UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

ESCUELA DE POSTGRADO

UNIDAD DE POST GRADO DE LA FACULTAD DE INGENIERIA ELECTRÓNICA

CONTRIBUCIÓN EN EL ANÁLISIS Y SIMULACIÓN DE UNA RED MPLS CON LA INTERNET DE SERVICIOS DIFERENCIADOS DIFFSERV

TESIS

PARA OPTAR EL GRADO DE MAGÍSTER EN TELECOMUNICACIONES

PRESENTADO POR

RAFAEL BUSTAMANTE ALVAREZ

LIMA – PERU

2007

Con cariño y gratitud a mis padres Uriel y Luz, a mis hermanas Ada y Lisbeth, a mi esposa Rossina y mi hija Andrea, quienes con su apoyo constante han hecho realidad mi objetivo de ser Magíster.

Rafael

AGRADECIMIENTOS

Para desarrollar un trabajo de investigación, son muy variados los recursos a los que debemos acceder y lograr resultados aceptables. Empezando por los conocimientos adquiridos, por las ideas precursoras del trabajo, pasando por la recopilación de la información y llegando a los medios físicos que posibilitan la realización práctica y posterior documentación del trabajo por lo cual debo hacer llegar el siguiente agradecimiento:

A Dios quien me dio fuerzas para poder seguir adelante con la presente tesis.

A la Universidad Nacional Mayor de San Marcos; mi ALMA MATER por haberme cobijado en sus aulas para seguir la noble carrera de Ingeniería Electrónica.

A mis Asesores Ing. Victor Cruz Ornetta, coasesores Ing. Daniel Díaz, y Msc. José Luis Muñoz meza, así como a todas las personas que me apoyaron en la elaboración de esta tesis.

INDICE

INDICE DE FIGURAS

GLOSARIO

6Bone	**Red IPv6 de carácter experimental creada para ayudar a los vendedores y usuarios**
3G	**Tercera Generación de Telefonia Celular**
ACL	**Control de Acceso a Listas**
AF	**Assured forwarding**
ATM	**Modo de Transferencia Asincrona**
Bandwidth	**Ancho de Banda**
Best Effort	**Nivel de servicio de Calidad de Servicio denominado el mejor esguerzo para la**
Buffers	**Dispositivo de almacenamiento de datos**
CAR	**Vlociada de Transmision entregada**
CBWFQ	**Class-based weighted fair queueing (CBWFQ**
Check Sum	**Parametro que calcula suma de datos de un mensaje o programa.**
CQ	**Custom Queuing**
CR-LDP	**Constraint-based Routing Label Distribution Protocol**
Destination Direction	**dirección de destino**
DiffServ	**Servicios diferenciados**
DS	**Difference service parametro**
DSCP	**Diffserv code oint parametro de la cabecera Ipv6**
EF	**Expedited forwarding**
EXP	**Campo expeimental del la etiqueta del protocolo mpls**
FEC	**Forwarding Equivalent Class**
FIB	**Forwarding information base es ub componente de un lsr**
FIFO	**First input first output forma de realizar colas en los nodos**
Frame Relay	**Tecnologia de conmutación de paquetes de tamaño variable**
FTP	**File Transfer Protocol**
GMPLS	**MPLS Genaralizado**
Hard State	**protocolos de señalizacion basados en TCP**
HLEN	**Longitu de la cabecera parametro de Ipv4**
HTTP	**El protocolo de transferencia de hipertexto**
IETF	**Internet Engineering Task Force, en castellano Grupo**
IGP	**Interior Gateway Protocol (IGP**
Interleaving	**Ordenamamiento no contiguo de data**

Internet	**Red mundial de computadoras conectadas usando protocolos como TCP/IP**
Internet2	**Red integrada por univesidades de EEUU para desarrollar tecnologias como IpV6**
IP	**Protocolo de Internet**
IP Precedence	**Priodidad de servicios**
IpV4	**Protocolo IP version 4**
IpV6	**Protocolo IP version 6**
IS-IS	**Intermediate system to intermediate system**
BGP	**Border Gateway Protocol**
LDP	**Protocolo de distribución de etiquetas**
Leaky Bucket	**Conformador de tráfico diferentes flujos a diferentes velocidades son uniformizadas**
L-LSP	**Label LSP que permite multiples LSPs según níveles de servicio**
LSR	**Label Switch Router**
MF	**Multifier Calssifier**
MIB	**Management Information Base**
MPLS	**Multiprotocol Label Switch**
NHLFE	**Next Hop Label Forwarding**
OSPF	**Open Shortest Path First**
Path Err	**Mensaje de error de RSVP para un enlace de origen a destino.**
PHB	**Per hop behavior**
PQ	**Priority queuing**
QoS	**Quality of Service**
Realibility	**Confiabilidad**
RED	**Random Early detection**
Resv Conf	**Mensaje de confirmación en RSVP**
Resv Err	**Mensaje de error de RSVP para un enlace de destino a origen.**
RFC	**Request for Comments**
RIP	**Routing Information Protocol**
RMON	**Monitoreo remoto**
RSPEC	**Especificaciones de reserva en RSVP**
RSVP	**Protocolo de reserva de recursos**
RTP-HC	**Real time protocol header cpmpression**
Soft State	**Se refiere al uso del protocolo IP para la señalización**
Source Direction	**direccion de origen**
TCP	**Transmission Control protocol**
TELNET	**Telecommunications Networking**
Throughput	**Rendimiento**
Token Bucket	**Mecanismo conformador de trafico permitiendo a la salida rafagas de trafico.**
ToS	**Type of service**

TSPEC	**Especificaciones de trafico en RSVP**
TTL	**Time to Live**
UDP	**User datagram Protocol**
URL	**Uniform Remote Localizer**
VER	**Version**
WAP	**Wireless access protocol**
WFQ	**Weighted Fair queing**
WRED	**Weighted random early detection**
WWW	**world wide web**

TABLAS

ABSTRAC

The advent of Internet and the globalization have modified the forms of communication in the world and the enterprises. All this has brought the search of facilities of communications permanents and secures with quality of service to improve the Internet. The studies of the news networks with MPLS are in a process that consists in finding news focus to improve the quality of service and rerouting. This thesis consists in analyzing and simulates MPLS with DiffServ to demonstrate that the use of the L-LSP is a good solution to improve the throughput in the network. The other side, it proposes a new method for updating time to rerouting in the network uses OSPF. Finally, it does an analysis and simulation of the network behavior, in case of the fail a link. It is to say the resilence of the network.

CAPITULO 1 Introducción y Objetivos

1.1 INTRODUCCIÓN

La Internet a desplazado a las tradicionales redes de datos y ha llegado a ser el modelo de red pública del presente. Pero si bien es cierto que la Internet ha llegado a consolidarse como el modelo de red pública de datos a gran escala; también lo es que no llega ha satisfacer ahora, todos los requisitos de los usuarios, principalmente de aquellos de entornos corporativos, que necesitan de la red para aplicaciones críticas tales como videoconferencias, telefonía IP entre otros. Una carencia fundamental de la Internet es la imposibilidad de seleccionar diferentes tipos de servicio para los diferentes aplicaciones de usuario es decir no existe diferenciación de servicios. La Internet se valora mas por el servicio de acceso y distribución de contenidos conocido como "best-effort". Si el Modelo de Internet ha de consolidarse como el modelo de la red de datos del siglo XXI entonces es necesario un modelo de RED como MPLS brindando servicios diferenciados.

El crecimiento permanente de la Internet así como sus aplicaciones y la búsqueda de una mayor calidad de servicio (QoS, Quality of Service) a permitido el desarrollo de nuevas arquitecturas como MPLS (Multiprotocol Label Switching), el cual se considera fundamental en la construcción de los nuevos cimientos para el desarrollo de la Internet actual y de la siguiente generación, el cual esta diseñado para poder dar servicios diferenciados según el modelo DIFFSERV del IETF, este modelo define mecanismos para poder clasificar el tráfico en un reducido número de clases de servicios, con diferentes prioridades. Según los requisitos de los usuarios, DIFFSERV permite diferenciar servicios tradicionales, tales como WWW, el correo electrónico, o

transferencia de archivos para los cuales el retardo no es crítico, de otras aplicaciones donde el retardo es importante como son la transmisión de video y voz interactiva. La arquitectura DIFFSERV se encuentra fuertemente ligada al protocolo IP versión 6 (IPV6) que se constituye como la siguiente generación de la INTERNET.

Sin embargo, a pesar de la implantación sistemática de MPLS en las grandes redes, todavía hay aspectos en el que se puede contribuir para su desarrollo, tal es el caso del campo EXP de la cabecera genérica de MPLS, mecanismos de enrutamiento y re-enrutamiento.

En esta tesis se realiza; un estudio de las nuevas arquitecturas como IntServ, DiffServ, MPLS, un análisis y simulación el comportamiento de una red MPLS con Servicios Diferenciados con L-LSPs con múltiples rutas en una red. También se propone un algoritmo predictivo para la determinación del ancho de banda disponible para cada enlace en cada nodo de una red MPLS-DiffServ y establecer la mejor ruta de extremo a extremo, durante el periodo de actualización de datos de ancho de banda disponible usando el protocolo OSPF lo que permite mejorar el re-enrutamiento y finalmente se realiza un análisis de la red MPLS-DiffServ para el re-enrutamiento en caso de pérdida de un enlace.

1.2 MOTIVACIÓN

La búsqueda de medios para dar una mejor calidad de servicios en las grandes redes conllevó a la aparición de nuevas arquitecturas como MPLS y DiffServ, siendo esta una búsqueda permanente, lo cual motiva su entendimiento mediante el estudio de estas nuevas arquitecturas y la necesidad de realizar nuevas propuestas, así como el análisis de la arquitectura MPLS con servicios diferenciados, ya que la arquitectura MPLS se considera fundamental en la construcción de las bases de la Internet del XXI.
Para el estudio de la Internet es necesario hacerlo realizando un análisis de los protocolos IPv4 e IPv6 asociado con las nuevas arquitecturas como MPLS, DiffServ e IntServ.

El uso del simulador ns basado en software libre como una herramienta de investigación; permite estudiar y analizar arquitecturas como MPLS y DiffServ,

además de otras aplicaciones de redes de datos. Finalmente, contribuir con su difusión en nuestro medio, mediante este trabajo de investigación

1.3 PROPUESTAS

- El entendimiento de las actuales arquitecturas de la Internet es clave para poder comprender el desarrollo y funcionamiento de la futura Internet con Calidad de Servicio, en este trabajo se analiza la arquitectura MPLS con DiffServ como una solución para proveer de Calidad de Servicio (QoS) a la Internet, para poder llevar a cabo este análisis se realizan análisis y simulaciones de MPLS con DiffServ, sobre la base de los L-LSPs, es decir, múltiples rutas, como una alternativa para mejorar la calidad de servicio en la red.

- La posibilidad de obtener un ancho de banda disponible más óptimo para la creación de un LSP en una red MPLS con DiffServ durante el tiempo de actualización con el protocolo OSPF, nos permite realizar una propuesta basado en conceptos de filtros adaptativos.

- Se propone un análisis de los procedimientos del re-enrutamiento basado en las técnicas propuestas en el IETF, que permitan mejorar la respuesta de la red MPLS con DiffServ frente a la pérdida de un enlace.

1.4 OBJETIVO GENERAL

Realizar un estudio y análisis de la simulación de una red MPLS con la Internet de los Servicios Diferenciados.

1.4.1 OBJETIVOS ESPECIFICOS

La presente tesis tiene como objetivos específicos:

- Realizar un análisis y simulación del comportamiento de una red MPLS con Servicios Diferenciados, mediante enrutamiento múltiple.
- Aportar en el entendimiento de nuevas soluciones en la Internet para soportar nuevas aplicaciones.
- Implementar un algoritmo predictivo para la determinación del ancho de banda disponible para cada enlace en cada nodo de una red MPLS-DiffServ y establecer la

mejor ruta de extremo a extremo, durante el periodo de actualización de datos de ancho de banda disponible usando el protocolo OSPF

- Realizar un análisis y simulación del re-enrutamiento como respuesta de la red MPLS con DiffServ frente a la pérdida de un enlace, es decir, cual es la mejor alternativa para mejorar la resiliencia de la red.

1.5 METODOLOGÍA

Se realiza un estudio de los temas concernientes a las propuestas, se plantea una hipótesis, luego se efectúa las simulaciones, los resultados de estas simulaciones son graficados, tabulados para su posterior análisis y conclusiones respectivas.

Las simulaciones se realizaron utilizando el programa ns (Network Simulator), el cual nos permite evaluar y analizar las distintas características de una red MPLS con Servicios Diferenciados tales como el ancho de banda, retardo, variación de retardo (jitter) y rendimiento (throughput). Este simulador fue desarrollado en el proyecto SAMAN (Simulation Augmented by Measurement and Analysis for Networks) **[1]** con el patrocinio de DARPA (Agencia de Investigación y Proyectos Avanzados de la Defensa de los Estados Unidos de Norteamérica) y existen versiones tanto para la plataforma Windows como Linux y se encuentra disponible en Internet.

