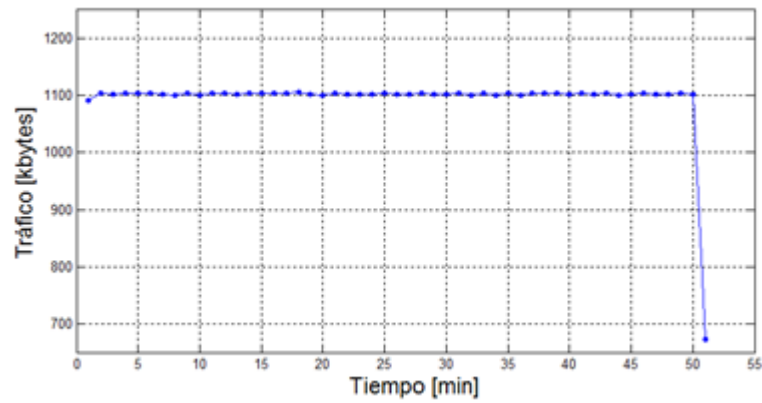


Gráfico 16: Tráfico entrada router 3 para escenario 1.

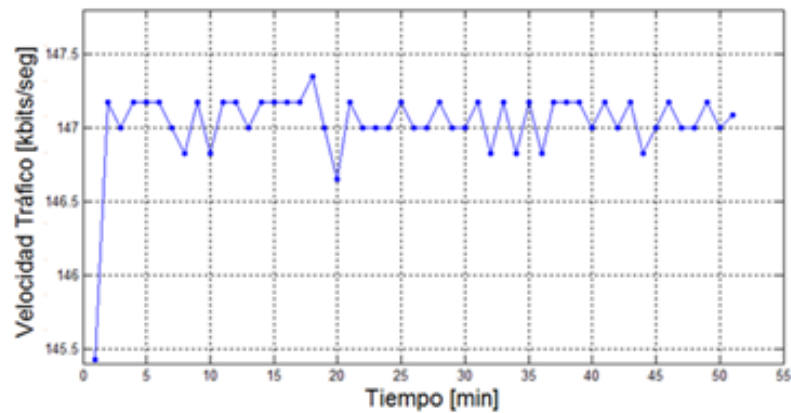
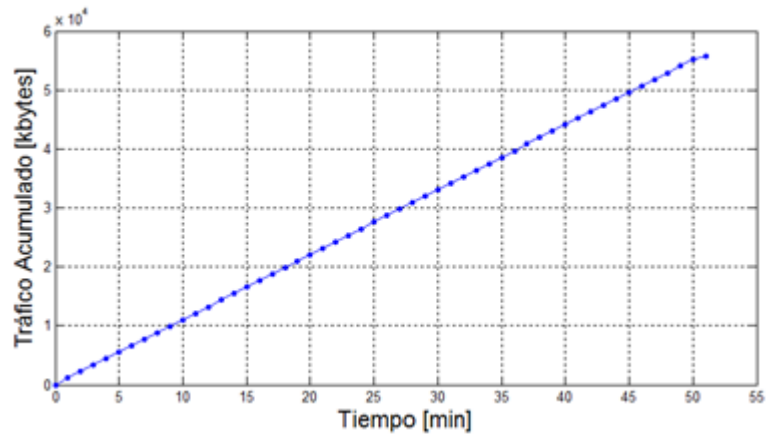


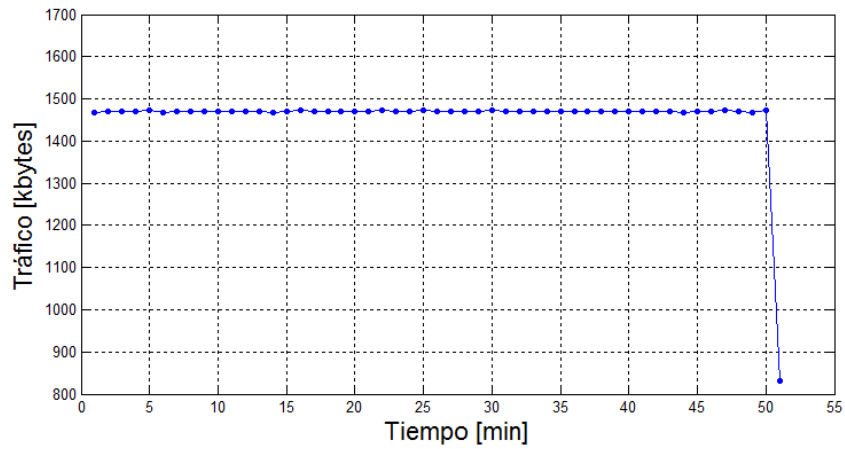
Gráfico 17: Velocidad de entrada del router 3 para escenario 1.



