UNIVERSIDAD PRIVADA DEL NORTE

ESCUELA DE POSGRADO

PROPUESTA DE IMPLEMENTACIÓN DE CONTROL DE TRÁFICO DE LA RED CON LINUX PARA MEJORAR LA CALIDAD DE SERVICIO DE LA RED LAN EN UNA UNIVERSIDAD PRIVADA DE LA CIUDAD DE CAJAMARCA.

Tesis para optar el grado de **MAGISTER** en:

Ingeniería de Sistemas con mención en Gerencia de Tecnologías de la Información.

Autor:

Bachiller. Juan Carlos Cabanillas Chávez

Asesora:

Dra. Ena Cecilia Obando Peralta

Cajamarca – Perú

2015

DEDICATORIA

A mis padres y hermano que me apoyaron y acompañaron durante todo este proceso, brindándome su apoyo constante durante toda mi carrera, a mis hijos Johao Mateo y Favio Angel, que son el motor de mi vida y la razón de mi existir.

AGRADECIMIENTO

Expreso mis agradecimientos a:

Dra. Ana Cecilia Obando por las horas dedicadas y por la dirección de este trabajo.

"A todas las personas que me han apoyado constantemente, a todas los que me han ayudado a crecer profesionalmente, a las que día a día me han hecho más feliz y a las que me han enseñado que "el éxito no es más que la constancia en el propósito"".

RESUMEN

En ésta tesis se plantea el análisis y estudio de las diferentes Arquitecturas de Calidad de Servicio (QoS), como: Servicios Integrados (IntServ), que se basa en garantizar QoS a través de la reserva de recursos extremo a extremo para cada flujo, y la arquitectura Servicios Diferenciados (DiffServ), en donde se provee a ciertas aplicaciones o protocolos, determinadas prioridades sobre el resto del tráfico en la red.

Se describirá las posibilidades que ofrece el sistema operativo LINUX, dentro del networking para mejorar la Calidad de Servicio (Quality of Service) en la red LAN, revisando detalladamente teoría de colas y los mecanismos usados para control de tráfico, como son: scheduling, policing, Clasifying, etc. También describiremos de forma detallada el tema de disciplina de colas, haciendo énfasis en aquellas que se pueden aplicar en el campo de control de tráfico de la red.

Abordaremos el tema de enrutamiento avanzado con LINUX, y su aplicabilidad al control de tráfico, presentando las herramientas con que cuenta el sistema operativo con Kernel 2.6.x., que incluye IPTABLES el cual utiliza métodos de marcado de paquetes en un firewall, e IPROUTE2, donde veremos el uso de la utilidad tc (Traffic Control) que permite hacer el control de tráfico de la red, aplicando los conceptos como disciplina de colas, clases y filtros.

Al final se planteara una propuesta de Calidad de Servicio (Quality of Service - QoS) con control de tráfico de la red con Linux. Se realiza la Implementación a través de las herramientas que posee el sistema operativo Linux para el control de tráfico en la interfaz de salida hacia Internet de la red LAN de la Universidad.

ABSTRACT

In this thesis the analysis and study of the different architectures Quality of Service (QoS), as arises: Integrated Services (IntServ), which is based on guaranteeing QoS by reserving resources end-to-end for each stream, and the Differentiated Services (DiffServ) architecture, which is provided to certain applications or protocols, certain priority over other traffic on the network.

The potential of the Linux operating system within the networking to improve QoS (Quality of Service) in the LAN, reviewing in detail queuing theory and the mechanisms used to control traffic, such as will be described: scheduling, policing , Clasifying, etc. We also describe in detail the issue queue discipline, emphasizing those that can be applied in the field of traffic control network.

We will address the issue of advanced routing with LINUX, and their applicability to traffic control, presenting the tools available to the operating system kernel 2.6.x, which includes IPTABLES which uses methods of marking packets on a firewall, and iproute2 where we will see the use of utility tc (Traffic Control) that allows control of network traffic, applying the concepts and queue discipline, classes and filters.

With traffic control network with Linux - a proposal to end Quality of Service (QoS Quality of Service) is raised. Implementation is through the tools that has the Linux operating system to control traffic on the Internet output interface to the LAN of the University.

ÍNDICE GENERAL

ÍNDICE DE TABLAS e ILUSTRACIONES

ÍNDICE DE GRÁFICOS Y FIGURAS

I. INTRODUCCIÓN

1) Problema de Investigación

A. Realidad Problemática

Las redes IP actuales (IPv4) proporcionan un envío de tráfico de mejor esfuerzo (Best-effort), en el cual se proporciona una mayor valorización por la distribución de información, sin ofrecer garantías de Calidad de Servicio (Quality of Service - QoS). De esta forma los usuarios que utilizan redes IP están demandando garantías de ancho de banda.

En la red LAN de la Universidad existen diversas aplicaciones como Base de datos, Aplicaciones Web, Correo corporativo, etc. De las cuales se necesitan tener tiempos de respuestas rápidos y garantías de acceso a estos servicios. En este contexto con la utilización de redes convencionales con una política de "best effort", sistema convencional que es la política de ”mejor esfuerzo”, con el que cuenta actualmente la red LAN de la Universidad, no permite garantizar Calidad de Servicio (Quality of Service - QoS). Aunado a esto se ha venido dando un crecimiento sostenido en número de usuarios y aplicaciones en la red LAN de la Universidad, los cuales utilizan gran cantidad de recursos de red (Networking), haciendo un mal uso de los recursos de la red, como el ancho de banda disponible (4 MB), accediendo a aplicaciones de Streaming, descarga masiva de archivos de gran tamaño y acceso a aplicaciones interactivas de audio y video. En consecuencia se genera la saturación de la red, retardando el envío y recepción de información, afectando en gran medida la realización de las actividades cotidianas de los usuarios de la red LAN, conformados por la plana docente y administrativa de la Universidad.

En base a esto surge la necesidad de mejorar la calidad y el acceso a estos servicios. Para brindar Calidad de Servicio (QoS), es necesario realizar un control sobre el tráfico saliente en la red LAN de la Universidad, para poder administrar el ancho de banda, filtrar, clasificar y aplicar prioridades distintas a los diferentes tipos de tráfico y servicios que brinda la Universidad y a las diferentes aplicaciones y servicios de Internet, donde acceden los usuarios pertenecientes a la red LAN. De esta manera se lograra mejorar el tráfico de re en la red LAN y el acceso a los recursos como son las aplicaciones internas u otras redes, como Internet, en forma eficiente.

B. Formulación del Problema

¿En qué medida el control de tráfico de la red con Linux mejorara la Calidad de Servicio (Quality of Service - QoS) de la red LAN en una Universidad Privada de la ciudad de Cajamarca, 2015?

C. Justificación de la Investigación

Teórica

La presente investigación encuentra su justificación teórica en la teoría desarrollada por Paul Rusty Russell sobre Netfilter/Iptables y la teoría desarrollado por Alexey Kuznetsov sobre Iproute2, herramientas que permiten el control de tráfico de la red en Linux.

Practica

La presente investigación encuentra su justificación práctica en que ayudara a mejorar Calidad de Servicio (Quality of Service - QoS) de la red LAN en una Universidad Privada de la ciudad de Cajamarca, 2015.

2) Objetivos

A. Objetivo General

Determinar en qué medida el control de tráfico de la red con Linux mejorara la Calidad de Servicio (Quality of Service - QoS) de la red LAN en una Universidad Privada de la ciudad de Cajamarca, 2015.

B. Objetivos Específicos

- Analizar el tráfico de la red LAN para poder implementar políticas de control de tráfico de la red.
- Analizar el control de tráfico de la red mediante las herramientas IProute2 y Netfilter en la red LAN.
- Medir la Calidad de Servicio (Quality of Service - QoS) en la red LAN.

3) Método

A. Tipo de Investigación

- Según su fin:

  Aplicada: Aquella investigación que tiene como objetivo práctico. Elaborar y/o aplicar propuestas prácticas para solucionar problemas específicos o investigar soluciones de uso inmediato. (Caballero Romero, 2000)

- Según nivel de alcance del conocimiento:

  Correlacional o causal: "Aquella investigación que busca conocer la relación o grado de asociación que exista entre dos o más conceptos, categorías o variables en un contexto en particular… para predecir cómo se puede comportar un concepto o una variable al conocer el comportamiento de otras variables vinculadas". (Hernandez Sampieri, Fernandez Collado, & Baptista Lucio, 2010).

B. Diseño de Investigación

Experimental: Se realiza en base a la demostración concreta de una hipótesis. Es la investigación que mayor control ejerce sobre las variables y aunque se realice en medio natural o artificial, siempre tiene la posibilidad de manipular la variable independiente (factores causales) para estudiar su influencia en la variable dependiente (consecuencias). Se trabaja con una muestra a la que se aplica el experimento y una muestra control a quien no se le aplica este, pero que sirve para contrastar los resultados. (Hernandez Sampieri, Fernandez Collado, & Baptista Lucio, 2010)

C. Método de la Investigación

Análisis – Síntesis

Análisis: Método lógico caracterizado por separar el todo en sus partes o en sus partes fundamentales para estudiar su naturaleza, función /o su significado. (Rodríguez, 1968)

Síntesis: Método lógico caracterizado por estudiar el fenómeno reuniendo el mismo en sus partes esenciales y sobresalientes. Es decir, se realiza a través de la conformación de algo completo a través de elementos específicos. (Rodríguez, 1968)

Deductivo – Inductivo

Deducción: Método lógico caracterizado por pasar de lo general a lo particular. Es decir que la conclusión no es nueva, se sigue necesariamente de las premisas.(Rodríguez, 1968).

D. Población

Red LAN de una Universidad Privada de la ciudad de Cajamarca, 2015**.**

E. Técnicas e Instrumentos

Técnicas:

- Análisis de datos: Análisis significa la categorización, ordenamiento, manipulación y resumen de datos para responder a las preguntas de investigación. El propósito del análisis es reducir los datos a una forma entendible e interpretable para que las relaciones de los problemas de investigación puedan ser estudiadas y probadas. La elección del tipo de análisis a utilizar depende de los datos que hayamos recolectado, del nivel de medición de las variables, de la manera como se hayan formulado las hipótesis, del interés del investigador. (Kerlinger, 1975)

Instrumentos: Se denomina instrumentos a aquellos elementos que permiten conducir el estudio y /o registrar los datos obtenidos en el proceso de investigación. En el caso de investigaciones prácticas, los instrumentos se refieren a aquellos materiales, equipos, máquinas, dispositivos, etc. usados para conducir el estudio.

- Software Ntopng para analizar tráfico de red.
- Herramienta tc (Traffic Control)
- Herramientas speedometer
- Herramienta Netfilter/iptables
- Herramienta Iperf.

F. Planteamiento de la Hipótesis

El control de tráfico de la red con Linux mejorara significativamente la Calidad de Servicio (Quality of Service - QoS) de la red LAN en una Universidad Privada de la ciudad de Cajamarca, 2015.

G. Variables

G.1. Variable Dependiente:
Calidad de Servicio (Quality of Service - QoS).

G.2. Variable Independiente:
Control de Tráfico de la red.

A. Operacionalización de Variables

VARIABLE	DEFINICION CONCEPTUAL	DEFINCIION OPERACIONAL	DIMENSION	INDICADOR
1.- Control de Tráfico de la Red.	El tráfico de red es el flujo de datos que se transfieren mediante un medio físico por la red. Flujo de datos en un contexto de dominios de colisión y de broadcast se centra en la forma en que las tramas se propagan a través de la red. Se refiere al movimiento de datos a través de los dispositivos de Capa 1, 2 y 3 y a la manera en que los datos deben encapsularse para poder realizar esa travesía en forma efectiva. (Cisco Systems, 1999)	Control de tráfico de la red con Linux hace referencia al subsistema de colas de paquetes en una red o Dispositivo de red y consiste de diversas operaciones: Clasificación, Policing, Scheduling, Shaping.	• Colas	• Disciplina de cola

2.- Calidad de Servicio (Quality of Service - QoS)	Calidad de Servicio (Quality of Service - QoS) se refiere a la capacidad de una red para proporcionar un mejor servicio al tráfico de red seleccionado sobre diversas tecnologías, incluyendo Frame Relay, modo de transferencia asíncrono (ATM), Ethernet y 802.1 redes y redes enrutadas en IP que pueden utilizar cualquiera o todas estas tecnologías subyacentes. (Cisco Systems, 1999)	Calidad de Servicio (Quality of Service - QoS) está relacionado con la planificación de los elementos de la red de acuerdo con la demanda de las aplicaciones, para ofrecer un mínimo nivel de garantía que satisfaga los requerimientos de tráfico; mínimo retraso de envió de los datos, mínima variación de los retrasos, ancho de banda adecuado para él envió satisfactorio de los datos.	• Ancho de banda	• Caudal máximo que se puede transmitir en Bytes/Segundo.
			• Tasa de perdidas	• Proporción de paquetes perdidos.
			• Retardo o latencia	• Tiempo medio que tarda en llegar los paquetes.

Fuente: Elaboración propia

II. MARCO TEÓRICO

1. ANTECEDENTES

- La tesis titulada Filtrado de paquetes IP a través de expresiones regulares sobre capa de aplicación y gestión del ancho de banda con Software Libre, en enero del 2010, publicado por la Universidad Mayor Facultad de Ingeniería, Temuco, Chile. (Inzunza Sanhueza & Marileo Ñancupil, 2010).
  Tiene como objetivo Implementar tecnologías basadas en herramientas de uso libre, sobre la plataforma GNU/Linux 2.6, destinadas a garantizar la calidad del servicio de los paquetes transmitidos en una red TCP/IP. El cual concluye que el sistema operativo GNU/Linux 2.4 y 2.6, tiene una capacidad ampliamente potenciable a través de la utilización de parches desarrollados exclusivamente para llevar a cabo tareas de implementación de Calidad de Servicio.
  El aporte de esta tesis a la investigación es que se da un claro ejemplo de que Iproute2 e Iptables, en su conjunto proporcionan potentes herramientas, en la tarea de brindar calidad de servicio (QoS) en una red de datos, a través del encolamiento, priorización, filtrado y clasificación de paquetes en una red TCP/IP.

- La tesis titulada Modelos de decision Linguistica para la QoS en Networking, en 2012, publicado por Universidad de Málaga, Málaga, España. (Gramajo, 2012)
  Tiene por objetivo es profundizar en la aplicación del modelado lingüístico difuso en procesos de análisis de decisión sobre procesos de QoS. El cual concluye que para lograr un rendimiento ´optimo se pueden implementar mecanismos y herramientas de QoS que realicen el control y coordinación de los recursos de conexión y de comunicación con redes externas e internet. El aporte de esta tesis a la investigación es que ayuda con las definiciones y a ampliar las bases teóricas de control de tráfico con software libre, herramientas para el control de tráfico y conceptos Básicos de Networking y QoS.

- La tesis titulada Medición y Análisis de Tráfico En Redes MPLS, en Septiembre del 2011, publicado por la Universidad Pontificia Universidad Católica del Perú, Lima, Perú. (Luna Victoria García, 2011). Tiene por objetivo medir y analizar los parámetros que definen la QoS en una red MPLS. El cual concluye en que las topologías de gran tamaño (caso BACKBONE) favorecen a MPLS, es decir, se presenta un mejor comportamiento con respecto al retardo de los tráficos así como del jitter que puede generarse. Gracias a esto, muchas clases de servicio propuestas por la ITU-T se pueden cumplir. El aporte de esta tesis a la investigación se da con las definiciones y a ampliar las bases teóricas de las arquitecturas para brindar calidad de servicio en una red IP.

- La tesis titulada Análisis y comparación de métodos de medición del desempeño de redes de computadores Ethernet y Metro Ethernet, en 2011, publicado por Universidad de Costa Rica, Costa Rica. (Vargas Piedra, 2011)
  Tiene por objetivo Investigar los métodos de medición del desempeño de redes de computadores Ethernet y Metro Ethernet mediante la revisión de recomendaciones, normas, legislaciones, RFCs y "benchmarks" relacionados. El cual concluye que las metodologías de pruebas de desempeño de red utilizadas ampliamente como marco de referencia para las redes Ethernet y Metro Ethernet son la recomendación de la UIT Y.1564 y la norma de la IETF RFC 2544.
  El aporte de esta tesis a la investigación es que ayuda a ampliar las bases teóricas para revisar las normas de regulación en relación con la calidad de servicio en redes de computadores Ethernet y Metro Ethernet y los métodos y herramientas de medición de desempeño que se solicitan para la comprobación de dichos parámetros de servicio.

- La tesis titulada Contribución en el Análisis y Simulación de una red MPLS con la Internet de Servicios Diferenciados Diffserv, en 2007, publicado por la Universidad Nacional Mayor de San Marcos, Lima, Perú. (Bustamante Alvarez, 2007).
  Tiene por objetivo Realizar un estudio y análisis de la simulación de una red MPLS con la Internet de los Servicios Diferenciados. El cual concluye que Diffserv puede ayudar a las redes a proveer un tratamiento preferencial a ciertas clases de tráfico, este tratamiento preferencial es provisto por acción del etiquetado de los paquetes IP en el campo DSCP, el tráfico condicionado y los mecanismos de planificación. El aporte de esta tesis a la investigación ayuda a ampliar las bases teóricas en cuando QoS y las arquitecturas para brindar calidad de servicio en una red IP.

- La tesis titulada Control de ancho de banda, en noviembre de 2005, publicado por la Universidad de Mendoza Facultad de Ingenieria, Mendoza, Argentina. (Navarro, 2005).
  Tiene como Objetivos; Desarrollar la problemática y los mecanismos existentes para poder "modelar" un servicio con QoS sobre una red de "Mejor Esfuerzo" como son las redes IP y Presentar el estado del arte de los algoritmos y mecanismos actuales disponibles para el control de ancho de banda sobre enlaces de datos en Linux. El cual concluye desde el punto de vista de las funcionalidades que Linux como sistema operativo, posee avanzadas capacidades para la implementación de mecanismos de control de ancho de banda. El aporte de esta tesis a la investigación es que ayuda a corroborar la hipótesis en que el sistema operativo Linux posee herramientas para controlar el tráfico y poder mejorar la calidad de servicio en una red de datos.

- La tesis titulada Estudio e Implementación de QoS mediante las Herramientas Iproute2 y Netfilter de Linux, en 2008, publicado por Universidad Tecnológica De La Mixteca, Huajuapan de Leon, Oaxaca, Mexico. (Silva Jiménez, 2008).
  Tiene por objetivo Analizar las herramientas IProute2 y Netfilter para marcar la prioridad de servicio en los paquetes de información trasmitidos en sistemas Linux. El cual concluye que las herramientas IProute2 y Netfilter son muy recomendables, pues permite una mejor forma de controlar el tráfico y la asignación de prioridades. La mejora de tener la red con QoS, fue aproximadamente de un 80% con respecto a no contar con ella. El aporte de esta tesis a la investigación es que se da un claro ejemplo de que Iproute2 e Iptables, en su conjunto proporcionan potentes herramientas para configurar y gestionar el ancho de banda de una red, por medio del sistema tc (Traffic Control, Control de Tráfico) de Linux, y Netfilter es una herramienta que permite filtrar paquetes o cambiar sus cabeceras haciendo uso de la herramienta Iptables.

2. BASES TEORICAS

2.1 Fundamento de Redes

Redes de Área Local (LAN)

(Tanenbaum, 2003), Redes de área local Las redes de área local (generalmente conocidas como LAN) son redes de propiedad privada que se encuentran en un solo edificio o en un campus de pocos kilómetros de longitud. Se utilizan ampliamente para conectar computadoras personales y estaciones de trabajo en oficinas de una empresa y de fábricas para compartir recursos (por ejemplo, impresoras) e intercambiar información. Las LANs son diferentes de otros tipos de redes en tres aspectos: 1) tamaño; 2) tecnología de transmisión, y 3) topología. Las LANs están restringidas por tamaño, es decir, el tiempo de transmisión en el peor de los casos es limitado y conocido de antemano.

Redes de Área Amplia (WAN)

(Tanenbaum, 2003), Una red de área amplia (WAN), abarca una gran área geográfica, con frecuencia un país o un continente. Contiene un conjunto de máquinas diseñado para programas (es decir, aplicaciones) de usuario. Los hosts están conectados por una subred de comunicación, o simplemente subred, para abreviar.

2.1.1 El modelo de referencia OSI

(Tanenbaum, 2003), Este modelo está basado en una propuesta desarrollada por la ISO (Organización Internacional deEstándares) como un primer paso hacia la estandarización internacional de los protocolos utilizados en varias capas. El modelo se llama OSI (Interconexión de Sistemas Abiertos) de ISO porque tiene que ver con la conexión de sistemas abiertos, es decir, sistemas que están abiertos a la comunicación con otros sistemas. Para abreviar, lo llamaremos modelo OSI.

Grafico 1: Modelo OSI

Fuente: http://tecnologia4elasalle.wikispaces.com/El+Modelo+OSI+de+ISO

La información contenida en una capa usualmente contiene cabeceras, referencias y datos encapsulados de una capa superior. La encapsulación es el proceso de ubicar los datos de una capa superior entre cabeceras y referencias de manera que cuando estos datos son recibidos por una capa, éstos son analizados, el protocolo de la capa respectiva remueve las cabeceas y referencias y entrega los datos a la capa superior en el formato en que esta puede comprenderlo.

Capa 7 (Aplicación): Esta capa contiene varios protocolos que los usuarios requieren con frecuencia. Un protocolo de aplicación de amplio uso es HTTP (Protocolo de Transferencia de Hipertexto), que es la base de World Wide Web. Cuando un navegador desea una página Web, utiliza este protocolo para enviar al servidor el nombre de dicha página. A continuación, el servidor devuelve la página.

Capa 6 (Presentación): Manipula el formato de presentación de los datos (HTML4, por ejemplo). Mientras se accede a una página Web, un computador puede estar enviando y recibiendo e-mails a la vez.

Capa 5 (Sesión): Esta capa permite que los usuarios de máquinas diferentes establezcan sesiones entre ellos. Las sesiones ofrecen varios servicios, como el control de diálogo (dar seguimiento de a quién le toca transmitir), administración de token y sincronización.

Capa 4 (Transporte): La función básica de esta capa es aceptar los datos provenientes de las capas superiores, dividirlos en unidades más pequeñas si es necesario, pasar éstas a la capa de red y asegurarse de que todas las piezas lleguen correctamente al otro extremo. Se encuentran los protocolos de comunicación que transfieren los datos (TCP, por ejemplo),

Capa 3 (Red): Esta capa controla las operaciones de la subred. Un aspecto clave del diseño es determinar cómo se enrutan los paquetes desde su origen a su destino. Las rutas pueden estar basadas en tablas estáticas (enrutamiento estático) codificadas en la red y que rara vez cambian. Se halla el direccionamiento lógico, el cual es usado para la determinación de las rutas (IP, por ejemplo).

Capa 2 (Enlace de datos): La tarea principal de esta capa es transformar un medio de transmisión puro en una línea de comunicación que, al llegar a la capa de red, aparezca libre de errores de transmisión. Si el servicio es confiable, el receptor confirma la recepción correcta de cada trama devolviendo una trama de confirmación de recepción. Se ubican los protocolos de red (Ethernet, por ejemplo).

Capa 1 (Física): En esta capa se lleva a cabo la transmisión de bits puros a través de un canal de comunicación. Los aspectos del diseño implican asegurarse de que cuando un lado envía un bit 1, éste se reciba en el otro lado como tal, no como bit 0. Se encuentran las especificaciones de cableado (RJ-45, por ejemplo).

2.1.2 El modelo de referencia TCP/IP

(Tanenbaum, 2003), Fue implementado por el Departamento de Defensa de los Estados Unidos y se originó como la necesidad de una red que pudiera sobrevivir ante cualquier condición, incluyendo una guerra nuclear. El modelo TCP/IP, consiste en cuatros capas como lo muestra el Gráfico Nº 2.

Algunas capas del modelo TCP/IP, comparten el mismo nombre que tienen algunas de las capas del modelo OSI.

En la capa de Aplicación no sólo se definen las aplicaciones, sino también, cómo los datos son formateados y cómo las sesiones son iniciadas y destruidas. En definitiva, todo lo concerniente a aplicaciones (Aplicación, Presentación y Sesión) del modelo OSI son combinadas en una sola capa en el modelo TCP/IP.

La capa de Transporte provee de servicios a la capa de aplicación, creando conexiones lógicas entre el origen y destino de los datos. En esta capa se encuentran los protocolos TCP (Transfer Control Protocol, Protocolo de Control de Transferencia) y UDP (User Datagram Protocol, Protocolo de Datagrama de Usuario). El Protocolo TCP es orientado a la conexión y brinda fiabilidad en la transferencia de los datos de extremo a extremo. Los mensajes son divididos en segmentos, que son reensamblados por el receptor de la capa superior (Aplicación).

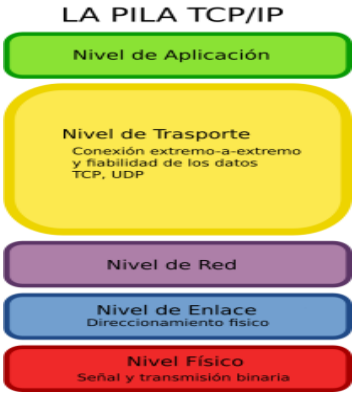


Grafico 2: Modelo TCP/IP

Fuente: http://modeloosiytcp.blogspot.pe/

Un segmento TCP se muestra en el Gráfico Nº 3, donde cada segmento se define a continuación:

- ✓ Source Port (Puerto de Origen): Es el número del puerto del sistema por el cual la fuente en envía los datos.
- ✓ Destination Port (Puerto de Destino): Es el número del puerto del sistema que recibirá los datos.
- ✓ Sequence Number (Número de Secuencia): Es el número del segmento por el cual se asegura que los datos lleguen en el orden correcto.
- ✓ Acknowledgement Number (Número de acuse de recibo): Es el siguiente octeto TCP proveniente del receptor.
- ✓ Header Length (Longitud de Cabecera) HLEN: Palabra de 32 bits en la cabecera. Code Bits (Bits de Código): Controla funciones tales como el inicio y término de una sesión.
- ✓ Reserved (Reservado): Los bits reservados son puestos a cero.
- ✓ Window (Ventana): El número de octetos que el origen acepta.
- ✓ Checksum (Suma de Verificación): Calcula una suma de verificación de la cabecera y del campo de datos.
- ✓ Urgent (Urgente): Indica el fin de un dato de carácter urgente.
- ✓ Options (Opciones): Es la única opción definida que especifica el tamaño del segmento TCP a enviar.
- ✓ Data (Datos): Contiene los datos de la capa superior (Aplicación).

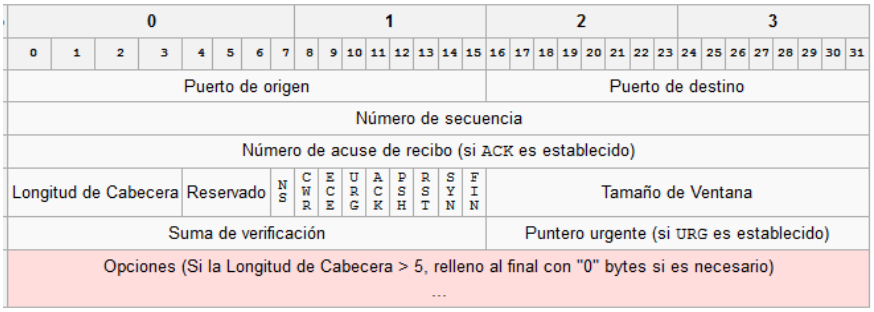


Grafico 3: Segmento TCP/IP

Fuente: http://www.arcesio.net/checksum/checksumTCP.html

Un protocolo orientado a conexión, significa que el protocolo TCP necesita establecer una conexión entre las dos partes antes de iniciar el envío de datos. Esto es conocido como the three-way handshake, o el apretón de manos en tres pasos, y significa que ambas partes (el que envía y el que recibe) se comunican usando una sincronización del protocolo de control de transmisión (SYN).

El Protocolo de Datagramas de Usuario (UDP), mostrado en el Gráfico Nº 4, es mucho más simple que TCP. UDP es un protocolo de la capa de transporte, no orientado a conexión y por ende no necesita establecer una conexión con otro host para enviar los datos. A continuación, se define cada segmento:

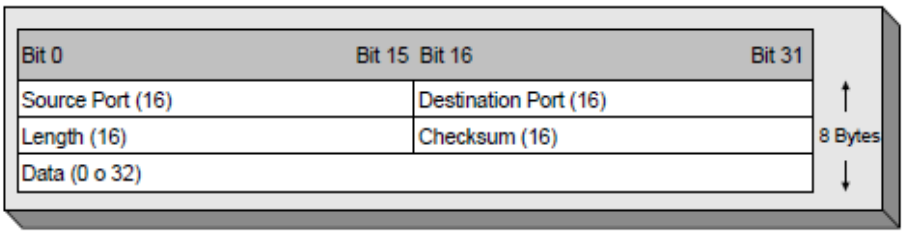


Grafico 4: Segmento UDP

Fuente: http://slidingwindowmorda.blogspot.pe/2015/05/sliding-window-udp.html

- Source Port (Puerto de Origen): Número del puerto usado por el emisor.
- Destination Port (Puerto de Destino): Es el número del puerto usado por el receptor para recibir los datos.
- Length (Longitud): Tamaño en bytes de la cabecera y los datos.
- Checksum (Suma de Verificación): Calcula una suma de verificación de la cabecera y el campo de datos.
- Data (Datos): Contiene los datos de la capa superior (Aplicación).

UDP no tiene ningún mecanismo para el control de flujo y no retransmite los datos si éstos se pierden. Esto significa que UDP no provee una entrega de datos fiable. Sin embargo, la manipulación de retransmisión y error pueden ser implementados en la capa de aplicación, si ello fuera necesario.

El protocolo principal hallado en la capa de Internet es IP (Internet Protocol, Protocolo de Internet), el cual provee conexión entre dos máquinas, entrega de datos de mejor esfuerzo (best-effort), algoritmo de ruteo de paquetes, manipula direcciones lógicas y la principal tarea es encontrar la mejor ruta entre dos terminales, sin preocuparse del contenido de los paquetes.

IP no emplea chequeos ni correcciones de error, y por esta razón es llamado un protocolo poco fiable. Sin embargo, estas funciones son manipuladas por la capa de transporte y la capa de aplicación El protocolo IP encapsula los datos de la capa de transporte en paquetes IP. El encabezado de un paquete IP se muestra en el Gráfico Nº 5 y se definen a continuación:

- Version (Versión): Especifica el formato de la cabecera IP del paquete. El campo contiene el número 4 si es IPv4 y 6 para IPv6.

- IP Header Length (Longitud de la Cabecera IP) HLEN: Indica la longitud del datagrama de la cabecera en 32 bits. Esta es la longitud total de toda la información de la cabecera e incluye dos variables de cabecera de longitud variable.
- Type of Service (Tipo de Servicio) ToS: Contiene 8 bits que especifican el nivel de importancia que ha sido asignado por un protocolo particular de capa superior.