En las simulaciones que se efectuaron se ha usado el simulador NS 2.1b6a con la integración **[2]** de MNS 2.0**.**

1.6 ANTECEDENTES

La optimización de los recursos de la red permite a MPLS suministrar mejor calidad de servicio entorno al cual existe un buen número de propuestas en esta línea. Sin embargo, MPLS por si solo no puede proveer diferenciación de tráfico, siendo este un requisito imprescindible para la provisión de garantía de servicio de calidad de servicio. Por esto, la sinergia entre MPLS y DiffServ tiene como antecedente la RFC 3270 además, de ser tratado en **[41], [42] y [43].** En el Perú, no existe un antecedente que no sea la publicación **[37]** y **[38].**

1.7 ORGANIZACIÓN DE LA TESIS

La Tesis esta estructurada de la siguiente manera. En el Primer Capitulo se realiza una introducción general, se indica la motivación que conlleva a desarrollar esta Tesis, se presenta las propuestas, los objetivos de la Tesis y la organización de la Tesis respectivamente. En el Segundo Capitulo se trata acerca de Calidad de Servicio, para poder comprender los conceptos y parámetros de la Calidad de Servicio (QoS) en las arquitecturas de protocolos de la Internet y luego un análisis de la Actual Internet. En el Tercer Capitulo se trata acerca de las Arquitecturas IntServ con el modelo de los Servicios Integrados y DiffServ con el modelo de los Servicios Integrados, luego se realiza una comparación entre ambas arquitecturas. En el Cuarto Capitulo se trata acerca de MPLS, que se considera fundamental en la construcción de las bases de la Internet del XXI, luego se trata sobre GMPLS y aplicaciones de MPLS. En el Quinto Capitulo se trata el estado del arte de MPLS y DiffServ en una misma red de MPLS con DiffServ. En el Sexto Capitulo se presenta un Algoritmo Predictivo para la Determinación de Ancho de Banda Disponible. En el Séptimo Capitulo se realiza un análisis y simulación de MPLS con DiffServ, incluyendo un análisis del re-enrutamiento en la red MPLS-DiffServ. En el Octavo capitulo se presenta las conclusiones recomendaciones del desarrollo de la tesis.

CAPITULO 2 Calidad de Servicio (QoS) y Análisis de la Internet Actual

2.1 INTRODUCCIÓN

El advenimiento de Internet y la globalización, han modificado las formas de comunicación en el mundo y también en las empresas; por lo tanto hoy en día es fundamental, para cualquier compañía, un medio que facilite dicha comunicación. Ésta demanda se irá incrementando en los próximos años con el objetivo de contar con comunicaciones permanentes y seguras.

Al mismo tiempo que aumenta la demanda de comunicaciones a través de Internet, también se busca la Calidad, es decir, el proceso de entrega de datos de manera fiable y/o "mejor de lo normal", basado en el uso eficiente de los recursos de la red y el Servicio como algo ofrecido al usuario final de la red, en aplicaciones cliente/servidor, enlaces de extremo a extremo de la red, transporte de datos, etc. Las aplicaciones actuales exigen cada vez mayores recursos de la red para brindar servicios con calidad, estas aplicaciones son de dos tipos; en primer lugar tenemos las aplicaciones No Elásticas aquellos que son sensibles a la llegada de los paquetes; en segundo lugar tenemos las aplicaciones Elásticas aquellas que esperan los datos cuando estas se retrasan. El crecimiento en el tráfico que circula por Internet así como la proliferación de servicios basados en ella han evidenciado las carencias del modelo **[6]** Best-Effort. En este modelo, cualquier tráfico es tratado de la misma forma. Es decir, no hay tráfico más prioritario que otro y la red solo se diseño para hacer lo mejor que pueda (best-

effort) para hacer llegar un paquete a su destino, el cual resulta apropiado para aplicaciones clásicas de redes de datos como correo electrónico, la transferencia de archivos, la navegación en la WWW y el comercio electrónico, que constituyen las aplicaciones Elásticas. Sin embargo, estas redes se utilizan cada vez más para transportar no sólo datos sino también flujos multimedia en difusión (por ejemplo, servicios de audio y video en streaming) e interactivos (por ejemplo, servicios de voz sobre IP) conocidas como aplicaciones No Elásticas.

Se hace necesario cambiar el modelo de servicio Best-Effort para adecuarlo a las nuevas aplicaciones, especialmente aquellas con gran consumo de ancho banda y requerimientos estrictos en cuanto a retardo y otros requerimientos. Esto significa, mejorar la calidad de servicio que se presta a los usuarios a través de la red, entendiéndose como mejorar los parámetros de retardo, rendimiento, ancho de banda entre otros parámetros. Los dos primeros modelos que se proponen para que Internet ofrezca una mejor QoS para aplicaciones de tiempo real son los modelos de IntServ o Servicios Integrados donde se crean estados en cada nodo para reservar recursos y los Servicios Diferenciados o DiffServ, donde se asignan las prioridades a cada paquete para que sean tratados por cada nodo de manera diferente respecto a los otros datos y recientemente MPLS (Multiprotocol Label Switching) que no solo se limita a ofrecer QoS en redes IP sino para optimizar la QoS en redes como ATM, y Frame Relay.

En este capitulo abordaremos el tema: Calidad de Servicio (QoS), para luego hacer un análisis de la Internet actual.

2.2 CALIDAD DE SERVICIO (QoS)

La Calidad de Servicio (QoS), se puede entender **[7]** como la medida del comportamiento de la bondad de la red con respecto a ciertas características de los servicios definidos, o como la capacidad de una red para proveer mejor servicio para un determinado tipo de trafico. Los parámetros relacionados con QoS son: Ancho de Banda, nivel de retardo o latencia, variación del retardo o jitter, rendimiento o throughput, pérdida de paquetes. Una red debe garantizar; que puede ofrecer un cierto nivel de Calidad de Servicio para un nivel de tráfico con un conjunto especificado de parámetros.

La implementación de Políticas de Calidad de Servicio se puede enfocar en varios aspectos según los requerimientos de la red, los principales son:

- Asignar ancho de banda en forma diferenciada.
- Evitar y/o administrar la congestión en la red.
- Manejar prioridades de acuerdo al tipo de tráfico.
- Modelar el tráfico de la red.

2.3 PARÁMETROS DE LA CALIDAD DE SERVICIO (QoS)

Tanto en el caso del tráfico que corresponden a aplicaciones elásticas y no elásticas, así la noción de QoS es muy importante y esta definida como un conjunto de parámetros que representan las propiedades de los **[8]** tráficos. En general existen los **[9]** siguientes parámetros:

2.3.1 RENDIMIENTO (Throughput)

Es el parámetro más importante y especifica cuantos datos (máximo o media) son transferidos a través de la red. En general no es suficiente expresar en término de bit por segundo, sino en unidades de paquetes, ya que el esquema de calidad de servicio debe ser aplicable a varias redes o sistemas de propósito general. El Throughput es medido después de la transmisión de datos porque un sistema añade retardo causado por limitaciones del procesador, congestión de la red, ineficiencias del proceso de almacenamiento de datos
en los buffers, transmisión de bits errados, carga de tráfico, hardware inadecuado. El Throughput varía con el tiempo durante la transmisión de datos debido al tráfico y la congestión. Cuando la data es paquetizada en tramas que contienen cabeceras, se quiere medir el Throughput respectivo es necesario despaquetizarlas, es decir extraer todos bits usados como cabeceras. La información de la cabeceras de las tramas (direcciones de origen y destino, parámetros para intercambio de información, código para chequeo de errores entre otros), reduce el Throughput.

2.3.2 RETARDO (Delay)

Se refiere al tiempo que dura en transmitirse un bit desde su origen hasta su destino. Es un parámetro que se emplea para medir el máximo retardo en una red de extremo a extremo. El retardo es ocasionado por la distancia, errores en la transmisión (bits errados), las capacidades de procesamiento de los sistemas que están involucrados en la transmisión, y otros factores. Aun si eliminaramos estos factores, el retardo siempre estará presente, es decir no puede ser eliminado.

2.3.3 VARIABILIDAD (Jitter)

Expresa la variación experimentada entre dos retardos consecutivos durante la transmisión y procesamiento de datos. El Jitter puede amortiguarse mediante el incremento de buffers (buffering) en los receptores lo que a su vez, incrementa el retardo extremo a extremo.

2.3.4 PERDIDA DE PAQUETES O FIABILIDAD (RELIABILITY)

Esta referida a la perdida de paquetes y corrupciones de datos durante la transferencia de datos. Cuando ocurre congestión en una red, los paquetes tienden a caerse debido a un sobre flujo del buffer o debido al esfuerzo limite de retardo. Las perdidas de paquetes afectan directamente la visión de la Calidad de Servicio en el lado del receptor extremo. Una ventaja del tráfico multimedia es su tolerancia a las perdidas donde las perdidas menores al 2 % suelen pasar inadvertidas. Para paliar los efectos en caso de perdidas elevadas se usan técnicas no excluyentes que permiten elevar el umbral de perdidas inclusive hasta el 20 % , dependiendo de la codificación entre estas técnicas podemos mencionar las siguientes: FEC basada en la redundancia, Interleaving basada en hacer compartir las perdidas entre todos los datos de un paquete y de esta forma aminorar los efectos de la perdidas bastante aplicable para los casos de transmisión de audio o video, Ocultación de perdidas (Loss Canceling) en el receptor basada en la Repetición, interpolación y predicción de los datos de llegada.

Algunos conceptos que están relacionados a los conceptos vertidos son los siguientes:

2.3.5 ANCHO DE BANDA (Bandwidth)

Es la capacidad de transportar información a través de un canal de comunicación. Este canal puede ser analógico o digital. En la transmisión analógica tal como en telefonía, radiodifusión (AM o FM), es medido en ciclos por segundo (hertz). En la transmisión digital (velocidad de transmisión) es medido en bits **[10]** por segundo. Para sistemas digitales, los términos "Ancho de Banda (Bandwidth)" o "Capacidad (Capacity)" se usan indistintamente, por ejemplo para indicar la capacidad de un canal o del enlace entre dos nodos en red. Aunque es más apropiado referirse a velocidad de transmisión cuando se trata de expresarse desde el punto de vista de la transmisión digital. Este concepto esta relacionado con el Throughput (el cual es medido cuando el retardo es considerado) ya que ambos se pueden expresar en las mismas magnitudes (bits/seg), empero Bandwidth se ha hecho más usual aplicarlo a la capacidad de los canales de comunicación de los enlaces entre los nodos en una red.

2.3.6 LATENCIA (Latency)

Un método para medir la Latencia es ver cuanto tiempo se demora un dispositivo en procesar un paquete. Este dispositivo puede ser un router, un sistema completo de comunicaciones que incluye routers y enlaces, en muchos casos hablar de Latencia es sinónimo de Retardo (Delay).

2.4 NIVELES DE QoS EN UNA RED DE EXTREMO A EXTREMO

Los niveles de servicio en una red de extremo a extremo se definen como la capacidad de una red para entregar servicio, especificando la necesidad de cada tráfico en una red. Los servicios difieren en el nivel de QoS, el cual describe, lo siguiente: como el servicio puede ser establecido por un específico Ancho de banda, Retardo, Jitter y Pérdida de datos. Existen tres niveles básicos de QoS que proveen las distintas redes, pueden ser definidas tal como se ilustra en la figura 2.1.

2.4.1 SERVICIO BEST EFFORT

También conocido como QoS deficiente. Best effort es un servicio que presenta una conectividad básica pero sin garantía. Es caracterizado por colas tipo FIFO los cuales no presentan diferenciación entre flujos.

2.4.2 SERVICIOS DIFERENCIADOS

Conocido como QoS blando, el tratamiento de los tráficos es según prioridades. Un tráfico puede ser tratado mejor que el resto (atención rápida, asignación de mayor ancho de banda y baja tasa de pérdidas). Esta provista por una clasificación de tráfico y el uso de herramientas de QOS tales como PQ (priority queuing), CQ (custom queuing), WFQ (weighted fair queuing) y WRED (Weighted Random Early Detection) estos últimos son tratados en el A**péndice C.**

2.4.3 SERVICIOS GARANTIZADOS

Conocido también como QoS duro. Se reserva recursos concretos para un tráfico específico. Este servicio esta provisto a través de herramientas de QoS como RSVP (Resources Reservation Protocol) y CBWFQ (Class Based -Weighted Fair Queuing).

Decidir cual de los tipos de servicio es apropiado para desplegarlo sobre la red depende de varios factores:

- La aplicación o problema que el usuario esta tratando de resolver. Cada uno de los tres tipos de servicio es apropiado para ciertas aplicaciones. Esto significa que el usuario no necesariamente debe migrar sus aplicaciones hasta Servicios Diferenciados o Servicios Garantizados aunque en muchos casos podría ser así, ya que un Servicio diferenciado puede brindar el mismo resultado que un Servicio Best Effort, por ejemplo para un paquete de correo electrónico se puede alquilar un servicio diferenciado o un Servicio Best Effort y en cualquiera de estos servicios se va ha brindar la misma calidad, porque el correo electrónico es una aplicación que puede tolerar la demora en su recepción.

- La necesidad de actualizar su infraestructura, de acuerdo a los nuevos requerimientos del nuevo servicio que esta alquilando por ejemplo cuando el

usuario pasa del Servicio Diferenciado al Servicio Garantizado, en este caso para el Servicio Garantizado necesita mayor equipamiento en cuanto a infraestructura.

- El costo de la implementación de depende del nivel de QoS que el usuario elija. Para aplicaciones con Servicio Garantizado requiere mayores recursos que para aplicaciones con Servicios Diferenciados.