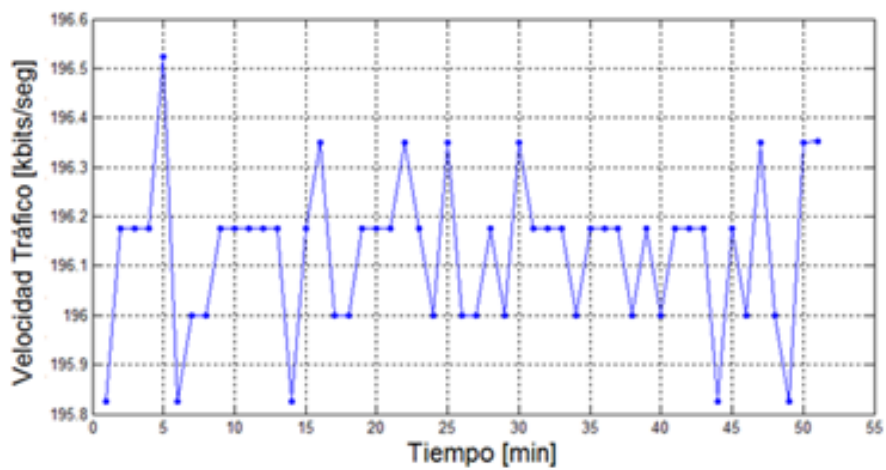


**Gráfico 18: Tráfico acumulado del router 3 para escenario 1.**

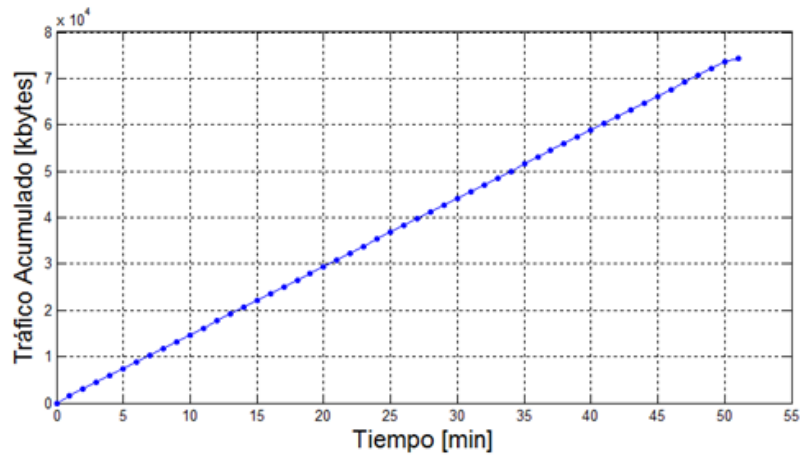
Gráficos para escenario 2:



**Gráfico 19: Tráfico entrada router 3 para escenario 2.**

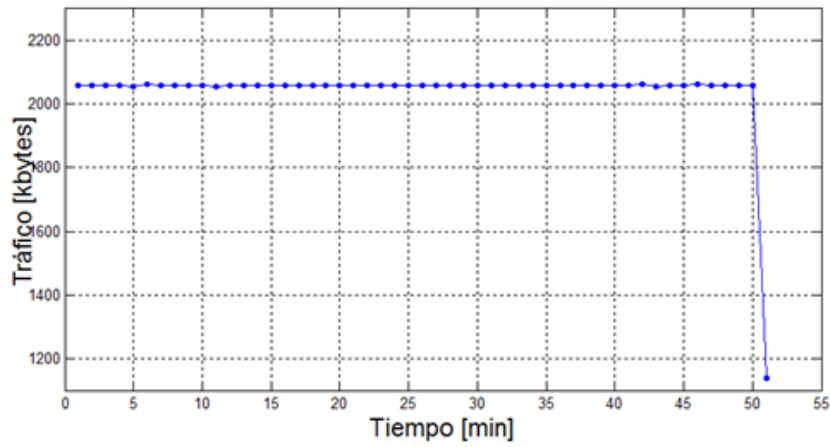


**Gráfico 20: Velocidad de entrada del router 3 para escenario 2.**

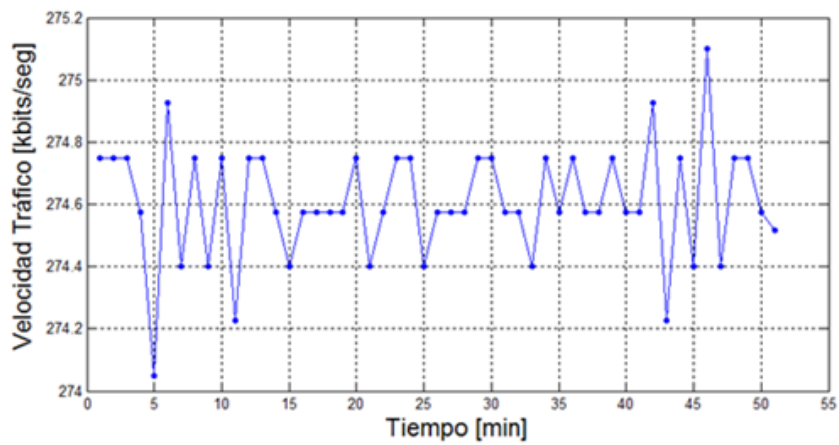


**Gráfico 21: Tráfico acumulado del router 3 para escenario 2.**

Gráficos para escenario 3:



**Gráfico 22: Tráfico entrada router 3 para escenario 3.**



**Gráfico 23: Velocidad de entrada del router 3 para escenario 3.**

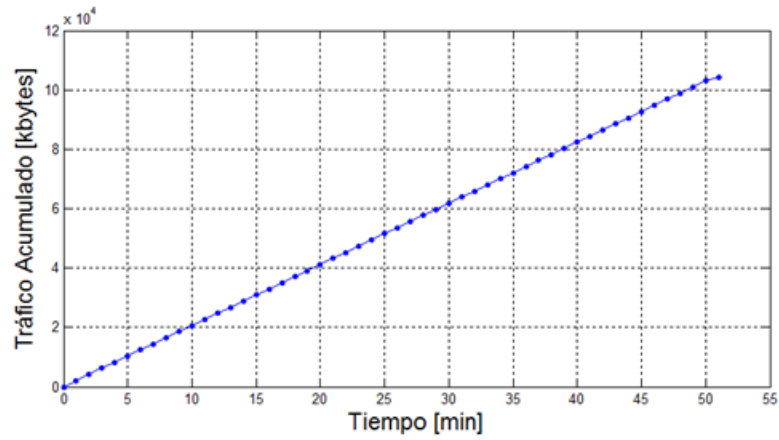


Gráfico 24: Tráfico acumulado del router 3 para escenario 3.

Gráficos para escenario 4:

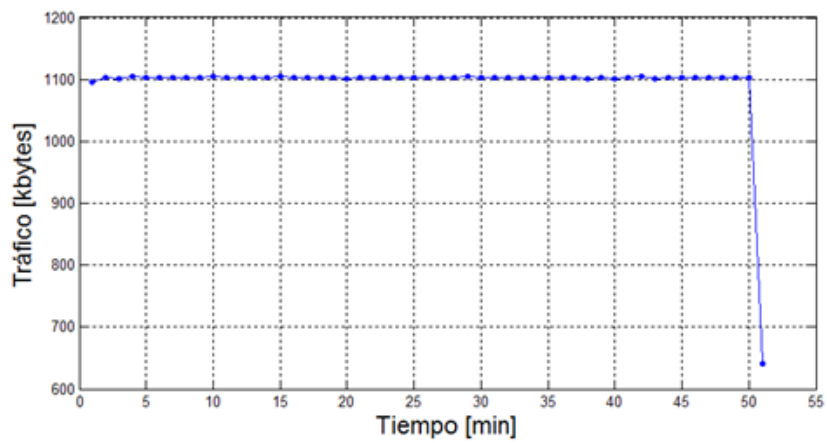


Gráfico 25: Tráfico entrada router 3 para escenario 4.