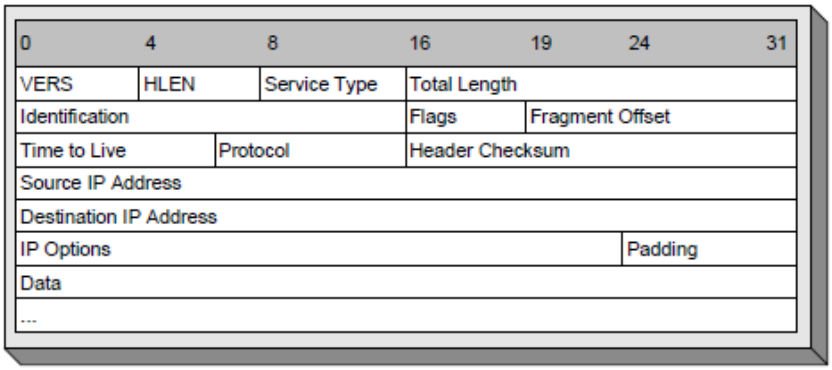


Grafico 5: Encabezado IP

Fuente: https://electrotelematica.wordpress.com/2011/02/12/datagrama-ip/

- Total Length (Longitud Total): 16 bits que determinan la longitud completa del paquete en bytes. Incluye los datos y la cabecera. Para obtener la longitud de los datos de carga útil, se substrae HLEN de la longitud total.
- Indentification (Identificación): 16 bits que especifican el datagrama actual (SEQ).
- Flags (Banderas): Un campo de 3 bits, en el cual, los dos bits menos significativos controlan la fragmentación. Un bit especifica si el paquete puede ser fragmentado, y los otros, indican si el paquete es el último fragmento de una serie de paquetes fragmentados.
- Fragment offset (Margen de Fragmentado): 13 bits que son usados para ayudar a unir los datagramas fragmentados. Este campo permite al siguiente campo a iniciar sobre un límite de 16 bits.
- Time to Live (Tiempo de Vida) TTL: Campo que especifica el número de saltos que un paquete puede recorrer. Este número es disminuido en uno cuando el paquete atraviesa un ruteador. Cuando el contador alcanza el cero, el paquete es descartado. Esto evita que los paquetes circulen indefinidamente.
- Protocol (Protocolo): 8 bits que indican el protocolo de la capa superior, como TCP o UDP, recibe paquetes entrantes luego de que el proceso IP haya sido completado.
- Header Checksum (Cabecera para la Suma de Verificación): 16 bits que ayudan a mantener la integridad de la cabecera IP.

- Source Address (Dirección de origen): 32 bits que especifican la dirección IP del nodo que envía el mensaje.
- Destination Address (Dirección de destino): 32 bits que especifican la dirección IP del nodo que recibe el mensaje.
- Options (Opciones): Permite al protocolo soportar varias opciones como seguridad. La longitud de este campo varía.
- Padding (Relleno): El protocolo de Internet agrega ceros extra a este campo, para asegurar que la cabecera IP sea siempre un múltiplo de 32 bits.

Los datos no forman parte de la cabecera IP. Esta contiene información de la capa superior (paquetes TCP o UDP) y tiene una longitud variable de hasta 64 bytes.

Si un paquete IP necesita llegar a una interfaz que tiene un tamaño de MTU (Maximum Transmission Unit, Unidad Máxima de Transmisión) menor que el tamaño del paquete IP, el protocolo fragmenta el paquete en paquetes más pequeños que quepan dentro de la MTU de la interfaz. Si el bit de no-fragmentación, en el campo Flags, está puesto a 1 y el paquete es más largo que la MTU de la interfaz, éste será descartado.

2.2 Trafico De Red

Es el flujo de datos que se transmiten mediante un medio físico por la red. Estos medios pueden ser guiados (cable coaxial, utp y fibra óptica), y no guiados (aire). El flujo de datos en el contexto de la colisión y los dominios de Difusión se centra en cómo las tramas de datos se propagan por la red. Hace referencia al movimiento de datos a través de los dispositivos de las capas 1, 2 y 3, y a cómo deben encapsularse los datos para efectuar este viaje de forma eficaz. (Cisco Systems, 1999)

2.3 Calidad de Servicio (Quality of Service)

Según (Cisco Systems, 1999), QoS se refiere a la capacidad de una red para proporcionar un mejor servicio a tráfico de red seleccionado sobre diversas tecnologías, incluyendo Frame Relay, modo de transferencia asíncrono (ATM), Ethernet y 802.1 redes y redes enrrutadas en IP que pueden utilizar cualquiera o todas estas tecnologías subyacentes.

2.4 FLUJOS

Un flujo es una conexión distinta o una conversación entre dos hosts. Cualquier conjunto único de paquetes entre dos hosts puede ser considerado como un flujo. Bajo el concepto TCP de una conexión con una IP origen, IP destino y puerto, representa un flujo (Kuznetsov, 1999).

2.5 PAQUETES Y TRAMAS

La palabra trama se utiliza normalmente para describir una unidad de datos de capa 2 (enlace de datos) unidad de datos que se remitirá a la siguiente destinatario. Interfaces Ethernet, interfaces PPP y T1 interconecta todas nombrar

su unidad de datos de capa 2 de un marco. El marco es en realidad la unidad en la que se realiza el control del tráfico. Un paquete, por otro lado, es un concepto de capa superior, que representa la capa 3 (red) unidades (Kuznetsov, 1999).

2.6 CLASIFICACIÓN:

Es el Mecanismo por el cual se identifican los paquetes y se los coloca en flujos o clases individuales. Los clasificadores (classifiers) ordenan o separan el tráfico entre colas. Clasificación es el mecanismo por el cual los paquetes son separados para tener diferente tratamiento, posiblemente diferente colas de salida. En Linux el modelo permite que un paquete "fluya" a través de una serie de clasificadores dentro de una estructura de control de tráfico y que sea clasificado de acuerdo a las políticas (policing) (Kuznetsov, 1999).

2.7 COLAS

Las Colas forman el telón de fondo para todos de control de tráfico y son el concepto integral detrás del Scheduling. Una cola es una ubicación (o buffer) que contiene un número finito de elementos en espera de una acción o servicio. En una red, una cola es el lugar donde nuestros paquetes (unidades) esperan a ser transmitida por el hardware (el servicio). En el modelo más simple, los paquetes se transmiten el primero que llega es el primero que sale (Kuznetsov, 1999).

2.8 CONTROL DE TRÁFICO

Control de tráfico es el término dado a todo el subsistema de gestión de colas de paquetes en una red o dispositivo de red. Control de tráfico se compone de varias operaciones distintas. Clasificar (Classifying) es un mecanismo mediante el cual identificar los paquetes colocarlos en los flujos o clases individuales. Vigilancia (Policing) es un mecanismo por el cual se limita el número de paquetes o bytes en una corriente de búsqueda de una clasificación particular. Scheduling es el proceso de toma de decisiones por el cual los paquetes se ordenan y reordenado para la transmisión. Shaping es el proceso por el cual los paquetes se retrasan y se transmiten para producir una tasa de flujo uniforme y predecible. (Kuznetsov, 1999).

El control de tráfico consiste de diversas operaciones, estas características del sistema de control de tráfico pueden combinarse de tal forma de reservar un determinado ancho de banda para un flujo de datos determinado (o una aplicación), o para limitar el ancho de banda disponible para un flujo o aplicación en particular.

2.9 CONTROL DE TRÁFICO EN LINUX

El núcleo (kernel) de Linux ofrece una amplia variedad de funciones de control de tráfico. Las partes del kernel, y varios programas de espacio de usuario para controlarlos han sido implementados por Alexey Kuznetsov. (Navarro, 2005)

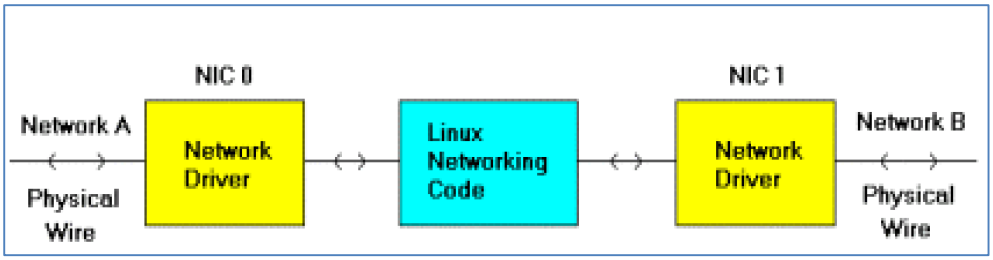


Grafico 6: Unix como Router

Fuente:http://tesis.udea.edu.co/dspace/bitstream/10495/145/1/SistemaOperativoLinuxControlTraficoRedes.pdf

En el grafico N° 6, se representa una maquina con sistema operativo Linux operando como un router puro, es decir, realizando el reenvío de paquetes de una interface a otra, sin generar o consumir paquete alguno, los paquetes pueden estar entrando desde la red A por la interface de red NIC 0 allí el controlador (Network driver) se encarga de entregar los paquetes al código de red de Linux (Linux Networking Code) este realiza el reenvío de estos paquetes, solicitando a la interface de red NIC 1 enviar estos paquetes a la red B. (Rengifo Salazar & Urrea Lujan, 2004)

El principio básico implicado en la aplicación de QoS en Linux se muestra en el Grafico N° 7. Esta figura muestra cómo el kernel procesa los paquetes entrantes, y cómo se genera paquetes que se enviará a la red. A la entrada se tiene un de-multiplexor que examina los paquetes entrantes para determinar si los paquetes están destinados para el nodo local. Si es así, se envían a la capa superior para su posterior procesamiento. Si no es así, envía los paquetes al bloque de reenvío. El bloque de reenvío, que también puede recibir los paquetes generados localmente desde la capa superior, busca la tabla de enrutamiento y determina el siguiente salto para el paquete. Después de esto, se pone en cola los paquetes para ser transmitidos en la interfaz de salida. Es en este punto que el control del tráfico linux entra en juego. Control de tráfico de Linux se puede utilizar para construir una compleja combinación de cola disciplinas, clases y filtros que controlan los paquetes que se envían en la interfaz de salida. (Navarro, 2005)

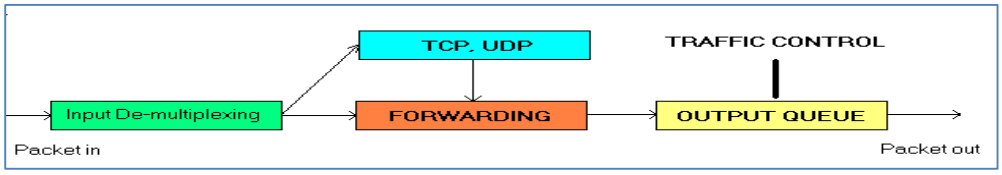


Grafico 7: Control de tráfico en Linux (Linux Traffic Control)

Fuente: http://qos.ittc.ku.edu/howto.pdf

El sistema operativo Linux a partir de la versión de kernel 2.2.x cuenta con un conjunto de herramientas o utilidades que permiten realizar tareas de enrutamiento avanzado, esto consiste en poder efectuar la gestión y manipulación de los paquetes que se transmiten. Opera como un router, realizando tareas de reenvío o rechazo de paquetes, veamos con detalle en el Grafico N° 8, cómo es posible llevar a cabo el control de tráfico gestionando las colas de paquetes:

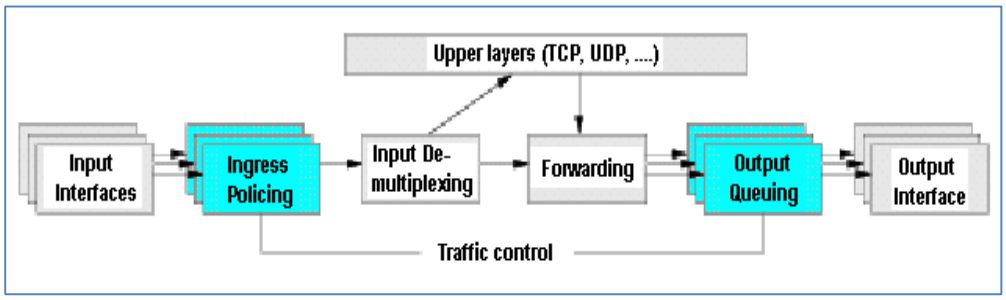


Grafico 8: Procesamiento de datos de Red

Fuente: http://qos.ittc.ku.edu/howto.pdf

Se puede a preciar que los paquetes que ingresan por las interfaces de entrada están sometidos a lo que el kernel puede decidir mediante una regulación de ingreso (Ingress Policing), es decir, permitir su entrada o descártarlos debido a que están llegando demasiado rápido, después de ello interesa los que son reenviados a la cola de salida (Output Queuing) donde nuevamente el kernel decide a que cola deben ir los paquetes, por cual interface van a salir, si los paquetes serán encolados o no, como serán enviados los paquetes según un orden, y si es posible retrasar su envío. Después de que el kernel realiza su control los paquetes son enviados a las interfaces de salida. En resumen lo que se tiene son dos etapas donde se realiza el control del tráfico por parte del kernel de Linux: una en la entrada Ingress Policing y la otra a la salida con las colas Output Queueing (Rengifo Salazar & Urrea Lujan, 2004).

El Control de tráfico en el Kernel de Linux consiste en los siguientes componentes principales:

- disciplinas de cola (queueing disciplines)
- clases (dentro de una disciplina de cola)
- filtros (filters)
- vigilancia (policing)

Según (Kuznetsov, 1999), mediante el empleo de las características del kernel de Linux para realizar control de tráfico, es posible llevar a cabo tareas tales como:

- Limitar el ancho de banda a un valor conocido
- Limitar el ancho de banda de un usuario o servicio
- Reservar ancho de banda para una aplicación en particular
- Priorizar el tráfico
- Permitir equilibrar el uso del ancho de banda entre diferentes usuarios
- Realizar balanceo de carga entre varios servidores
- Rechazar cierto tráfico en particular

2.9.1 DISCIPLINAS DE COLAS

(Rengifo Salazar & Urrea Lujan, 2004). Las disciplinas de colas (queuing disciplines) son algoritmos que se encargan de gestionar las colas en todo el proceso de ingreso (ingress) y salida (egress) de los paquetes sobre un dispositivo de red, estas son la base sobre la que se fundamenta la gestión de tráfico en Linux, cada dispositivo de red tiene asociado un proceso de ingreso y de egreso.

Existen dos grupos de disciplinas de colas, unas con clases y las otras sin clases. Aquellas con clases permiten al usuario crear subdivisiones para realizar diferenciación en el tratamiento del tráfico, mientras que en las disciplinas de colas sin clases esto no es posible. Se debe tener en cuenta que Linux define el concepto root para referirse a la etapa egress, es decir, tráfico saliente sobre el cual es posible aplicar toda la potencia del control de tráfico y está el concepto ingress que se refiere al tráfico que ingresa, y su única utilidad es limitar el tráfico entrante.

Linux soporta varias disciplinas de colas y son:

- Class Based Queue (CBQ)
- Token Bucket Flow (TBF)
- Clark-Shenker-Zhang (CSZ)
- First In First Out (FIFO)
- Stochastic Fair Queuing (SFQ)
- Diff-Serv Marker (DS-MARK)
- HTB (Hierarchical Token Bucket)

Las funciones que soportan las diferentes disciplinas de colas son:

- Enqueue: Encola un paquete con la disciplina de cola.
- Dequeue: Desencola los paquetes para que sean enviados.
- Requeue: Reencola un paquete para su transmisión, después de desencolado el paquete, si por alguna razón, el paquete no se transmite, el paquete tiene que ser puesto de nuevo en la cola en la misma posición donde fue desencolado
- Drop: Descarta los paquetes de la cola.
- Init Inicializa y configura los parámetros de una disciplina de cola cuando es creada.

2.9.2 CLASES

Cada clase posee una cola, que por defecto es una cola FIFO. Cuando la función de puesta en cola de una disciplina de colas se llama, la disciplina de colas aplica los filtros a determinar la clase a la que pertenece el paquete.

Hay dos maneras en que una clase puede ser identificada. Una es a través de la clase identifier, que es especificado por el usuario. El otro identificador, que se utiliza dentro del kernel para identificar una clase, que se conoce como la identificador interno.

Este ID es único y es asignado por la disciplina de colas. El ID de clase es un tipo de datos U32, mientras que el ID interno es un entero largo sin signo. La mayoría de las funciones en clases utilizan el ID interno para identificar la clase. Ahora bien, hay algunas funciones (como el get y la función de cambio, que serán se verá más adelante) que utilizan el ID de clase también. (Saravanan, 1999).

2.9.3 FILTROS

Los filtros se utilizan para clasificar los paquetes en base a ciertas propiedades del paquete, por ejemplo, TOS byte en el encabezado IP, direcciones IP, números de puerto, etc. Trabajó cuando se invoca la función de puesta en cola de una disciplina de colas. Queuing disciplinas utilizan filtros para asignar los paquetes entrantes a una de sus clases (Saravanan, 1999).

Los filtros están asociados a las disciplinas de colas pues es a través de estos que se define a que clase serán asignados los paquetes que ingresan, un filtro básicamente se usa para clasificar los paquetes basados en propiedades del campo TOS, la dirección IP, el número de puerto (Rengifo Salazar & Urrea Lujan, 2004).

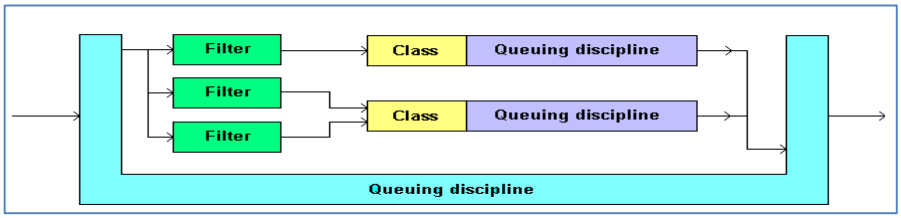


Grafico 9: Modelo para control de tráfico del kernel de Unix

Fuente: https://docencia.etsit.urjc.es

En el Grafico N° 9 se puede apreciar como el kernel de Linux maneja el tráfico donde la disciplina de cola tiene asociada un filtro/regulador y unido a esto puede venir una clase.

Los filtros también pueden tener una estructura interna: puede controlar los elementos internos, que están referenciados por un mango. Estas asas son de 32 bits de largo, pero no se dividen en jor ma y los números de menor importancia como los ID de clase. El mango se refiere 0 al propio filtro. Al igual que las clases, filtros también tienen una ID interna, que se puede obtener con la ayuda de una función get. La estructura básica de los filtros es muestra en la Figura 10.

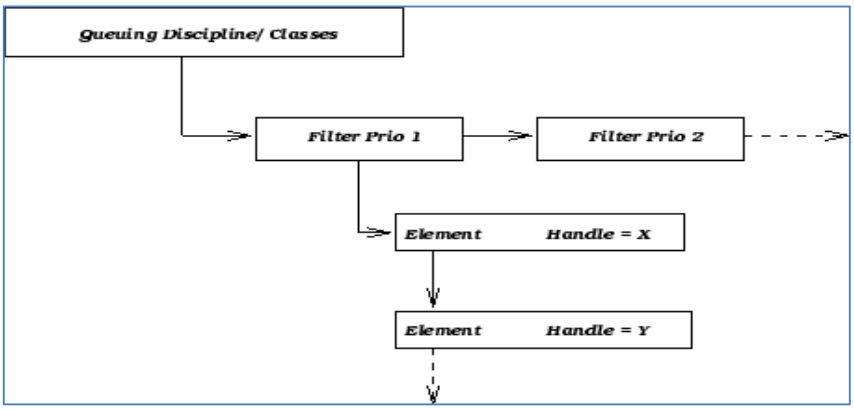


Grafico 10: Estructura de los Filtros

Fuente: http://qos.ittc.ku.edu/howto.pdf

En el Grafico N° 10

Esta estructura se utiliza para representar listas ltro y es mantenido por la clases y disciplinas de colas. Como ejemplo, la estructura de clases en cbq / sched / sch cbq.c neta mantiene la lista de filtro mediante el uso de la estructura proto tcf. La función de la cadena TCF en clases, que se describe en la sección anterior, se utiliza para devolver el anclaje a una lista de filtro, que se puede usar para atravesar el filtro lista. Las listas de filtros están clasificadas por prioridad, en orden ascendente. También, las entradas tienen la forma adecuada por el protocolo para el que se aplican, por ejemplo, IP, UDP, etc.

major: este parámetro es libre de significado para el kernel. Se usa un esquema arbitrario de numeración

minor: identifica sin ambigüedad el objeto como una qdisc si tiene un valor de cero, cualquier otro valor identifica el objeto como una clase.

Un manejador (handle) está reservado para el ingress, es usado como una etiqueta en classid y flowid de la utilidad tc son identificadores externos usados por el usuario, el kernel cuenta con identificadores internos para cada objeto (Saravanan, 1999).

2.9.3.1 Tipos de Filtros

Para el caso de los filtros, lo que se hace es llamar una cadena clasificadora que se encarga de tomar la decisión sobre a qué clase debe procesar el paquete, esta cadena está formada por todos los filtros asociados a la disciplina de cola. Algunos filtros disponibles son:

- **fw:** se fundamenta en la manera en que el firewall (cortafuegos) marca el paquete, es sencillo y no requiere el uso de la utilidad tc.
- **u32:** su funcionamiento está basado en los campos del paquete IP.
- **route:** basa su funcionamiento por la ruta que tomara el paquete según la tabla de enrutamiento.

Los argumentos más comunes de estos filtros son:

protocol: define el protocolo que aceptará el filtro.

parent: define el manejador (handle) al que está asociado el filtro, debe ser una clase ya existente.

prio: establece la prioridad del filtro, los números menores se verifican primero.

handle: es el manejador y difiere según el filtro.

A. Filtro U32: Es el filtro más avanzado, pues gracias a su versatilidad es posible definir muchos criterios para lograr el filtrado, lo cual lo hace el más utilizado. Este permite definir estructuras basadas en cualquier conjunto de bits del paquete IP, ya sea en su encabezado o en campo de datos, algunos de los criterios son:

- Dirección IP de origen, dirección de destino del paquete.
- Puerto fuente, puerto destino de todos los protocolos IP.
- Protocolo utilizado: tcp, udp, icmp, gre, ipsec.

El campo TOS.

La forma más simple del filtro u32 es una lista de registros consistentes en dos campos: un selector y una acción, se realiza entonces una comparación entre los selectores y el paquete IP que está siendo procesado hasta que ocurra una coincidencia y entonces la acción asociada se ejecuta.

Cuando se usa la utilidad tc para definir un filtro, esta consiste de tres partes: especificación del filtro, selector y acción.

tc filter add dev INTF [protocol PROTO] [(preference1 priority) PRIO] [parent Q_KIND]

Se tiene entonces que protocol define el protocolo que se aplicará al filtro, preference (priority) establece la prioridad del filtro seleccionado, se puede disponer de varios filtros (listas de reglas) con diferentes prioridades, se pasará por cada lista en el orden en que las reglas fueron agregadas, entonces las listas con menor prioridad (con numero de preferencia mas alto) serán procesadas, el campo parent define la disciplina de cola raíz.

Selector U32: este selector contiene la definición del patrón con el cual será comparado el paquete actualmente procesado, concretamente lo que hace es definir cuales bits han de ser comparados en el encabezado del paquete, un ejemplo:

tc filter add dev eth0 protocol ip parent 1:0 pref 10 u32 \ match u32 00100000 00ff0000 at 0 flowid 1:0

Al revisar esta línea desde la palabra clave match, se establece que el selector coincidirá con cabeceras IP cuyo segundo byte sea 0x10(0010), el numero 00ff es la mascara de comparación, que le indica al filtro que bits tiene que comparar, at define desde donde inicia la coincidencia con respecto al desplazamiento especificado en este caso el desplazamiento cero, iniciándose así desde el principio del paquete, la coincidencia del paquete ocurrirá si su campo TOS tiene los bits de menor retraso activos.

La sintaxis del selector es como sigue:

match [u32 1 u16 1 u8] PATTERN MASK [at OFFSET 1 nexthdr + OFFSET]

Aquí u32, u16 y u8 definen la longitud en bits del patrón, PATTERN y MASK deben tener esta misma longitud, at OFFSET establece el ajuste o desplazamiento en bytes, para dar inicio a la comparación y nexthdr + OFFSET define de acuerdo al valor del desplazamiento el inicio del encabezado de la capa superior.

La descripción anterior corresponde a la definición de un selector general, como se puede ver es más complejo, existe el selector especifico que relaciona ítems tales como: dirección fuente/destino, puerto fuente/destino, campo TOS y protocolo.

B. Filtro Route: Este filtro basa su funcionamiento en los resultados de las tablas de enrutamiento, cuando un paquete que viaja a través de las clases llega a una que tiene marcado con el filtro route, entonces separa los paquetes basado en la información de las tablas de enrutamiento, la configuración de este filtro, se usa en combinación con la utilidad ip con la cual se generan las diferentes tablas de enrutado, la sintaxis es:

tc filter [add 1 del 1 change 1 get] dev STRING [parent PARENTID] [protocol PROTO] [prio PRIORITY] route

Un ejemplo:

tc filter add dev eth0 parent 1:0 protocol ip prio 100 route

Se definió un filtro tipo route, que se agregó al nodo padre con una prioridad 100, básicamente si el paquete llega a este nodo consultará la tabla de rutas y si coincide una, lo enviará a la clase dada con la prioridad especificada.

2.9.4 Elementos del Control de Tráfico

2.9.4.1 Shaping

Shaping es el mecanismo por el cual los paquetes se retrasan antes de la transmisión en una cola de salida para satisfacer una tasa de salida deseada. Este es uno de los deseos más comunes de los usuarios que buscan soluciones de control de ancho de banda. El acto de retrasar un paquete como parte de una solución de control de tráfico hace que cada mecanismo de conformación en un mecanismo no-trabajo de conservación, lo que significa más o menos: "Se requiere trabajo con el fin de retrasar los paquetes." Shapers intentan limitar o tráfico ración para satisfacer, pero no superar una velocidad configurado (medida con frecuencia en paquetes por segundo o bits / bytes por segundo) (Kuznetsov, 1999).

2.9.4.2 Scheduling

La programación es el mecanismo por el cual los paquetes se disponen (o reordenan) entre la entrada y la salida de una cola particular. El planificador abrumadoramente más común es el FIFO (primero en entrar, primero en salir) planificador. Desde una perspectiva más amplia, cualquier conjunto de mecanismos de control de tráfico en una cola de salida puede ser considerado como un planificador, porque los paquetes están dispuestos para la salida (Kuznetsov, 1999).

2.9.4.3 Classifying

Clasificadores tipo u separar el tráfico en las colas.

Clasificar es el mecanismo por el cual los paquetes se separan para el tratamiento diferente, posiblemente diferentes colas de salida. Durante el proceso de aceptación, encaminamiento y transmisión de un paquete, un dispositivo de red puede clasificar el paquete un número de diferentes maneras. La clasificación puede incluir marcando el paquete, lo que sucede generalmente en el límite de una red bajo un único control administrativo o la clasificación puede ocurrir en cada salto individualmente (Kuznetsov, 1999).

2.9.4.4 Policing (Regulación o Control)

(Rengifo Salazar & Urrea Lujan, 2004), cuando se habla de regulación o control (Policing), para efectuar control de tráfico, se refiere a regular el tráfico de los paquetes para que estos no excedan un límite.

En principio se consideran cuatro tipos de mecanismos de regulación o control:

- Decisiones de regulación por filtros
- No aceptar encolar un paquete
- Rechazo de un paquete desde una cola interior
- Rechazo de un paquete cuando se encola uno nuevo.

Cuando hablamos de desiciones de regulación por filtros, estos últimos los que deciden qué hacer con el paquete, ya sea no realizar nada, reclasificarlo si este excede ciertos límites o simplemente descartarlo porque el paquete violó los límites.

Cuando no se acepta encolar un paquete, se presenta la disciplina de cola falla en encolar el paquete y este es descartado, algunas disciplinas de colas tienen mecanismos de realimentación que permiten volver a llamar la disciplina de cola, dando una segunda oportunidad al paquete para que sea nuevamente encolado.

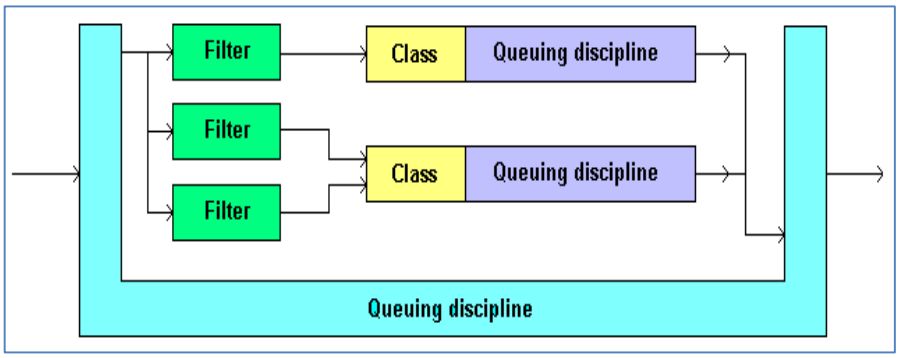


Grafico 11: Manejo de disciplinas de colas con clases y filtros

Fuente: http://qos.ittc.ku.edu/howto.pdf

El Grafico N° 11, resume los temas tratados hasta el momento, disciplina de colas, clases y filtros, Linux dispone entonces de un conjunto de componentes para realizar el control de tráfico, que tienen su análogo en los elementos tradicionales de control de tráfico, se listan a continuación:

Elementos Tradicionales	Componentes de Linux
shaping (ajuste)	class, ofrece Capacidades de ajuste.
Scheduling (ordenación o gestión)	qdisc es en sí un scheduler (ordenador de los paquetes para salgan de cierta forma)
Classifing (clasificación)	filter, realiza la clasificación, pues esta es parte del filtro.
policing (regulación o limitación)	policer hace parte de filter, como mecanismo de regulación.
dropping (descarte)	filter con un policer, permite descartar el tráfico.
Marking (Marcado)	dsmark, mediante esta disciplina de colas se realiza el marcaje del paquete, es decir, puede ser alterado.

Tabla 1: Comparación entre control de tráfico tradicional y Linux

Fuente: Elaboración propia

Hasta aquí se ha revisado el tema de control de tráfico con Linux presentándose la estructura general a nivel de kernel, se mostró como se realiza el proceso, que objetos utiliza Linux para llevar a cabo dicha tarea que funciones están asociadas, interesa ahora es cual es la interface de usuario disponible para llevar a cabo el control de tráfico, en este trabajo se revisó la herramienta IPROUTE2 con su utilidad tc (Traffic Control) que gracias a Alexey Kuznetsov y la existencia de NETLINK, pues por medio de este IPROUTE2 y sus utilidades se comunican con el kernel, lo que sigue es aplicar los conceptos ya vistos usando las utilidades de IPROUTE2, para ello se revisarán algunas disciplinas de colas mostrando como es su configuración con tc, lo primero que se debe saber es que la configuración del kernel debe brindar soporte al control del tráfico, se presenta a continuación lo requerido para esto:

2.10 Disciplinas de Cola sin Clases y con Clases En Linux

2.10.1 Disciplinas de colas sin clases.

Con las disciplinas de cola, cambiamos el modo en que se envían los datos. Las disciplinas de cola sin clases son aquellas que, mayormente, aceptan datos y se limitan a reordenarlos, retrasarlos, o descartarlos.