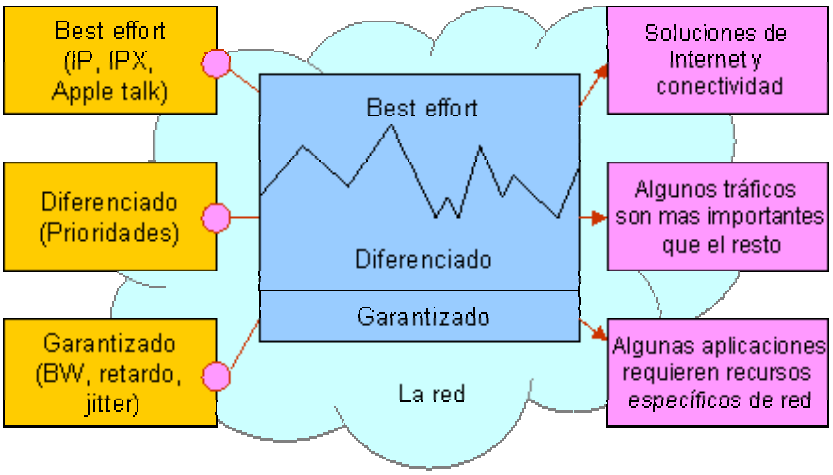

Figura 2.1

2.5 ARQUITECTURA BASICA DE QoS

La arquitectura básica de la calidad de servicio introduce tres componentes fundamentales para la implementación de QoS en una red de Internet, tal como se ilustra en la figura 2.2.

- QoS en la identificación y técnicas de marcado de paquetes (clasificación y reservación) para la coordinación de QoS de extremo a extremo entre los elementos de la red (señalización).

- QoS dentro de un solo elemento de la red que puede ser un router (enrrutador o encaminador), equipo de acondicionamiento de tráfico entre otros, que depende del tipo de tecnología implementada.

- QoS en las políticas de manejo y acondicionamiento de las funciones de la red para el control y administración de tráfico de extremo a extremo en una red.

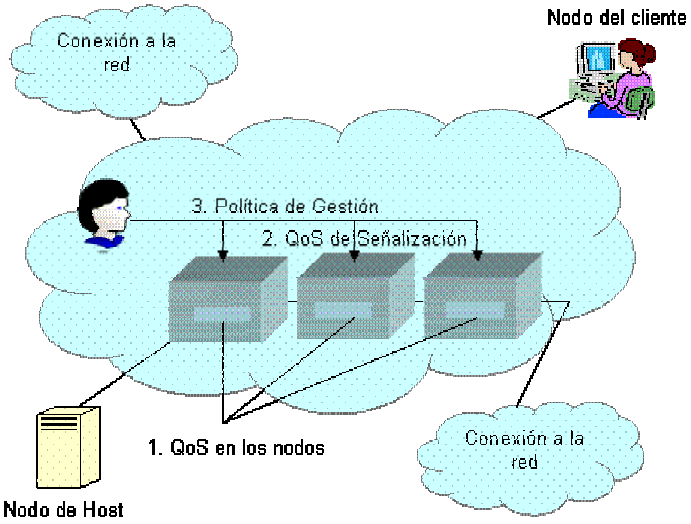

Figura 2.2 Arquitectura Básica de una Red con QoS

2.5.1 IDENTIFICACION Y MARCADO

Se cumple mediante la Clasificación y Reservación.

2.5.1.1 CLASIFICACIÓN

Para proveer prioridad a ciertos flujos, primero el flujo tiene que ser identificado y si se desea puede ser marcado. Estas dos tareas son comúnmente referidas como solo clasificación.

Históricamente, la identificación fue realizada usando el control de Acceso a Listas (ACLs). ACL identifica los tráficos para la administración de la congestión mediante herramientas como PQ y CQ. Porque PQ y CQ son usados en routers que están basados en salto por salto (hop by hop) donde las políticas de prioridad que están vinculadas al Precedence, son propias en cada router, el proceso de identificación de paquetes es solamente usado en cada router.

Típicamente esta funcionalidad (Clasificación) esta implementada en un extremo de la red o cerca del dominio administrativo, para que cada elemento subsiguiente en la red pueda proveer un servicio basado en determinada política. Por ejemplo cuando un URL (Uniform Resources Localizer) es identificado en un paquete http, este paquete puede ser marcado según IP Precedence y ser enviado a través de la red.

Algunos aspectos de la Clasificación:

De acuerdo a los niveles de IP Precedence establecidos sobre los tráficos, se puede crear Servicios Diferenciados. Estas herramientas potentes proveen simpleza y flexibilidad para la implementación de políticas de QoS en una red.

IP utiliza los 3 bits de Precedence en la cabecera de IPv4 en el campo Type of Service (ToS) para especificar la clase de servicio para cada paquete, como es mostrado en la figura 2.3 se puede dividir el tráfico en seis clases de servicio usando los IP Precedence y los otros dos están reservados para uso interno de la red. Las tecnologías de manejo de colas en la red pueden usar esta referencia indicada para proveer un manejo apropiado del tráfico.

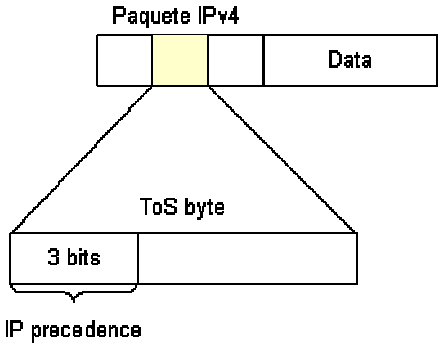

Figura 2.3 sub IP precedence

En la figura 2.4 se observa la configuración del subcampo Precedence cuando toma el valor de 5 y el campo ToS toma el valor de 160.

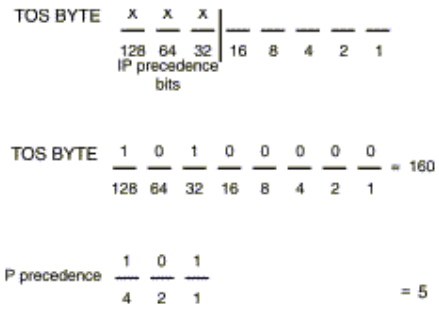

Figura 2.4

En el RFC2475 que trata sobre la arquitectura de los Servicios Diferenciados se define una extensión de 3 a 6 bits para IP Precedente como DS Code Point donde los 2 bits

menos significativos se reservan para usos futuros. Esta especificación es comúnmente llamada DiffServ.

El aspecto de Reservación de recursos para QoS esta vinculado con RSVP que es un estándar de IETF (RFC 2205) para aplicaciones que puedan reservar recursos de QoS en cada router. Es decir RSVP está disponible para aplicaciones que requieren un QoS específico como se puede ilustrar en la figura 2.5 donde se observa que mediante los Servicios Garantizados de RSVP se puede garantizar un Ancho de Banda. Existen otros protocolos que permiten la reserva de recursos en cada nodo como CR-LDP de MPLS, BGP en sistemas autonomos.

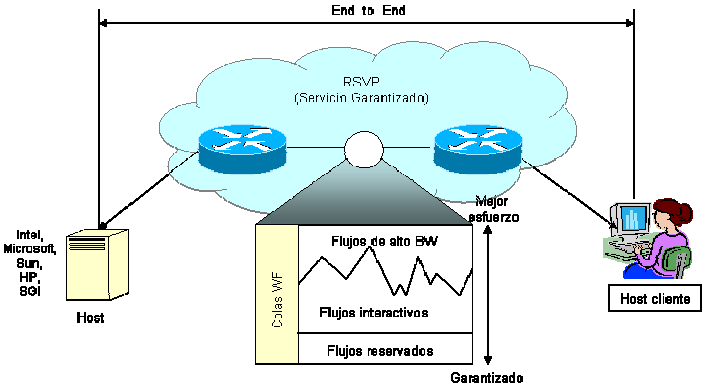

Figura 2.5 RSVP con QoS

RSVP será tratado con mayor profundidad en los capítulos III y IV

2.5.2 QOS DENTRO DE UN NODO DE LA RED

Son varias las funciones del router (nodo) de la red como son: enlace eficiente (fragmentación e Interleaving), la administración de la congestión, administración de colas, establecer las herramientas de conformación y políticas de tráfico de QOS.

2.5.2.1 ENLACE EFICIENTE (Link Efficient)

Los paquetes pequeños tienen problemas en los enlaces de baja velocidad. Por ejemplo, en el caso de los retardos de serialización para un paquete de1500 bytes sobre un enlace de 56 kbps, son de 214 milisegundos. Si un paquete de voz va detrás de este gran paquete, su retardo será excesivo. Para evitar esto, se produce la Fragmentación y el Intercalado (Interleaving), es decir, se segmenta el paquete grande, en pequeños

paquetes e intercala con los paquetes de voz. El Interleaving es tan importante como la Fragmentación.

Ejemplo del cálculo del retardo por serialización:

Tamaño del paquete: 1500 – bytes / paquete x 8 bits / byte = 12000 bits.

Velocidad de transmisión de l enlace: 56000 bps.

Resultado: 12000 bits / 56,000 bps = 0.214 segundos

= 214 mili segundos

Otro aspecto para el enlace eficiente es la eliminación de los bits de overhead.

2.5.2.1.1 MECANISMOS DE ENLACE EFICIENTE

Existen dos mecanismos que permiten el Enlace Eficiente: LFI (Link Fragmentation Interleaving) y RTP-HC (Real Time Protocol Header Compresion). Los cuales mejoran la eficiencia y la predictibilidad de las aplicaciones de niveles de servicio.

LFI reduce el retardo y el Jitter en los enlaces de baja velocidad; fraccionando los paquetes grandes e intercalando con los paquetes pequeños, LFI fue diseñado para los enlaces de bajo Ancho de Banda.

RTP-HC (Real Time Protocol and Header Compression) fue diseñado para transportar aplicaciones de multimedia incluyendo audio y video paquetizado sobre redes IP. RTP-HC comprime las cabeceras de los paquetes reduciendo el Overhead y disminuye significativamente los retardos.

2.5.2.2 CONFORMACION Y POLÍTICAS DE TRÁFICO

2.5.2.2.1 SHAPPING

Es usado para crear tráfico de flujo que limita el ancho de banda de los flujos y prevenir el problema de OVERFLOW (ver acápite 2.5.2.4). Por ejemplo cuando un trafico comienza con T1 y termina con 384 kbps, debido a que la demanda de ancho de manda haya aumentado durante el proceso de transmisión entonces a este trafico se le tuvo que limitar su ancho de Banda.

2.5.2.2.2 POLICING

Es similar a Shapping, pero que defiere en que el exceso de tráfico no es guardado en el buffer (según el ejemplo anterior T1-384 kbps, no es guardado en el buffer). Es decir que esta información no es almacenada para una posterior transmisión en otras condiciones de tráfico.

2.5.2.2.2.1 HERRAMIENTAS DE POLITICAS DE TRÁFICO

Entre las herramientas de políticas de flujo se tiene CAR (Committed Access Rate) fundamentalmente, provee la mayor prioridad posible a un flujo y limita la prioridad a otro flujo. CAR es usado para limitar el Ancho de Banda de un Flujo para favorecer otro flujo se basa en el algoritmo de Token Bucket, que permite regular el tráfico de un flujo en función de la tasa trasmisión en cada momento a una tasa fijada que varia con sobrepicos por un cierto intervalo. Otro mecanismo, el Leaky Bucket que permite regular el trafico de un flujo a una determinada tasa de transmisión prefijada, al final no resulta ser eficiente si la tasa de los flujos son demasiados bajos, ocupando un ancho de banda mayor en canal de transmisión, que se merece. Un mecanismo que combina ambas características de los mecanismos mencionados anteriormente es lo mas podría resultar mas eficiente.

2.5.2.3 ADMINISTRACIÓN DE LA CONGESTIÓN

La Administración de la Congestión contempla mecanismos que hacer cuando, un trafico excede al Ancho de Banda, entonces se toma las decisiones del caso, como por ejemplo cuando el tráfico (voz, video, datos) excede el Ancho de Banda del enlace, en esta situación, ¿qué debe hacer el router?, quizá permitir que haya una sola cola y que el primer paquete que llega debe ser el primero que salga o que deben formarse varias colas. Las herramientas de Administración de congestión responden a la pregunta antes mencionada.

Un método para manejar un OVERFLOW (ver acápite 2.5.2.4) de tráfico es el uso de algoritmos de colas, para clasificar el tráfico, y determinar algunos métodos de priorización de flujo de tráfico.

Existen las siguientes herramientas:

- First in-First out (FIFO) Queuing

- Priority Queuing (PQ)
- Custom Queuing (CQ)
- Flow Based -Weighted Fair Queuing (WFQ)
- Class Based -Weighted Fair Queuing (CBWFQ)

Estas herramientas solamente se usan cuando hay congestión, es decir cuando se forman colas. En ausencia de congestión, todos los paquetes son distribuidos directamente a la interfaz de salida de un router.

2.5.2.4 ADMINISTRACIÓN DE COLAS (Queue Management)

La administración de colas es necesaria porque la capacidad de las colas tiene un límite, ya que si se sobrepasa este límite se produce el OVERFLOW. Cuando una cola esta llena, un paquete más, ya no puede ingresar, entonces este paquete se perderá. Esta pérdida es previsible. Los routers no pueden evitarlo aún si se trata de paquetes considerados de alta prioridad. Entonces un mecanismo es necesario para hacer lo siguiente:

- Tratar de asegurar que las colas no se llenen, de tal forma que los paquetes de alta prioridad puedan seguir su curso normal.
- Permitir cierta clase de pérdidas de paquetes, que sean los de baja prioridad, antes que los de alta prioridad.

Los mecanismos para evitar la congestión son formas de administración de colas. Estas técnicas lo que hacen son monitorizar la carga de tráfico de la red en un esfuerzo para anticipar y evitar la congestión en los “cuellos de botellas”, como oposición a las técnicas de Administración de Congestión que opera para controlar después que se ha producido la congestión.

Entre las principales técnicas para evitar la congestión se tiene el RED (Weighted Random Early Detection).

2.5.3 ADMINISTRACIÓN DE QOS (QoS Management)

Ayuda a evaluar las políticas y objetivos de QOS mediante la siguiente metodología:

- Contar con un programa de monitoreo de la red que nos ayude a determinar las características del tráfico de la red. Los objetivos de las aplicaciones para QOS son establecidos como base, los cuales pueden medirse en términos de su respuesta en el tiempo.
- Despliegue de las técnicas de QOS cuando las características del tráfico han sido obtenidas.
- Evalúa los resultados de la prueba, la respuesta de las aplicaciones sobre las cuales se ha aplicado QOS y ver si se han conseguido los resultados esperados.