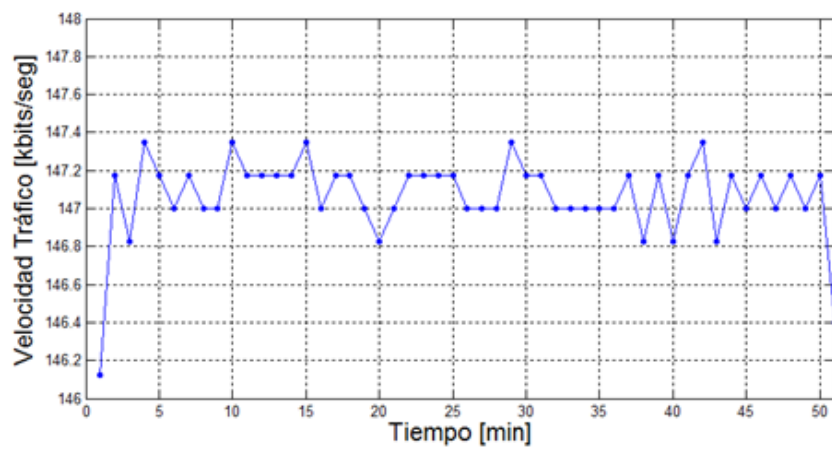
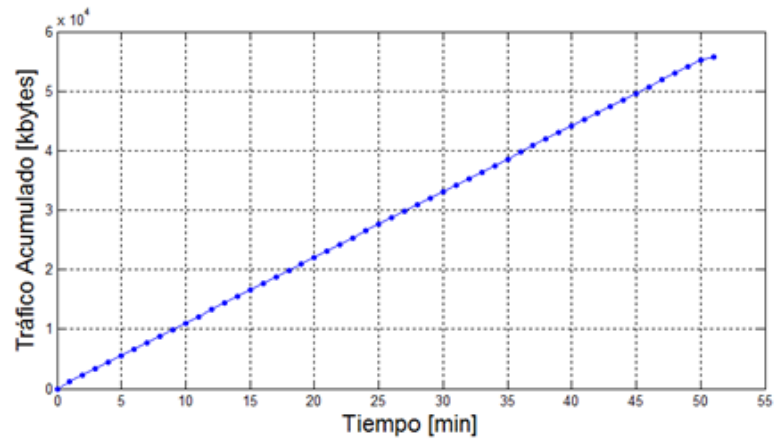
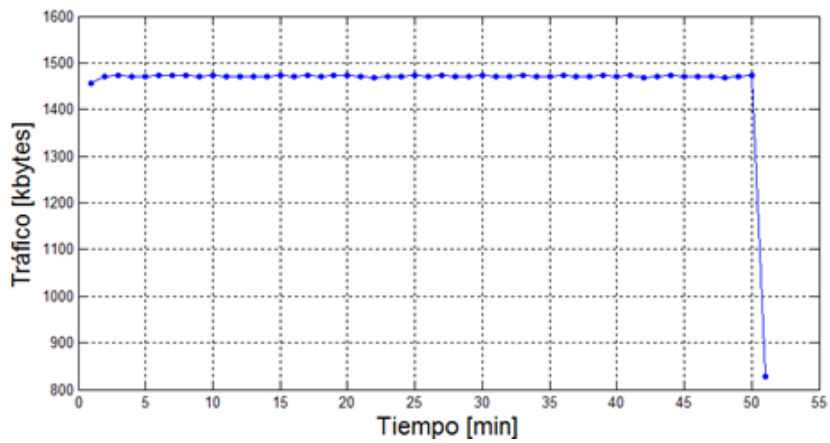


Gráfico 26: Velocidad de entrada del router 3 para escenario 4.