La disciplina más usada, con mucho, es la qdisc pfifo_fast (se usa por defecto). Esto también explica por qué estas características avanzadas son tan robustas. No son más que «simplemente otra cola» (Hubert, y otros, 2004).

A. pfifo_fast

La qdisc (disciplina de colas) sin clases, Pfifo_fast es el algoritmo de planificación por defecto utilizado cuando ningún otro se define de forma explícita.

pfifo_fast, es básicamente una cola FIFO, esta disciplina tiene tres bandas con prioridades 0, 1, 2 dentro de cada banda se aplica la regla FIFO, los paquetes que vengan con mayor prioridad van a la banda 0, los siguientes a la banda 1 y el resto a la banda 2, Sin embargo, no se procesará la banda 1mientras haya paquetes esperando en la banda 0. Lo mismo se aplica para las bandas 1 y 2. Así que tenemos 3 prioridades: los paquetes en cola 0 serán enviados antes de que los paquetes en la cola 2 (Kuznetsov, 1999), ver Gráfico N° 12.

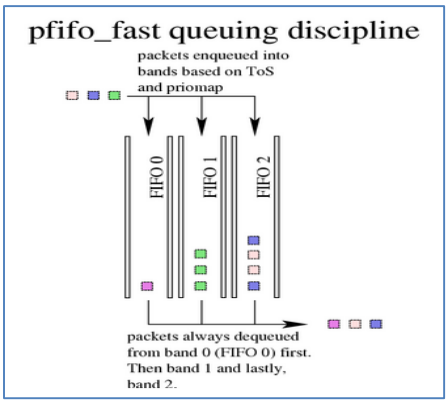


Grafico 12: Disciplina de cola pfifo_fast

Fuente:http://www.ctr.unican.es/asignaturas/dec/Doc/dec_seminario_TrafficControl.pdf

Esta disciplina de cola no puede ser manipulada por el usuario, pues tiene ya una configuración por defecto, para priorizar el tráfico el kernel se basa en la campo TOS del paquete IP.

Configurar una FIFO con tc es muy simple:

tc qdisc add dev eth0 root pfifo limit 10

Lo que se hizo fue agregar "add" una nueva disciplina de cola "qdisc" al dispositivo de red "dev" eth0 (interfaz Ethernet 0), root se refiere a disciplina de cola raíz (egress) se requiere a pesar que pfifo es una disciplina de cola sin clases por el formato de la utilidad tc, pfifo (disciplina de cola PFIFO) y limit establece la longitud de la cola, que es el número de paquetes a tener

en la cola, en este caso 10 paquetes en cola, este parámetro es el único a configurar en esta disciplina de cola.

B. TBF (Token Bucket Filter)

El Token Bucket Filter (Navarro, 2005) (TBF), es un qdisc sencillo que se limita a dejar pasar paquetes que lleguen a una tasa que no exceda una impuesta administrativamente, pero con la posibilidad de permitir ráfagas cortas que excedan esta tasa.

La implementación de TBF consiste en un búfer (el bucket o balde), que se llena constantemente con piezas virtuales de información denominadas tokens, a una tasa específica (token rate). El parámetro más importante del bucket es su tamaño, que es el número de tokens que puede almacenar.

Cada token que llega toma un paquete de datos entrante de la cola de datos y se elimina del bucket. Asociar este algoritmo con los dos flujos (tokens y datos), nos da tres posibles situaciones:

- Los datos llegan a TBF a una tasa que es igual a la de tokens entrantes. En este caso, cada paquete entrante tiene su token correspondiente y pasa a la cola sin retrasos.
- Los datos llegan al TBF a una tasa menor a la de los token. Sólo una parte de los tokens se borran con la salida da cada paquete que se envía fuera de la cola, de manera que se acumulan los tokens, hasta llenar el bucket.
- Los datos llegan al TBF a una tasa mayor a la de los token. Esto signigica que el bucket se quedará pronto sin tokens, lo que causará que TBF se acelere a sí mismo por un rato. Esto se llama una «situación sobrelímite». Si siguen llegando paquetes, empezarán a ser descartados.

PARÁMETROS Y USO

limit o latency: limit es el número de bytes en la cola que pueden estar esperando por tokens disponibles, también es posible definirlo de otra manera, especificando el tiempo máximo (latency) que puede estar un paquete en la TBF, este cálculo tiene en cuenta el tamaño del bucket, su velocidad y velocidad pico (peakrate, si está configurada).

- burst/buffer/maxburst: define el tamaño de la cola de tokens, es decir, el tamaño del bucket en bytes, este es la máxima cantidad de bytes para la cual existen tokens disponibles de manera inmediata.
- mpu: establece el número mínimo de tokens por paquete (minimum packet unit.
- rate: velocidad.

- peakrate: cuando existen tokens disponibles los paquetes que llegan son enviados inmediatamente, pero es posible regular esto si se desea, configurando la velocidad pico (peakrate) que establece la velocidad a la cual el bucket puede vaciarse.
- mtu/minburst: define la unidad máxima de transferencia.

Configuración de ejemplo:

tc qdisc add eth0 root tbf rate 250kbit latency 50ms burst 1540

Se configura la qdisc tbf como tipo egress y se define como velocidad 250kbit, con un tiempo de permanencia de los paquetes de 50ms y un buffer de 1540 bytes.

C. SFQ (Stochastic Fairness Queueing)

Stochastic Fairness Queueing (SFQ) (Navarro, 2005), es una implementación sencilla de la familia de algoritmos de colas justas (fair queueing). Es menos preciso que los otros, pero también necesita menos cálculos mientras que resulta ser casi perfectamente justo.

Con esta disciplina de cola se busca distribuir de manera equitativa los datos transmitidos a la red, basa su funcionamiento en flujos, pues la distribución la realiza generando un número arbitrario de estos, para lograr esto usa una función hash la cual separa el tráfico en colas tipo FIFO que son mantenidas internamente, el tráfico es enviado en modo round-robin (por turnos), gráficamente se tiene:

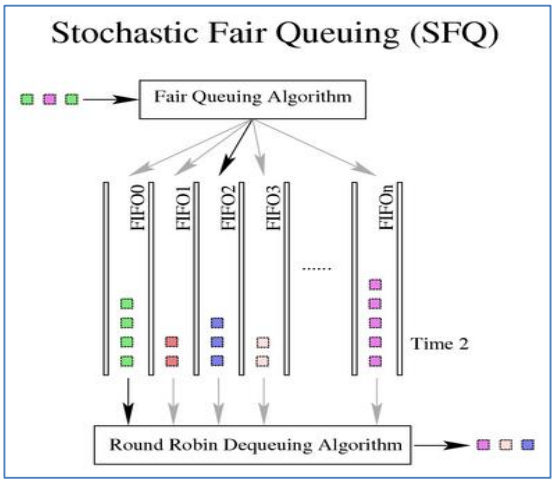


Grafico 13: Disciplina de cola SFQ

Fuente:http://www.ctr.unican.es/asignaturas/dec/Doc/dec_seminario_TrafficControl.pdf

Parámetros y uso

SFQ es mayormente autoajustable:

- Perturb: Reconfigurar el hash una vez cada estos segundos. Si no se indica, el hash no se reconfigurará nunca. No es recomendable. 10 segundos es probablemente un buen valor.
- Quantum: Cantidad de bytes de un flujo que se permiten sacar de la cola antes de que le toque el turno a la siguiente cola. Por defecto es 1 paquete de tamaño máximo (tamaño MTU). ¡No lo ponga por debajo del MTU!
- Limit: El número total de paquetes que serán encolados por esta SFQ (tras empezar a descartarlos).

Configuración de ejemplo

Si tiene un dispositivo que tenga velocidad de enlace y tasa actual disponible idénticas, como una línea telefónica, esta configuración ayudará a mejorar la equitatividad:

tc qdisc add dev ppp0 root sfq perturb 10

tc -s -d qdisc ls qdisc sfq 800c: dev ppp0 quantum 1514b limit 128p flows 128/1024 perturb 10sec Sent 4812 bytes 62 pkts (dropped 0, overlimits 0)

El número 800c: es el número de manejador asignado de forma automática, limit indica que pueden esperar 128 paquetes en esta cola. Hay 1024 hashbuckets disponibles para la contabilidad, de los cuales puede haber 128 activos al mismo tiempo.

2.10.2 Disciplinas de colas con clases.

Las qdisc con clases son muy útiles si tiene diferentes tipos de tráfico a los que quiere dar un tratamiento separado. Este tipo de disciplinas brindan un mayor nivel de control y adquieren su valor cuando se trata de gestionar tráfico con diferentes prioridades, pues permiten al usuario configurarlas para dar tratamiento a cada tipo de tráfico (Rengifo Salazar & Urrea Lujan, 2004).

El flujo dentro de las qdisc con clases y sus clases
(Hubert, y otros, 2004), cuando entra tráfico dentro de una qdisc con clases, hay que enviarlo a alguna de las clases que contiene (se necesita «clasificarlo»). Para determinar qué hay que hacer con un paquete, se consulta a los «filtros». Es importante saber que los filtros se llaman desde dentro de una qdisc, Los filtros asociados a esa qdisc devuelven entonces una decisión, y la qdisc la usa para encolar el paquete en una de las clases. Cada subclase puede probar otros filtros para ver si se imparten más instrucciones. En caso contrario, la clase encola el paquete en la qdisc que contiene.

Aparte de contener otras qdisc, la mayoría de las qdisc con clases también realizan «shaping». Esto es útil tanto para reordenar paquetes (con SFQ, por ejemplo) como para controlar tasas. Necesitará esto en caso de tener una interfaz de gran velocidad (por ejemplo, ethernet) enviando a un dispositivo más lento (un cable módem).

La familia qdisc: raíces, controladores, hermanos y padres

Cada interfaz tiene una «qdisc raíz» de salida. Por defecto, es la disciplina de colas pfifo_fast sin clases que mencionamos anteriormente. A cada qdisc y clase se le asigna un controlador (handle), que puede usar en posteriores sentencias de configuración para referirse a la qdisc. Aparte de la qdisc de salida, la interfaz también puede tener una de entrada, que dicta las normas sobre el tráfico que entra.

Los controladores de estas qdisc consisten en dos partes, un número mayor y un número menor: <mayor>:<menor>. Es costumbre darle a la qdisc de raíz el nombre «1:», que es lo mismo que «1:0». El número menor de una qdisc siempre es 0.

Las clases deben tener el mismo número mayor que sus padres. Este número mayor tiene que ser único dentro de una configuración de salida o entrada. El número menor debe ser único dentro de una qdisc y sus clases.

Cómo se usan los filtros para clasificar el tráfico

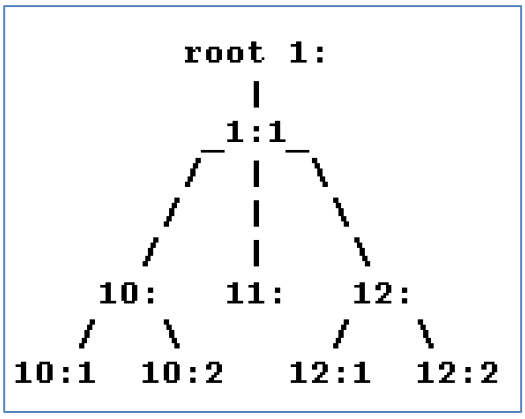


Grafico 14: Ejemplo de estructura jerárquica de clasificación

Fuente: Elaboración propia.

No debe interpretar que el núcleo está en la cima del árbol y la red abajo, ya que no es el caso. Los paquetes se encolan y desencolan en el qdisc raíz, que es la única cosa con la que habla el núcleo. En la figura 10: El kernel trata directamente con la qdisc raíz (root) pues es allí donde los paquetes entran o salen de la cola, se pueden tener esquemas como 1: > 1:1 > 12: > 12:2, o este 1: > 12:2.

Para enviar los paquetes a la interfaz, la disciplina de colas qdisc 1: recibe la petición de dar salida a los paquetes, esta se pasa a 1:1 que enseguida se pasa a 10: 11: y 12: los cuales consultan a sus descendientes para desencolar los paquetes.

Cómo se desencolan los paquetes para enviarlos al hardware

Cuando el núcleo decide que necesita extraer paquetes para enviarlos a la interfaz, la qdisc 1: raíz recibe una petición de desencolar, que se pasa a 1:1, que a su vez la pasa a 10: 11:, y 12:, cada una de las cuales consulta a sus descendientes, e intenta hacer dequeue() sobre ellos. En este caso, el núcleo necesita recorrer todo el árbol, porque sólo 12:2 contiene un paquete.

En breve, las clases anidadas SOLO hablan a sus qdisc paternas, y nunca a una interfaz.

La consecuencia de esto es que las clases nunca desencolan más rápido de lo que sus padres permiten. Y esto es exactamente lo que queremos: de esta manera, podemos tener SFQ como clase interna, que no hace ajustes, sólo reordena, y tenemos una qdisc externa, que es la que hace los ajustes.

A. CBQ

Se trata de una qdisc bastante más compleja pero muy potente al mismo tiempo permite hacer tanto shaping como scheduling. Es decir, podremos limitar el tráfico a la tasa que deseemos y además clasificarlo gracias a las clases que creemos con ayuda de los filtros. Esta qdisc funciona como una combinación de PRIO y TBF.

Para desencolar los paquetes se usa un WRR (Round Robin por pesos) o lo que es lo mismo, siempre que haya paquetes en las clases con más prioridad, éstos saldrán antes hacia la interfaz (Álvarez Martín & Nazareno, 2010).

B. HTB

HTB (Hierarchical Token Bucket) (Kuznetsov, 1999), es una disciplina de cola con clases similar a CBQ pero tienen sus diferencias, es más sencilla de configurar, intuitiva y comprensible, es usada para controlar el ancho de banda dado en un enlace, permite usar un enlace físico para simular varios enlaces y enviar diferentes tipos de tráfico en los distintos enlaces simulados, requiere que se especifique como dividir el enlace físico en los enlaces simulados y decidir cuál enlace simulado usar para que un paquete dado sea enviado.

HTB asegura que la cantidad de servicio proporcionado a una clase sea la cantidad mínima requerida por esta y que efectivamente se le asignó, cuando una clase solicita menos de la cantidad asignada, el ancho de banda restante es distribuido a otras clases las cuales solicitaron el servicio.

HTB basa su principio de funcionamiento en los conceptos de tokens y buckets de un extremo a otro de un sistema basado en clases y filtros lo cual permite un control de tráfico más complejo y fino. Bajo un esquema complejo de préstamo, HTB brinda la posibilidad de realizar sofisticados técnicas de control de tráfico y para las formas fáciles se puede usar HTB para ajustarlo.

Lo que permite hacer esta disciplina de cola al usuario es definir las características de los tokens y buckets a usados y le posibilita anidar los buckets de manera arbitraria, cuando se acompaña de clasificación el tráfico puede ser controlado en forma granular.

El principio de funcionamiento de HTB se comprende mejor si se visualiza un árbol de clases, cada clase tiene su velocidad y prioridad, los paquetes son divididos en flujos los cuales son ligados a las hojas del árbol, cuando se habla de ajuste (shaping), este ocurre en las clases hoja (classe leaf), que están al extremo del árbol, es decir, no se lleva acabo ajuste alguno en las clases interiores o en las clases raíz (root), estas solamente existen para sugerir cómo el modelo de préstamo debe distribuir los tokens disponibles.

El mecanismo de préstamo es fundamental en esta disciplina de cola, pues ocurre que las clases hijas prestan tokens de sus padres una vez ellas han excedido la velocidad (rate), una clase hija continúa intentando prestar hasta que alcance un límite (ceil), en este punto empezará a encolar paquetes hasta que hallan tokens/ctokens disponibles.

En el Grafico N° 16, se presenta gráficamente como ocurren los préstamos, además se puede ver la existencia de varias clases root que simulan circuitos virtuales:

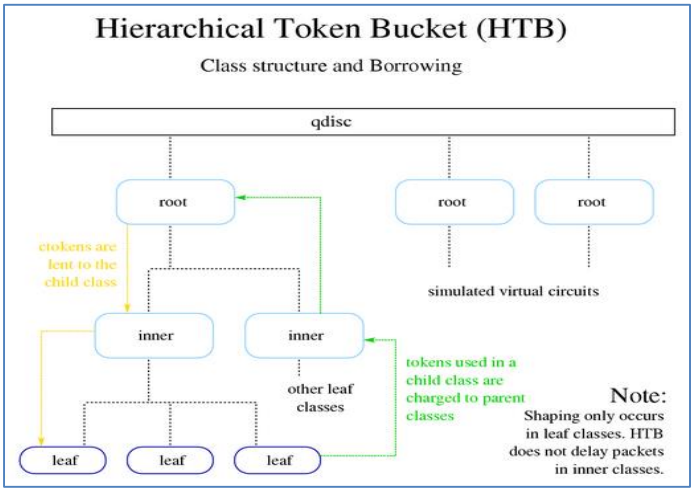


Grafico 15: Estructura de clase y préstamo

Fuente: http://www.um.edu.ar/catedras/PPT03/document/Modulo_III/Traffic-Control/ppt0c3-tc-02.html

Los parámetros de configuración de esta disciplina cola son:

default: si este parámetro está en default 0 define que por cualquier causa un tráfico sin clasificar abandone la cola a velocidad del hardware, pasando por encima de cualquier clase ligada a la disciplina de cola root, este valor puede apuntar a una clase en particular, que se encargará de sacar dicho tráfico.

- rate: establece la velocidad límite mínima del tráfico transmitido, se puede entender como el ancho de banda garantizado para la clase leaf.
- ceil: usado para configurar la máxima velocidad deseada la cual será el límite del tráfico transmitido.
- burst: define la velocidad del bucket, HTB sacará de la cola burst bytes antes de esperar la llegada de mas ctokens.
- cburst: configura la máxima velocidad del bucket, HTB sacará de la cola cburst bytes antes de esperar la llegada de mas ctokens.
- quantum: este es el parámetro clave que usa HTB para controlar el préstamo, normalmente el valor correcto del quantum es calculado por HTB, si no es especificado por el usuario. Este parámetro es muy delicado pues tiene un gran efecto sobre el mecanismo de préstamo y el ajuste, pues es usado para dividir el tráfico entre las clases hijas sobre la velocidad y para transmitir los paquetes de esas mismas clases.
- r2q: este valor es calculado por el usuario como pista para ayudar HTB en la determinación de quantum óptimo para una clase en particular.
- mtu: define la máxima unidad de transferencia.
- prio: establece la prioridad.

Un ejemplo de configuración:

```
# tc qdisc add dev eth0 root handle 1: htb default 11

# tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbps \ ceil 100kbps

# tc class add dev eth0 parent 1:1 classid 1:10 htb rate 80kbps \ ceil 100kbps

# tc class add dev eth0 parent 1:1 classid 1:11 htb rate 20kbps \ ceil 100kbps
```

La primera línea agrega la qdisc HTB al dispositivo de red como root (egress) le asigna el handle 1: que se usa mas abajo como referencia y asigna a la clase 11 todo el tráfico que no sea clasificado, la segunda línea define la clase root 1:1 que está debajo de la qdisc 1: , que es la que permitirá a las clases hijas prestarse entre si, las restantes líneas configurar clases hijas en las que se configura el ancho de banda 80kbps y 20kbps.

2.11 ENRUTAMIENTO AVANZADO CON LINUX

2.11.1 Enrutamiento

(Hubert, y otros, 2004), El enrutamiento es el proceso de selección de un camino por el cual se mandarán paquetes, también se le conoce con el nombre de ruteo o encaminamiento, en general se le conoce como el proceso de dirigir y transferir un paquete por medio de la red.

El enrutamiento consta de dos características que son: Determinación de trayectorias y conmutación, la conmutación es cuando los paquetes se distribuyen por la red pero no existe ninguna trayectoria definida, y la determinación de trayectorias se refiere a lo que es la métrica, longitud de trayectoria, algoritmos de enrutamiento y tablas de enrutamiento, la métrica es un estándar de medición en cuanto a la longitud, para determinarse cuál es la mejor ruta.

2.11.2 NETFILTER

(Russell, 2002), Netfilter se trata de un framework o interfaz completa, dentro del núcleo de Linux, que permite interceptar y manipular paquetes de red. El módulo más conocido construido sobre Netfilter es IPtables. Esta herramienta nos permite manipular Netfilter desde el espacio de usuario. A veces se usa IPtables para referirse a toda la infraestructura ofrecida por Netfilter.

Con esto conseguimos que un módulo del kernel, más una utilidad de usuario controlen el flujo de paquetes que van desde y hacia las interfaces de red.

Con Netfilter, usando IPtables podemos realizar:

- filtrado de paquetes
- traducción de direcciones IP y puertos (NAT Network Address Translation)
- Manipulaciones sobre datagramas IP (packet MANGLING).

Toda la gestión de paquetes la lleva el kernel, y Netfilter es el encargado de aplicar las reglas de filtrado que previamente le habremos definido con iptables. En resumen, diremos que Netfilter es la parte del kernel encargada de filtrar tráfico e iptables la herramienta con la que configuraremos Netfilter.

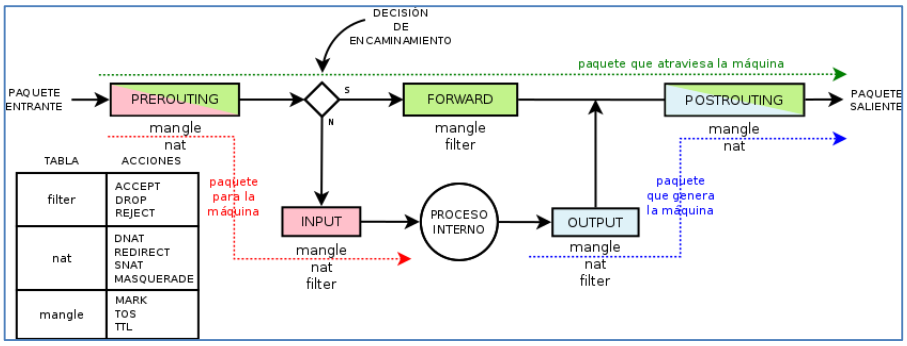


Grafico 16: tratamiento de paquetes en Iptables.

Fuente: http://docs.iescdl.es/~josem/netfilter.html

Cuando llega un paquete, primero se mira si va destinado a la máquina local, o no. Si está destinado a la máquina local, pasará a la cadena INPUT en caso de ser un paquete entrante, y pasará a OUTPUT en caso de ser un paquete saliente. Si el paquete no es para la máquina local, sino que va destinada a otra máquina y se tiene activado el forwarding, entonces pasará a la cadena FORWARD, en caso contrario se eliminará.

IPTABLES: La heramienta iptables inserta y elimina reglas de la tabla de filtrado de paquetes del núcleo.

2.11.2.1 Arquitectura de Netfilter

Netfilter realiza la gestión del filtrado mediante tablas organizadas en cadenas (chains) y éstas a su vez compuestas por reglas que se evalúan sobre los paquetes analizados en busca de condiciones según unos parámetros y, en caso de darse alguna, ejecutando la acción asociada.

A. Cadenas (Chains)

Las cadenas son agrupaciones de reglas que se deben aplicar a los paquetes según momentos concretos del flujo de los paquetes a su paso por el sistema Netfilter. Indican CUANDO actuar sobre los paquetes. Las posibles cadenas que nos encontramos son:

- INPUT: Determina la acción realizar cuando un paquete coincide con la regla, a la entrada de la interfaz (filtrado de tráfico entrante). Se aplica a los paquetes destinados a la propia máquina.
- OUTPUT: Determina la acción a realizar cuando un paquete coincide con la regla, a la salida de la interfaz (filtrado de tráfico saliente). Se aplica a los paquetes originados en la propia máquina.
- FORWARD: Determina la acción a tomar cuando un paquete se envía desde una interfaz a otra.

Se trata de una cadena transversal que se encuentra de forma intermedia entre las dos siguientes.

- PREROUTING: Es el primer estadio en el sistema Netfilter. Determina la primera acción a realizar antes de que el paquete entre en el sistema.
- POSTROUTING: Determina la acción a realizar, justo antes de enviar el paquete a la interfaz destino.

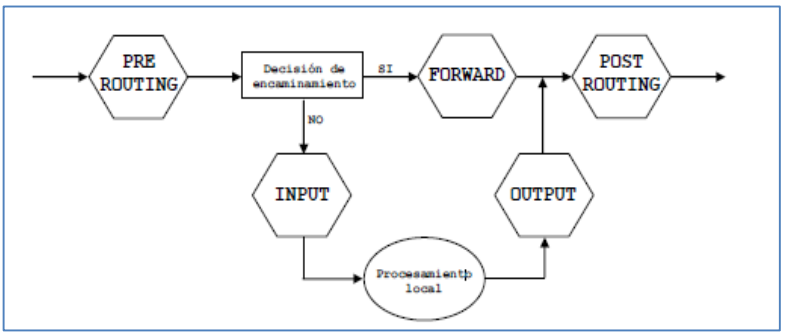


Grafico 17: Esquema de procesamiento de Netfilter

Fuente: http://docs.iescdl.es/~josem/netfilter.html

Cuando se recibe un paquete por cualquier interfaz se lleva a cabo, en primer lugar, una suma de comprobación (Checksum). Si es correcta, los paquetes transitan hacia la evaluación de la cadena **PREROUTING** se encargarán de determinar el tratamiento que deberá de darse al paquete en función de la dirección destino:

- ✓ Si el paquete va dirigido a la propia máquina, éste es enviado a la cadena INPUT, que en caso de superarse será procesado localmente.
- ✓ Si la dirección destino del paquete es distinta de la local, y está activada la función de reenvío, se evalúa la cadena FORWARD. Si se superan las reglas de esta cadena se reenvía el paquete (si la función de reenvío no estaba habilitada, el paquete se descarta DROP).

NOTA: los paquetes que no son considerados “locales” son aquellos que, por lo general, pertenecen a otra subred.

Antes de que los paquetes abandonen el sistema Netfilter y sean enviados a la interfaz destino son recibidos por la cadena POSTROUTING.

La cadena OUTPUT solamente es utilizada cuando los paquetes han sido originados localmente. Además, los paquetes que pasen por la cadena OUTPUT necesariamente pasan por POSTROUTING.

B. Tablas

Tablas en Netfilter Las tablas son usadas para indicar al sistema de filtrado el tipo de procesamiento que se debe aplicar a los paquetes. Una tabla maneja una cierta cantidad de reglas internas que se organizan en cadenas y existen cuatro tablas por defecto: filter, mangle, nat y raw.

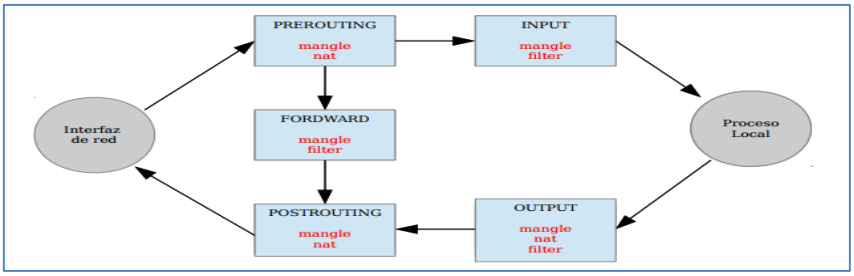


Grafico 18: Tablas en el esquema del procesamiento de Netfilter.

Fuente: http://sw-libre.blogspot.pe/2011/10/hola-basandome-en-la-documentacion-de.html

FILTER: Se usa para el filtrado general de paquetes y es la tabla predeterminada de Netfilter. Decide qué es lo que entra y qué no. Está compuesta por las siguientes cadenas:

- INPUT
- OUTPUT
- FORWARD

NAT: Controla la traducción de direcciones. Permite alterar las direcciones origen y destino del datagrama, analizando algunas propiedades del mismo. Está compuesta por las cadenas:

- PREROUTING
- POSTROUTING
- OUTPUT

Algunos usos típicos de NAT son:

- SNAT (Source NAT): También conocido como IP Masquerading. Se cambia la dirección IP origen del datagrama al pasar por el router (después del encaminamiento y antes de su reenvío).
- DNAT (Destination NAT): Se modifica la dirección IP destino antes del encaminamiento.

MANGLE: Analiza ciertas características del paquete y lo marca, en función de su naturaleza, para que reciba cierto tratamiento específico (ej: diferenciar tráfico en función de los servicios…). Mangle permite la reescritura completa de paquetes (o tramas completas). En definitiva la tabla mangle controla los procesos de modificación del contenido y las opciones del paquete. Las cadenas que se agrupan en esta tabla son:

- INPUT
- OUTPUT
- FORWARD
- PREROUTING
- POSTROUTING

C. Reglas IPtables

Como se indicó anteriormente, para el uso de Netfilter desde el espacio de usuario es necesario utilizar, además de los módulos del kernel, la utilidad IPtables. A través del comando iptables es posible insertar, eliminar y modificar reglas dentro de Netfilter.

Sintaxis

Un comando básico de iptables está compuesto por 7 partes, cada una de los cuales indican:

- El comando iptables como punto de partida.
- La tabla a usar (filter, nat, mangle). Si no se pone nada se usa por defecto filter.
- El comando que deseamos aplicar a una cadena (insertar regla, modificar una existente, eliminar, etc.). Para definir la operación se usan una serie de comandos.
- La cadena a usar, que puede ser una de las cadenas por defecto (INPUT, OUTPUT, FORWARD, PREROUTING….) o bien aquellas creadas por el usuario.
- La condición que especifica los criterios que debe de cumplir un paquete (los campos que lo componen) para que sea aplicable la acción.
- La acción a realizar para aquellos paquetes que cumplan la condición.
- Una serie de opciones que podemos aplicar para ajustar la acción.

La sintaxis de un comando iptables es la siguiente:

iptables -t [tabla] -[regla] [criterio] -j [acción]

Donde:

- t [tabla]: Determina a qué tabla se desea añadir una regla.

  Existen 3 tipos de tablas: NAT, filter y mangle, siendo filter la tabla por defecto por omisión. La tabla NAT se usa para modificar las conexiones, como el enmascaramiento de conexiones, re direccionamiento de puertos, entre otras acciones. En la tabla filter, es donde se añaden las reglas relacionadas con el filtrado. La tabla mangle modifica los paquetes a nivel de cabecera IP (ToS, TTL, por ejemplo).