Para poder administrar QOS a través de la red se cuenta con una serie de herramientas entre las cuales podemos mencionar RMON (Cisco), MRTG, entre otros que nos puede suministrar información referente a las características del tráfico en la red. La información suministrada por ejemplo por RMON (programa) nos ayuda a validar cualquier despliegue de QOS sobre la red así como poder establecer políticas de QOS.
En las redes basadas en MPLS, en la actualidad, basadas en las clases de servicios, se puede distinguir como influye la QOS.

2.6 ANÁLISIS DE LA ACTUAL INTERNET Y LA FALTA DE QOS

Cuando se diseño la Internet con el protocolo IPv4, fue diseñada para el transporte de aplicaciones en los cuales el tiempo de envío y respuesta no es un problema importante, esto sucede en aplicaciones como, correo electrónico, ftp, telnet, http. La actual Internet ofrece un servicio básico conocido como “Best Effort”, donde la red no hace ninguna distinción entre cada aplicación. En la concepción inicial de la Internet la idea fue de dotar de inteligencia a los Hosts extremos, y a los nodos de la red dotarlos con la mínima capacidad de procesamiento, estos debían solo realizar una labor de encaminamiento de paquetes sin ninguna acción adicional, pero en la actualidad esta forma de trabajo es inadecuada para tráficos en tiempo real que corresponden a aplicaciones como video conferencia, voz sobre IP (VoIP), telecontrol, telemedicina, teleducación, entre otros, lo que muestra la falta de calidad de servicio de la Internet actual para este tipo aplicaciones.

La Internet actual necesita de cambios y tiende a la obsolescencia debido a que no soporta las nuevas aplicaciones de tiempo real, entre otros. De allí que se han desarrollado nuevos modelos para que la Internet de la siguiente generación asegure una buena calidad de servicio (QoS) dependiendo de los datos que se envían a través de la red.

En la actualidad para el tipo de tráfico de tiempo real que circula por la red, no solo se requiere un tratamiento especial en cada nodo de la red, sino que los protocolos actuales sean redefinidos y otros nuevos sean definidos para ofrecer un tratamiento diferencial a los datos que fluyen a través de la red. Todo lo anterior conduce a plantear nuevos modelos en la Internet, manteniendo el protocolo IP en su versión actual (IPv4), así como en su nueva versión (IPv6), que es una redefinición del protocolo IP.

Los primeros modelos que se proponen para que la Internet del futuro asegure una buena calidad de servicio (QoS) para aplicaciones de tiempo real son los Servicios Integrados o IntServ, donde se crean estados en cada nodo de la red para reservar recursos, y los servicios diferenciados o Diffserv, donde se asignan prioridades a cada paquete para que sea tratado por cada nodo de manera diferente respecto a otros paquetes. Pero también se propone un nuevo enfoque para no solo ofrecer QoS en las redes IP sino también para optimizar la QoS en redes como ATM, Frame Relay y se trata de Multiprotocol Label Switching o MPLS, todos estos modelos los trataremos con mayor detalle en los capítulos 3 y 4 respectivamente.

La Internet de hoy está experimentando un cambio respecto a las aplicaciones que transporta, por ejemplo ahora ya es una realidad el uso de teléfonos móviles con acceso a Internet con tecnología WAP, GPRS y redes celulares de tercera generación UMTS (3G), televigilancia basado en redes IP, teleducación, telemedicina entre otros. Por lo tanto es necesario disponer de una nueva Internet con capacidad de soportar más aplicaciones del futuro. Por otro lado, todas estas nuevas aplicaciones podrán ofrecer un mejor funcionamiento si a cada uno de los terminales que lo soportan se les asigna una dirección IP propia. La razón de cambiar el protocolo IPv4 a IPv6, se sustenta en que IPv4 no fue diseñado para soportar nuevas aplicaciones.

Una de las primeras limitaciones detectadas al protocolo IPv4 fue las pocas direcciones IP que se pueden definir (2^{32} = 4,294'967,296 direcciones). Se estima que para el año 2010 el número de usuarios sobrepase estas direcciones IP, estas aproximaciones son hechas sin considerar el impacto que la Internet Móvil ya esta generando, con el crecimiento vertiginoso de la telefonía celular que esta teniendo últimamente. De otro lado el procesamiento de los paquetes de datos, que se lleva a cabo en los nodos de la red, necesita ser procesados de manera inmediata, realizándose para esto previamente, mecanismos de clasificación y planificación en cada nodo. El proceso de Clasificación se realiza para identificar los paquetes de datos que requieren un tratamiento especial de la red y el mecanismo de Planificación, para definir como serán tratados los paquetes de datos clasificados para ofrecer, la QoS solicitada para las aplicaciones de tiempo real. En este punto surgen nuevas limitaciones del protocolo IPv4. En primer lugar la cabecera IPv4 (ver Figura 2.18) presenta 12 campos y su procesamiento en cada nodo de la red limita el envío rápido de paquetes, sobre todo teniendo en cuenta que alguno de estos campos, son redundantes como el chequeo de cabecera, y otros innecesarios para el tipo de aplicaciones de tiempo real, como son los campos relacionados al proceso de fragmentación lo que implica agregar tiempo de procesamiento en los nodos de la red, es así que en la futura Internet solo el host emisor realizará la fragmentación. En segundo lugar, el campo ToS (Type of Service) en la actualidad es un campo poco usado, por lo que se redefinió para ofrecer una alternativa de prioridad a los paquetes que ingresan a la red, con el fin de ofrecer un trato diferencial. En tercer lugar no hay mecanismos para distinguir o clasificar flujos de datos de otros. La identificación de flujos de datos resulta ser otra alternativa de para asignar recursos de la red a determinados flujos de datos, bien podrían ser estos flujos a aplicaciones de tiempo real como aplicaciones de videoconferencia, etc. En cuarto lugar nuevas mejoras al actual protocolo son prácticamente imposibles, ya que el campo asignado para ello es de solo 40 bytes como máximo. Por todas estas limitaciones que presenta IPv4; la IETF en **[4]** ha definido el protocolo IPv6 que será el soporte de la Internet del futuro, al cual se le considera la base de la Internet de la siguiente generación. Una de las características más notorias de este protocolo reside en el campo de direcciones que es de 128 bits que define hasta 2^{128} direcciones, asegurando de esta manera direcciones IP para futuras aplicaciones que accedan a Internet. Otro cambio radica en la eliminación de campos innecesarios, en la cabecera de este protocolo IP, ahora se dispone de 08 campos, de

esta manera contribuye a mejorar el tiempo de procesamiento de los paquetes en los nodos de la red.

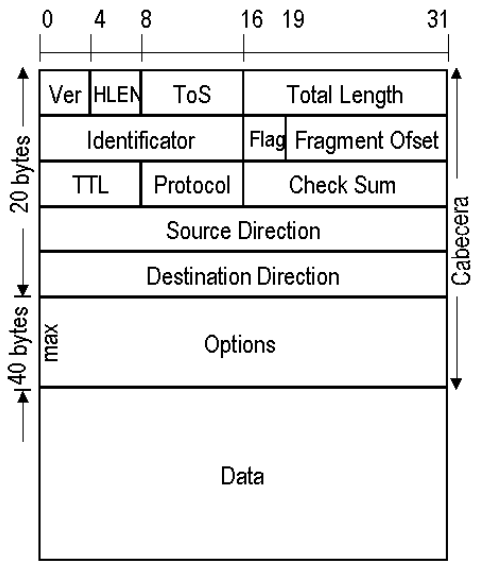

Figura 2.6 Formato del Protocolo IPv4

Para las aplicaciones multimedia, en IPv6 se han definido dos campos: Etiqueta de Flujo y Servicios Diferenciados o DS, como se ilustra en la figura 2.19. El campo etiqueta de flujo permite identificar a los paquetes pertenecientes a un flujo de tráfico en particular, para lo cual se requiere un trato especial. Este campo define la arquitectura de Internet de Servicios Integrados o IntServ. El campo Servicios Diferenciados o DS, que es una mejora de la opción ToS (Type of Service) definido en IPv4, asigna niveles de prioridades a cada paquete de datos para que cada router asigne un determinado comportamiento por salto o PHB (Per-Hop-Behavior). Este campo define la arquitectura de Servicios Diferenciados o DiffServ.

El IPv6 presenta muchas características **[5]** importantes. Por ejemplo, un usuario se podrá unir a una red IPv6 de manera inmediata (Plug&Play) o ante la presencia de varios servidores con igual contenido de información (mirrow), un usuario podrá acceder a aquel que se encuentre más próximo, sin necesidad de realizar ningún proceso adicional que seleccionar la dirección de los (direccionamiento anycast) hosts. Este protocolo IPv6 se encuentra bastante estable y ya existen redes con el protocolo IPv6 funcionando como Internet2, 6Bone y empresas como Microsoft tienen incorporado

BIBLIOGRAFÍA

[1] SAMAN Simulation Augmented by Measurement and Analysis for Networks http://www.isi.edu/nsnam/ns/index.html.

[2] "MNSver2.0 Manual", http://flower.ce.cnu.ac.kr/~fog1 /mns/mns2.0/manual/manual.htm

[3] Software de instalación del ns para el sistema operative Linux http://www.isi.edu/nsnam/dist/index.html

[4] S. Deering, R. Hinden, "Internet Protocol, Versión 6 (IPv6) Specification", IETF Standards Track RFC 2460, December 1998

[5] Alberto López Toledo, "Calidad de Servicio en Ipv6", Madrid Global IPv6 Summit, Enero 2001

[6] Centro de Comunicaciones Avanzadas de Banda Ancha, "Internet 2 a

Catalunya (I2CAT)", Febrero 1999 http://www.ccaba.upc.es

[7] Cisco, "Internetworking Technology Handbook", capitulo "Quality of Service (QoS)" Diciembre 2003

[8] Rec. I.350 "Aspectos generales de Calidad de Servicio y de Calidad de Funcionamiento en las Redes Digitales Incluidas las Redes Digitales de Servicios Integrados", ITU-T, 1993

[9] Rec. E.800 "Terms and Definitions Related To The Quality of Telecommunications Services", CCITT, 1988.

[10] Tom Sheldon " Encyclopedia of Networking and Telecommunications" Mc Graw Hill, 2001

[11] Rec. I.363.5, "Capa de Adaptación del Modo de Transferencia Asíncrono Tipo 5",ITU-T, Agosto 1996.

[12] Rec. I.371, "Control de Trafico y Control de Congestión en la RDSI-BA", ITU-TS, Agosto 1996

[13] S. Van Luinen, Z. Budrikis, and A. Cantóni, "The Contolled Cell Transfer Capability", ACM Sigcomm Computer Comunication Review, pp. 55-71, 1977

[14] ITU-T SG/13 "General Network Aspects", Plenary Meeting, ITU-T,

Junio 1998

[15] R.Braden, Ed., L.Zhang, S.Berson, S.Herzog, S.Jamin RFC 2205 Resource ReSerVation Protocol (RSVP). Version 1 Functional Specificacion. September 1997

[16] E. Rosen, A. Viswanathan, R. Callon Multiprotocol Label Switching Architecture, RFC3031, Enero 2001

[17] C. Semería, J.W Stewart III,"Optimizing Routing Software for Reliable Internet Growth", Juniper Network Inc., White Paper, Julio 1999

[18] J. Barbero "Una Arquitectura de Backbone Para la Internet del Futuro" 2001 Unisource Iberia

[19] Rob Redford Enabling Bussiness IP Services With Multiprotocol Label Switching http://www.cisco.com/warp/public/cc/so/neso/vvda/ipatm/mpls_wp.htm 2001

[20] William Stallings. High-speed networks. TCP/IP and ATM design principles Prentice Hall. 1998

[21] William Stallings. Comunicaciones y redes de computadores. Sexta edición Prentice Hall. 2000.

[22] T.Bates, Y. Rekhter, Cisco Systems, D. Katz Juniper Networks RFC2283 "Multiprotocol Extensions for BGP-4" Junio 2000

[23] L. Anderson, Nortel Networks, P. Doolan, Innovate Networks, N. Feldman, IBM Corp, A. Fredette, PhotonEx Corp , B. Thomas Cisco Systems, Inc. RFC3036 "LDP Specification", Enero 2001

[24] Brittain, Farrel. "MPLS traffic engineering: A choice of signaling protocols" Data Connection. Enero 2000

[25] B. Jacomoussi, Nortel Network, L. Andersson, Utfors BA, R. Callon, Juniper Networks, RFC3212 "Constraint-Based LSP Setup using LDP", Enero 2002.

[26] J. Ash, AT&T, M. Girish, Atoga Systems, RFC3213 "Applicability Statement for CR-LDP", Enero, 2002]

[27] D. Awduche, Movaz Networks, L. Berger, D. Gan, Juniper Networks, RFC3209 "RSVP-TE: Extensions to RSVP for LSP Tunnels", Diciembre 2001

[28] Chuck Semería, Juniper Networks "VPN Fundamentals", 2001.

[29] L.Berger, Movaz Networks, RFC3471"Generalized Multi-Protocol Label Switching (GMPLS), Signaling Functional Description", Enero, 2003

[30] E. Mannie, RFC3445 “Generalized Multi-Protocol Label Switching (GMPLS) Architecture”, Octubre, 2004

[31] C, RFC3473 “Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions”, Enero,2003

[32] P. Ashwood, Nortel Networks, L.Berger, Movaz Networks, RFC3472” Generalized Multi-Protocol Label Switching (GMPLS) Signaling Constraint-based Routed Label Distribution Protocol (CR-LDP) Extensions”, Enero, 2004

[33] M. Victoria de Diego Bartolomé, Diego Gallego Pérez, José Antonio López Mora, Alberto Gómez Vicente, Telefónica Investigación y Desarrollo “ UMTS: Hacia una Red Todo IP”, Enero, 2002.