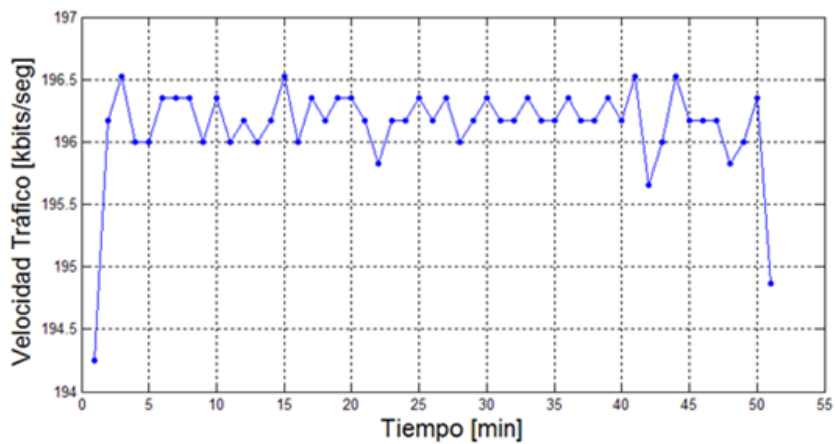


**Gráfico 27: Tráfico acumulado del router 3 para escenario 4.**

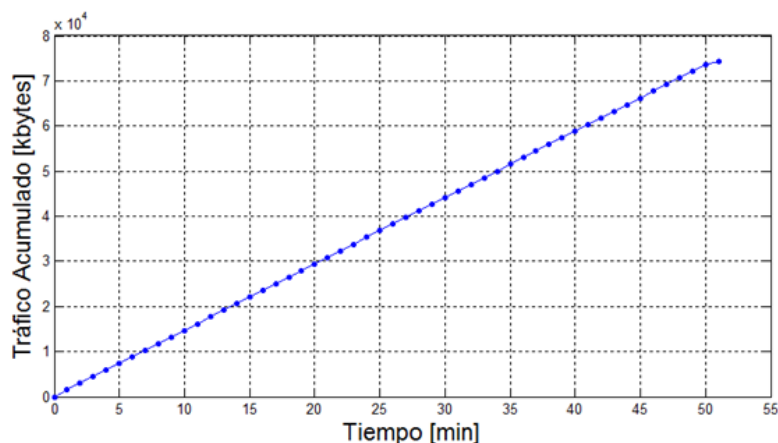
Gráficos para escenario 5:



**Gráfico 28: Tráfico entrada router 3 para escenario 5.**



**Gráfico 29: Velocidad de entrada del router 3 para escenario 5.**



**Gráfico 30: Tráfico acumulado del router 3 para escenario 5.**

## 7.2 Instalación y Configuración Software

### 7.2.1 Instalación y Configuración GNS3

Paso 1: Descargar GNS3

Se descarga el archivo GNS3-0.7.4-win32-all-in-one desde la dirección web: <http://www.gns3.net/download>, el cual incluye Dynamips, Qemu/Pemu, Putty, VPCS, WinPCAP y Wireshark

Paso 2: Instalar GNS3

Se realiza doble click al archivo que se descarga y aparecen las ventanas típicas de instalación de programas. GNS3 depende de otros programas para operar. Estas dependencias incluyen WinPCAP, Dynamips, y Pemuwrapper. Estos componentes juntos con GNS3 son todos elegidos por defecto para la instalación.

Paso 3: Instalar la IOS

Una vez que se instala GNS3, se crea una carpeta llamada IOS en el disco C donde se guarda el archivo de la IOS del *router* Cisco 2600s. Luego en *edit* de GNS3 se selecciona “*IOS images and hypervisors*” y se selecciona el archivo en el directorio donde se guarda la imagen del *router* Cisco y se pone guardar.

Con estos 3 pasos se puede empezar a crear distintos escenarios de red.

### 7.2.2 Instalación y Configuración VirtualBox

Paso 1: Descargar VirtualBox

Se descarga el archivo VirtualBox-4.1.6-74713-Win desde la dirección web: <https://www.virtualbox.org/wiki/Downloads>

## Paso 2: Instalar VirtualBox

La instalación es simple, se tiene que elegir la ubicación donde se instala y aceptar.

## Paso 3: Agregar Máquinas Virtuales

Una vez instalado el programa, en la opción Máquina se selecciona agregar y se buscan las máquinas virtuales Cliente\_Tesis, Servidor\_Tesis y Medidor1, en las cuales se tiene instalado el sistema operativo Ubuntu y flow-tools.

### 7.2.3 Instalación y Configuración Flow-tools

#### Paso 1: Descargar e Instalar Flow-tools

Se utiliza el gestor de paquetes de Ubuntu por lo que la descarga e instalación es simple y rápida.