- [regla]: Son opciones a realizar sobre las tablas. A (Append, Añadir), añade una regla valida como INPUT, FORWARD y OUTPUT. L (List, Listar), lista las reglas existentes en una tabla. F (Flush, Vaciar), elimina todas las reglas a menos que se especifique eliminar una regla específica. P (Policy, Política), establece la política del firewall. Por defecto es ACCEPT (Aceptar todas las conexiones).
- [criterio]: Especifica las características del paquete a rastrear. Contiene las siguientes
- opciones: -s (Source, Fuente), la cual determina la dirección IP de origen, -d (Destination, Destino), la cual determina la dirección IP de destino, --sport (Source Port, Puerto Fuente), especifica el puerto origen, --dport (Destination Port, Puerto Destino), señala el puerto de destino y -p (Protocol, Protocolo), indica el protocolo de transmisión utilizado.
- -j [acción]: Establece la acción a realizar con el paquete. Existen cinco acciones básicas: ACCEPT, REJECT, DROP, REDIRECT y LOG. ACCEPT (Aceptar), acepta el paquete. REJECT (Rechazar), rechaza el paquete replicando un paquete ICMP del tipo Port Unreachable (Puerto inalcanzable). DROP (Descartar), descarta el paquete sin enviar respuesta ICMP. REDIRECT (Redirigir), redirige el paquete a la dirección especificada.

Una regla de iptables especifica una condición y una acción:

Condición: Características que debe cumplir un paquete para que la regla le sea aplicable. Ejemplos de condiciones:

-p tcp --dport 80: el protocolo es TCP y el puerto destino es 80

-s 13.1.2.0/24: la direccion de origen es de la subred 13.1.2.0/24.

Acción: indica lo que se hace con el paquete si cumple la condición de la regla. Ejemplos de acciones:

ACCEPT: el paquete se acepta
DROP: el paquete se descarta

SNAT --to-source 13.1.2.1: se cambia la IP origen del paquete

Las reglas se agrupan en listas de reglas, llamadas cadenas. Las cadenas se agrupan en tablas.

Ejemplo de la regla de Iptables para marcar los paquetes:

```
iptables -t mangle -A PREROUTING -p tcp --dports
80,443 -j MARK --set-mark 1
```

2.11.3 IPROUTE2

(Kuznetsov, 1999), Iproute2 es un conjunto de utilidades para el control de TCP / IP de redes y control de tráfico en Linux. En la actualidad se mantiene por Stephen Hemminger.

A partir del kernel 2.2.X, se cuenta con el routing, cortafuegos y código de clasificación de tráfico. En IProute2 se utilizará la herramienta tc que sirve para manipular el control de tráfico, en combinación con las opciones:

- tc qdisc, permite establecer la disciplina de colas.
- tc class, permite establecer clases basadas en la planificación de las disciplinas de colas.
- tc filter, permite establecer el filtrado de paquetes QoS/CoS (Class of service, Clase de servicio).
- tc policy, permite a los usuarios establecer las políticas QoS/CoS.

En resumen el tc se usa para hacer la configuración de las colas y clases, estos componentes se combinan para controlar el flujo de paquetes entrantes y salientes en una interfaz de red.

IPROUTE2 tiene dos herramientas con las que implementar todas estas nuevas facilidades:

A. COMANDO IP

Se encarga de monitorear las direcciones e interfaces de red, gestionar las tablas ARP (Address Resolution Protocol, Protocolo de Resolución de Direcciones), del uso de tablas de enrutamiento y de la creación de túneles IP.

B. tc (traffic control)

Herramienta que permite implementar toda la Gestión de Tráfico en linux. Con el comando tc se manipula y controla el ancho de banda y la priorización de paquetes, a través de algoritmos de disciplinas de colas para el control del tráfico de la red, tc se usa para hacer la configuración de las colas y clases, estos componentes se combinan para controlar el flujo de paquetes entrantes y salientes en una interfaz de red.

tc es una herramienta que permite a los administradores de red compilar distintas políticas de calidad de servicio. Su utilización permite establecer las jerarquías de clases, disciplinas de cola y filtros para la distribución, encolado y control de los paquetes sobre una red TCP/IP.

Los comandos que tc utiliza son:

- tc class: Establece la jerarquía de clases.
- tc qdisc: Configura la disciplina de cola para una clase.
- tc filter: Determina los filtros para la identificación de paquetes.

Las sintaxis para estos comandos son las siguientes:

tc class add dev [INTERFAZ] parent [CLASE_PADRE] / classid [CLASEID][QDISC] rate /ANCHO_BANDA]

Donde:

- INTERFAZ: Es el nombre de la interfaz sobre la cual está implementándose la jerarquía.
- CLASE_PADRE: Corresponde al nombre de la clase a la que pertenece.
- CLASEID: Es el nombre identificador de la clase respectiva.
- QDISC: Describe disciplina de cola establecida para esa clase.
- ANCHO_BANDA: Especifica el ancho de banda máximo para los paquetes en Mbits.

tc qdisc add dev [INTERFAZ] parent [CLASE_PADRE][QDISC] / quantum [BYTES] perturb [SEG]

Donde:

- INTERFAZ: Es el nombre de la interfaz sobre la cual esta implementándose la jerarquía.
- CLASE_PADRE: Corresponde al nombre de la clase a la que pertenece.
- QDISC: Describe disciplina de cola establecida para esa clase.
- BYTES: Especifica la cantidad de bytes que se debe sacar de la cola para su clasificación.
- SEG: Tiempo en segundos para reconfigurar el algoritmo de hash que divide el tráfico en colas.

tc filter add dev [INTERFAZ] protocol [PROTOCOLO] / parent [CLASE_PADRE] prio / [ORDEN] [FILTRO]

Donde:

- INTERFAZ: Es el nombre de la interfaz sobre la cual está implementándose la jerarquía.
- PROTOCOLO: Corresponde al protocolo utilizado.

- CLASE_PADRE: Nombre de la clase a la que pertenece.
- ORDEN: Especifica la prioridad de la clase con respecto a las demás.
- FILTRO: Determina el filtro utilizado para la identificación del tráfico dentro de la clase.

2.12 Firewalls

Un firewall o cortafuegos se definen como un organizador cuyo propósito o función principal es proteger la red, un firewall restringe ciertos tipos de tráfico desde Internet a la red local y viceversa, esto quiere decir, que restringe el flujo de información entre dos redes.

Para determinar qué tipo de tráfico se permitirá se utiliza el firewall usando un conjunto de reglas que son predefinidas por el administrador, tomando en cuenta sus necesidades y la de sus usuarios. El tráfico de red está constituido por paquetes IP, los cuales son datos que viajan en flujos desde un lugar origen hasta un lugar destino, y dichos paquetes tiene cabeceras que contienen información sobre el paquete fuente, el paquete destino, tipos de protocolos, etc., esto sirve para que el firewall con las reglas compruebe las cabeceras y determine así que paquetes se aceptan (ACCEPT), cuales se niegan (REJECT) o se rechazan (DROP).

2.13 Parámetros de Calidad de Servicio

La recomendación realizada por (UNIÓN INTERNACIONAL DE TELECOMUNICACIONES, 2001), define un modelo de categorías de calidad de servicio (QoS) para servicios multimedios desde el punto de vista del usuario extremo. Teniendo en cuenta las expectativas del usuario con respecto a diversas aplicaciones multimedia, se determinan diferentes categorías según toleren o no las pérdidas de información y de retardo. Se pretende que estas categorías sirvan de base para definir clases QoS realistas para las redes de transporte subyacentes y los mecanismos de control de la QoS correspondientes.

La finalidad de esta Recomendación es proporcionar orientación sobre los factores clave que inciden en la calidad de servicio (QoS) desde la perspectiva del usuario extremo. Se elabora una clasificación amplia de categorías QoS conforme al usuario extremo teniendo en cuenta una gama de aplicaciones que utilizan como media la voz, el vídeo, la imagen y datos, y los parámetros que permiten conocer la satisfacción del usuario en cuanto a esas aplicaciones. El objetivo es que estas categorías sirvan de base para determinar clases de la QoS realistas y los mecanismos de control QoS correspondientes para las redes de transporte subyacentes.

2.13.1 Necesidades de calidad de funcionamiento determinadas por el usuario.

Un reto importante para las redes alámbricas e inalámbricas con IP de reciente instalación es proporcionar la calidad de servicio (QoS, quality of service) adecuada para los servicios diferentes.

Esto exige un conocimiento profundo de los requisitos de calidad de funcionamiento de los servicios y las aplicaciones. El punto de partida para determinar estas necesidades de calidad de funcionamiento debe ser el usuario. Al cliente no le interesa saber cómo se implementa un servicio determinado. Pero lo que sí le interesa es comparar la manera en que diferentes proveedores ofrecen el mismo servicio según parámetros de calidad de funcionamiento universal y centrado en el usuario. Esto significa que la calidad de funcionamiento se debería expresar mediante parámetros que:

- Tienen en cuenta todos los aspectos del servicio desde el punto de vista del usuario.
- Se centran en los efectos percibidos por el usuario más que en sus causas dentro de la red.
- Son independientes de la arquitectura o tecnologías de la red.
- Se pueden medir objetiva o subjetivamente en el punto de acceso al servicio.
- Se pueden relacionar fácilmente con los parámetros de calidad de funcionamiento de la red.

2.13.2 Parámetros de QoS en redes LAN

1. **Tiempo de transmisión (Retardo o Latencia)**

   (UNIÓN INTERNACIONAL DE TELECOMUNICACIONES, 2001). Hay diversos tiempos de transmisión o retardo, como el tiempo que lleva establecer un servicio determinado a partir de la solicitud de alta del usuario y el tiempo para recibir información específica una vez que el servicio está dado de alta. El retardo tiene un impacto muy directo en la satisfacción del usuario según la aplicación, y se puede producir en el terminal, la red o cualquier servidor. Obsérvese que, desde el punto de vista del usuario, el retardo también tiene en cuenta el efecto en otros parámetros de red, como el caudal.

Retardo en un sentido	Características de Calidad
De 0 a 150 ms	Aceptable para la mayoría de las aplicaciones de usuario
150 a 400 ms	Aceptable siempre y cuando las Administraciones conozcan la influencia del tiempo de transmisión en la calidad de transmisión de las aplicaciones de usuario.
Por encima de 400 ms	Inaceptable a efectos de planificación general de una red

Tabla 2: ITU-T G.114 Recomendaciones de Retardo o Latencia

2. **Ancho de Banda**
   (UNIÓN INTERNACIONAL DE TELECOMUNICACIONES, 2001). Una medida de la capacidad de transmisión de datos, expresada generalmente en Kilobits por segundo (kbps) o en Megabits por segundo (Mbps). Indica la capacidad máxima teórica de una conexión, pero esta capacidad teórica se ve disminuida por factores negativos tales como el retardo de transmisión, que pueden causar un deterioro en la calidad. Es determinante para analizar el comportamiento de la calidad en Redes de Área Local, en estas redes se debe de tener en cuenta la utilidad de los dispositivos de interconexión según las capas del modelo OSI y su relación con la QoS.

3. **Tasa de Perdidas o Pérdida de Información**
   (UNIÓN INTERNACIONAL DE TELECOMUNICACIONES, 2001). La pérdida de información tiene un efecto muy directo en la calidad de la información que se presenta al usuario, se trate de voz, imagen, vídeo o datos. En este contexto, la pérdida de información no se limita a los errores de bit o a la pérdida de paquetes durante la transmisión, sino también a los efectos de cualquier degradación introducida por la codificación del medio para conseguir una transmisión más eficaz.

   VALORES RECOMENDADOS: La pérdida de paquetes máxima admitida para que no se degrade la comunicación deber ser inferior al 1%.

2.14 Herramientas básicas para la medición de desempeño de la red

Es importante utilizar estas herramientas con discreción ya que la mayoría implican tráfico dentro de la red, lo que sacrificaría su desempeño. Estas herramientas básicas junto con sus algoritmos representan la base de otras utilidades más robustas para medición de desempeño de red que al fin de cuenta sólo las implementan incorporándoles mayores índices de funcionalidad.

2.14.1 Ntopng

(Luca, 2015), Ntopng es una poderosa herramienta que permite analizar en tiempo real el tráfico de red. Esto te permite evaluar el ancho de banda utilizado por ips o hosts individuales, por puertos, e identificar los protocolos de red más utilizados. Ntopng, la siguiente generación (next generation), la evolución natural del ntop original, se basa en libpcap y se ha escrito en una manera portátil para ejecutar virtualmente en todas las plataformas Unix, MacOSX.

Ntopng (de Network Top) es una herramienta que permite monitorizar en tiempo real una red. Es útil para controlar los usuarios y aplicaciones que están consumiendo recursos de red en un instante concreto y para ayudarnos a detectar malas configuraciones de algún equipo, (facilitando la tarea ya que, justo al nombre del equipo, aparece sale un banderín amarillo o rojo, dependiendo del volumen de tráfico), o a nivel de servicio.

Ntopng analiza por defecto el tráfico de red una interfaz de red (eth0, eth1, etc) pero también puede analizar y mostrar los resultados vía web (con el comando -i fichero.pcap) de un archivo capturado de tráfico con wireshark, tcpdump, etc en formato pcap

La herramienta Ntopng nos permitirá obtener lo siguiente:

- Mostrar la distribución del tráfico IP entre los diversos protocolos
- Identificar los protocolos de red más utilizados
- Distribución de protocolos.
- Consumo de ancho de banda.
- Puertos más utilizados.
- Un resumen general de tráfico.

2.14.2 PING

PING (Packet Internet Groper), (Snader, 2000), una de las herramientas más fundamentales y útiles para la depuración de las redes y para las aplicaciones que se ejecutan en ellos es la utilidad ping. Su función principal es verificar la conectividad entre dos hosts. Como tal, es una herramienta muy valiosa para los problemas de la red de depuración.

Es una utilidad para el diagnóstico de la conexión entre dos dispositivos en una red TCP/IP mediante el envío de paquetes ICMP (de sus siglas en inglés “Internet Control Message Protocol”) de solicitud y respuesta.

Además del diagnóstico de conexión, la utilidad PING permite obtener el RTT; es decir, es una herramienta que puede utilizarse para la medición de la métrica de latencia. También especifica la pérdida de paquetes.

La versión de PING para sistemas basados en Unix muestra la siguiente información:

- El nombre del “host” destino.
- El número de paquetes transmitidos.
- El número de paquetes recibidos.
- La pérdida porcentual de paquetes.
- El promedio, el valor máximo y el valor mínimo del RTT.

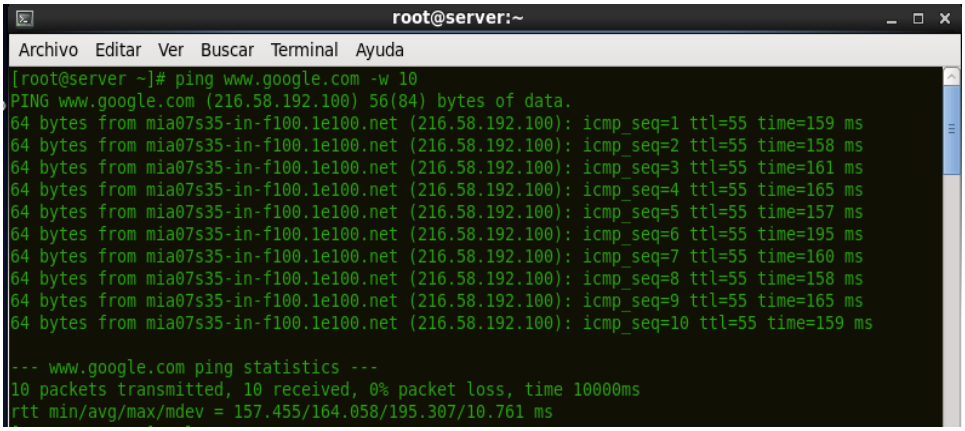


Grafico 19: Utilidad PING

Fuente: Elaboración propia.

En Grafico N° 19, se muestra la salida de la utilidad PING. Se utilizó la herramienta de consola del sistema operativo Centos 6.5, para ejecutar el PING. La opción “w” utilizada define que se efectuarán 10 pruebas; por defecto cada prueba se realiza en intervalos de 1 s. Se pueden utilizar varias opciones para el PING las cuales están documentadas en la página “man” de Linux para PING (ejecutando man ping en la consola).

Nótese que al final de la ejecución del PING se brinda la estadística de la cantidad de paquetes transmitidos, la cantidad de paquetes recibidos, el porcentaje de pérdida de paquetes y el valor promedio, máximo y mínimo del RTT.

2.14.3 Iperf

IPerf (Dugan, Elliott, Mah, Poskanzer, & Prabhu, 2009) es una herramienta para medidas activas del ancho de banda máximo alcanzable en redes IP. Es compatible con diversos parámetros relacionados con la temporización, buffers y protocolos (TCP, UDP, SCTP con IPv4 e IPv6). Para cada prueba se informa el ancho de banda, pérdida, y otros parámetros.

Iperf permite medir el ancho de banda disponible hacia un “host” destino. Permite configurar múltiples parámetros de TCP y UDP. También puede utilizarse para determinar el “jitter”. Se basa en un sistema cliente-servidor, por lo tanto, en ambos “hosts” debe instalarse la aplicación.

2.15 Arquitecturas de QoS sobre IP

Hay dos arquitecturas de QoS que han sido propuestas y estandarizadas por la IETF. La primera se denomina Servicios Integrados (IntServ) y la segunda, Servicios Diferenciados (DiffServ). IntServ provee QoS a través de la reserva de recursos a lo largo del camino de datos, antes de iniciar la transmisión de los paquetes. Por otra parte, DiffServ incluye un conjunto de herramientas de clasificación y de mecanismos de cola que proveen a ciertas aplicaciones o protocolos, determinadas prioridades sobre el resto del tráfico en la red.

2.15.1 SERVICIOS INTEGRADOS (Integrated Services)

El esquema de trabajo de IntServ (Braden .R, 1994) preserva la semántica extremo a extremo de QoS para IP. Los puntos extremos clave son las aplicaciones emisoras y receptoras que solicitan un nivel de servicio deseado a la red para un conjunto de flujos, definidos por sus direcciones de origen, destino, puertos origen, destino y protocolo de transporte.

El modelo se sustenta en los siguientes supuestos:

- ✓ Los recursos se deben gestionar de forma direc- ta y explícita para satisfacer los requerimientos de las aplicaciones. Esto implica la utilización de mecanismos para control de admisión y reservación de recursos.

- ✓ Internet debe ser la infraestructura común para el tráfico normal y el de tiempo real, ya que construir una nueva red para el tráfico de tiempo real sería demasiado complejo. Esto implica que se debe unificar la pila de protocolos para cualquier tipo de tráfico, es decir, IP debe ser utilizado también para el transporte de datos de tiempo real.

Cada flujo se debe atender independientemente y no puede influenciar a otros. La arquitectura define dos clases de servicio adicionales al mejor esfuerzo: servicio garantizado y servicio de carga controlada, las cuales especifican el tratamiento que deben reci- bir los flujos a lo largo del camino. Además, IntServ requiere que los recursos necesarios para satisfacer los requerimientos de una aplicación o servicio se reserven sobre el trayecto con anticipación, para lo cual es necesario el Protocolo de Reservación de Recursos (RSVP). Éste utiliza un conjunto de mensajes de señalización para transportar infor- mación sobre los requerimientos y propiedades de cada flujo, la cual se utiliza para mantener tablas de estado en cada uno de los nodos, generando así un alto tráfico de señalización y ocupación de recursos en los dispositivos.

A. Clase de Calidad de Servicio

IntServ describe tres tipos principales de servicio que puede solicitar una aplicación:

- Guaranteed Services (Servicio garantizado) – [RFC2212]) – Provee límites firmes (matemáticamente probables) sobre las demoras de encolado de paquetes extremo a extremo , haciendo lo posible por proveer un servicio que garantice el ancho de banda y la demora.
- Controlled load (Carga Controlada) - [RFC2211]) – Provee al flujo de aplicación una calidad de servicio equivalente al QoS que recibiría el mismo flujo en un elemento de red que se encuentra bajo muy poca carga, pero utiliza control de capacidad (adminisión) para asegurar que el servicio es recibido aún si el elemento de red está sobrecargado.
- Besteffort service (Servicio de Mejor Esfuerzo) - no provee ninguna garantía de servicio de ningún tipo.

B. RSVP

(Braden, Zhang, Berson, Hersog, & S, 1997)El Protocolo de Reservación de Recursos se ha diseñado para permitir a los nodos finales y enrutadores de las secciones de comunicación (tanto multicast como unicast) comunicarse con el resto para establecer una ruta que pueda soportar la Calidad de Servicio requerida. La Calidad de Servicio se define en una especificación de flujo denominada flowspec.

RSVP identifica una sección por medio de una dirección, un tipo de protocolo de

Transporte y un número de puerto destino. RSVP no es un protocolo de enrutamiento, se usa para reservar recursos a través de la ruta que se establece por cualquiera de los protocolos de niveles inferiores.

Objetivo del diseño de RSVP

- ✓ Proporcionar la posibilidad de que receptores heterogéneos puedan hacer reservas de acuerdo a sus necesidades. No se debe asumir que todos los receptores tienen las mismas capacidades ni que requieran la misma Calidad de Servicio.
- ✓ Debe adaptarse a las variaciones de miembros en grupos multicast. La conexión o desconexión de los miembros de un grupo multicast debe ser dinámica.
- ✓ Permitir a los usuarios especificar sus necesidades al nivel de aplicación para que los recursos reservados para un grupo multicast pueda reflejar con precisión los recursos necesitados.
- ✓ Permitir a los receptores seleccionar entre varios canales. Un receptor debe ser capaz de controlar que paquetes son llevados en sus recursos reservados.
- ✓ Debe adaptarse a los cambios en las rutas unicast y multicast, RSVP utiliza el nivel de red para estos propósitos y mantiene un estado de las rutas.
- ✓ Controlar la sobrecarga que produce el protocolo en la red para que no crezca linealmente o peor con el número de participantes.
- ✓ Favorece el diseño modular para acomodar distintas tecnologías.

Tipos de enrutamiento para RSVP

Aunque se ha visto que RSVP no es un protocolo de enrutamiento si que hay cuatro

Problemas que se deben tratar con el protocolo de enrutamiento:

- Encontrar una ruta que soporte la reserva de recursos.
- Encontrar una ruta que tenga la suficiente capacidad disponible para un nuevo flujo. Se puede optar por dos formas diferentes de encontrar esta ruta. Una podría ser la de modificar los protocolos de enrutamiento y gestionarlos de acuerdo a un mecanismo de control del tráfico.
- Adaptarse a un fallo de ruta. Cuando un nodo falla, el enrutamiento adaptativo encontrará una ruta alternativa. El refresco periódico de RSVP automáticamente hará una reserva en la nueva ruta.
- Adaptarse a un cambio de ruta (sin fallo). Los cambios de ruta pueden ocurrir sin que se produzcan fallos. Aunque RSVP podría usar las mismas técnicas de reparación que las descritas en el punto 3, esta solución podría producir una merma en la Calidad de Servicio.

RSVP está actualmente diseñado para trabajar con cualquier protocolo de enrutamiento disponible sin modificación. Esto puede provocar que se produzcan ciertas degradaciones en la Calidad de Servicio al no cumplirse los anteriores requerimientos. Se espera que las futuras generaciones de protocolos de enrutamiento incluirán mecanismos que en conjunción con RSVP resolverán los problemas enumerados.

2.15.2 SERVICIOS DIFERENCIADOS

Esta propuesta plantea, asignar prioridades a cada uno de los paquetes que son enviados a la red. Cada enrutador deberá analizar y dar un tratamiento diferencial a cada uno de estos paquetes. En este enfoque no se necesita asignar ningún estado ni establecer algún proceso de señalización en cada nodo. Esta es la razón principal porque DiffServ ofrece mejor escalabilidad que IntServ. En el grupo de trabajo DiffServ de la IETF (Nichols .K, 1998), se define en el campo (DS Differentiated Services), válido tanto para IPV4 e IPV6, como el campo donde se asignará las prioridades a los paquetes, como se muestra en el Grafico N° 20.

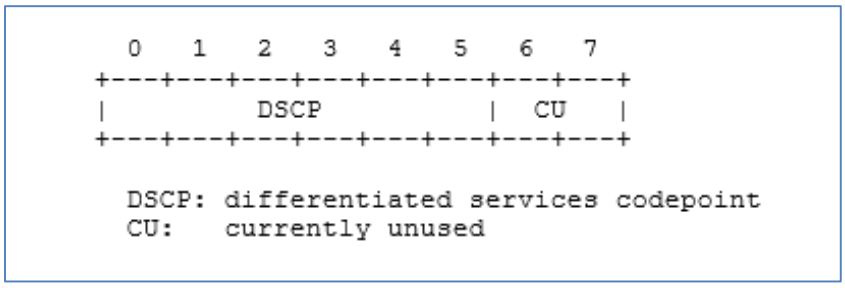


Grafico 20: Estructura del campo DSCP

En el sub-campo DSCP (Differentiated Services Code Point, Punto de Codificación de Servicios Diferenciados) es donde se especifica la prioridad de cada paquete de datos. La arquitectura DiffServ consta de nodos extremos DS de entrada y salida, así como nodos DS internos, tal como se muestra en el Grafico N° 21.

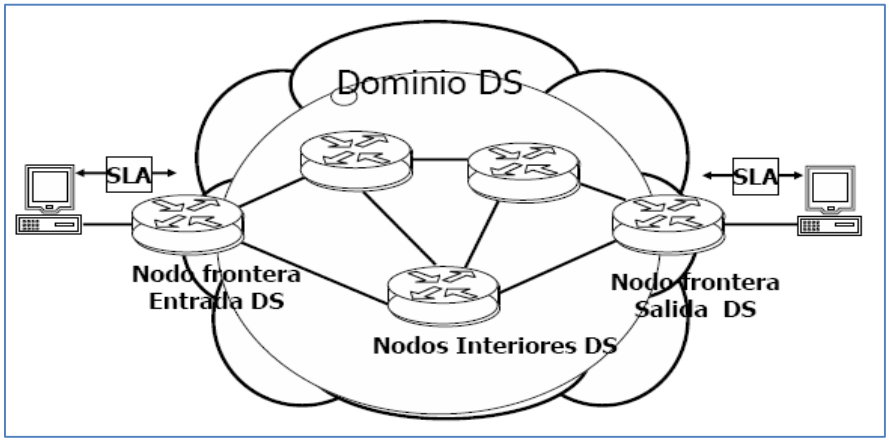


Grafico 21: Arquitectura de diffserv

Estos nodos DS conforman un dominio de servicio diferenciado y tienen un grupo de políticas y grupos de comportamiento por salto o (PHB, Comportamiento por Salto) bien definidos. Un dominio DS puede estar constituido por una o más redes bajo la misma administración, siendo responsable de asegurar que los recursos sean adecuadamente suministrados y reservados de acuerdo a un contrato de servicio o (SLA, Acuerdo de Nivel de Servicio) entre el cliente y el proveedor de servicio.

El Nodo DS extremo debe ser requerido para realizar funciones de acondicionamientos de tráfico entre los dominios DS que conecta, clasifica y establece las condiciones de ingreso de los flujos de tráfico, teniendo en cuenta el contenido de los campos: dirección IP de origen y destino, identificador de protocolo, puerto de origen y destino, y DS Code Point, Punto de Codificación, conocido como clasificador MF (clasificador Multi Campo).

Diffserv construye servicios desde los PHB clasificando paquetes en “clases” en el borde, o límite, de la red y marcando los paquetes correctamente. Opcionalmente, los paquetes pueden ser medidos con técnicas de vigilancia o modelado. El núcleo de la red implementa los PHBs y utiliza las marcas de los paquetes para hacer el encolado y las decisiones de descarte de paquetes (Nichols .K, 1998).

3. BASES CONCEPTUALES

3.1 Disciplina de colas (qdisc):

Un algoritmo que controla la cola de un dispositivo, sea de entrada (ingress) o de salida (egress). (Brown, 2006)

3.2 qdisc raíz (root qdisc):

La qdisc raíz es la que está adjunta al dispositivo.
Qdisc sin clases: Una qdisc sin subdivisiones internas configurables.

3.3 Qdisc con clases:

Una qdisc con clases contiene múltiples clases. Algunas de ellas contienen otra qdisc, que a su vez puede ser con clases, pero no tiene por qué. De acuerdo con la definición estricta, pfifo_fast *es* con clases, porque contiene tres bandas que son, en realidad, clases. Sin embargo, desde la perspectiva de configuración del usuario, no tiene clases ya que las clases no se pueden tocar con la herramienta tc. (Brown, 2006)

3.4 Clases:

Una qdisc con clases puede tener muchas clases, cada una de las cuales es interna a ella. A una clase, a su vez, se le pueden añadir varias clases. De manera que una clase puede tener como padre una qdisc u otra clase. Una clase terminal (o clase "hoja": leaf class) es una clase que no tiene clases hijas. Esta clase tiene 1 qdisc adjunta. Esta qdisc es responsable de enviar datos a la clase. Cuando creas una clase, se le adjunta una qdisc fifo. Cuando añades una clase hija, se elimina esta qdisc. Para clases terminales, esta qdisc fifo puede ser reemplazada con otra más adecuada. Incluso se puede reemplazar esta qdisc fifo por otra con clases de manera que se puedan añadir más clases. (Brown, 2006)

3.5 Clasificador:

Cada qdisc con clases necesita determinar a qué clase necesita enviar un paquete. Esto se hace usando el clasificador. (Brown, 2006)

3.6 Filtro:

La clasificación se puede realizar usando filtros. Un filtro contiene varias condiciones que pueden ser cumplidas. (Brown, 2006)

3.7 Best-Effort o Mejor Esfuerzo

Redes IP se describen a menudo como "mejor esfuerzo" redes. Esto se refiere al enfoque de la calidad del servicio en la red misma no distingue activamente en

su tratamiento de los servicios que el tránsito de la red. En una mejor red IP esfuerzo todos los paquetes IP son tratados de la misma manera.

La red se compromete su "mejor esfuerzo" para entregar cada paquete lo más rápido que puede, pero no hace ninguna empresa para tratar cualquier clase de paquetes de forma preferente a cualquier otra. Si bien esto suena como un enfoque perfectamente imparcial y justo, se dice a menudo de las redes de Internet que, para ciertas aplicaciones, "mejor esfuerzo" simplemente no es lo suficientemente bueno. Se afirma que lo que las redes IP necesitan alguna manera de proporcionar una respuesta superior para apoyar a ciertas clases de aplicaciones de alguna manera especial. "Mejor que mejor esfuerzo" es una manera de describir esta forma de la calidad del servicio (Geoff, 2001).

3.8 Sistema Operativo Linux

El actual kernel de Linux, viene con una serie de características que nos permitirá realizar un control avanzado del tráfico aplicando QoS (Calidad de Servicio).

En las versiones anteriores del Kernel 2.2.x, se tenía que recompilar el Kernel para poder tener las funciones de enrutamiento. Hoy en día el sistema operativo Linux y cualquier distribución Linux a partir de la versión de kernel 2.2.x, vienen instaladas de forma predeterminada las funciones necesarias de enrutamiento avanzado, por lo que se cuenta con un conjunto de herramientas o utilidades que permiten realizar tareas de enrutamiento avanzado, lo que permite efectuar la gestión y manipulación de los paquetes que se transmiten.

El servidor tiene instalado el sistema operativo GNU/Linux Centos 6.5 el cual tiene los siguientes paquetes, para proveer los servicios requeridos:

- Iproute2
- Netfilter

Todas las opciones que vienen instaladas de manera predeterminada, son necesarias para utilizar las herramientas Netfilter, IProute2, y las distintas clases y colas que permiten implementar la propuesta de QoS.