[34] Omar A. Walid Llorense, Departamento de Ingeniería de Sistemas Telemáticos de la Universidad Politécnica de Madrid” Convergencia Sobre MPLS en la Red UTRAN”, Junio, 2004

[35] K. Kompella, Y. Rekhter, Juniper Networks, RFC3477 “Signalling Unnumbered Links in Resource ReSerVation Protocol-Traffic Engineering (RSVP-TE)”, Enero, 2003.

[36] Y. Rekhter, K. Kompella, Juniper Networks, A. Kullberg, NetPlane Systems, RFC3480 " Signalling Unnumbered Links in CR-LDP", Febrero, 2003.

[37] Daniel .Diaz A Revista de Telecomunicaciones. "Visión Actual y Futura de Internet: IntServ, DiffServ y MPLS" AHCIET Octubre 2002.

[38] Rafael Bustamante Alvarez, "Protocolo de Señalización CR-LDP en la Arquitectura MPLS y un Análisis con Simulación de Calidad de Servicio y Prioridad de Recursos", Agosto 2003

[39] G. Ahn, The MPLS Network Simulator http://www.raonet.com/introduction.shtml

[40]. S. Raghava "Simulation resources", http://csgrad.csvt.edu/~sraghava/qos/

[41] Raúl Jiménez Mateo, Cristina Paniagua Paniagua, Alfonso Gazo Cervero José Luis Gonzáles Sánchez, Francisco J. Rodríguez, "Integración de MPLS y DiffServ en una Arquitectura para la Provisión de QoS 2003

[42] Kulkarni, Amit Narayan "An Invetigation of Forwarding in The MPLS Support for Diferentiated Services", Agosto, 2002

[43] Timea Dreilinger "Diffserv and MPLS",

http://saturn.acad.bg/bis/pdfs/04_doklad.pdf, 2006

[44] Kim, Sangmin, "Source Driven MPLS Multicast" North Carolina State University, Master of Science, 05-06-2003

[45] Wu, Kehang, "Flow Aggregation Based Lagrangian Relaxation with Applications to Capacity Planning of IP Network with Multiple Classes of Service", North Carolina State University Phd Degree, 05-01-2004.

[46] Desai, Vinay Kumar, "Traffic Engineering in Multiprotocol Label Switching VPNs", Luisiana State University, Master Thesis, 02-12-2004.

[47] Aniker, Pooja S, "Traffic Engineering and Path Protection in MPLS Virtual Private Networks", Luisiana State University, Master Thesis, 29-103-2005

[48] Rojanarowan, Jerapong, "MPLS-Based-Best Effort Traffic Engineering", Doctoral Thesis, Giorgia Tech, 12-09-2005

[49] Jhon A. Proakis, Dimitris G, Manolakis, "Tratamiento Digital de Señales, Principios , Algoritmos y Aplicaciones", 1998., Prentice Hall

[50] Alan V. Oppemheim, Ronald Schafer con Jhon Rbuck , " Tratamiento de Señales en Tiempo Discreto", 2000, Prentice Hall

[51] Vigía K. Madisetti Douglas B. Williams, "The Digital Signal Processing" Handbook, 1998, IEEE PRESS.

[52] Boaz Porat, "A Course in Digital Signal Proceessing", 1997, Jhon Wiley & Sons INC.

[53] R. Braden, D Clark, S. Shenker, "Integrated Services in the Internet Architecture", RFC1633, 1994

[54] J. Wroclawski, "The Use of RSVP with IETF Integrated Services", RFC2210, 1997

[55] J. Wroclawski, Specification of the Controlled-Load Network Element Service RFC2211

[56] Sally Floid, Edie Koller "Internet Research Needs Better Models",ICSI Center for Internet Research, Berkeley , California, 2002.

[57] Stuart Kurkowski, Tray Camp, Michael Colagrosso, "MANET Simulation Studies: The Incredibles", MCS Departament Colorado School Mines Golden, Colorado USA, 2005

APENDICE A

Los códigos fuentes para las simulaciones desarrolladas están en lenguaje TCL que es un lenguaje orientado a objetos, cuyas librerías están en lenguaje C con programación orientado objetos, los cuales son susceptibles de ser modificados de acuerdo a las simulaciones que se implementan.

Para el desarrollo de las simulaciones en MPLS se emplea MNS 2.0 que es un conjunto de librerías publicadas en [2]. En el siguiente script se presenta la simulación correspondiente al caso en que la red No DiffServ y No MPLS para una trafico Extra Muy Alto de UDP en cada caso haremos los comentarios pertinentes, asi como los gráficos que podemos obtener de mediante el XGRAPH que es una herramienta que tiene el NS para la graficación de los resultados de las simulaciones, también presentaremos los archivos de AWK que permitieron calcular la cantidad de información transferida por el trafico TCP medido en el destino.

SIMULACION DE NO DIFFSERV NO MPLS DE UN EXTRA MUY ALTO TRAFICO DE UDP

 # Autor: Ing. Rafael Bustamante

 # Universidad Nacional Mayor de San Marcos

 # Facultad de Ingeniería Electrónica

 # Maestría de Telecomunicaciones

```
#
#
# Title : no_ds_no_mpls_huge-01.tcl
#
#         This particular file simulates the following features
#              1. No Diffserv enabled
#              2. No MPLS enabled
#              3. Both UDP and TCP traffic mixed together on the same
link
#              4. 'huge' means very high UDP traffic, some TCP traffic
#              5. TCP traffic is from an infinite FTP source
#              6. The simulation is run for a duration of 20 seconds
of NS
#                 simulation time. Throughput is collected at the
destination
#              7. All the parameters are set upfront and is
changeable.
#

#Create a simulator object

set ns [new Simulator]

#Trace, Nam, and other files

#source side view of number of bytes sent over time.
set trace_file_snd_tcp [open no_ds_no_mpls_huge_send_side.tr w]

#sink side view of number of bytes recieved over time.
#added a bounded variable, ndatabytesrecv_ to TCPSink in NS code to
handle this.
#set trace_file_rcv_tcp  [open no_ds_no_mpls_huge_recv_side.tr w]

#UDP bw recieved at sink
set trace_file_udp [open no_ds_no_mpls_huge_udp.tr w]

#NAM trace
set nam_file [open no_ds_no_mpls_huge_nam.nam w]

#Trace file all for future use
set trace_file_all [open no_ds_no_mpls_huge_trace.tr w]

#Total number of bytes sent to the application above TCP at the
sender.
#added a bounded variable, totalbytes_ to TCPSink in NS code to handle
this.
set trace_file_tcp_bw_total [open no_ds_no_mpls_tcp_bw_total.tr w]

#Trace, Nam file settings
$ns namtrace-all $nam_file
$ns trace-all $trace_file_all

#Variables
```

```
#All links have 1Mb capacity.
set link_capacity 1Mb

#I am using a link capacity of 5Mb for the last step to the
destination.
set link_capacity1 5Mb

set link_delay    10ms

set udp_active 1
set tcp_active 1

set udp_packet_size 1000
set udp_interval 0.005
#1.6Mbps

set udp_flow_id 1

set tcp_flow_id 2

set udp_start_time 5.0
set udp_end_time 20.0
set tcp_start_time 5.0
set tcp_end_time 20.0
set finish_time 25.0
set record_time 0.0

set record_proc_call_interval 0.055

#settings
$ns color $udp_flow_id Green
$ns color $tcp_flow_id Blue



#topology creation

set node0 [$ns node]
set node1 [$ns node]
set node2 [$ns node]

#Router1
set lsrnode3 [$ns node]

#Router2
set lsrnode4 [$ns node]

#Router3
set lsrnode5 [$ns node]

#Router4
set lsrnode6 [$ns node]

#Router5
set lsrnode7 [$ns node]

#Router6
set lsrnode8 [$ns node]

#Router7
set lsrnode9 [$ns node]
```

```
#Router8
set lsrnode10 [$ns node]

#Router9
set lsrnode11 [$ns node]

set node12 [$ns node]



#Links
$ns duplex-link $node0 $lsrnode3 $link_capacity $link_delay DropTail
$ns duplex-link $node1 $lsrnode3 $link_capacity $link_delay DropTail
$ns duplex-link $node2 $lsrnode3 $link_capacity $link_delay DropTail

$ns duplex-link $lsrnode3  $lsrnode5  $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode5  $lsrnode7  $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode7  $lsrnode9  $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode9  $lsrnode11 $link_capacity $link_delay
DropTail

$ns duplex-link $lsrnode3  $lsrnode4   $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode4  $lsrnode6   $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode6  $lsrnode8   $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode8  $lsrnode10  $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode10  $lsrnode11 $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode5  $lsrnode6   $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode7  $lsrnode8   $link_capacity $link_delay
DropTail
$ns duplex-link $lsrnode9  $lsrnode10  $link_capacity $link_delay
DropTail

$ns duplex-link $lsrnode11  $node12    $link_capacity1 $link_delay
DropTail

# Topology  is this:
#
$ns duplex-link-op $node0 $lsrnode3 orient right-down
$ns duplex-link-op $node1 $lsrnode3 orient right
$ns duplex-link-op $node2 $lsrnode3 orient right-up

$ns duplex-link-op $lsrnode3 $lsrnode5 orient right
$ns duplex-link-op $lsrnode3 $lsrnode4 orient right-up
$ns duplex-link-op $lsrnode4 $lsrnode6 orient right
$ns duplex-link-op $lsrnode5 $lsrnode6 orient right-up
$ns duplex-link-op $lsrnode6 $lsrnode8 orient right
$ns duplex-link-op $lsrnode5 $lsrnode7 orient right
$ns duplex-link-op $lsrnode7 $lsrnode8 orient right-up
$ns duplex-link-op $lsrnode8 $lsrnode10 orient right
$ns duplex-link-op $lsrnode7 $lsrnode9 orient right
$ns duplex-link-op $lsrnode9 $lsrnode10 orient right-up
```

```
$ns duplex-link-op $lsrnode10 $lsrnode11 orient right-down
$ns duplex-link-op $lsrnode9 $lsrnode11 orient right
$ns duplex-link-op $lsrnode11 $node12 orient right


#Agents and Sinks

# UDP agent and sink

set udp_agent_cbr [new Agent/CBR]
$ns attach-agent $node0 $udp_agent_cbr
$udp_agent_cbr set packetSize_ $udp_packet_size
$udp_agent_cbr set interval_ $udp_interval

$udp_agent_cbr set fid_ $udp_flow_id

set sink_udp [new Agent/LossMonitor]
$ns attach-agent $node12 $sink_udp
$ns connect $udp_agent_cbr $sink_udp

# TCP agent and sink

set tcp_agent_ftp [new Agent/TCP]
set sink_tcp [new Agent/TCPSink]

$ns attach-agent $node1 $tcp_agent_ftp
$ns attach-agent $node12 $sink_tcp
$ns connect $tcp_agent_ftp $sink_tcp
$tcp_agent_ftp set fid_ $tcp_flow_id
set tcp_ftp_src [new Application/FTP]
$tcp_ftp_src attach-agent $tcp_agent_ftp
#set tcp_ftp_src [$tcp_agent_ftp attach-source FTP]


#Procedures

# Finish procedure: Called at the end of the simulation. finish_time
controls
# the call-time of this procedure

proc proc_finish {} {
      global trace_file_snd_tcp trace_file_rcv_tcp trace_file_udp
nam_file trace_file_all ns
      global trace_file_tcp_bw_total

      close $trace_file_snd_tcp
      #close $trace_file_rcv_tcp
      close $trace_file_udp
      close $trace_file_tcp_bw_total

      $ns flush-trace
      close $nam_file
      close $trace_file_all

        exec ../nam-1.0a8/nam no_ds_no_mpls_huge_nam.nam &

        exec ../xgraph-12.1/xgraph no_ds_no_mpls_huge_udp.tr
no_ds_no_mpls_huge_send_side.tr no_ds_no_mpls_huge_recv_side.tr -
geometry 800x400

        }
```

```
proc proc_record {} {
        global tcp_agent_ftp sink_udp sink_tcp trace_file_snd_tcp
trace_file_rcv_tcp trace_file_udp  record_proc_call_interval

        global trace_file_tcp_bw_total

        set ns [Simulator instance]

        set time $record_proc_call_interval


      set bw_tcp_sent [$tcp_agent_ftp set ndatabytes_]
      #set bw_tcp_recv  [$sink_tcp set ndatabytesrecv_]

      set bw_udp       [$sink_udp set bytes_]
      #set total_tcp_bw [$sink_tcp set totalbytes_]

      set time_now [$ns now]

      #Calculate BW in Mbit/sec and write it to files.
        #puts $trace_file_rcv_tcp "$time_now [expr
$bw_tcp_recv/$time*8/1000000]"

      puts $trace_file_udp "$time_now [expr $bw_udp/$time*8/1000000]"

      puts $trace_file_snd_tcp "$time_now [expr
$bw_tcp_sent/$time*8/1000000]"
        #puts $trace_file_tcp_bw_total "$time_now [expr
$total_tcp_bw/$time*8/1000000]"



      $sink_udp set bytes_ 0
      $tcp_agent_ftp set ndatabytes_ 0
      #$sink_tcp set ndatabytesrecv_ 0

      #not setting the totalbytes_ variable because it should not be
reset to 0

      $ns at [expr $time_now + $time] "proc_record"

}

# main events

$ns at $record_time "proc_record"


$ns at $udp_start_time "if {$udp_active != 0} {$udp_agent_cbr start}"
$ns at $udp_end_time   "if {$udp_active != 0} {$udp_agent_cbr stop}"

$ns at $tcp_start_time "if {$tcp_active != 0} {$tcp_ftp_src start}"
$ns at $tcp_end_time "if {$tcp_active != 0} {$tcp_ftp_src stop}"

$ns at $finish_time "proc_finish"

# start the simulation

$ns run
```

Los resultados de esta simulación se muestran a continuación en la siguiente gráfica.