#### Paso 2: Configuración de Flow-capture

Se configura el recolector *flow-capture* desde el paquete de instalación de flow-tools. Su configuración se establece en el archivo `/etc/flow-tools/flow-capture.conf`. En dicho archivo, el contenido por defecto es:

```
# Configuration for flow-capture
#
# Robin Elfrink <robin@a1.nl>
#
# Every line is basically just the options to flow-capture, see
# flow-capture(1) for explanation.

# Example 1:
# Capture flows from router at 10.1.1.10, listening at port 3000.
# Store flows in /var/flow/myrouter.
-w /var/flow/myrouter 0/10.1.1.10/3000

# Example 2:
# Capture flows from router at 10.3.2.6, listening at port 3002.
# Store flows in /var/flow/mysecondrouter. Rotate files every
# 5 minutes.
-w /var/flow/mysecondrouter -n 275 0/10.3.2.6/3002

# Example 3:
# Same as above, but only listen at address 10.3.2.5, and store
# files under 'YYYY/YYYY-MM/YYYY-MM-DD' directories.
-w /var/flow/mysecondrouter -n 275 -N 3 10.3.2.5/10.3.2.6/3002
```

Se puede ver que el archivo está claramente comentado y facilita su comprensión, pero aún así se describen algunos de sus parámetros de configuración principales en la Tabla 19. Para más información, consultar manual [20].

**Tabla 19: Parámetros de flow-capture.**

Parámetro	Descripción
-w workdir	Específica el directorio principal donde el recolector almacena la información proveniente de la sonda.
-n rotations	Indica el número de archivos que almacena flow-capture durante un día, o lo que es lo mismo el intervalo temporal para guardar la información en el disco. El valor por defecto es 95, es decir, crea un archivo cada 15 minutos.
-e expire count	Número de archivos máximo a almacenar por el recolector. En caso de sobrepasarse se procede a la eliminación de los más antiguos. Se comprueban todos los subdirectorios del directorio principal del recolector.
-E expire size	Tamaño máximo ocupado por los archivos capturados por el recolector. En caso de sobrepasarse se procede a la eliminación de los más antiguos. Se comprueban todos los subdirectorios del directorio principal del recolector.
-v pdu version	Específica la versión de NetFlow a usar.
-N nesting level	Indica el formato de anidamiento de los archivos: -3 YYYY/YYYY-MM/YYYY-MM-DD/flow-file -2 YYYY-MM/YYYY-MM-DD/flow-file -1 YYYY-MM-DD/flow-file 0 flow-file 1 YYYY/flow-file 2 YYYY/YYYY-MM/flow-file 3 YYYY/YYYY-MM/YYYY-MM-DD/flow-file Por defecto el valor es 3.
localip/remoteip/port	Establece la dirección local donde se escucha, la dirección remota a la que se aceptarán los flujos y el puerto donde se atienden las llegadas. Un valor 0 en los dos primeros hará válida cualquier dirección.

El archivo de configuración *flow-capture.conf* del Servidor es

```
-p /var/run/flow-capture.pid -n 287 -w /var/db/flows/router1 -S 5 192.168.50.50/0/5678
```

Y el contenido del archivo de configuración *flow-capture.conf* en el Cliente:

```
-p /var/run/flow-capture.pid -n 287 -w /var/db/flows/router1 -S 5 192.168.60.60/0/9996
```

## 7.2.4 Instalación y Configuración CyNC

Paso 1: Descargar CyNC

Se descarga la última versión de CyNC desde la dirección [http://www.control.aau.dk/~henrik/CyNC\\_v0.2/CyNC\\_v0.2.rar](http://www.control.aau.dk/~henrik/CyNC_v0.2/CyNC_v0.2.rar)

## Paso 2: Descargar e instalar RTC toolbox

Para instalar CyNC se tiene que descargar adicionalmente la toolbox RTC desde la dirección web: <http://www.mpa.ethz.ch/Rtctoolbox/Overview>.

Una vez descargado este archivo, se descomprime en un directorio, por ejemplo: C:\Users\Documents\MATLAB\RTCToolbox\rtc.

Luego, se abre Matlab y se cambia al directorio rtc, y se ejecuta rtc\_install.

## Paso 3: Instalar CyNC

Una vez instalado correctamente RTC toolbox, se descomprime el archivo descargado en el Paso 1 en un directorio por ejemplo C:\Users\Documents\MATLAB\CyNC\_v0.2, con lo cual ya se puede utilizar todas las funciones de esta toolbox.

## 7.3 Descripción DVD

Los DVD contienen las carpetas:

- 1- Escenarios: Todos los escenarios con los archivos de configuración de los *routers*.
- 2- IOS: Imagen de la IOS del *router* 2600.
- 3- Mediciones: En varios archivos Microsoft © Excel, todos los resultados de las mediciones.
- 4- Software: GNS3, VirtulBox, flow-tools, CyNC y RTC toolbox.
- 5- Video: Clip de video que utiliza para el *streaming*.
- 6- Máquinas Virtuales: Servidor, Cliente y Medidor.