III. PROPUESTA DE SOLUCIÓN

1. DESARROLLO DE LA PROPUESTA

Hemos visto la situación actual de la problemática de las redes convencionales con política de mejor esfuerzo, frente a la posibilidad de mejorar la Calidad de Servicio (Quality of Service – QoS), describiendo las capacidades de Linux como sistema operativo y herramienta para aplicar control de tráfico de la red, para solucionar el problema presente en la red LAN de la Universidad y mejorar la Calidad de Servicio (Quality of Service – QoS).

Con estas herramientas (Iproute2/tc y Netfilter/Iptables), se propone desarrollar un esquema de control de tráfico de la red, con características de distribución de ancho de banda y priorización de tráfico.

Para la implementación del control de tráfico de la red propuesto, se optó por utilizar una herramienta de software libre (Sistema Operativo Linux - Distribución Centos Version 6.5), para lo cual se estudiaron detalles de operación de la misma. Para realizar el control de tráfico de la red, se utilizara la herramienta tc (Traffic Control) escrita por Alexey N. Kuznetsov, el cual permite manipular y realizar configuraciones en el Sistema Operativo GNU/Linux, con el manejo de los dispositivos de red (interfaz de red “eth0”), así como el flujo de paquetes a través de los mismos.

Para la siguiente propuesta de implementación de control de tráfico de la red con Linux en la red LAN se tomara en cuenta los siguientes puntos:

- Análisis del tráfico de la red LAN, de tal manera que se identifique cuáles son los servicios y aplicaciones de mayor demanda, los puertos más utilizados y el consumo de ancho de banda, para aplicar las políticas de control de tráfico de la red.
- Configurar las herramientas Netfilter/Iptables e Iproute2, para poder implementar el control de tráfico de la red. Para implementar la propuesta y aplicar el control de tráfico de la red, se debe configurar de forma apropiada la estructura de colas en el equipo con sistema operativo Linux el cual funciona como Router. Concretamente la configuración se llevará a cabo en el interfaz de salida hacia Internet, la interfaz de red “eth0”.
- Medir la Calidad de Servicio (Quality of Service – QoS) en la red LAN.

1.1 RED DE DATOS ACTUAL

La red LAN de la Universidad es una red Ethernet tipo IEEE 802.3 con las siguientes características:

- Su cableado estructurado actual esta implementado con cable categoría 5, transmite datos a velocidades de hasta 100 Mbps.
- Cuenta con un cuarto de telecomunicaciones central (Data Center), donde se encuentran los equipos de comunicación de datos de la Universidad (Servidores, Router, Switch).
- La troncal de la red esta interconectada a la red principal de Internet por el ISP (Proveedor de Servicio de Internet) de Movistar mediante un enlace de fibra óptica.
- Las áreas de trabajo o tomas de usuario (Oficinas administrativas, aulas y laboratorios) se conectan a la red de datos mediante el cableado horizontal, los cuales se conectan hacia los diferentes gabinetes de telecomunicaciones ubicados en los diversos ambientes de la Universidad.
- Se cuenta con un enlace a Internet dedicado de 4 Mbps.
- La red LAN de la Universidad cuenta en promedio con 170 usuarios (Oficinas Administrativas, Aulas y Laboratorios).
- Los gabinetes repartidores de planta se conectan hacia el cuarto de telecomunicaciones (Gabinete central), mediante el Backbone (cableado vertical), con enlaces de fibra óptica multimodo o mediante cable UTP de par trenzado.

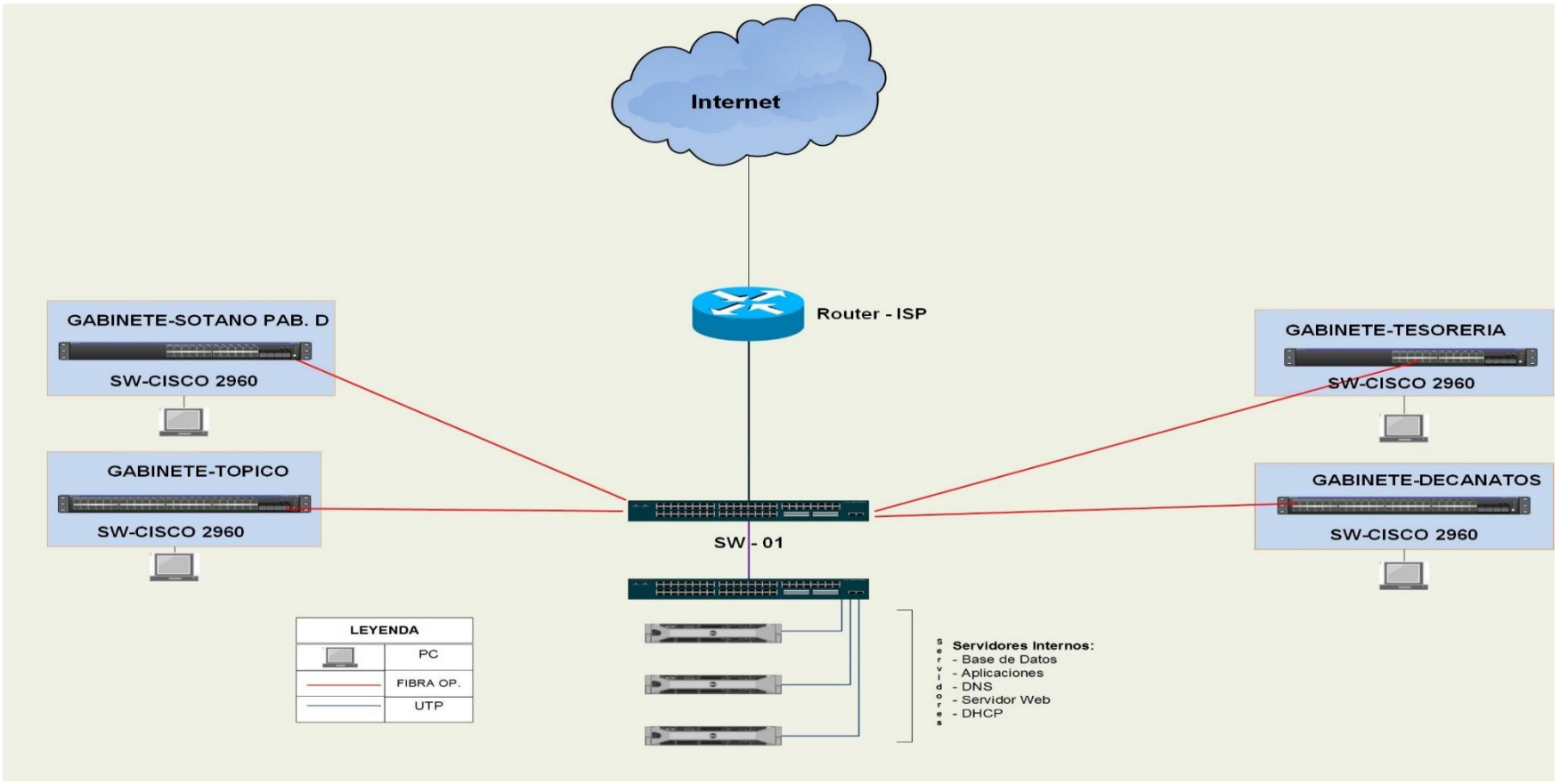


Ilustración 1: Topología de Red de datos actual de la Universidad

Fuente: Elaboración propia.

1.2 Propuesta de Arquitectura de red LAN para Implementar control de tráfico de la red.

En el Grafico N° 22, se presenta la propuesta de la arquitectura de red para implementar el control de tráfico de la red. Se tienen dos segmentos de red diferentes (eth0, eth1), en la interfaz de red “eth0” interfaz de salida hacia internet de los usuarios de la red LAN, está conectado hacia el router del ISP (Proveedor de servicio Internet) y en interfaz “eth1”, es la interfaz del servidor GNU/Linux que conecta hacia la red LAN. El servidor con sistema operativo Linux sirve de Firewall y Gateway a ambas redes y que a su vez servirá como dispositivo de control de tráfico de la red.

Para la propuesta de implementación, cuya arquitectura de red se muestra en el Grafico N° 22, se realiza el control de tráfico sobre la interfaz de red “eth0” del equipo GNU/Linux, es decir, se controlará el tráfico de la red en la interfaz de salida hacia internet por parte de los usuarios.

Este tipo de arquitectura se propone en base a diffserv, donde existen dos nodos extremos DS de entrada y salida, en la siguiente propuesta se realiza el control de tráfico en el nodo de salida, que en el grafico N° 22 está representado por el servidor GNU/Linux con QoS (Quality of Service).

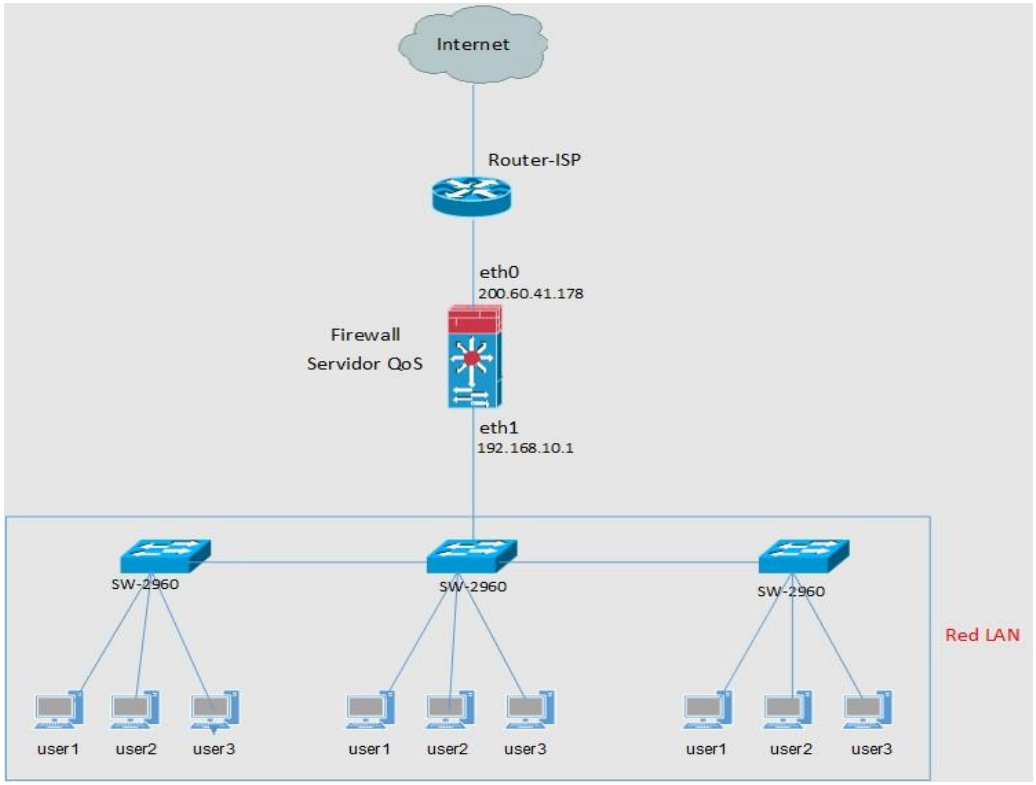


Grafico 22: Arquitectura de Red con Calidad de Servicio (QoS)

Fuente: Elaboración propia.

IV. RESULTADOS

A. DIAGNOSTICO

1. Analizar el tráfico de la red para implementar las políticas de control de tráfico de la red.

Para efectos del presente análisis, se realiza bajo el Sistema Operativo Linux, con la distribución CentOs 6.5 y con la versión del Kernel 2.6.32.

En el mismo equipo de cómputo se tiene implementado el Firewall: donde estará configurado NAT, filtrado y clasificación de tráfico con la herramienta IPTABLES.

Para llevar a cabo dicho análisis se hace uso de la herramienta **Ntopng** Versión 2, el cual permite analizar el tráfico de la red en tiempo real a través de una interfaz web. A partir de los datos obtenidos es posible implementar las políticas de control de tráfico, de esta manera se mejorara la Calidad de Servicio (Quality of Service - QoS) en la red LAN de la Universidad. Las mediciones se realizan en la interfaz de red “eth0” interfaz de salida hacia Internet, perteneciente al equipo con sistema operativo Linux el cual cumple la función de Router para la red LAN.

A. En el Grafico N° 23: se indica el tipo de protocolo más usado, a fin de obtener una estadística y saber cuál es el porcentaje de uso de cada uno de ellos.

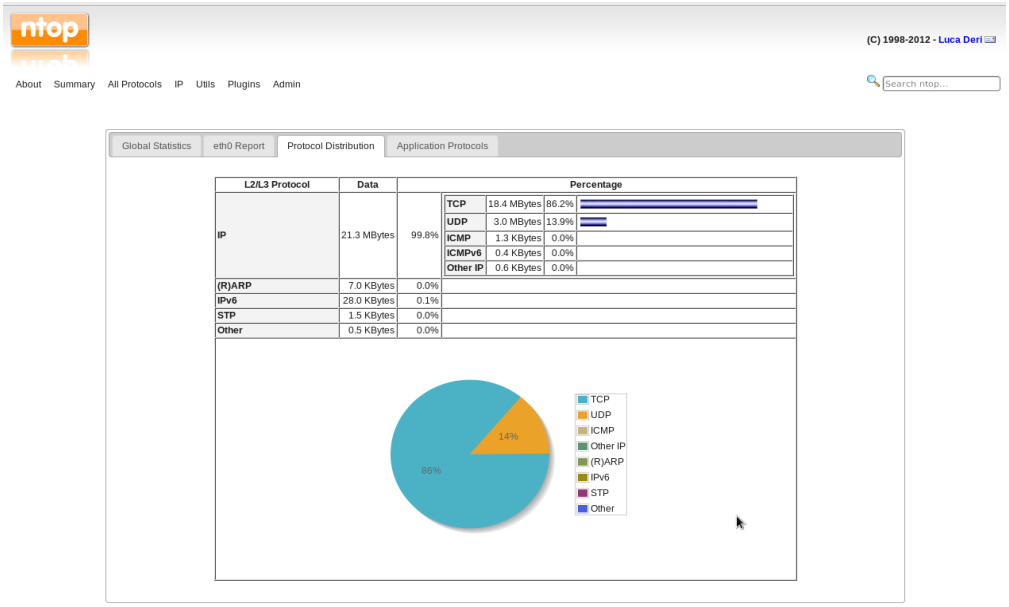


Grafico 23: Distribución global de los protocolos TCP/UDP

Fuente: Elaboración propia.

- En esta distribución global, el tráfico se clasifica de acuerdo al protocolo de red (IP, NetBios, etc.). Se observa que los protocolos de mayor consumo son el TCP que ocupa el 86.0% y el UDP que ocupa un 13.0%.

B. En el Grafico N° 24: se observa de manera detallada los tipos de protocolo que consumen mayor ancho de banda, en base a porcentajes.

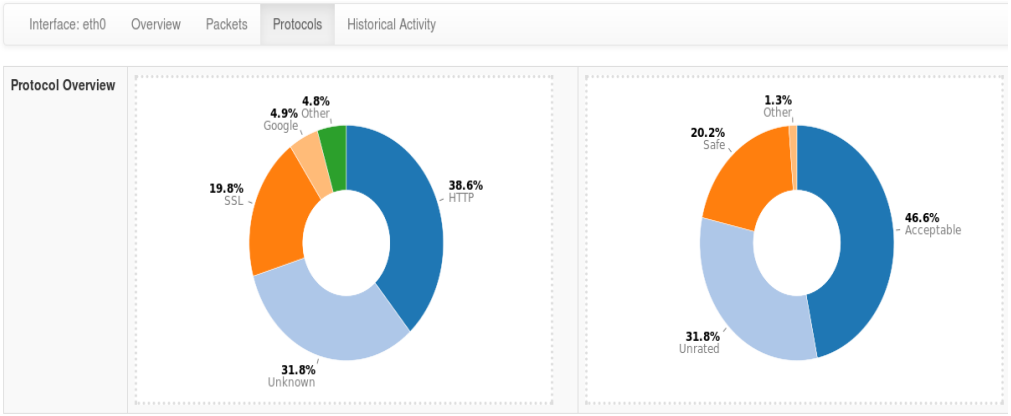


Grafico 24: Distribución de protocolo IP

Fuente: Elaboración propia.

- La clasificación del tráfico se hace de acuerdo al protocolo TCP en sus protocolo de aplicación como: (HTTP, FTP, etc.) como puede verse en el Grafico N° 24. Podemos ver que el protocolo TCP correspondiente al HTTP, tiene un mayor consumo de ancho de banda, esto se debe a que los usuarios principalmente realizan consultas de páginas Web, Correo, Chat, etc.

- También se puede visualizar que hay un gran porcentaje de consumo de ancho de banda 31.8% de protocolos desconocidos (Unknow), más se adelante se aplicara reglas de control de tráfico para este tipo de protocolos.

C. En el gráfico N° 25: se indica el tráfico total por puerto.

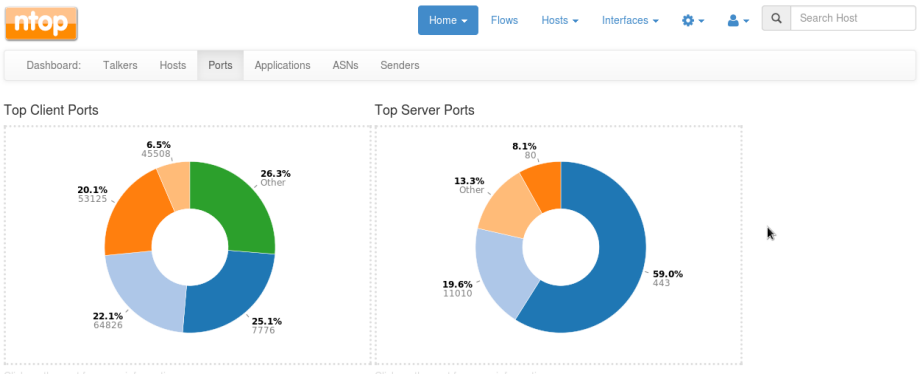


Grafico 25: Tráfico total por puerto

Fuente: Elaboración propia.

- En el grafico podemos analizar que el tráfico generado hacia el puerto 7776, correspondiente a las aplicaciones internas de la Universidad (Intranet Docentes, Intranet Administrativos – Workflow, Intranet Alumnos), es de 25.1%.
- El tráfico interno se da en mayor medida hacia la intranet de la Universidad http://intranet.edu.pe (puerto 7776), debido a que los usuarios (Plana Administrativa, Docentes y Alumnos) consultan estas aplicaciones para realizar tareas administrativas, subir notas, consultar notas y consultar los diferentes trabajos dejados por los docentes de la Universidad.

- Detalle de los aplicativos internos alojados en intranet.edu.pe:

  - ✓ Intranet Docente: Registro de notas, tareas asignadas a los alumnos, información relacionada con los cursos.
  - ✓ Intranet Alumno: El alumno puede revisar sus calificaciones, horarios, trabajos o tareas asignadas por el docente.
  - ✓ Intranet de Administrativos (Workflow): Se realizan tareas por parte del personal administrativo, como rendiciones y solicitudes de fondo, etc.

- Se tiene que aplicar reglas de control de tráfico hacia http://intranet.edu.pe (Puerto 7776), de tal manera que se priorice el tráfico hacia este puerto y asigne un porcentaje de ancho de banda, para que no se presente ningún inconveniente al consultar dichos aplicaciones y servicios.

2. Analizar el control de tráfico de la red, mediante la configuración de las herramientas Iproute2/Nerfilter.

Se realiza el análisis del control de tráfico de la red, en base al indicador de la primera variable (Disciplina de Cola), esta disciplina de cola está configurada en la interfaz de red “eth0”, interfaz de red instalada en el equipo con sistema operativo Linux que a su vez funciona como Router de la red LAN para la salida hacia Internet.
Este análisis se ejecuta una vez se tiene configurado las herramientas Iproute2 y Netfilter/Iptables, herramientas para control de tráfico de la red. El análisis se realiza antes de realizar el control de tráfico en la red y después de haber implementado el control de tráfico en la red LAN.

2.1 Configuración de Iproute2 con Herramienta Traffic Control (tc).

2.1.1 Creación de Políticas de control de tráfico de la red en Linux

Las reglas para controlar el tráfico de la red con Linux, se crean en base al resultado del monitoreo y análisis de tráfico de la red LAN, realizado en el punto N° 1 (Análisis de tráfico de la red LAN), en consecuencia, se consideran los siguientes puntos para crear las políticas de Calidad de Servicio (Quality of Service - QoS).

a) Criterios para crear las políticas de Calidad de Servicio (Quality of Service - QoS).

- ❖ Dar mayor prioridad al tráfico SSH, SYN, DNS, se consideran de alta prioridad debido a que la transmisión de datos es orientada a conexión, primero se tienen que enviar paquetes para comprobar si se puede o no establecer la comunicación, una vez que se comprueba se envía la respuesta y se inicia la comunicación de ambas partes Usuario - Servidor. Generalmente son aquellas aplicaciones donde la transmisión de datos se requiere de bajos delays (Perdida de Paquetes). Como el tráfico interactivo (telnet, SSH).

- ❖ El tráfico HTTP, HTTPS tendrá una prioridad media, ya que la mayoría de los usuarios utilizan el correo como parte fundamental y funcional de su trabajo para mantener una comunicación constante y fluida. Esto se comprobó en el monitoreo de la red LAN según las gráficas estadísticas presentadas anteriormente.

- ❖ Como tercera prioridad se tomara al tráfico generado hacia el puerto 7776, que corresponde al puerto que utilizan las aplicaciones internas de la Universidad, los cuales son consultados en gran medida por los usuarios tanto administrativos como alumnos de la Institución.

- ❖ El tráfico generado por aplicaciones Interactivas como chat e intercambio de archivos, BitTorrent, Streaming y cualquier otro tipo de tráfico no contemplado anteriormente, tendrá la prioridad más baja, por estar considerados dentro del tipo de servicios de ocio los cuales saturan la red.

b) Se generan 4 clases, esta distribución de clases se crea de acuerdo a los criterios analizados anteriormente en el punto (a):

Primera Clase: Esta clase se implementa debido a que la transmisión de información es orientada a conexión, por lo tanto se tiene que dar mayor prioridad a los paquetes SYN, ACK, SSH, etc.

Segunda Clase: Los usuarios necesitan hacer uso del Internet para la búsqueda de información, consulta de páginas web, envío de correos, de tal manera que realicen sus labores académicas y administrativas, por lo que se tiene que dar prioridad al protocolo HTTP y HTTPS.

Tercera Clase: Esta clase se implementa para dar prioridad al puerto 7776, el cual corresponde a las aplicaciones internas de la Universidad, donde los alumnos y administrativos realizan consultas a estas aplicaciones.

Cuarta Clase: Esta clase es para las aplicaciones de ocio y aplicaciones con puertos desconocidos que consumen un gran porcentaje de ancho de banda saturando la red.

Teniendo identificadas las clases anteriores, se especifican las aplicaciones, la asignación del ancho de banda y la prioridad correspondiente a cada una de estas clases, lo cual se detalla en la siguiente tabla N° 5.

CLASE	TIPO DE APLICACIÓN	ANCHO DE BANDA ASIGNADO	PRIORIDAD
:10	SSH, SYN, Trafico DNS	600 Mb	1
:11	HTTP, HTTPS	1800 Mb	2
:12	HTTP: Puerto 7776	1200 Mb	3
:13	BiTorrent, Streaming, Trafico en general sin clasificar.	200 Mb	4

Tabla 3: Especificaciones de QoS

Fuente: Elaboración propia.

c) Especificaciones generales para la creación de las políticas de calidad de servicio.

- La prioridad se toma desde el número más bajo hacia adelante, el número menor indica una prioridad alta. La prioridad 0 es la prioridad más alta y de mayor importancia.
- Las clases se forman de acuerdo a la prioridad que se van aplicar a cada una de ellas, cabe resaltar que mientras mayor sea la prioridad otorgada al servicio es menor el número que se asigna.
- El total de ancho de banda que se tiene para repartir entre las diferentes clases es de 4 MB.
- La qdisc principal se implementara en la interfaz de red eth0, esta interfaz es por donde se transmiten los paquetes hacia internet.
- La tarjeta de red eth0, tiene una qdisc raíz asociada a ella, y cuando se reciben los paquetes en la tarjeta, la qdisc recibe la petición de desencolar, tomando en cuenta las marcas en el paquete que se haya establecido con Netfilter/Iptables, es así como la qdisc lo enviará a alguna de las clases que contiene.
- Cada qdisc y cada clase tienen asignado un controlador el cual consta de 2 partes separadas por 2 puntos, un número mayor y un número menor (a:b).
- Las clases deben tener el mismo número mayor que sus padres y el número menor debe ser único dentro de una qdisc y sus clases.

2.1.2 Configuración de las reglas de control de tráfico de la red.

En esta parte se crean las reglas con las políticas de control de tráfico de la red analizadas anteriormente utilizando Iproute2 con su herramienta Traffic Control (t)c para crear las reglas de control de tráfico, luego se realizara la configuración de la herramienta Netfilter/Iptables para filtrar y marcar los paquetes de datos.

Inicialmente se comprueba la disciplina de cola que actualmente está configurada, en la interfaz de red "eth0" (interfaz de salida hacia Internet),

Verificamos con la siguiente instrucción:

```
# tc qdisc show dev eth0
```

Se muestra una salida similar a lo siguiente:

```
qdisc pfifo_fast 0 bands 3 priomap 1222120011111111
```

Donde:

- **tc qdisc:** indica la disciplina de colas que existen.
- **show dev:** es la instrucción para mostrar las colas que hay en el dispositivo.
- **eth0:** es la interfaz o tarjeta de red que se tiene configurada.

A. Diagrama del diseño de las colas.

Dentro de las múltiples disciplinas de cola, en esta propuesta de implementación se escoge HTB (Hierarchical Token Bucket), que se ajusta a los requerimientos planteados (ver en detalle HTB en el capítulo 2 dentro de la disciplina de colas con clase), El cual es útil para cuando se tienen diferentes tipos de tráfico y se les quiere dar un tratamiento por separado asignando prioridades diferentes.

(Navarro, 2005), esta disciplina de cola cuenta con scripts de configuración estándares (htb.init) que permiten definir las clases y su jerarquía, no obstante ello, las posibilidades de clasificación de los filtros de la arquitectura de control de tráfico (tc) son básicas. Es por esto que se utiliza como herramienta para realizar la etapa de clasificación, Iptables, que permite mayores capacidades de selección de paquetes. Para realizar la identificación y determinación del contexto de los paquetes (flujos) el controlador (Traffic Control - tc) utiliza iptables con marcas de firewall, las cuales existen solo durante el viaje del paquete dentro del kernel de Linux (marcas que viajan en algún campo de la cabecera IP tipo DSCP). Cuando el paquete es identificado (de acuerdo a la información de configuración) es marcado y su marca de firewall se asocia a

una clase de tráfico dentro de la jerarquía HTB. Un servicio o aplicación tiene asociada una marca de firewall y una clase de tráfico. Al servicio se le definen parámetros de control de tráfico (ancho de banda máximo y ancho de banda mínimo [rate y ceil de HTB]) y se le pueden definir múltiples filtros de paquete IP, denominados condiciones IP.

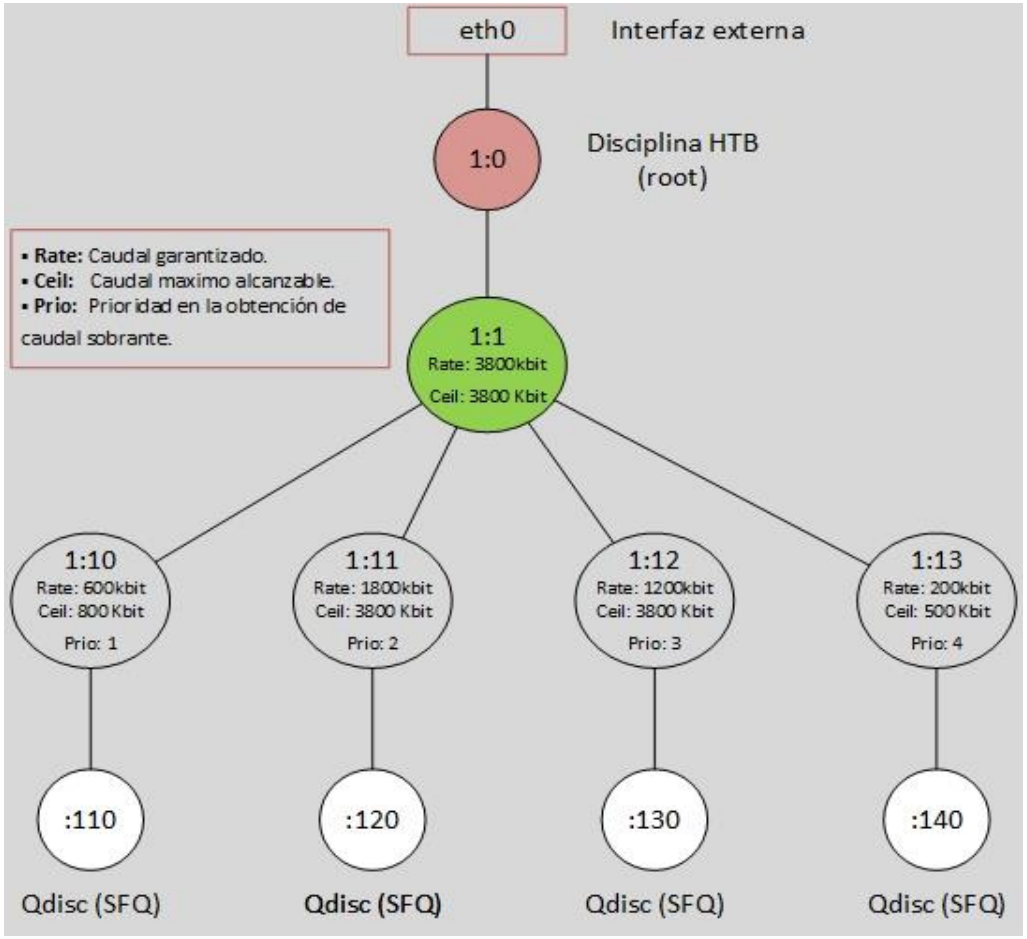


Grafico 26: Jerarquía de Control de Tráfico.

Fuente: Elaboración propia.

La clase raíz 1: se configura con la siguiente instrucción:

```
# tc qdisc add dev eth0 root handle 1: htb default 13
```

Donde:

$IF_EXT = eth0 (Tarjeta de red)

tc qdisc add añade a la interfaz de red “eth0” una qdisc raíz (root) con disciplina de cola htb, (1:), tambie se hace referencia a la clase 13, donde se irá el tráfico por defecto.

HTB, crea subcolas para garantizar y reservar ancho de banda, esto se da cuando se crea la clase 1:1, para organizar la llegada de los paquetes a la cola qdisc, como vemos en la siguiente instrucción:

```
# tc class add dev eth0 parent 1: classid 1:1 htb rate ${MAX}kbit ceil ${MAX}kbit
```

Donde:

tc class add crea una clase dentro de la qdisc raiz, donde la cola 1 es padre, indicado con parent 1: y el nombre se define con classid y es 1:1 (significa que es la hija 1 de la raíz 1).