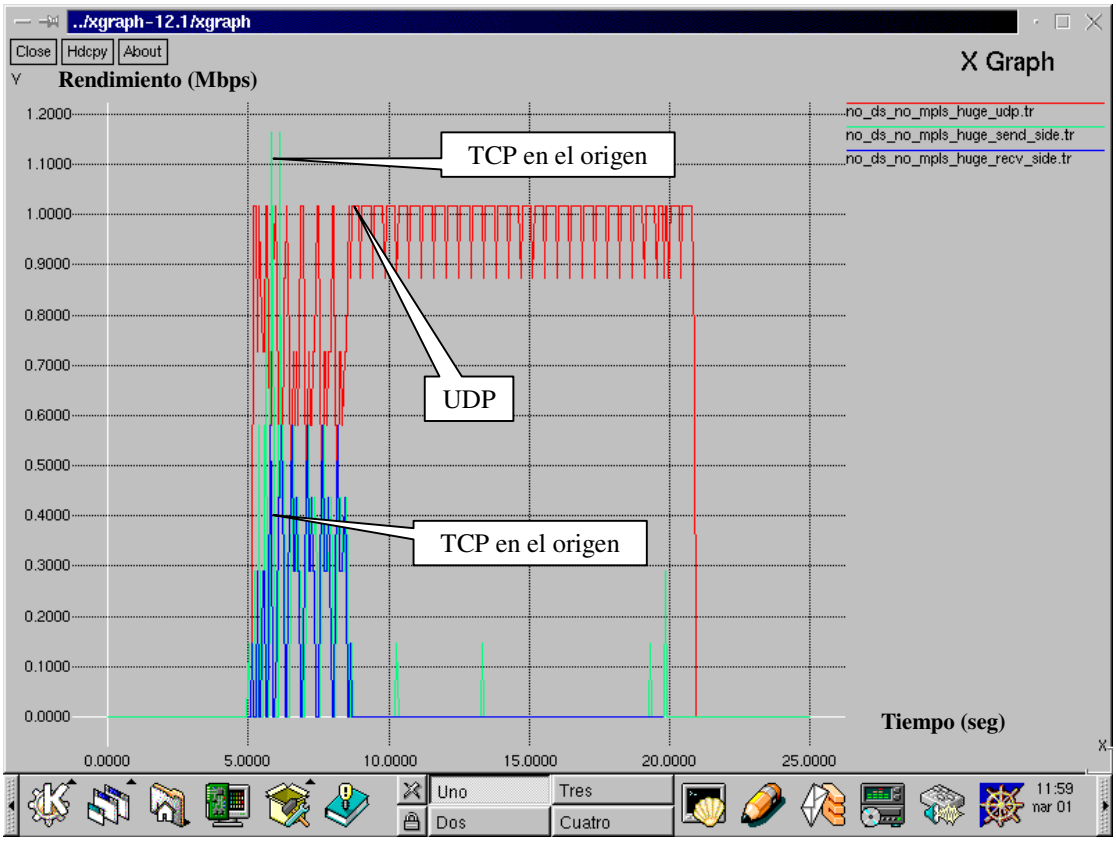

FIGURA A-1

SIMULACIÓN DE NO DIFFSERV - MPLS DE UN EXTRA MUY ALTO TRAFICO DE UDP

Los resultados de esta simulación se muestran a continuación en la siguiente gráfica.

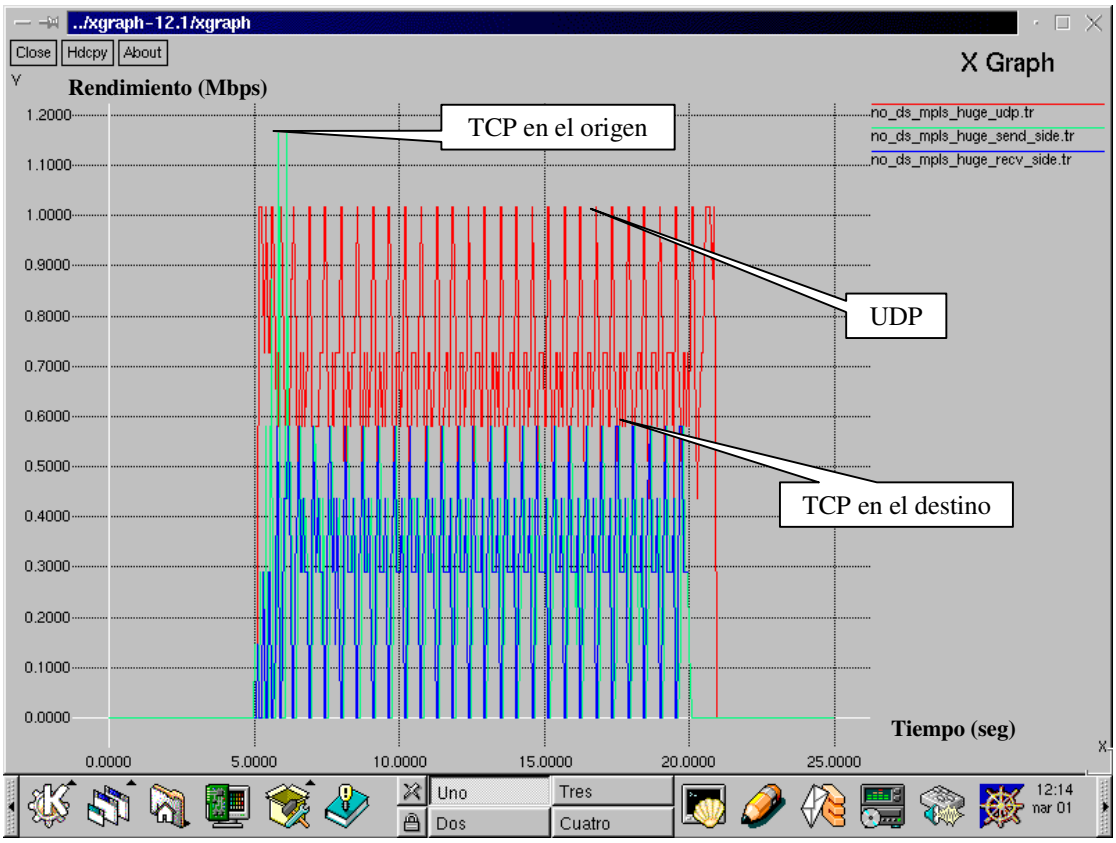
../xgraph-12.1/xgraph
Close
Hdcpy
About
X Graph
Rendimiento (Mbps)
TCP en el origen
UDP
TCP en el destino
no_ds_mpls_huge_udp.tr
no_ds_mpls_huge_send_side.tr
no_ds_mpls_huge_recv_side.tr
1.2000
1.1000
1.0000
0.9000
0.8000
0.7000
0.6000
0.5000
0.4000
0.3000
0.2000
0.1000
0.0000
0.0000
5.0000
10.0000
15.0000
20.0000
25.0000
Tiempo (seg)
Uno
Tres
Dos
Cuatro
12:14
mar 01

FIGURA A-2

SIMULACIÓN DE DIFFSERV – NO MPLS DE UN EXTRA MUY ALTO TRÁFICO DE UDP

Los resultados de esta simulación se muestran a continuación en la siguiente gráfica.

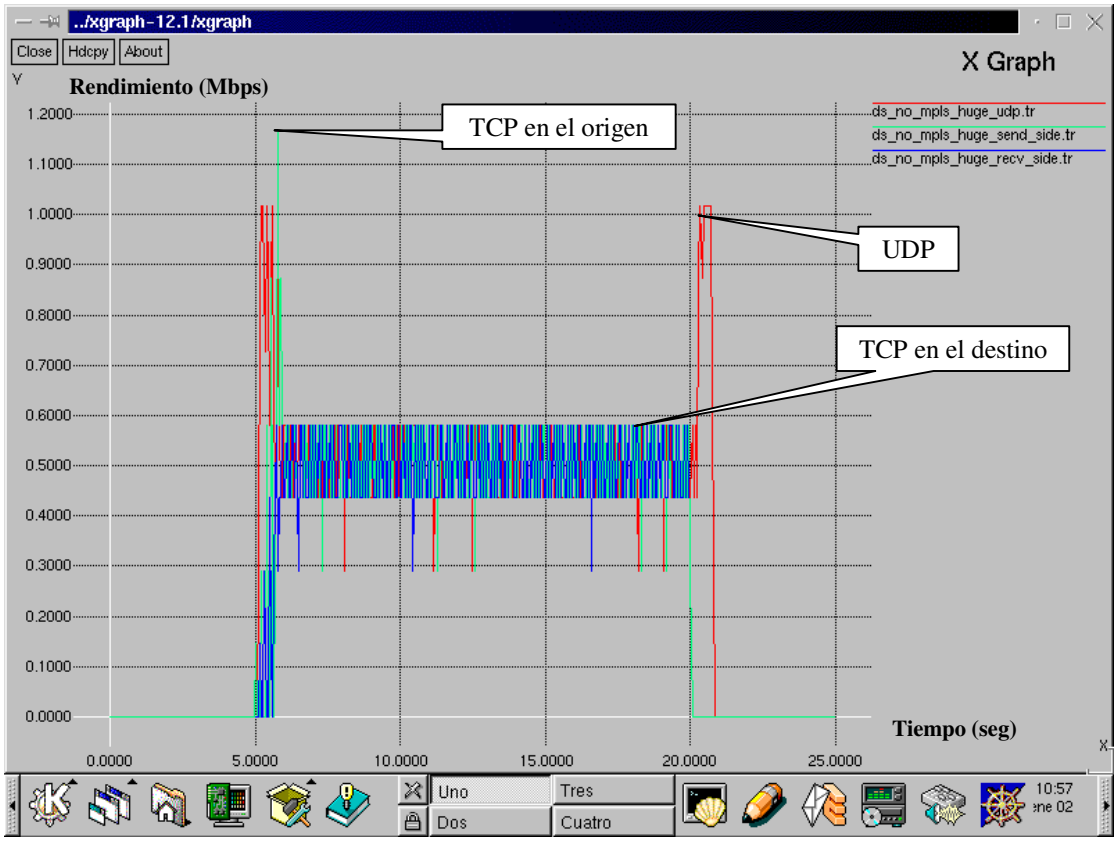

FIGURA A-3

SIMULACIÓN DE DIFFSERV – NO MPLS EN UN LSP DE UN EXTRA MUY ALTO TRÁFICO DE UDP

Los resultados de esta simulación se muestran a continuación en la siguiente gráfica.

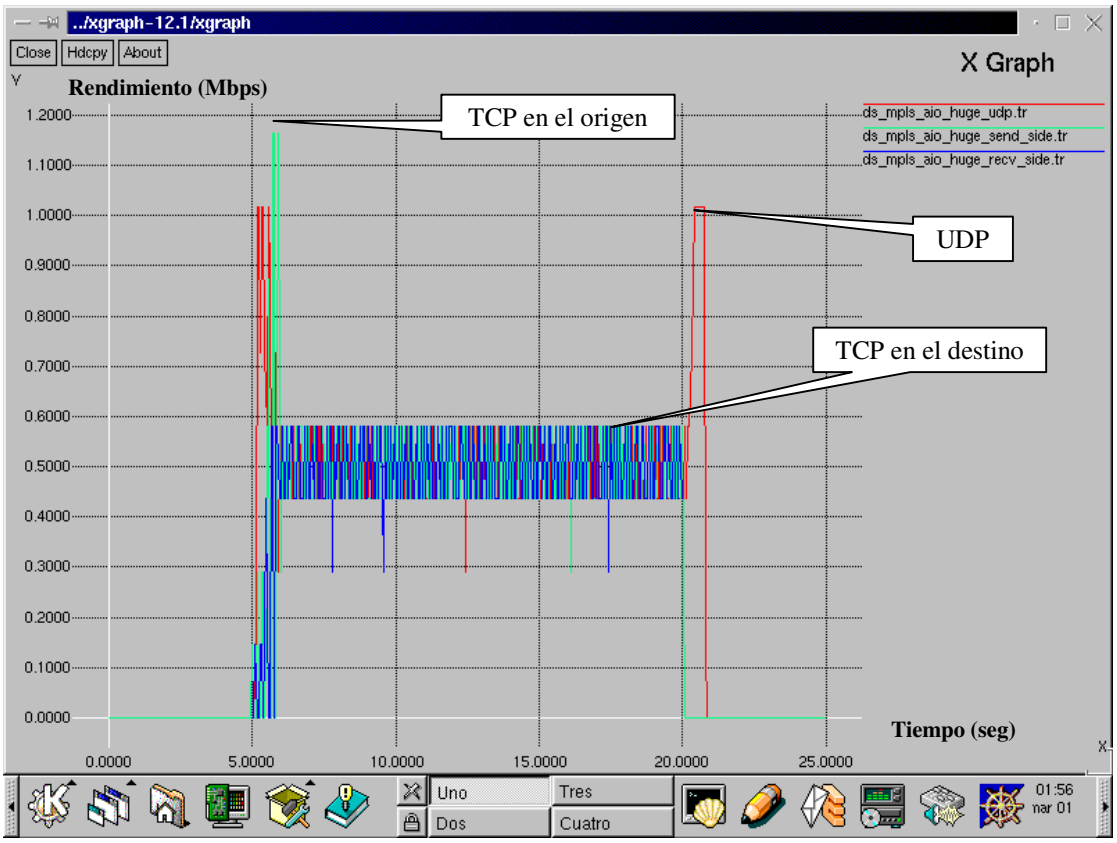

FIGURA A-4

SIMULACIÓN DE DIFFSERV – NO MPLS EN MULTIPLES LSPs DE UN EXTRA MUY ALTO TRÁFICO DE UDP

Los resultados de esta simulación se muestran a continuación en la siguiente gráfica.

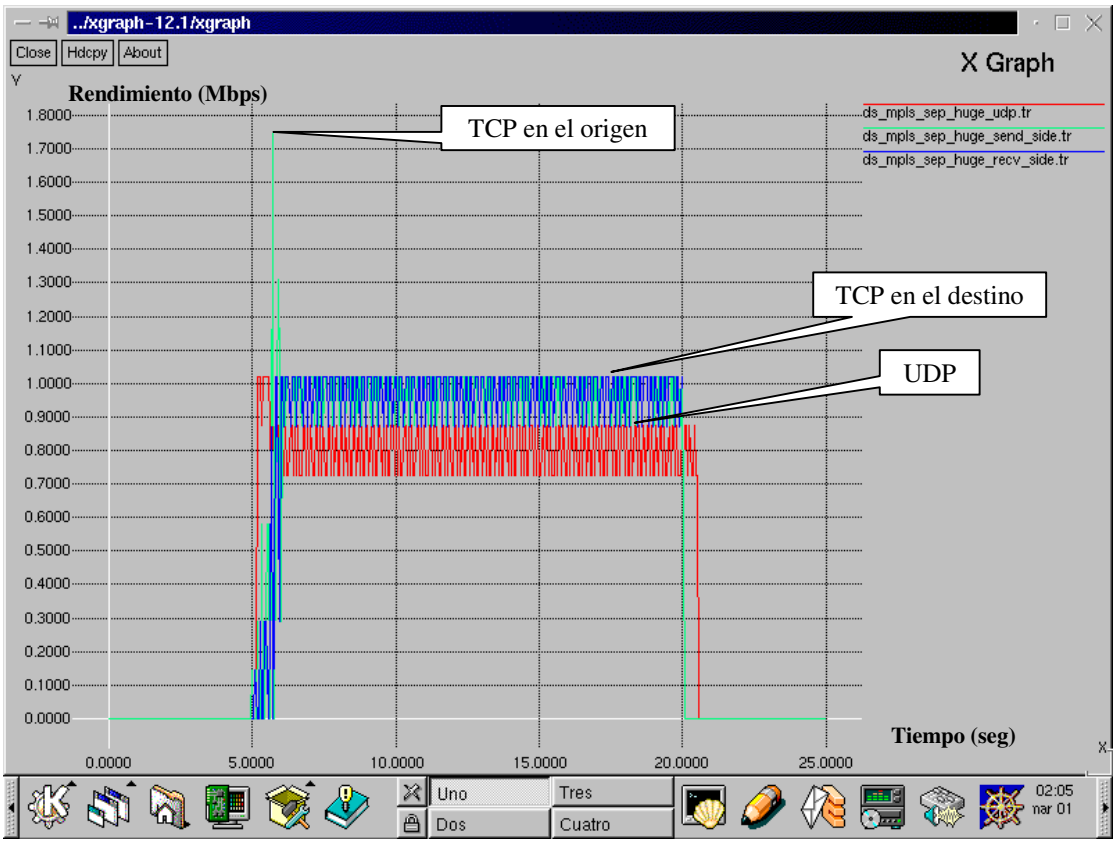

FIGURA A-5

Los resultados A-2 hasta A-5 son el resultado del desarrollo de código, que permite dar las condiciones de simulación de acuerdo al tipo de arquitectura.

Para las simulaciones efectuadas se tomó como referencia las propuestas de [GEIL01] y [SRGH02]. Se desarrollo por cada grafico un programa de simulación en NS haciendo un total de 5500 lineas de programación.

A continuación se presenta el código fuente en AWK para determinar la transferencia de información por el trafico TCP en el destino para No DiffSErv y No MPLS, procedimiento que se repite para los otros casos.

GENERACION DEL ARCHIVO PARA GRAFICACIÓN DEL THROUGHPUT PARA CADA EXPERIMENTO.

 # Autor: Ing. Rafael Bustamante

 # Universidad Nacional Mayor de San Marcos

 # Facultad de Ingeniería Electrónica

 # Maestría de Telecomunicaciones

```
BEGIN {
   highest_packet_id = 0;
   #inicio del contador de paquetes que salen del nodo1 al nodo3
   n = 0;
   #inicio del contador de paquetes que llegan al nodo12
   m = 0;
}

{
   action = $1;
   time = $2;
   node_1 = $3;
   node_2 = $4;
   src = $5;
   size_pqt = $6;
   flow_id = $8;
   node_1_address = $9;
   node_2_address = $10;
   seq_no = $11;
   packet_id = $12;

   if (flow_id == FID) {

      if ( action != "d" ) {
         if ( action == "r" ) {

             ;#if ( node_1 == 1 && node_2 == 3 ) {
              ;#  n = n + 1;
               ;# packet_start[n] = packet_id;
               ;# start_size_pqt[n]= size_pqt;
               ;# start_time[n] = time;
               ;# nn = n;
             ;#}
             if ( node_1 == 11 && node_2 == 12 ) {
                m = m + 1;
                packet_end[m] = packet_id;
                end_size_pqt[m] = size_pqt;

                end_time[m] = time;
                mm = m;
             }
         }
      }
   }
}
END {
```

```
    m = 1;
    p = 0;
    for ( i = 5.055; i < 20; i += 0.055 ) {
         while ( end_time[m] <= i ) {
             p = end_size_pqt[m] + p;
             m = m + 1;
         }
         printf("%f %f\n", i, 8*p/(0.055*1000000));
         ;#p= 0;
    }
}
```

El archivo destinatario del procesamiento del archivo TRACE que es el que graba todos los eventos durante la simulación, es graficado mediante XGRAPH.

El grafico correspondiente a cinco archivos procesados se presenta a continuación correspondiente a No DiffServ No MPLS.

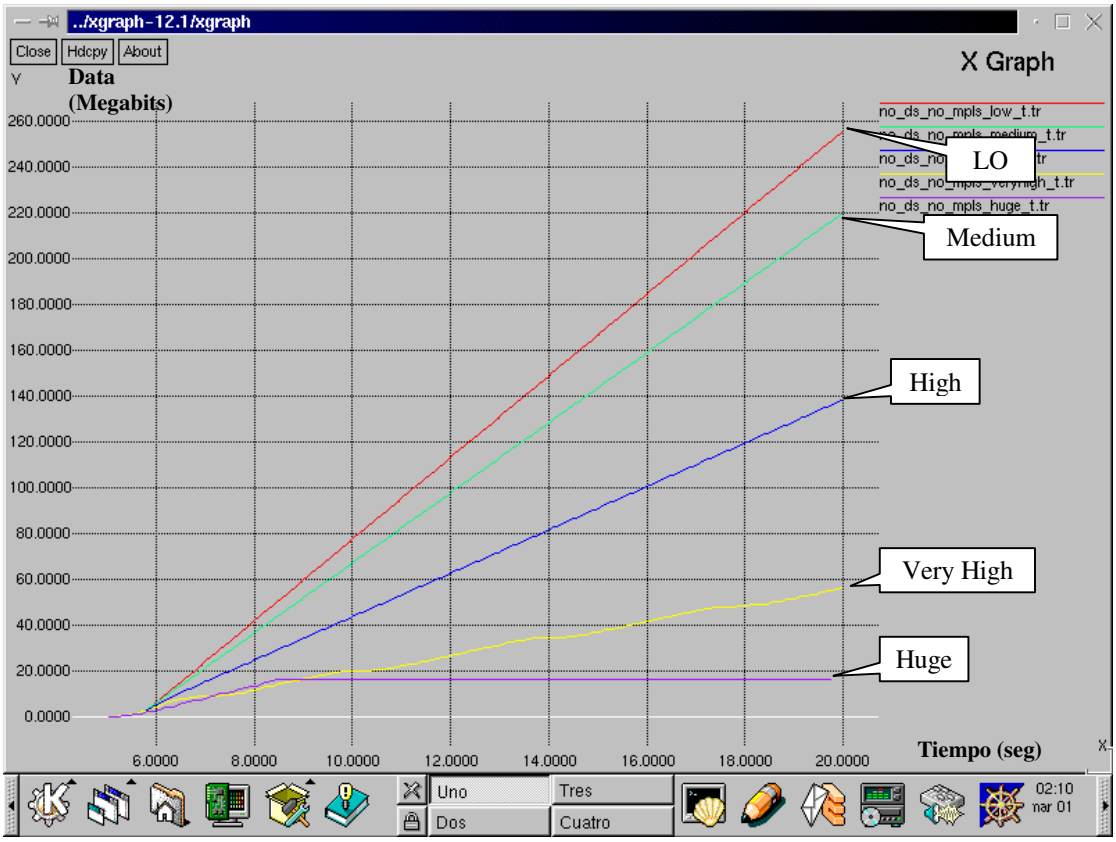

FIGURA A-6

NOTA: Los archivos TRACE es procesado mediante los programas desarrollados en AWK, cuyos resultados fueron graficados con el XGRAPH y exportados a WINDOWS y graficados con MATLAB.

APENDICE B

A partir de los archivos generados con el programa efectuado en AWK se grafica con el programa MATLAB tal comos se ha mostrado en el capitulo 5.

Lo primero que se presenta es el programa que genera la tabla de las estadísticas del rendimiento del tráfico en cada simulación. Aquí se presenta el código correspondiente a la simulación de No DiffServ No MPLS.

```
;% PROGRAMA QUE CALCULA LOS VALORES DE THROUGHPUT
%
================================================================================
==============================
;% AUTOR: Ing. RAFAEL BUSTAMANTE ALVAREZ
;%
;% TITULO: SIMULACION DE NO MPLS NO DIFFSERV
;%==============================================================================
==============================
;% LOW
[t11,BW11]=textread('no_ds_no_mpls_low_udp.tr','%f %f');
[t12,BW12]=textread('no_ds_no_mpls_low_send_side.tr','%f %f');
[t13,BW13]=textread('no_ds_no_mpls_low_recv_side.tr','%f %f');
;% MEDIUM
[t21,BW21]=textread('no_ds_no_mpls_medium_udp.tr','%f %f');
[t22,BW22]=textread('no_ds_no_mpls_medium_send_side.tr','%f %f');
[t23,BW23]=textread('no_ds_no_mpls_medium_recv_side.tr','%f %f');
;% HIGH
[t31,BW31]=textread('no_ds_no_mpls_high_udp.tr','%f %f');
[t32,BW32]=textread('no_ds_no_mpls_high_send_side.tr','%f %f');
[t33,BW33]=textread('no_ds_no_mpls_high_recv_side.tr','%f %f');
;% VERY HIGH
[t41,BW41]=textread('no_ds_no_mpls_veryhigh_udp.tr','%f %f');
[t42,BW42]=textread('no_ds_no_mpls_veryhigh_send_side.tr','%f %f');
[t43,BW43]=textread('no_ds_no_mpls_veryhigh_recv_side.tr','%f %f');
;% HUGE
[t51,BW51]=textread('no_ds_no_mpls_huge_udp.tr','%f %f');
[t52,BW52]=textread('no_ds_no_mpls_huge_send_side.tr','%f %f');
[t53,BW53]=textread('no_ds_no_mpls_huge_recv_side.tr','%f %f');
;%==============================================================================
==============================
;% CALCULOS ESTADISTICOS
;%==============================================================================
==============================
;% LOW
u11=mean(BW11(92:365));m11=median(BW11);ds11=std(BW11(92:365));mn11=mi
n(BW11);mx11=max(BW11);
u12=nanmean(BW12(92:365));m12=median(BW12);ds12=std(BW12(92:365));mn12
=min(BW12);mx12=max(BW12);
u13=nanmean(BW13);m13=median(BW13);ds13=std(BW13);mn13=min(BW13);mx13=
max(BW13);
;% MEDIUM
u21=mean(BW21(92:365));m21=median(BW21);ds21=std(BW21(92:365));mn21=mi
n(BW21);mx21=max(BW21);
u22=mean(BW22(92:365));m22=median(BW22);ds22=std(BW22(92:365));mn22=mi
n(BW22);mx22=max(BW22);
u23=mean(BW23);m23=median(BW23);ds23=std(BW23);mn23=min(BW23);mx23=max
(BW23);
;% HIGH
u31=mean(BW31(92:365));m31=median(BW31);ds31=std(BW31(92:365));mn31=mi
n(BW31);mx31=max(BW31);
u32=mean(BW32(92:365));m32=median(BW32);ds32=std(BW32(92:365));mn32=mi
n(BW32);mx32=max(BW32);
u33=mean(BW33);m33=median(BW33);ds33=std(BW33);mn33=min(BW33);mx33=max
(BW33);
;% VERY HIGH
u41=mean(BW41(92:365));m41=median(BW41);ds41=std(BW41(92:365));mn41=mi
n(BW41);mx41=max(BW41);
```

```
u42=mean(BW42(92:365));m42=median(BW42);ds42=std(BW42(92:365));mn42=mi
n(BW42);mx42=max(BW42);
u43=mean(BW43);m43=median(BW43);ds43=std(BW43);mn43=min(BW43);mx43=max
(BW43);
;% HUGE
u51=mean(BW51(92:365));m51=median(BW51);ds51=std(BW51(92:365));mn51=mi
n(BW51);mx51=max(BW51);
u52=mean(BW52(92:365));m52=median(BW52);ds52=std(BW52(92:365));mn52=mi
n(BW52);mx52=max(BW52);
u53=mean(BW53);m53=median(BW53);ds53=std(BW53);mn53=min(BW53);mx53=max
(BW53);
;%=================================================================
================================
;% MOSTRANDO LOS RESULTADOS EN TABLAS
;%=================================================================
================================
fprintf('  \n    ESTADISTICAS DEL RENDIMIENTO DE TRAFICO EN NO
DIFFSERV Y NO MPLS\n');
fprintf('\n               TRAFICO BAJO DE UDP\n');
fprintf('               Prom.     Mediana    Desv.Estd. Minimo
Maximo\n');
fprintf('   UDP CBR     %f  %f   %f   %f
%f\n',u11,m11,ds11,mn11,mx11);
fprintf('   TCP Fuente  %f  %f   %f   %f
%f\n',u12,m12,ds12,mn12,mx12);
fprintf('   TCP Destino %f  %f   %f   %f
%f\n',u13,m13,ds13,mn13,mx13);

fprintf('\n               TRAFICO MEDIO DE UDP\n');
fprintf('               Prom.     Mediana    Desv.Estd. Minimo
Maximo\n');
fprintf('   UDP CBR     %f  %f   %f   %f
%f\n',u21,m21,ds21,mn21,mx21);
fprintf('   TCP Fuente  %f  %f   %f   %f
%f\n',u22,m22,ds22,mn22,mx22);
fprintf('   TCP Destino %f  %f   %f   %f
%f\n',u23,m23,ds23,mn23,mx23);

fprintf('\n               TRAFICO ALTO DE UDP\n');
fprintf('               Prom.     Mediana    Desv.Estd. Minimo
Maximo\n');
fprintf('   UDP CBR     %f  %f   %f   %f
%f\n',u31,m31,ds31,mn31,mx31);
fprintf('   TCP Fuente  %f  %f   %f   %f
%f\n',u32,m32,ds32,mn32,mx32);
fprintf('   TCP Destino %f  %f   %f   %f
%f\n',u33,m33,ds33,mn33,mx33);

fprintf('\n               TRAFICO MUY ALTO DE UDP\n');
fprintf('               Prom.     Mediana    Desv.Estd. Minimo
Maximo\n');
fprintf('   UDP CBR     %f  %f   %f   %f
%f\n',u41,m41,ds41,mn41,mx41);
fprintf('   TCP Fuente  %f  %f   %f   %f
%f\n',u42,m42,ds42,mn42,mx42);
fprintf('   TCP Destino %f  %f   %f   %f
%f\n',u43,m43,ds43,mn43,mx43);

fprintf('\n               TRAFICO EXTRA MUY ALTO DE UDP\n');
fprintf('               Prom.     Mediana    Desv.Estd. Minimo
Maximo\n');
```

```
fprintf('   UDP CBR     %f  %f   %f   %f
%f\n',u51,m51,ds51,mn51,mx51);
fprintf('   TCP Fuente  %f  %f   %f   %f
%f\n',u52,m52,ds52,mn52,mx52);
fprintf('   TCP Destino %f  %f   %f   %f
%f\n',u53,m53,ds53,mn53,mx53);
```