En esta clase ningún paquete excede la velocidad de lo que se ha definido como máximo, para este caso se maneja un MAX de 3800kbit, y donde ceil se es el máximo posible de transferencia y rate el mínimo garantizado.

Adicionalmente se crean las clases denominadas 1:10, 1:11, 1:12, 1:13, con su máximo de transferencia (ceil) y su mínimo (rate) y su prioridad:

```
# tc class add dev eth0 parent 1:1 classid 1:10 htb rate 600kbit ceil 800kbit prio 1
```

```
# tc class add dev eth0 parent 1:1 classid 1:11 htb rate 1800kbit ceil 3800kbit prio 2
```

```
# tc class add dev eth0 parent 1:1 classid 1:12 htb rate 1200kbit ceil 3800kbit prio 3
```

```
# tc class add dev eth0 parent 1:1 classid 1:13 htb rate 200kbit ceil 600kbit prio 4
```

Dónde:

Las clases (classid) son hijas de 1:1 (parent 1:1), htb es la disciplina que se utiliza en cada una de las clases, con prio se indica la prioridad para cada una de las clases. Esta prioridad indica que si una de las clases no usa todo su ancho de banda asignado y garantizado, este ancho de banda sobrantes es repartido entre las clases restantes de acuerdo a sus prioridades.

La asignación de ancho de banda asignado a cada clase, se hace desde unos cuantos Kbytes hasta la capacidad máxima (4 MB), del ancho de banda, menos el 20% del total, esto por perdidas en las transmisiones de entrada y salida (Tx/Rx) de la señal.

Finalmente asignamos una cola (qdisc) a cada clase, para esto usamos la disciplina sfq, con esta disciplina de colas todas las clases son tratados de manera justa, cada qdisc 110: 120: 130: y 140: está asociada con las clases 1:10, 1:11, 1:12 y 1:13.

```
# tc qdisc add dev eth0 parent 1:10 handle 110: sfq
```

```
# tc qdisc add dev eth0 parent 1:11 handle 120: sfq
```

```
# tc qdisc add dev eth0 parent 1:12 handle 130: sfq
```

```
# tc qdisc add dev eth0 parent 1:13 handle 140: sfq
```

Filtrado de Paquetes

Sabemos que en las disciplinas de colas con clases es necesario llevar a cabo una clasificación del tráfico. Es decir, es necesario determinar a qué clase debe ir cada paquete IP. Para esto utilizamos el concepto de 'filtro' que asociamos a cada disciplina de cola si es necesario llevar a cabo una clasificación.

Las posibilidades a la hora de filtrar los paquetes son múltiples, incluso hay algunos tipos de filtros que son específicos de determinadas disciplinas de colas. En esta propuesta sólo haremos uso del filtro fwmark, el cual nos deja usar el código de netfilter (iptables) para marcar el tráfico:

Cuando se tenga el tráfico marcado y las clases de tráfico creado, establecemos las reglas para encolar el tráfico en su clase correspondiente, dentro de la clasificación de colas predefinidas.

Para ello, de nuevo utilizamos la herramienta tc:

```
# tc filter add dev eth0 protocol ip parent 1: prio 1 handle 1 fw classid 1:10
```

```
# tc filter add dev eth0 protocol ip parent 1: prio 2 handle 2 fw classid 1:11
```

```
# tc filter add dev eth0 protocol ip parent 1: prio 3 handle 3 fw classid 1:12
```

```
# tc filter add dev eth0 protocol ip parent 1: prio 4 handle 4 fw classid 1:13
```

Con estas reglas indicamos que vamos a crear un filtro "tc filter" asociado a la interfaz de red eth0 "add dev eth0" y a la clase root "parent 1:0" (la qdisc que hemos creado antes). Este filtro es sobre el protocolo ip "protocol ip" y le fijamos una prioridad para indicar en qué orden se aplican los filtros "prio 1" (mayor número, mayor prioridad). A continuación viene qué condiciones debe cumplir nuestro paquete (con el filtro fwmark), el cual se encargará de manejar los paquetes (handle 1, handle 2, handle 3, handle 4) y dirigirlos (fw) a la cola

adecuada gestionada por la clase 1:1x (classid 1:10, classid1:11, classid 1:12, classid 1:13).

Cabe mencionar que el fw clasifica en base a la marca del paquete que esté hecha con IPtables.

B. Script diseñado en base a las políticas mencionadas anteriormente. Para este caso, se divide todas aquellas aplicaciones o tipo de tráfico a las que se le dará un trato o prioridad especial.

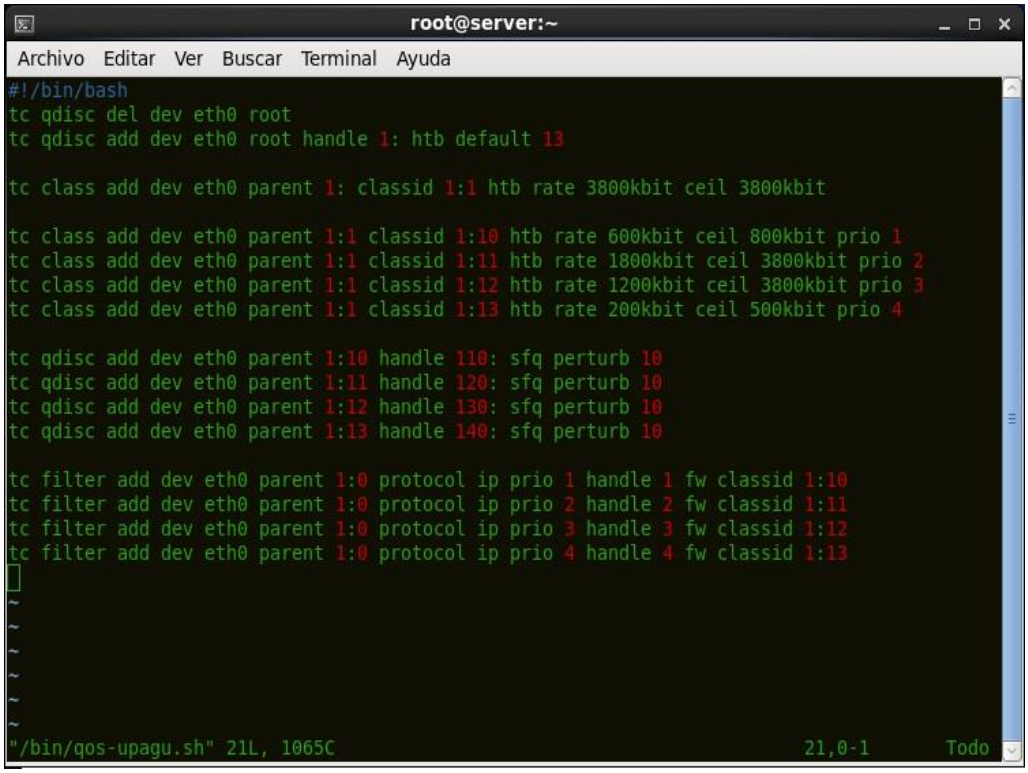


Grafico 27: Script qos.sh reglas QoS

Fuente: Elaboración propia.

Al terminar de crear las reglas se inicia el script con la siguiente instrucción, desde una terminal de Linux.

```
# root@localhost:/bin/qos #  ./qos.sh
```

Donde:

root@localhost:/bin/qos muestra la ruta donde se encuentra el script dentro del servidor, llamado también prompt y qos.sh es el nombre que se le da a dicho script.

Y con la siguiente instrucción: tc -s class show dev eth1

Se muestra (show dev) que realmente el tráfico esté en las clases que se definieron anteriormente, con la opción -s se pueden ver las estadísticas de una interfaz de red (eth0).

En el Grafico N° 28, se observa que las clases creadas contienen información sobre cuantos bytes y paquetes se están enviando en tiempo real, en resumen el tráfico que pasa en cada una de las clases es debido al trato que se le da en las prioridades de acuerdo a la marca que se le asigna a cada paquete.

Grafico 28: Flujo de tráfico en las clases creadas

Fuente: Elaboración propia.

2.2 Configuración de Firewall con Netfilter/Iptables

Una vez creadas las clases y definido sus prioridades se procede al marcado de cada uno de los paquetes, para cada tipo de tráfico definido anteriormente, de acuerdo al tipo de protocolo o puerto. Es aquí donde se hace uso del Netfilter al invocar IPtables en cada una de las políticas.

El árbol que se ha creado en el punto anterior, no es capaz de clasificar el tráfico. Entonces hacemos uso del firewall de linux (Iptables) para hacer unas "marcas" a los paquetes con el fin de que la qdisc raíz envíe el tráfico a las clases adecuadas.

Para etiquetar o marcar los paquetes utilizaremos la tabla mangle de Iptables sobre la cadena PREROUTING, Con -j MARK y -set-mark [id] le ponemos el número de marca al paquete, esto se realiza antes de tomar la decisión de encaminamiento, de nada servirá marcar los paquetes después de que se hubiera tomado la decisión de encaminamiento.

La configuración de Iptables se realiza en el servidor con sistema operativo Linux Centos 6.5 que tiene la función de Router, por lo tanto dentro de este equipo se realiza la configuración del Firewall.

En el punto de Netfilter se analizaron los parámetros de configuración de iptables de tal modo que se pueda marcar los paquetes en la tabla mangle sobre la cadena PREROUTING, por lo que se obvia la explicación del siguiente script de Iptables.

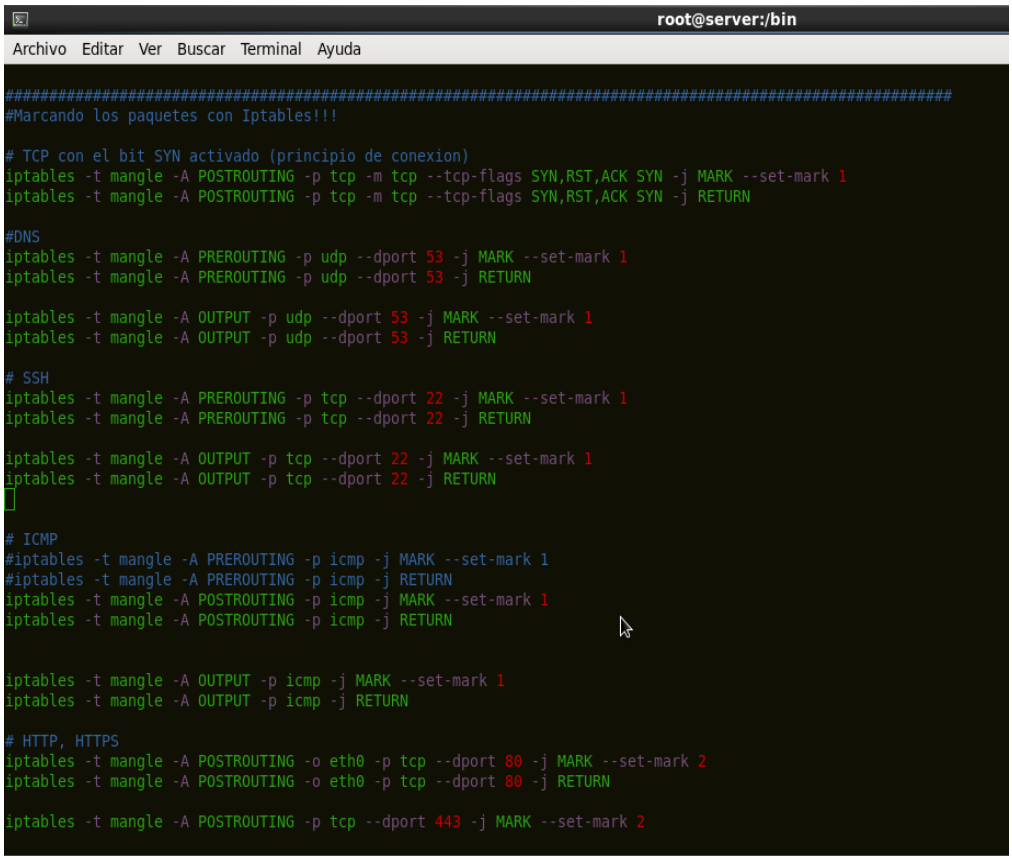


Grafico 29: Script de Iptables (iptables.sh).

Fuente: Elaboración propia.

Todos los equipos de los usuarios de la red LAN, de esta configuración están utilizando direcciones IPv4 privadas según la RFC 1918 y por tanto no son enrutables directamente a Internet. Damos por hecho que el equipo con función de Router está configurado para hacer SNAT, realizando una traducción de su dirección IP privada por una dirección IP pública de los paquetes que vengan de su interfaz interna de la red LAN(interfaz de red "eth1").

2.3 Resultados en la red LAN sin tener control de tráfico de la red.

2.3.1 Primera Variable

La prueba se realizó en base a la variable: "Control de Tráfico de la Red", la cual referencia al disciplina de colas configurada en la interfaz de red "eth0", interfaz de red de salida hacia Internet, esta interfaz de red está instalada en el Servidor con Sistema Operativo Linux que funge como Router de la red LAN.

La disciplina de cola más usada e instalada por defecto en Linux es pfifo_fast, aquí ningún paquete recibe un tratamiento especial, el primer paquete que

llega es el primero que sale y todos los paquetes son tratados exactamente igual, sin ningún trato especial. Esta disciplina de cola tiene tres bandas o clases con prioridades, la prioridad 0 es de mayor prioridad que 1 y 1 que 2. El kernel toma en cuenta el campo ToS (Tipo de Servicio) e inserta los paquetes de mínimo retardo en la banda 0.

- **Indicador**

  El indicador para esta dimensión es: **"La disciplina de cola"**, esta se verifica antes de realizar el control de tráfico de la red en Linux para mejorar la Calidad de Servicio (QoS) en la red LAN de la Universidad.

  Para ver la disciplina de cola configurada en la interfaz de red "eth0", hacemos uso del comando tc (Traffic Control) herramienta de Iproute2 a través de la terminal de Linux (S.O CentOs):

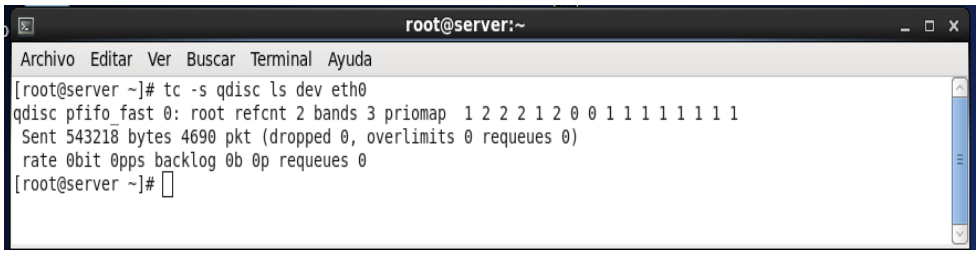


Grafico 30: Disciplina de Cola

Fuente: Elaboración propia.

Detalle:

Disciplina de cola	: **qdisc pfifo_fast**
Bandas	: 3 priomap
Bytes transmitidos	: 543218 Bytes
Paquetes	: 4690 pkt

2.4 Resultados con control de tráfico de la red implementado.

En este punto hacemos uso del comando tc (Traffic Control), donde verificamos que la disciplina de cola cambia en relación a la disciplina de cola utilizada antes de aplicar control de tráfico de la red con Linux.

❖ **Indicador**

El indicador para esta dimensión es: **"La disciplina de cola"** utilizada después de realizar el control de tráfico de la red con Linux para mejorar la Calidad de Servicio (QoS) en la red LAN de la Universidad.

Disciplina de cola en la interfaz de red "eth0".

Comando utilizado:

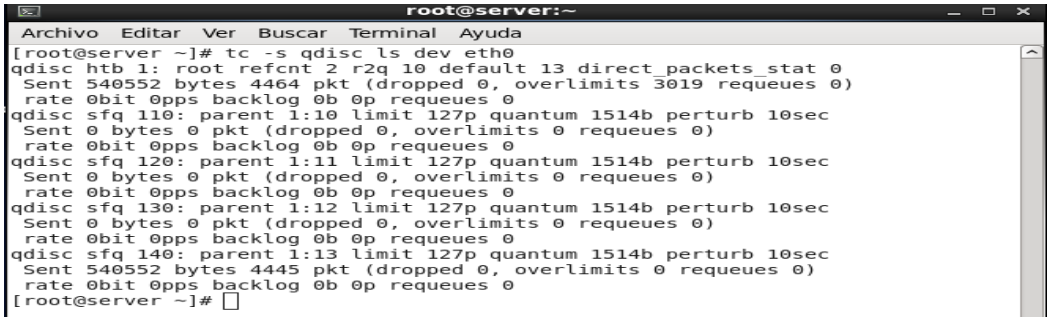


Grafico 31: Disciplina de cola HTB

Fuente: Elaboración propia.

Detalle:

Disciplina de cola	: **qdisc htb**
Cola por defecto	: 13
Bytes transmitidos	: 5430552 Bytes
Paquetes	: 4464 pkt

La administración de colas en Linux en la interfaz de salida "eth0", aplicando el control de tráfico de la red, cambia con respecto a la disciplina utilizada anteriormente, la disciplina de cola en funcionamiento es la htb (Hierarchical Token Bucket).

Indicador:

Disciplina de Cola = htb

En el análisis de las pruebas realizadas en este punto se puede verificar que la disciplina de cola cambia con respecto a la disciplina utilizada, antes de aplicar un control de tráfico a la red LAN.

3. Medición de la Calidad de Servicio (QoS) en la red LAN.

En esta parte se realiza las mediciones, en cuanto a los parámetros recomendados por UIT-T en su apartado G.1010, el cual hace referencia al Ancho de banda, Latencia y Tasa de pérdidas, las mediciones se realizan antes de realizar el control de tráfico en la red y después de haber implementado el control de tráfico en la red LAN para mejorar la Calidad de Servicio (QoS).

A. Resultados sin tener Calidad de Servicio (QoS) en la red LAN

1) Ancho de Banda

La tasa de transferencia depende del ancho de banda, la cual esta compartido con múltiples aplicaciones al mismo tiempo. La tasa de transferencia es la cantidad de información o datos que se envían a través de una conexión de red en un periodo de tiempo establecido. Por lo general se mide en bits por segundo (bps), kilobits por segundo (Kbps) o megabits por segundo (Mbps). (Espinosa Ceniceros, 2010)

Indicador: El indicador para esta dimensión es: **"**caudal máximo o taza de transferencia que se puede transmitir en Bytes/segundo**"**, en esta parte el indicador se mide antes de realizar el control de tráfico de la red con Linux.

➢ **Prueba 1**

Esta prueba se realiza al protocolo TCP en base a su protocolo de aplicación HTTP, que de acuerdo al análisis de tráfico de la red LAN, realizado en el punto N° 1 (Análisis de trafico de Red), es el protocolo más usado por los usuarios de la red LAN de la Universidad. Para medir la taza de transferencia del protocolo HTTP, se realiza en el servidor que cumple la función de Router y firewall, en su tarjeta de interfaz "eth0".

- **Trafico HTTP**

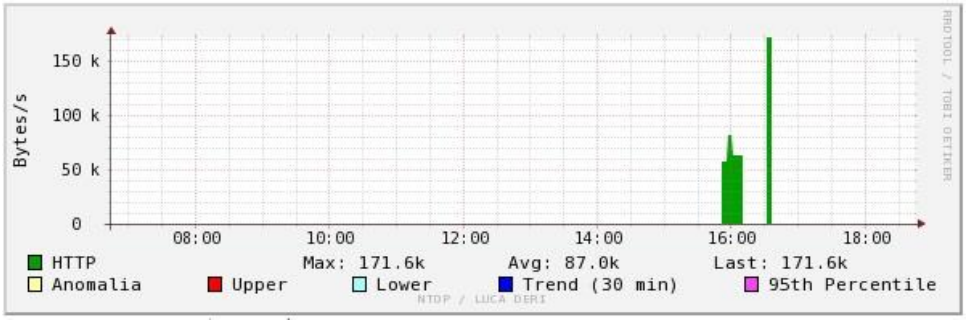


Grafico 32: Trafico HTTP

Fuente: Elaboración Propia

- El consumo de los usuarios de la red LAN, hacia el protocolo http, como se puede apreciar en el Grafico N° 32, es de 171.6 Kbps.

  Caudal máximo protocolo HTTP = 171.6 Kbps

- **Trafico Protocolos desconocidos**

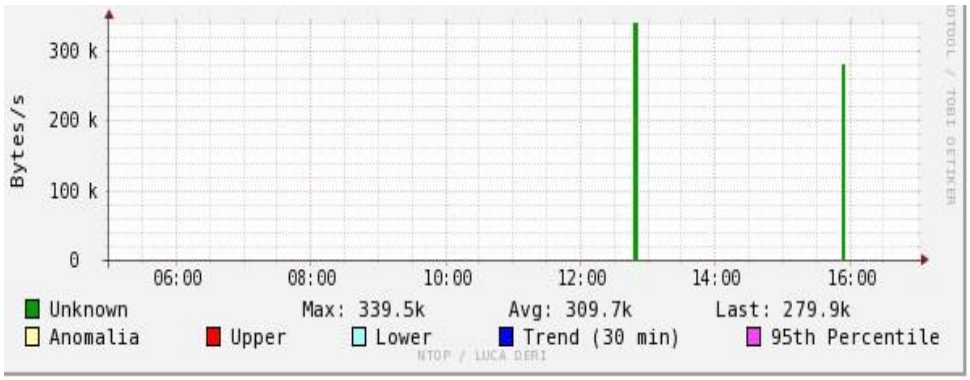


Grafico 33: Trafico Unknow

Fuente: Elaboración propia.

- El consumo de los usuarios de la red LAN, hacia protocolos desconocidos, Ver el Grafico N° 33, llega a los 339 Kbps.

  Caudal máximo protocolo Unknown = 339 Kbps

➢ **Prueba 2.**

Se realizan las pruebas al protocolo TCP en su protocolo de aplicación HTTP, esta prueba se realiza en base al porcentaje que consume este tipo de protocolo.

En el Grafico N° 34, se muestra el tráfico correspondiente al protocolo de aplicación HTTP que se analizó en la red LAN. Se observa que el máximo valor de tráfico HTTP alcanza el 14.6 % y el consumo por parte de los protocolos desconocidos (Unknown) alcanza un porcentaje de 50.4%, estas pruebas se realizaron antes de tener implementada la Calidad de Servicio (QoS).

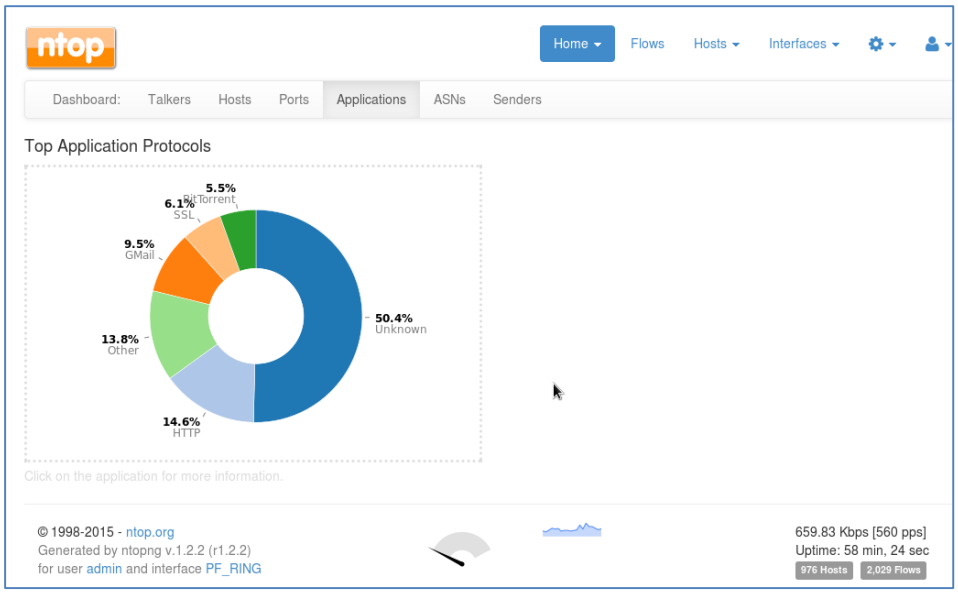


Grafico 34: Porcentajes de tipos de tráfico.

Fuente: Elaboración propia.

2) Latencia

Latencia: Tiempo entre el envío de un mensaje por parte del equipo transmisor y la recepción del mensaje por parte del equipo receptor.

Las pruebas se realizan bajo un esquema de red de área local (LAN) con tecnología Ethernet y TCP/IP como sistema de transporte, una configuración ampliamente difundida a nivel de cliente-servidor.

Indicador:

El indicador para esta dimensión es: **"Tiempo promedio que tarda en llegar los paquetes"**, en esta parte el indicador se mide antes de realizar el control de tráfico en Linux.

- **Prueba 1**

  En esta prueba para medir la latencia de la red se realizó bajo el esquema de red del grafico N° 35, esta se realiza entre un equipo cliente de la red LAN de la Universidad y un host ubicado en la red de Internet.

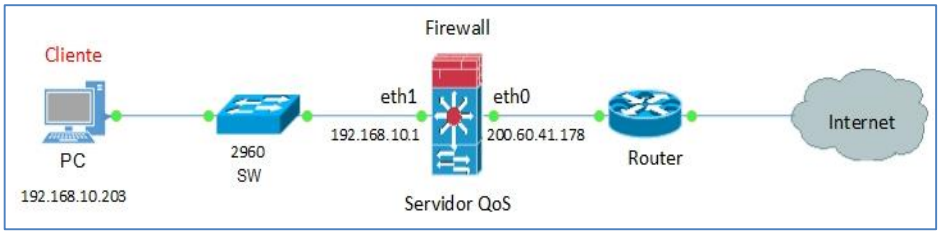


Grafico 35: Esquema de red para pruebas de latencia.

Fuente: Elaboración propia.

La prueba se realizó ejecutando un ping en la PC con dirección IP (192.168.10.203) de la red LAN, hacia un host remoto (www.google.com) ubicado en la red de Internet.

Resultados: Se realizó un ping hacia la página www.google.com, por ser una de los buscadores que es consultado constantemente por parte de los usuarios de la red LAN, arrojando los siguientes resultados:
Las pruebas se realizaron con tamaños de tramas Ethernet de 32 Bytes. Hacia la dirección IP: 173.194.219.104, ver Anexo N° 1.

Sitio Web	Dirección IP	Bytes	Time
www.google.com	173.194.219.104	32 Bytes	169 ms
		32 Bytes	167 ms
		32 Bytes	203 ms
		32 Bytes	173 ms
		32 Bytes	205 ms
		.	.
		.	.
		.	.
		32 Bytes	208 ms
		32 Bytes	119 ms
		32 Bytes	202 ms

Tabla 4: detalle de la latencia de la red de datos.

Fuente: Elaboración propia.

Estadísticas:

- Numero de paquetes transmitidos = 33
- Numero de paquetes recibidos = 33
- Paquetes perdidos = 0%
- Tiempo aproximado de ida y vuelta (round-trip):
  - Mínimo = 88 ms
  - **Media = 166 ms**
  - Máximo = 208 ms

Al realizar esta prueba se obtiene los siguientes resultados:

Indicador:

Tiempo promedio = 166 ms

En base a los parámetros recomendados por la ITU-T G.114, el resultado de la medición del indicador presenta un nivel inaceptable para el funcionamiento adecuado de la red de datos. Este resultado puede afectar a algunas de las aplicaciones de la red de datos.

- **Prueba 2**

En esta prueba para medir la latencia de la red se realizó bajo el esquema de red del Grafico N° 36, se realiza entre equipo cliente de la red LAN de la Universidad y un host ubicado en la red de Internet.

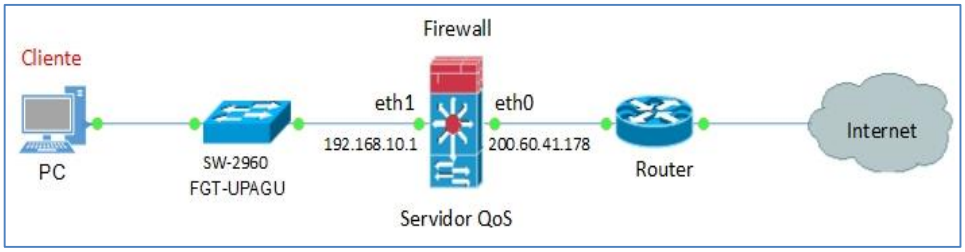


Grafico 36: Esquema de red para medir la latencia de la red.

Fuente: Elaboración propia.

La prueba se realizó ejecutando un ping en la PC con dirección IP (192.168.10.200) de la red LAN, arrojando lo siguientes resultados antes de implementar un control de tráfico en la red de datos.

La prueba se hizo realizando un ping hacia la página www.facebook.com, arrojando los siguientes resultados:

Las pruebas se realizaron con tamaños de tramas Ethernet de 64 Bytes. Hacia la dirección IP: 179.60.192.13, ver Anexo N° 2.

Dirección IP	Bytes	Time
179.60.192.13	64 Bytes	564.3 ms
	64 Bytes	565.1 ms
	64 Bytes	559.8 ms
	64 Bytes	571.1 ms

Tabla 5: detalle de la latencia de la red de datos.

Fuente: Elaboración propia.

Estadísticas:

- Numero de paquetes transmitidos = 5
- Numero de paquetes recibidos = 4
- Paquetes perdidos = 20%
- Tiempo aproximado de ida y vuelta (round-trip):
  - Mínimo = 559.8 ms
  - **Media = 565.5 ms**
  - Máximo = 571.1 ms

Al realizar esta prueba se obtiene los siguientes resultados:

Indicador:

Tiempo promedio = 565.5 ms

En base a los parámetros recomendados por la ITU-T G.114, Observando que la latencia de la red está por encima de los 400 ms. Esta es una mala performance de la red excediendo los parámetros recomendados por la ITU-T G.114, teniendo un nivel inaceptable para el funcionamiento adecuado de la red de datos.

B. Resultados aplicando control de tráfico para mejorar la Calidad de Servicio (Quality of Service - QoS).

1) Ancho de Banda

Para medir el ancho de banda (Tasa de transferencia Kbytes/s) se vuelve a utilizar la herramienta Ntopng, se mide el consumo de ancho de banda, con respecto al protocolo http y protocolos desconocidos, estas mediciones se realizan teniendo implementada el control de tráfico de la red, con las diferentes reglas para mejorar la Calidad de Servicio (QoS) en la red LAN.

Indicador: El indicador para esta dimensión es: **"**caudal máximo o taza de transferencia que se puede transmitir en Bytes/segundo**"**.

- **Prueba 1**

Se hacen las pruebas al protocolo TCP en base a su protocolo de aplicación HTTP, que de acuerdo al análisis de tráfico de la red realizado en el punto N° 1 (Análisis de tráfico de la Red), es el protocolo más usado por los usuarios de la red LAN de la Universidad y el tráfico de los protocolos desconocidos que son los que generan congestión en la red LAN. Para medir la taza de transferencia de estos protocolos, se realiza en el servidor que cumple la función de Router y firewall, en su tarjeta de interfaz de red "eth0", sobre esta interfaz fluye todo del tráfico de la red LAN hacia Internet.

- **Trafico HTTP**

En el Grafico N° 37, se muestra el tráfico correspondiente al protocolo HTTP que se analizó en la red LAN. Se observa que el máximo valor de tráfico alcanza los 328.0 Kbps.

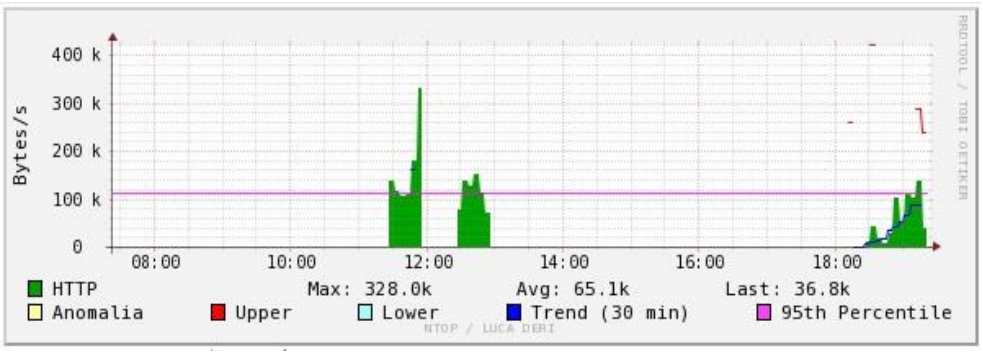


Grafico 37: Trafico HTTP con QoS

Fuente: Elaboración propia.

Caudal máximo protocolo HTTP = 328.0 Kbps

- **Trafico desconocido**

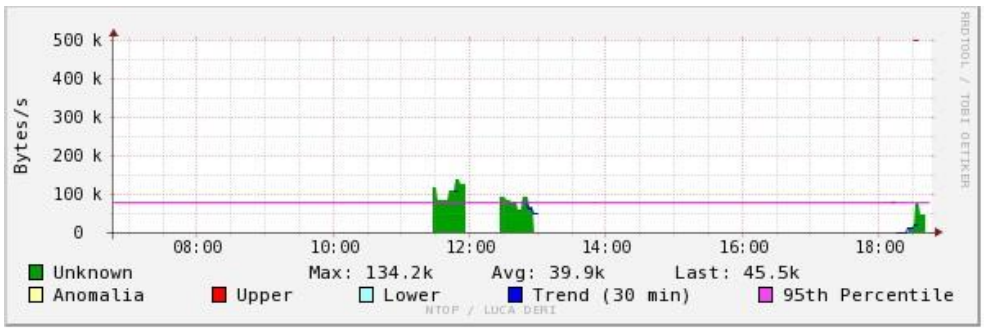


Grafico 38: Trafico Unknow con QoS

Fuente: Elaboración propia.

Caudal máximo protocolo Unknown = 134.2 Kbps

➢ **Prueba 2.**

Se hacen las pruebas al protocolo TCP en su protocolo de aplicación HTTP, esta prueba se realiza en base al porcentaje que consume este tipo de protocolo. En el Grafico N° 39, se muestra el tráfico correspondiente al protocolo de aplicación HTTP que se analizó en la red LAN. Se observa que el máximo valor de tráfico HTTP alcanza el 60.7% y los protocolos desconocidos 11.3% de consumo de ancho de banda.

Grafico 39: Trafico de Protocolo HTTP y Unknow

Fuente: Elaboración propia.

Resultados:

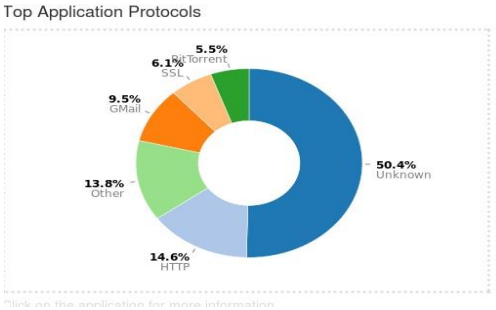
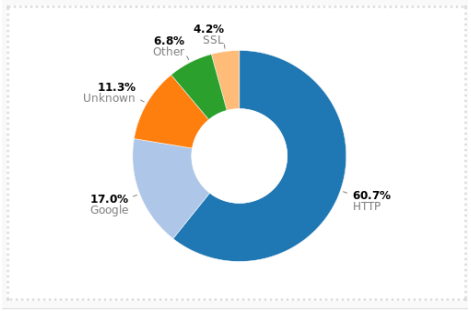
Sin QoS	Con QoS
	

Grafico 40: Porcentaje de Trafico HTTP

Fuente: Elaboración propia.

2) Latencia

A. Prueba 1

Esta prueba para medir la latencia de la red se realizó bajo el esquema de red del grafico N° 4, esta se realiza entre un equipo cliente de la red LAN de la Universidad y un host ubicado en la red de Internet.

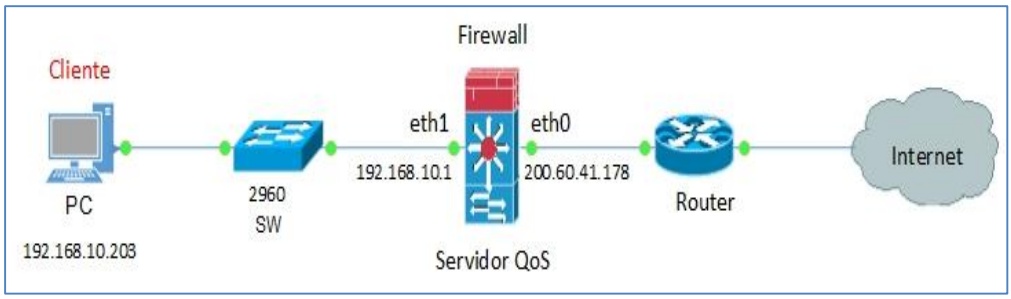


Grafico 41: Esquema de red para medir la latencia.

Fuente: Elaboración propia.

La prueba se realizó en uno de las PC´s con dirección IP (192.168.10.203) de la red LAN, arrojando lo siguientes resultados. Se realizó esta prueba ya teniendo implementado el control de tráfico en la red de datos. Para esto se hizo uso del comando ping.
La prueba se hizo realizando un ping hacia la página www.google.com, arrojando los siguientes resultados:

Las pruebas se realizaron con tamaños de tramas Ethernet de 32 Bytes. Hacia la dirección IP: 173.194.219.104, ver Anexo 3.

Sitio Web	Dirección IP	Bytes	Time
www.google.com	216.58.219.132	32 Bytes	67 ms
		32 Bytes	67 ms
		32 Bytes	67 ms
		32 Bytes	69 ms
		32 Bytes	78 ms
		.	.
		.	.
		.	.
		32 Bytes	68 ms
		32 Bytes	69 ms
		32 Bytes	67 ms

Tabla 6: detalle de la latencia de la red de datos.

Fuente: Elaboración propia.

Estadísticas:

- Numero de paquetes transmitidos = 14
- Numero de paquetes recibidos = 14
- Paquetes perdidos = 0%
- Tiempo aproximado de ida y vuelta (round-trip):
  - Mínimo = 67 ms
  - Media = 68 ms
  - Máximo = 78 ms

Al realizar esta prueba se obtiene los siguientes resultados:

Indicador:

Tiempo promedio = 68 ms

Se observa que la latencia de la red tiene un promedio de 68 ms y un máximo de 78 ms. En tal sentido el retardo está por debajo de los 150 ms, de los parámetros recomendados por la ITU-T G.114, teniendo un nivel aceptable para el funcionamiento adecuado de la red de datos.
Esta reducción en la latencia de la red de datos es producto de la implementación de QoS con control de tráfico en Linux.

B. Prueba 2

En esta prueba para medir la latencia de la red se realizó bajo el esquema de red del grafico N° 42, se realiza entre un equipo cliente de la red LAN de la Universidad y un host ubicado en la red de Internet.

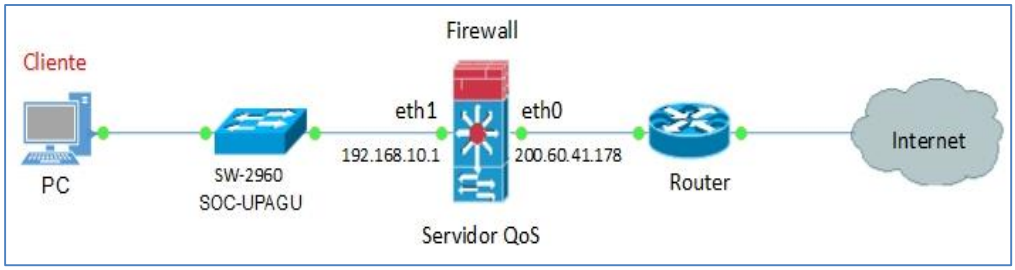


Grafico 42: Esquema de red para medir la latencia de la red.

Fuente: Elaboración propia.

La prueba se realizó ejecutando un ping en la PC con dirección IP (192.168.10.200) de la red LAN. Se realizó esta prueba ya teniendo implementado el control de tráfico en la red de datos. Para esto se hizo uso del comando ping.

Una de las pruebas se hizo realizando un ping hacia la página www.facebook.com, arrojando los siguientes resultados:

Las pruebas se realizaron con tamaños de tramas Ethernet de 64 Bytes. Hacia la dirección IP: 179.60.192.13, ver Anexo N° 4.

Direccion Web	Dirección IP	Bytes	Time
www.facebook.com	31.13.73.1	64 Bytes	88.9 ms
		64 Bytes	67.6 ms
		64 Bytes	66.8 ms
		64 Bytes	66.8 ms
		64 Bytes	66.6 ms

Tabla 7: detalle de la latencia de la red de datos.

Fuente: Elaboración propia.

Estadísticas:

- Numero de paquetes transmitidos = 5
- Numero de paquetes recibidos = 5
- Paquetes perdidos = 0%
- Tiempo aproximado de ida y vuelta (round-trip):
  - Mínimo = 66.6 ms
  - **Media = 71.3 ms**
  - Máximo = 88.9 ms

Al realizar esta prueba se obtiene los siguientes resultados:

Indicador:

Tiempo promedio = 71.3 ms

3) Tasa de Pérdidas

Prueba de implementación: Iperf

Tal y como se expuso en el Capítulo 2, Iperf es una herramienta que permite efectuar mediciones de desempeño de red.

Iperf trabaja en modo cliente-servidor y puede operar las pruebas sobre varios protocolos como UDP o TCP. Iperf es una herramienta para medir el máximo ancho de banda sobre los protocolos TCP y UDP, permitiendo modificar y optimizar diferentes parámetros. Iperf reporta el BW, perdida de paquetes (para UDP).

Para las pruebas se implementó un escenario que simula un servicio de red Ethernet sobre TCP/IP para el transporte del tráfico; específicamente en una red LAN.

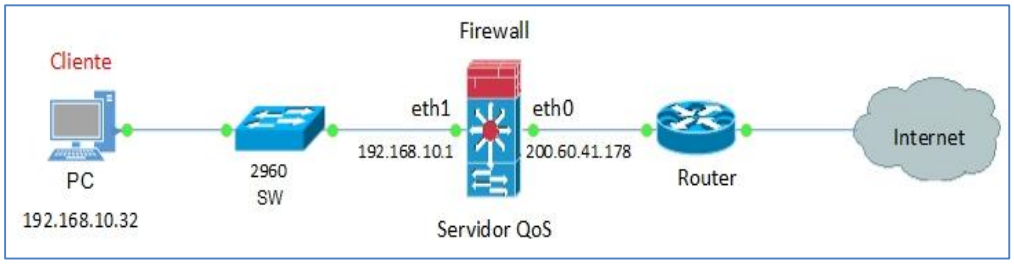


Grafico 43: Esquema de red para medir taza de pérdidas

Fuente: Elaboración propia.

En el grafico N° 43, se muestra el diagrama de conexión e interfaces utilizados para las pruebas. Debe notarse que la red simulada es un esquema de red, en el cual se aplica calidad de servicio en el servidor 1, para los clientes de la red LAN.

Se utilizó un esquema de Red de Área Local Ethernet para las pruebas propuestas y TCP/IP como sistema de transporte, una configuración ampliamente difundida a nivel de cliente-servidor.

Se utiliza un direccionamiento privado del rango de direcciones 192.168.10.0/24 para las terminales de los clientes. Este rango no tiene relación con el utilizado en la configuración de las direcciones IP de los Routers o Firewall utilizados.

Las pruebas se efectuaron bajo una plataforma Linux instaladas en el cliente y servidor. Dicha plataforma se basa en el sistema operativo Centos con kernel 2.6.32.

Iperf como Servidor: para el servidor usamos la opción -s.

Por lo tanto en el servidor ejecutamos el siguiente comando como se muestra la figura:

[root@server~]# iperf -s u -i 5

Se efectuó una prueba con Iperf bajo el esquema anterior, ejecutando Iperf en el cliente con dirección IP (192.168.10.32), hacia la pc que esta como servidor con dirección IP (192.168.10.1). El resultado obtenido se muestra en el grafico N° 44.

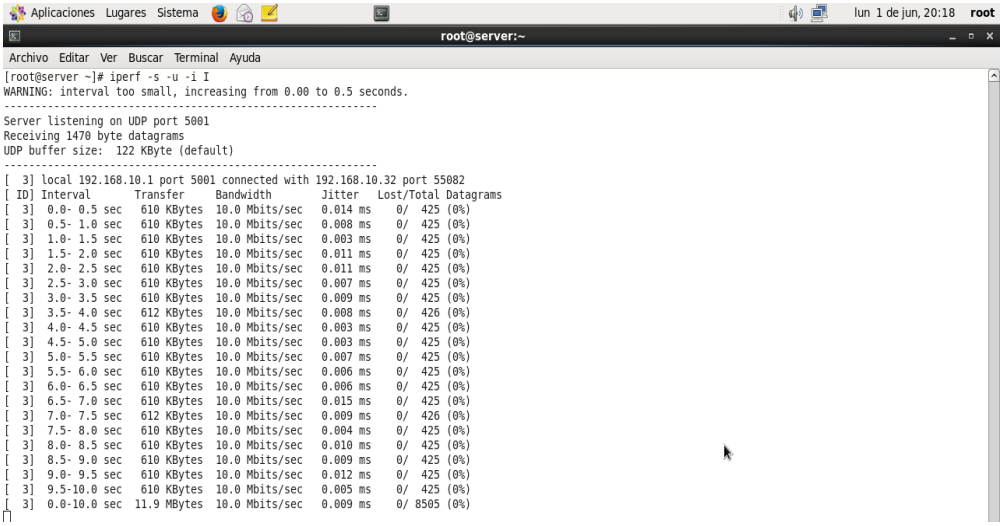


Grafico 44: Resultados Iperf como Servidor.

Fuente: Elaboración propia.

Las pruebas se realizó utilizando segmento UDP, para ello se utiliza el argumento -u, estas pruebas proporcionan información importante sobre la pérdida de paquetes.

Opciones:

- -s : modo servidor
- - i : intervalo de muestra de estadísticas cada 5 segundos por defecto.
- puertos: por defecto el servidor escucha en el puerto tcp 5001 y el cliente usa un puerto dinámico > 1024 como puerto origen.
- opción -u se indica que se utiliza segmentos UDP.

Iperf como Cliente: En el lado cliente se indica la IP del servidor con la opción - c.

Por lo tanto en el cliente que tiene la dirección IP (192.168.10.32), ejecutamos el siguiente comando como se muestra en el grafico N° 45:

[root@server~]# iperf -c 192.168.10.1 -u -b 10m

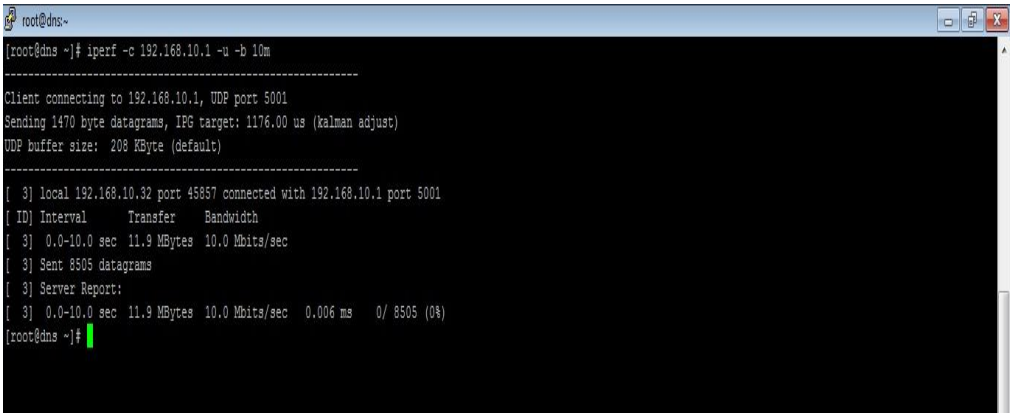


Grafico 45: Resultados Iperf como Cliente.

Fuente: Elaboración propia.

Opciones:

- -c 192.168.10.1 : en el cliente (que tiene IP 192.168.10.32) le decimos que se conecte al servidor con IP 192.168.10.1
- el tiempo total del test mediante la opción -t y el intervalo para ir viendo cómo va el test mediante la opción -i.

La orden anterior lanza una medida de prestaciones entre el cliente con ip 192.168.10.32 y la máquina servidor con dirección IP 192.168.10.1 (indicada con la opción -c). La medida de prestaciones es bidireccional y simétrica, pero no simultánea: primero se transmite en un sentido y luego en el contrario. Se va a usar UDP como protocolo de transporte (opción -u) y la velocidad a la que las fuentes de datos generan tráfico es de 10 Mbps (opción -b 10M). En la máquina remota debe haber otro programa iperf lanzado para recibir el tráfico (en el servidor). La salida que produce la orden anterior se muestra en la imagen anterior.

Obtenemos los siguientes resultados:

- Intervalo: 0.00 – 10.00 s
- Transferencia: 11.9 Mbytes
- Lost/Total: 0/8505
- **Datagramas: 0%**

- Para mantener una buena calidad en el enlace, la pérdida de paquetes no debe ser mayor a 1%. Una taza de pérdida de paquetes alta generará muchas retransmisiones de segmentos TCP/UDP, lo cual afectará el ancho de banda.

Indicador:

Porcentaje de paquetes perdidos= 0%

B. DISCUSIÓNES

A. Analizar el tráfico de la red LAN para implementar políticas de control de tráfico en la red LAN.

En el resultado 1 alineado al objetivo 1 se puede apreciar que con el análisis del tráfico de la red LAN se determina cuáles son los servicios y las aplicaciones de mayor demanda, los puertos más utilizados, el consumo de ancho de banda en la red LAN de la Universidad, en base a esto se implementara las políticas de control de tráfico de la red para mejorar la Calidad de Servicio (Quality of Service - QoS). Esto se ve corroborado con el análisis realizado a través de la herramienta Ntopng, arrojando diversos resultados con gráficos estadísticos. En el Grafico N° 23 y Grafico N° 24 sobre el protocolo HTTP, se obtiene que es el protocolo de mayor consumo del ancho de banda 86.2% del total disponible (4 Mb) de la red LAN. En base a estos resultados se crean las reglas para controlar el tráfico de la red LAN, otorgándole una prioridad media (Prioridad 2) y asignando un ancho de banda de 1800 Mb al protocolo HTTP.

B. Analizar el control de tráfico de la red mediante las herramientas IProute2 y Netfilter en la red LAN.

En el resultado 2 alineado al objetivo específico 2, se puede apreciar que teniendo configurados las herramientas Iproute2 e Iptables, se controla el tráfico de la red, este control de tráfico se realiza a través de la interfaz de red “eth0”, interfaz de salida hacia Internet, en esta interfaz de red se configura la “disciplina de cola” HTB (Hierarchical Token Bucket). Esto se ve corroborado en el Grafico N° 28, script donde se observa las clases creadas las cuales contienen información sobre el total de bytes enviados y paquetes transmitidos en tiempo real, el tráfico de red que pasa por cada una de las clases (1:10, 1:11, 1:12, 1:13) con sus respectivas prioridades y el ancho de banda asignado a cada clase (clase 1:10 - 600 Mb, clase 1:11 - 1800 Mb, clase 1:12 - 1200 Mb, clase 1:13 - 200 Mb), de acuerdo al tipo de tráfico. El tráfico generado en cada clase es de acuerdo a la marca que se le asigna a cada paquete a través de la herramienta Iptables..

Teniendo implementado el control de tráfico en la red LAN permite, el control del ancho de banda disponible (4Mb), asignando un determinado ancho de banda a cada clase creada, permite tener diferentes clases para diferentes tipos de tráfico, asignando a cada clase un ancho de banda garantizado. Cuando una clase solicita menos de la cantidad de ancho de banda asignada, el ancho de banda restante es distribuido a otras clases las cuales solicitan el servicio.

La herramienta Iptables permite marcar los paquetes para encolar el tráfico de la red en sus respectivas clases (Trafico SYN ACK DNS - marca 1, Trafico HTTP HTTPS - maca 2, trafico HTTP con destino al puerto 7776 - marca 3, Trafico desconocido (Unknow) - marca 4). Los paquetes irán a sus respectivas clases de acuerdo a las marcas que tienen (marca 1 ----> clase 1:10, marca 2 ----> clase 1:11, marca 3 ----> clase 1:12, marca 4 ----> clase 1:14), esto se corrobora en el script de Iptables Grafico N° 29.

C. Medición de la Calidad de Servicio (Quality of Service - QoS) en la red LAN.

En el resultado 3 alineado al objetivo específico 3 se puede apreciar que la medición de la Calidad de Servicio (Quality of Service - QoS) en la red LAN se ha realizado en base a tres indicadores: Ancho de Banda, Latencia y Tasa de pérdidas.

1. En la medición del indicador ancho de banda, se observa el comportamiento del tráfico HTTP cuando se tiene implementado el control de tráfico de la red en la red LAN. Con respecto a los resultados obtenidos cuando no se tiene aplicado el control de tráfico de la red, el consumo de ancho de banda (Caudal Maximo = 171.6 Kbps), vemos que ha mejorado el consumo de ancho de banda en el protocolo http. Esto se ve corroborado en el Grafico N° 37, medición realizada con la herramienta Ntopng, en la cual se observa un incremento de tráfico por parte del protocolo http, alcanzando un valor máximo de 328.0 kbps. Esta mejora se obtiene por haber realizado clasificación de tráfico, asignando un ancho de banda garantizado 1800 Mb del total disponible 4 Mb para este protocolo y por haberle asignado una prioridad alta (Prioridad 2) con respecto al resto de tráfico.

   - Para el tráfico http, https, se estableció una prioridad 2, lo que indica que este tipo de tráfico saldrá hacia internet mucho más rápido con respecto a los otros tipos de tráfico generado por las diferentes aplicaciones, de esta manera los usuarios tendrán una respuesta rápida al navegar por internet y al consultar las diferentes páginas web.

   - Con respecto a los resultados obtenidos cuando no se tiene aplicado el control de tráfico de la red, se observa que se tiene un menor consumo de ancho de banda (Caudal Maximo = 171.6 Kbps), por el protocolo http, esto se obtiene, debido a que los clientes o usuarios hacen uso del ancho de banda disponible en la red LAN al mismo tiempo. Este es el estado "normal" de la red, donde se tienen a todos los usuarios accediendo al único recurso disponible, ancho de banda total de 4 MB. Conllevando a no tener una distribución equitativa del ancho de banda disponible.

En la medición del indicador “Ancho de banda” en base al porcentaje de consumo de ancho de banda por parte del protocolo HTTP, los resultados obtenidos cuándo no se tiene aplicado el control de tráfico de la red se obtiene un consumo de ancho de banda de 14.6% (HTTP). Cuando se tiene implementado el control de tráfico de la red el consumo de ancho de banda por el protocolo HTTP es de 60.7%, por lo tanto ha mejorado el consumo de ancho de banda con respecto a cuándo no se tiene implementado el control de tráfico de la red. Esto se ve corroborado en el Grafico N° 40, se puede apreciar una mejora notable del consumo de ancho de banda por parte del protocolo HTTP (60.7 %); aumentando en un 46.1 % con respecto a los resultados sin aplicar control de tráfico a la red LAN de la Universidad.

En la medición del consumo de ancho de banda por los protocolos desconocidos (Unknown) bajo en un 39.1 %, esto se ve corroborado en el Grafico N° 40, medición realizada con la herramienta Ntopng. Este resultado se obtiene ya que se estableció una prioridad baja (Prioridad 4) y se estableció un ancho de banda 200 Mb determinado en la clase N° 4, para el tráfico de los 4 MB disponibles, por lo tanto este tipo de tráfico hacia esos puertos tiene un ancho de banda limitado.

2. En la medición del indicador “Latencia” Se observa que la latencia de la red tiene un promedio de 68 ms y un máximo de 78 ms cuando se tiene implementado el control de tráfico de la red con respecto a la latencia de la red sin control de tráfico en el cual se obtiene un promedio de 565.5 ms y un máximo de 571.1 ms. Esto se ve corroborado en la Tabla N° 6 y Tabla N° 4 respectivamente, medición realizada con la herramienta Ping. Producto de la implementación de control de tráfico de la red en Linux la latencia de la red está por debajo de los 150 ms, de los parámetros recomendados por la ITU-T G.114, teniendo un nivel aceptable para el funcionamiento adecuado de la red de datos.

3. En la medición del indicador “Tasa de Perdidas” se puede verificar que la perdida de paquetes (Taza de Perdidas) tiene un valor de 0%. Esto se ve corroborado en el Grafico N° 44, medición realizada con la herramienta Iperf. En tal sentido la Tasa de Perdidas está por debajo del 1%, parámetro recomendado por la ITU-T G.114. Lo que indica que existe una buena calidad de transmisión de datos en el enlace, esto debido a que se está aplicando control de tráfico en la interfaz de red “eth0” que corresponde a la salida de los datos por parte de los clientes, de esta manera se mejora Calidad de Servicio (QoS) en la red LAN.

V. CONCLUSIONES Y RECOMENDACIONES

1) CONCLUSIONES

A. Los resultados obtenidos al implementar control de tráfico de la red con Linux para mejorar la Calidad de Servicio (Quality of Service - QoS) en la red LAN de la Universidad, fueron satisfactorios, ya que en uno de los puntos aumento el consumo de ancho de banda del protocolo HTTP en un 46.1%. Se aplicaron conceptos básicos de control de tráfico en un sistema operativo Linux, posibilidades que brinda GNU/Linux como instrumento para la implementación de mecanismos de control de tráfico de la red para mejorar la Calidad de Servicio (Quality of Service - QoS), confirmándose la hipótesis de la investigación.

B. El análisis del tráfico de red LAN de la Universidad Privada, es necesario para poder identificar cuáles son los servicios y aplicaciones de mayor demanda, los puertos más utilizados, el consumo de ancho de banda. Con estos resultados a partir del análisis de los gráficos estadísticos obtenidos con la herramienta Ntopng, se aplicara las políticas de control de tráfico, a los diferentes tipos de tráfico transmitidos por los usuarios de la red LAN de la Universidad, estas políticas de control de tráfico se implementa en base a las necesidades del entorno de trabajo de la Universidad.

C. Las versiones del kernel de Linux, de la 2.4 en adelante cuentan con herramientas Netfilter/Iptables e Iproute2, que posibilitan la implementación de control del tráfico por medio de marcas en un firewall y gestionan el control de tráfico a través de tareas basadas en la gestión de colas. Dichas herramientas permiten definir clases, políticas y filtros a través de los cuales podemos manejar algoritmos de encolamiento, permitiéndonos priorizar tráfico, distribución equitativa de ancho de banda, Reserva de ancho de banda, para mejorar la calidad de servicio en una red LAN.

D. Se mejoró la Calidad de Servicio (Quality of Service - QoS) en la red LAN, resultados satisfactorios obtenidos en base a los parámetros recomendados por la ITU-T en su apartado G.114 (Ancho de Banda, Latencia y Tasa de pérdidas), mejorando el consumo del ancho por parte de las aplicaciones y protocolos específicamente en la utilización del protocolo http en un 46.1%. En la medición de la latencia de la red se obtiene que está por debajo de los 150 ms teniendo un nivel aceptable para el funcionamiento adecuado de la red de datos y la Tasa de Perdidas en la red LAN está por debajo del 1% lo que indica que existe una buena calidad de transmisión de datos en el enlace de la red.

2) RECOMENDACIONES

A. El análisis de tráfico de la red LAN, permite implementar las políticas de calidad de servicio, por lo tanto es muy importante que este análisis se realice de una manera más detallada, por lo que se recomienda hacer uso de software especializados en el análisis de los datos para el tráfico de la red, que permitan obtener gráficos y resultados estadísticos más detallados de perfiles de tráfico.

B. Si la propuesta se va a aplicar a redes con gran cantidad de usuarios, se haría muy difícil mantener los scripts de configuración, los cuales crecen en tamaño y especifican una estructura de árbol compleja (por ejemplo: la numeración de los nodos del mismo). Por otra parte, el uso de filtros para poder establecer prioridades sobre cada tipo de traficó tendrían una mayor complejidad. Se dificulta entonces la configuracion de estos scripts. Teniendo en cuenta todas estas consideraciones, se recomienda utilizar la herramienta denominada tcng (TC Next Generation), escrita por Werner Almesberger, el cual contiene una combinación de scripts ya configurados, para lograr la automatización y la configuraciones necesarias para el control de traficó propuesto. Al utilizar tcng, podemos escribir los mismos scripts hechos con el comando tc.

C. Cuando se implemente el control de tráfico de la red LAN en el equipo GNU/Linux se debe configurar de forma apropiada la disciplina de colas. Específicamente la configuración se debe realizar en la interfaz de salida hacia el router del operador de Internet (ISP), en la interfaz de red (“eth0”) por el que sale a Internet los usuarios de la red LAN.

D. La presente tesis demostró la implementación de un controlador de borde en la interfaz de salida hacia Internet, el sistema operativo Linux posee todas las capacidades para implementar completamente QoS extremo a extremo. La propuesta desarrollada no implementa un modelo de QoS extremo a extremo, pudiendo, en caso de ser necesario extender sus funcionalidades dado que el Sistema Operativo Linux, soporta los estándares de DiffServ e IntServ. En la propuesta presentada solo podemos controlar lo que enviamos a internet, pero no lo que nos llega de internet, por lo tanto se recomienda complementar la propuesta con el modelo para poder controlar el tráfico entrante desde internet y así tener un control más completo tanto en el tráfico de salida como en el tráfico de entrada.

REFERENCIAS BIBLIOGRÁFICAS

Almesberguer, W. (11 de Marzo de 2013). *Linux Network Traffic Control .* Obtenido de Linux Network Traffic Control : http://www.almesberger.net/cv/papers/tcio8.pdf

Braden .R, C. .. (Junio de 1994). *Integrated Services in the Internet Architecture:*. Obtenido de Integrated Services in the Internet Architecture:: http://www.ietf.org/rfc/rfc1633.txt

Braden, R. E., Zhang, L., Berson, S., Hersog, S., & S, J. (Septiembre de 1997). *Resource ReSerVation Protocol (RSVP).* Obtenido de http://www.ietf.org/rfc/rfc2205.txt.pdf

Brown, M. A. (2006). *Traffic Control HOWTO*. Obtenido de http://linux-ip.net/articles/Traffic-Control-HOWTO/index.html

Bustamante Alvarez, R. (2007). *Contribución en el Análisis y Simulación de una red MPLS con la Internet de Servicios Diferenciados Diffserv.* Lima.

Caballero Romero, A. (2000). *Metodologìa de la Investigación Cientìfica. Diseños con Hipótesis Explicativas.* Lima: Udegraf.

Christian, B. (2006). *Understanding Linux Network Internals.* O'Reilly Median, Inc.

Cisco Systems, I. (Junio de 1999). *Internetworking Technology Overview.* Obtenido de http://www-2.dc.uba.ar/materias/tc/downloads/apuntes/qos.pdf

Dugan, J., Elliott, S., Mah, B. A., Poskanzer, J., & Prabhu, K. (2009). *iperf.fr*. Obtenido de https://iperf.fr/

Espinosa Ceniceros, J. C. (2010). *Parámetros de medición en QoS*. Obtenido de http://juankenny.blogspot.pe/2013/02/lab-rt-parametros-de-medicion-en-qos.html

Fuster Monzó, J. (2004). *Filtrado de paquetes y QoS.* Obtenido de http://www.redes-linux.com/manuales/ancho_banda/filter_qos.pdf

Geoff, H. (Julio de 2001). *Best Efforts Networking.* Obtenido de Best Efforts Networking: http://www.potaroo.net/ispcol/2001-09/2001-09-best.pdf

Gramajo, S. (2012). *Modelos de decision Linguistica para la QoS en Networking.* Malaga.

Hernandez Sampieri, R., Fernandez Collado, C., & Baptista Lucio, M. d. (2010). *Metodología de la Investigación. (5.a Edición).* Mexico: McGRAW-HILL / INTERAMERICANA EDITORES, S.A.

Hubert, B., Maxwell, G., Thomas, G., Van Mook, R., Van Oosterhout, M., Schroeder, P. B., . . . Cárdenes, R. J. (17 de Diciembre de 2004). *Linux Advanced Routing & Traffic Control HOWTO.* Obtenido de Linux Advanced Routing & Traffic Control HOWTO: http://www.lartc.org/lartc.pdf

Inzunza Sanhueza, D. H., & Marileo Ñancupil, O. S. (2010). *Filtrado de paquetes ip a través de expresiones regulares sobre capa de aplicacion y gestion del ancho de banda con software libre.* Temuco Chile.

Kerlinger, F. N. (1975). *Investigación del comportamiento. Técnicas Y Metodología.* México, D.F: Nueva Editorial Interamericana.

Kuznetsov, A. N. (14 de Abril de 1999). *IP Command Reference.* Obtenido de http://www.linuxfoundation.org/collaborate/workgroups/networking/iproute2

Luca, D. (2015). *ntopng High-speed web-based traffic analysis*. Obtenido de http://www.ntop.org/

Luna Victoria García, J. I. (2011). *Medición y Análisis de Tráfico En Redes MPLS.* Lima.

Navarro, D. F. (2005). *Controlador de Ancho de Banda.* Mendosa Argentina.

Nichols .K, B. S. (Diciembre de 1998). *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*. Obtenido de Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers: http://www.ietf.org/rfc/rfc2474.txt.pdf

Russell, R. (24 de Enero de 2002). *Netfilter.* Obtenido de Netfilter: http://www.netfilter.org/documentation/HOWTO//netfilter-hacking-HOWTO.html

Saravanan, R. (22 de Agosto de 1999). *Linux - Advanced Networking Overview.* Obtenido de http://qos.ittc.ku.edu/howto.pdf

Silva Jiménez, M. M. (2008). *Estudio e Implementacion de QoS mediante las Herramientas Iproute2 y Netfilter de Linux.* Mexico.

Snader, J. C. (2000). *Effective TCP/IP Programming: 44 Tips to Improve Your Network Programs.* ADDISON-WESLEY.

Systems, A. d. (2004). *CCNA 1 y 2, tercera edición.* Pearson Educación.

Tanenbaum, A. S. (2003). *Redes de Computadoras.* Mexico: Pearson Educación de México S.A.

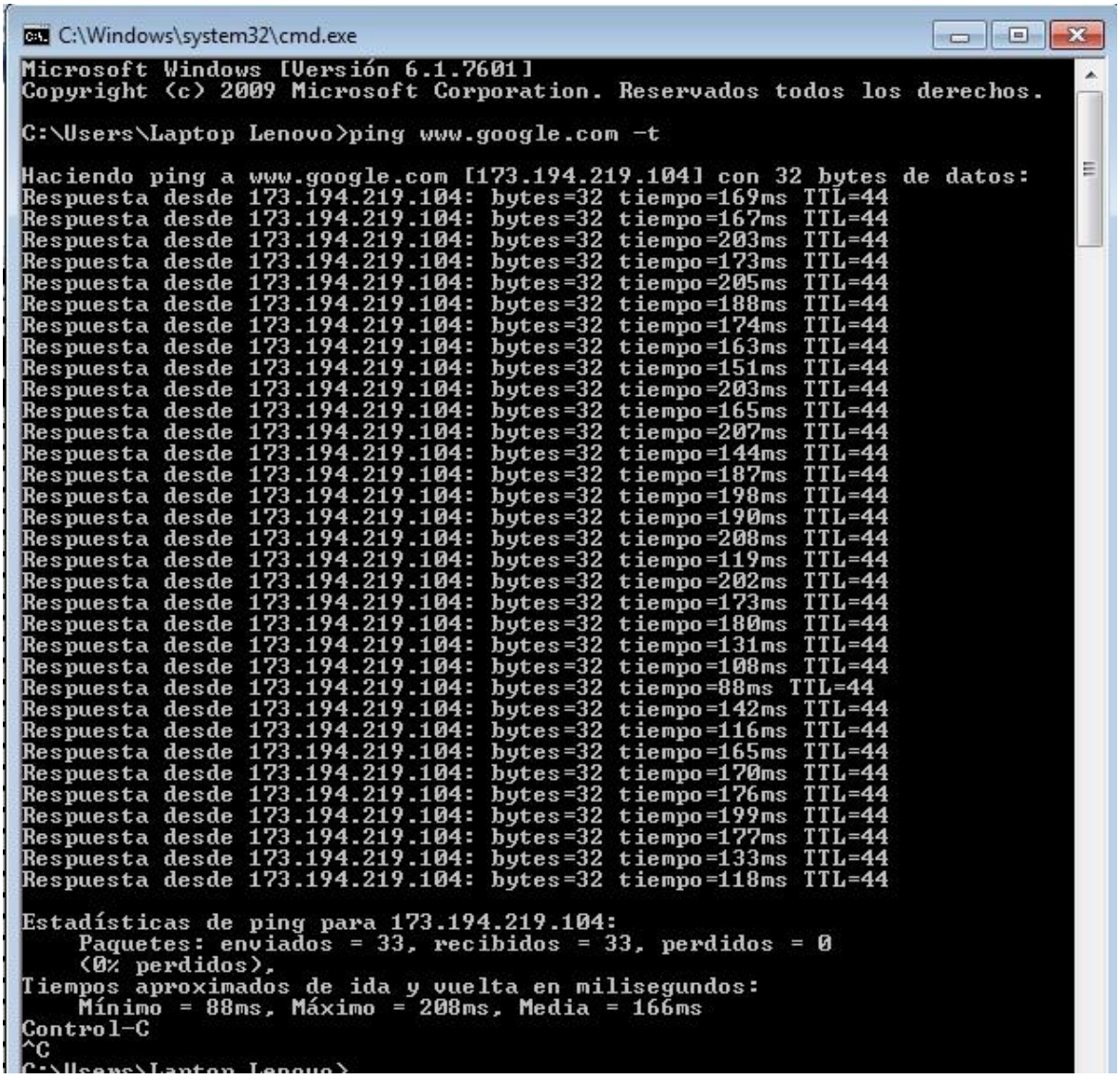
UNIÓN INTERNACIONAL DE TELECOMUNICACIONES. (Noviembre de 2001). *Categorías de calidad de servicio para los usuarios de extremo de servicios multimedios.* Obtenido de https://www.itu.int/rec/T-REC-G.1010-200111-I/

Vargas Piedra, F. A. (2011). *Análisis y comparación de métodos de medición del desempeño de redes de computadores Ethernet y Metro Ethernet.* Costa Rica.

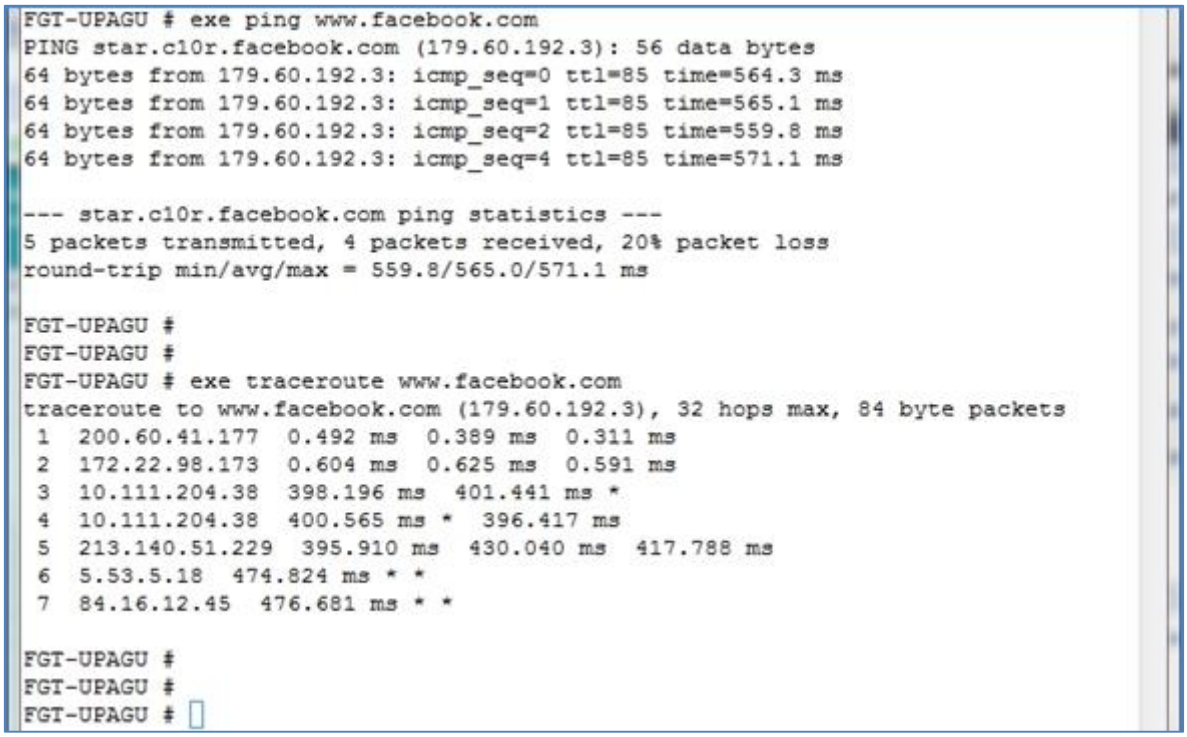
Welte, H., & Neira Ayuso, P. (1999). *NETFILTER*. Obtenido de http://www.netfilter.org/documentation/HOWTO/es/

ANEXOS

Anexo 1: Pruebas con herramienta Ping.

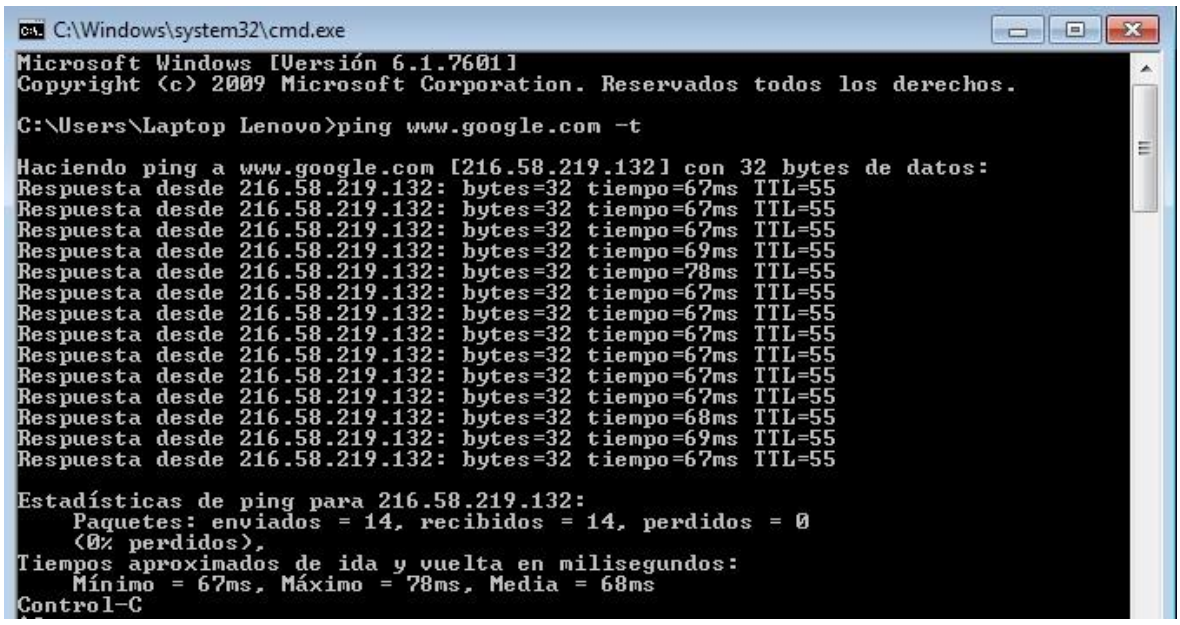


Anexo 2:



```
FGT-UPAGU # exe ping www.facebook.com
PING star.c10r.facebook.com (179.60.192.3): 56 data bytes
64 bytes from 179.60.192.3: icmp_seq=0 ttl=85 time=564.3 ms
64 bytes from 179.60.192.3: icmp_seq=1 ttl=85 time=565.1 ms
64 bytes from 179.60.192.3: icmp_seq=2 ttl=85 time=559.8 ms
64 bytes from 179.60.192.3: icmp_seq=4 ttl=85 time=571.1 ms

--- star.c10r.facebook.com ping statistics ---
5 packets transmitted, 4 packets received, 20% packet loss
round-trip min/avg/max = 559.8/565.0/571.1 ms

FGT-UPAGU #
FGT-UPAGU #
FGT-UPAGU # exe traceroute www.facebook.com
traceroute to www.facebook.com (179.60.192.3), 32 hops max, 84 byte packets
 1  200.60.41.177  0.492 ms  0.389 ms  0.311 ms
 2  172.22.98.173  0.604 ms  0.625 ms  0.591 ms
 3  10.111.204.38  398.196 ms  401.441 ms *
 4  10.111.204.38  400.565 ms *  396.417 ms
 5  213.140.51.229  395.910 ms  430.040 ms  417.788 ms
 6  5.53.5.18  474.824 ms * *
 7  84.16.12.45  476.681 ms * *

FGT-UPAGU #
FGT-UPAGU #
FGT-UPAGU #
```

Anexo 3:



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\Laptop Lenovo>ping www.google.com -t

Haciendo ping a www.google.com [216.58.219.132] con 32 bytes de datos:
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=69ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=78ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=68ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=69ms TTL=55
Respuesta desde 216.58.219.132: bytes=32 tiempo=67ms TTL=55

Estadísticas de ping para 216.58.219.132:
    Paquetes: enviados = 14, recibidos = 14, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 67ms, Máximo = 78ms, Media = 68ms
Control-C
```

Anexo 4:

```
SOC_UPAGU $ exe ping www.facebook.com
PING star.c10r.facebook.com (31.13.73.1): 56 data bytes
64 bytes from 31.13.73.1: icmp_seq=0 ttl=87 time=88.9 ms
64 bytes from 31.13.73.1: icmp_seq=1 ttl=87 time=67.6 ms
64 bytes from 31.13.73.1: icmp_seq=2 ttl=87 time=66.8 ms
64 bytes from 31.13.73.1: icmp_seq=3 ttl=87 time=66.8 ms
64 bytes from 31.13.73.1: icmp_seq=4 ttl=87 time=66.6 ms

--- star.c10r.facebook.com ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 66.6/71.3/88.9 ms

SOC_UPAGU $ exe ping www.facebook.com
PING star.c10r.facebook.com (31.13.73.1): 56 data bytes
64 bytes from 31.13.73.1: icmp_seq=0 ttl=87 time=70.1 ms
64 bytes from 31.13.73.1: icmp_seq=1 ttl=87 time=67.6 ms
64 bytes from 31.13.73.1: icmp_seq=2 ttl=87 time=69.4 ms
64 bytes from 31.13.73.1: icmp_seq=3 ttl=87 time=66.7 ms
64 bytes from 31.13.73.1: icmp_seq=4 ttl=87 time=78.9 ms

--- star.c10r.facebook.com ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 66.7/70.5/78.9 ms

SOC_UPAGU $
SOC_UPAGU $
SOC_UPAGU $
SOC_UPAGU $ exe traceroute www.facebook.com
traceroute to www.facebook.com (31.13.73.1), 32 hops max, 84 byte packets
 1  200.60.41.177  0.418 ms  0.415 ms  0.311 ms
 2  172.22.98.173  0.617 ms  5.821 ms  5.962 ms
 3  10.111.204.38  16.038 ms  43.580 ms  42.026 ms
 4  10.111.204.38  47.480 ms  16.998 ms  24.443 ms
 5  213.140.51.229  29.975 ms  36.419 ms  40.513 ms
 6  5.53.5.6  88.024 ms  85.488 ms  93.051 ms
 7  * * *
 8  84.16.12.238  67.814 ms  83.709 ms  74.005 ms
 9  5.53.0.121  188.607 ms  66.590 ms  66.800 ms
10  74.119.79.181 <po102.psw01b.mia1.tfbnw.net>  66.691 ms  66.777 ms  67.173 ms
11  173.252.65.139 <msw1ao.01.mia1.tfbnw.net>  68.019 ms  70.390 ms  85.545 ms
12  31.13.73.1 <www.facebook.com>  76.514 ms  67.350 ms  70.036 ms

SOC_UPAGU $
```

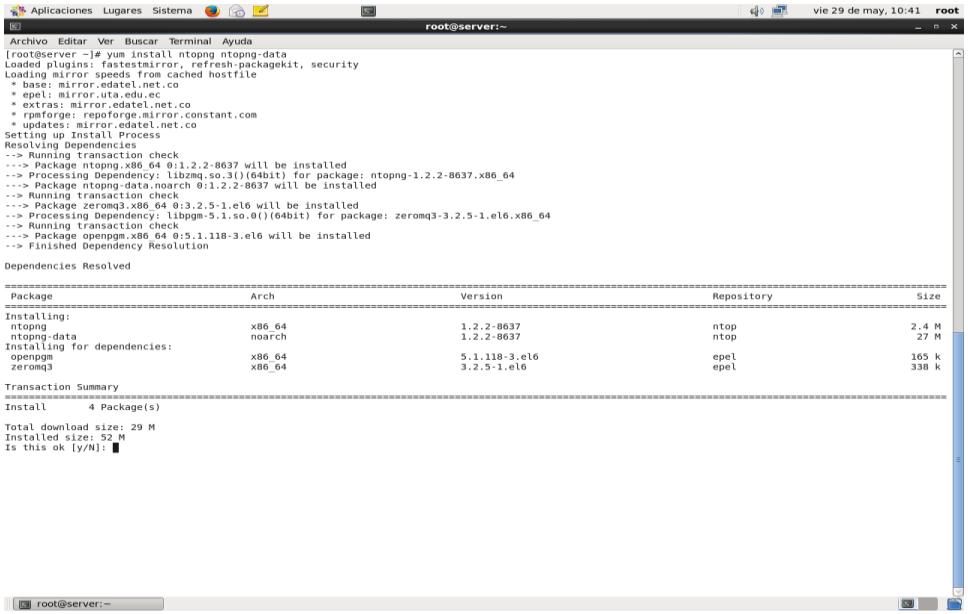
Anexo 5: Script Integral de Iptables (iptables.sh)

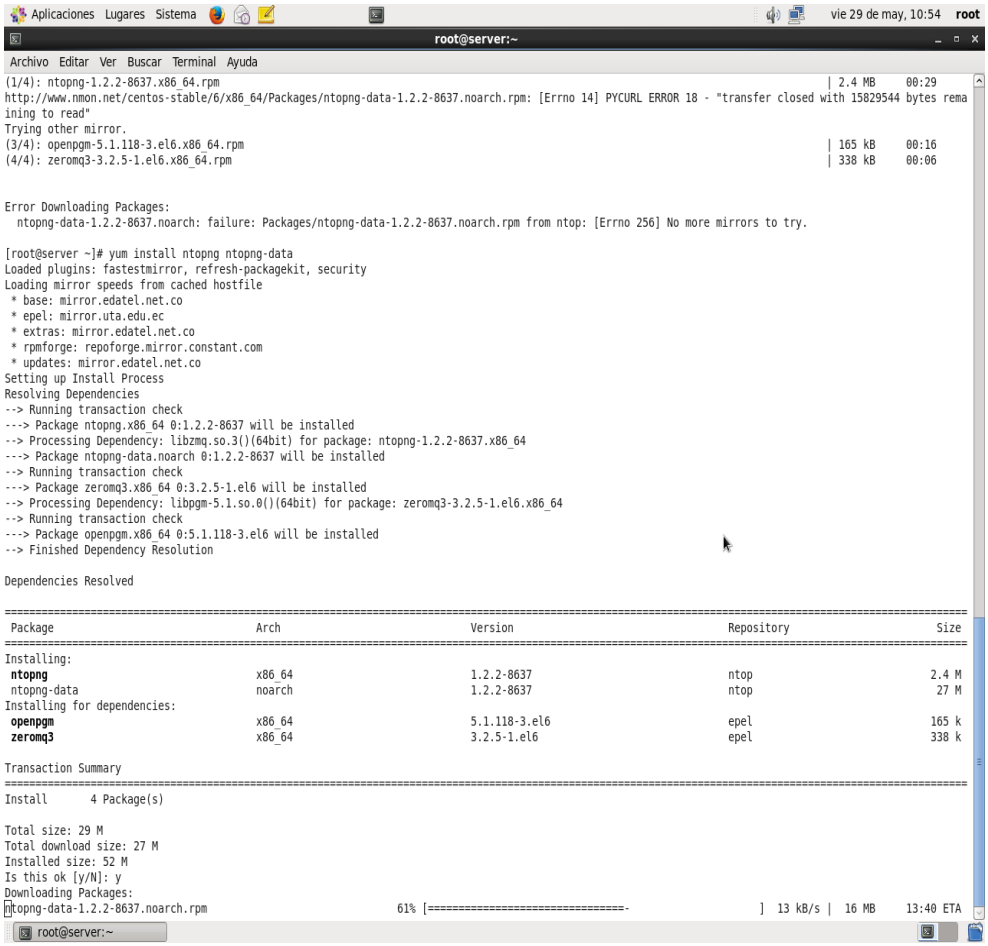
```
#!/bin/bash
##--------------- Script de Configuracion de  IPTABLES -----------
## Juan Carlos Cabanillas Chavez
##
#Squid Server IP
SQUID_SERVER="200.60.41.178"
#Interfaz conectada a Internet
INTERNET="eth0"
#Interfaz Conectada a la red Lan
LAN_IN="eth1"
#Puerto Squid
SQUID_PORT="3128"

#Limpiando todas las Reglas antiguas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

## Establecemos politicas por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

##-----TABLA FILTER-------
##Reglas para la Cadena INPUT
# Aceptamos el trafico de entrada y salida por la loopback
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
# Aceptamos las conexiones pendientes(RELATED) o ya Establecidas
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

#aceptamos las conexiones establecidas desde el firewall
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

#DHCP
iptables -A INPUT -m state --state NEW -m udp -p udp --dport 67 -j
ACCEPT
iptables -A INPUT -m state --state NEW -m udp -p udp --dport 68 -j
ACCEPT

#SSH
iptables -A INPUT -p tcp -i eth0 --dport 22 -j ACCEPT
iptables -A INPUT -p udp -i eth0 --dport 22 -j ACCEPT



##----TABLA NAT---------
```

```
#Redireccionamos al Proxy-Transparente con Squid
iptables -t nat -A PREROUTING -i $LAN_IN -p tcp -s
192.168.10.0/255.255.255.0 --dport 80 -j REDIRECT --to-port 3128

#Cambiamos la direccion IP-origen por la IP-publica para poder
salir a Internet----(SNAT)
iptables -t nat -A POSTROUTING -s 192.168.10.0/255.255.255.0 -o
eth0 -j SNAT --to 200.60.41.178
##
###iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 7776 -j
DNAT --to 192.168.10.116:7776 ###esta regla cumple la misma
funcion que la de abajo
####-----iptables -t nat -A PREROUTING -i $LAN_IN -p tcp -s
192.168.10.0/255.255.255.0 --dport 7776 -j DNAT --to
192.168.10.116:7776

# redireccionames las peticiones hacia el servidor de aplicaciones
iptables -t nat -A PREROUTING -i eth0 -p tcp -d 200.60.41.178 --
dport 7778 -j DNAT --to 192.168.10.116:7776


####################################
#Marcando los paquetes con Iptables!!!

# TCP con el bit SYN activado (principio de conexion)
iptables -t mangle -A POSTROUTING -p tcp -m tcp --tcp-flags
SYN,RST,ACK SYN -j MARK --set-mark 1
iptables -t mangle -A POSTROUTING -p tcp -m tcp --tcp-flags
SYN,RST,ACK SYN -j RETURN

#DNS
iptables -t mangle -A PREROUTING -p udp --dport 53 -j MARK --set-
mark 1
iptables -t mangle -A PREROUTING -p udp --dport 53 -j RETURN

iptables -t mangle -A OUTPUT -p udp --dport 53 -j MARK --set-mark
1
iptables -t mangle -A OUTPUT -p udp --dport 53 -j RETURN

# SSH
iptables -t mangle -A PREROUTING -p tcp --dport 22 -j MARK --set-
mark 1
iptables -t mangle -A PREROUTING -p tcp --dport 22 -j RETURN

iptables -t mangle -A OUTPUT -p tcp --dport 22 -j MARK --set-mark
1
iptables -t mangle -A OUTPUT -p tcp --dport 22 -j RETURN

# ICMP
iptables -t mangle -A PREROUTING -p icmp -j MARK --set-mark 1
iptables -t mangle -A PREROUTING -p icmp -j RETURN
```

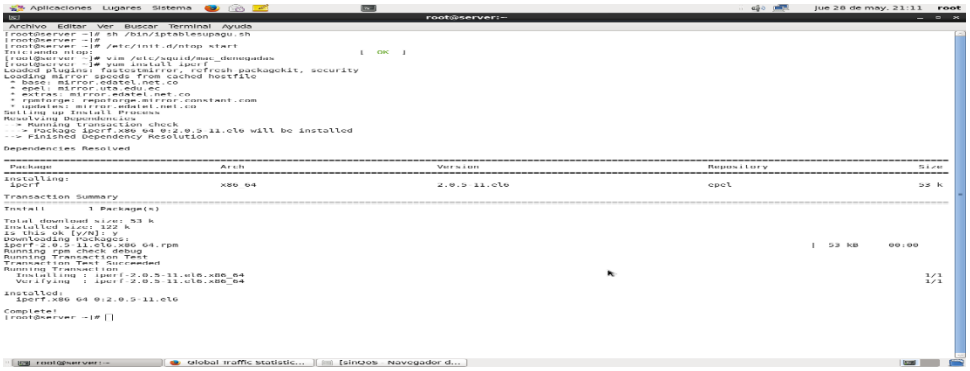
```
iptables -t mangle -A OUTPUT -p icmp -j MARK --set-mark 1
iptables -t mangle -A OUTPUT -p icmp -j RETURN

# HTTP, HTTPS
iptables -t mangle -A POSTROUTING -o eth0 -p tcp --dport 80 -j
MARK --set-mark 2
iptables -t mangle -A POSTROUTING -o eth0 -p tcp --dport 80 -j
RETURN

iptables -t mangle -A POSTROUTING -p tcp --dport 443 -j MARK --
set-mark 2
iptables -t mangle -A POSTROUTING -p tcp --dport 443 -j RETURN

# SMTP
iptables -t mangle -A POSTROUTING -p tcp --dport 25 -j MARK --set-
mark 2
iptables -t mangle -A POSTROUTING -p tcp --dport 25 -j RETURN

#POP3
iptables -t mangle -A POSTROUTING -p tcp --dport 110 -j MARK --
set-mark 2
iptables -t mangle -A POSTROUTING -p tcp --dport 110 -j RETURN

# SERVIDORES INTERNOS

iptables -t mangle -A PREROUTING -p tcp --sport 7776 -j MARK --
set-mark 3
iptables -t mangle -A PREROUTING -p tcp --sport 7776 -j RETURN

iptables -t mangle -A OUTPUT -p tcp --sport 7776 -j MARK --set-
mark 3
iptables -t mangle -A OUTPUT -p tcp --sport 7776 -j RETURN


#TrÃ¡co en general (emule....)
iptables -t mangle -A PREROUTING -j MARK --set-mark 4
iptables -t mangle -A PREROUTING -j RETURN
```

Anexo 6: Proceso de Instalacion de Ntopng.



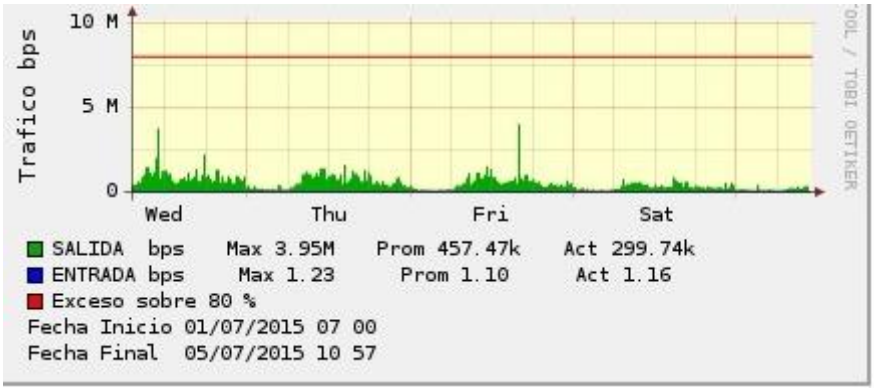


Anexo 7: Instalación de Iperf



Anexo 8: Trafico medido

- Trafico medido desde el 01 de Julio hasta el 05 de Julio 2015



- Trafico medido desde el 19 de Octubre hasta el 21 de Octubre 2015