ESTADISTICAS DEL RENDIMIENTO DE TRAFICO EN NO DIFFSERV Y NO MPLS

	TRAFICO BAJO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.000000	0.000000	0.000000	0.000000	0.000000
TCP Fuente	0.939084	0.872727	0.205699	0.000000	1.745455
TCP Destino	0.933467	1.018182	0.205779	0.000000	1.018182

	TRAFICO MEDIO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.158301	0.130909	0.024966	0.000000	0.261818
TCP Fuente	0.809025	0.727273	0.174618	0.000000	1.745455
TCP Destino	0.653214	0.727273	0.142818	0.000000	0.872727

	TRAFICO ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.471082	0.436364	0.074724	0.000000	0.785455
TCP Fuente	0.512807	0.436364	0.134181	0.000000	1.454545
TCP Destino	0.504562	0.581818	0.120922	0.000000	0.727273

	TRAFICO MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.782482	0.581818	0.183296	0.000000	1.018182
TCP Fuente	0.222429	0.000000	0.228253	0.000000	1.163636
TCP Destino	0.205128	0.145455	0.167814	0.000000	0.581818

	TRAFICO EXTRA MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.930060	0.872727	0.173724	0.000000	1.018182
TCP Fuente	0.072727	0.000000	0.180522	0.000000	1.163636
TCP Destino	0.060561	0.000000	0.142540	0.000000	0.581818

FIGURA B-1 Tabla de las estadísticas de No DiffServ No MPLS

A continuación se presenta el programa en MATLAB para la graficación de los archivos procesados,

```
;% PROGRAMA QUE CALCULA LOS VALORES DE THROUGHPUT
%
================================================================================
==============================
```

```
;% AUTOR: Ing. RAFAEL BUSTAMANTE ALVAREZ
;%
;% TITULO: SIMULACION DE NO MPLS NO DIFFSERV
;%=================================================================
================================
;% LOW
[t11,BW11]=textread('no_ds_no_mpls_low_udp.tr','%f %f');
[t12,BW12]=textread('no_ds_no_mpls_low_send_side.tr','%f %f');
[t13,BW13]=textread('no_ds_no_mpls_low_recv_side.tr','%f %f');
;% MEDIUM
[t21,BW21]=textread('no_ds_no_mpls_medium_udp.tr','%f %f');
[t22,BW22]=textread('no_ds_no_mpls_medium_send_side.tr','%f %f');
[t23,BW23]=textread('no_ds_no_mpls_medium_recv_side.tr','%f %f');
;% HIGH
[t31,BW31]=textread('no_ds_no_mpls_high_udp.tr','%f %f');
[t32,BW32]=textread('no_ds_no_mpls_high_send_side.tr','%f %f');
[t33,BW33]=textread('no_ds_no_mpls_high_recv_side.tr','%f %f');
;% VERY HIGH
[t41,BW41]=textread('no_ds_no_mpls_veryhigh_udp.tr','%f %f');
[t42,BW42]=textread('no_ds_no_mpls_veryhigh_send_side.tr','%f %f');
[t43,BW43]=textread('no_ds_no_mpls_veryhigh_recv_side.tr','%f %f');
;% HUGE
[t51,BW51]=textread('no_ds_no_mpls_huge_udp.tr','%f %f');
[t52,BW52]=textread('no_ds_no_mpls_huge_send_side.tr','%f %f');
[t53,BW53]=textread('no_ds_no_mpls_huge_recv_side.tr','%f %f');
;%=================================================================
================================
;% GRAFICANDO
;%=================================================================
================================
m1=ones(1,455);
m2=2*ones(1,455);
m3=3*ones(1,273);
t14=0:0.055:5;
z1=zeros(size(t14));
m4=3*ones(size(t14));
t15=20:0.055:25;
z2=zeros(size(t15));
m5=3*ones(size(t15));
;%=================================================================
================================
;% LOW
figure(1)
subplot(2,1,1),plot(t11,BW11,t12,BW12,t13,BW13);
;%plot(t11,BW11,t12,BW12,t13,BW13);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO BAJO DE UDP');
ylabel('Rendimiento ( Mbps )');
xlabel('Tiempo ( seg )');
grid;
subplot(2,1,2),plot3(m1,t11,BW11,m2,t12,BW12,m3,t13,BW13,m4,t14,z1,'r'
,m5,t15,z2,'r');
;%plot3(m1,t11,BW11,m2,t12,BW12,m3,t13,BW13,m4,t14,z1,'r',m5,t15,z2,'r
');
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS BAJO');
zlabel('Rendimiento ( Mbps )');
ylabel('Tiempo ( seg )');
xlabel('Traficos UDP= 1 TCP Fuente= 2 Destino= 3');
grid;
;%=================================================================
================================
;% MEDIUM
```

```
figure(2)
subplot(2,1,1),plot(t21,BW21,t22,BW22,t23,BW23);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO MEDIO DE
UDP');
ylabel('Rendimiento ( Mbps )');
xlabel('Tiempo ( seg )');
grid;
subplot(2,1,2),plot3(m1,t21,BW21,m2,t22,BW22,m3,t23,BW23,m4,t14,z1,'r'
,m5,t15,z2,'r');
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO MEDIO DE
UDP');
zlabel('Rendimiento ( Mbps )');
ylabel('Tiempo ( seg )');
xlabel('Traficos UDP= 1 TCP Fuente= 2 Destino= 3');
grid;

;%=================================================================
==============================
;% HIGH
figure(3)
subplot(2,1,1),plot(t31,BW31,t32,BW32,t33,BW33);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO ALTO DE UDP');
ylabel('Rendimiento ( Mbps )');
xlabel('Tiempo ( seg )');
grid;
subplot(2,1,2),plot3(m1,t31,BW31,m2,t32,BW32,m3,t33,BW33,m4,t14,z1,'r'
,m5,t15,z2,'r');
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO ALTO DE UDP');
zlabel('Rendimiento ( Mbps )');
ylabel('Tiempo ( seg )');
xlabel('Traficos UDP= 1 TCP Fuente= 2 Destino= 3');
grid;

;%=================================================================
==============================
;% VERY HIGH
figure(4)
subplot(2,1,1),plot(t41,BW41,t42,BW42,t43,BW43);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO MUY ALTO DE
UDP');
ylabel('Rendimiento ( Mbps )');
xlabel('Tiempo ( seg )');
grid;
subplot(2,1,2),plot3(m1,t41,BW41,m2,t42,BW42,m3,t43,BW43,m4,t14,z1,'r'
,m5,t15,z2,'r');
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO MUY ALTO DE
UDP');
zlabel('Rendimiento ( Mbps )');
ylabel('Tiempo ( seg )');
xlabel('Traficos UDP= 1 TCP Fuente= 2 Destino= 3');
grid;

;%=================================================================
==============================
;% HUGE
figure(5)
subplot(2,1,1),plot(t51,BW51,t52,BW52,t53,BW53);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO EXTRA MUY ALTO
DE UDP');
ylabel('Rendimiento ( Mbps )');
xlabel('Tiempo ( seg )');
```

```
grid;
m3=3*ones(size(BW53));
subplot(2,1,2),plot3(m1,t51,BW51,m2,t52,BW52,m3,t53,BW53,m4,t14,z1,'r'
,m5,t15,z2,'r');
axis([1 3 0 25 0 1.5]);
title('RENDIMIENTO DE NO-DIFFSERV-NO-MPLS EN UN TRAFICO EXTRA MUY ALTO
DE UDP');
zlabel('Rendimiento ( Mbps )');
ylabel('Tiempo ( seg )');
xlabel('Traficos UDP= 1 TCP Fuente= 2 Destino= 3');
grid;
```

APENDICE C

C.1. HERRAMIENTAS DE ADMINISTRACIÓN DE CONGESTION (Algoritmos)

Un método para manejar un OVERFLOW (la capacidad de las colas tiene un límite, ya que si se sobrepasa este límite se produce el OVERFLOW) de tráfico es el uso de algoritmos de colas, para clasificar el tráfico, y determinar algunos métodos de priorización de flujo de tráfico.

Existen las siguientes herramientas:

- First in-First out (FIFO) Queuing
- Priority Queuing (PQ)
- Custom Queuing (CQ)
- Flow Based -Weighted Fair Queuing (WFQ)
- Class Based -Weighted Fair Queuing (CBWFQ)

Estas herramientas solamente se usan cuando hay congestión, es decir, cuando se forman colas. En ausencia de congestión, todos los paquetes son distribuidos directamente a la interfaz de salida de un router.

C.2 FIFO (First In – First Out)

En su más simple forma, el encolamiento FIFO, involucra almacenar paquetes cuando la red está congestionada y enviarlas de acuerdo al orden de arribo cuando la red no está muy congestionada. FIFO es por defecto el algoritmo de encolamiento en algunas instancias; no requiere configuración sin embargo presenta varias desventajas. La más importante, es que el encolamiento FIFO no existe priorización en el tratamiento de los paquetes; el orden de arribo determina el Ancho de Banda, la prontitud en asignación

del BUFFER respectivo. No provee protección contra las aplicaciones que causan perturbación en el tráfico.

Las ráfagas pueden causar grandes retardos y causar problemas a las aplicaciones que son sensibles al retardo, y también potencialmente a los mensajes de control y señalización en la red. Un encolamiento FIFO presenta pérdidas de paquetes debido al OVERFLOW lo cual no es deseable porque los paquetes que se pierden podrían ser paquetes de alta prioridad; el encolamiento FIFO no puede evitar esta situación.

C.3 PQ (Priority Queuing)

PQ asegura que los tráficos importantes sean manejados, controlados y atendidos de la manera más rápida posible en cada nodo donde se use. PQ fue diseñado para dar una estricta prioridad a los tráficos más importantes. El encolamiento por prioridades puede flexiblemente priorizarse de acuerdo al protocolo de red (IP, IPX, APPLE TALK), interfaz de entrada, el tamaño del paquete, las direcciones de fuente o destino.

En PQ cada paquete es ubicado en uno de las cuatro colas: high, médium, normal y low, basado en la asignación de prioridades. Los paquetes que no son clasificados, se le considera en la cola de normal. Durante la transmisión PQ da prioridad primero a la cola High Priority sobre todas las demás.

PQ es útil para asegurar que una misión crítica de tráfico atraviese varios enlaces WAN donde recibirán un tratamiento de prioridad. Por ejemplo PQ asegura que los reportes de ventas de una empresa basadas en ORACLE sean transmitidos con mayor prioridad que otras aplicaciones menos críticas a través de una red WAN.

PQ esa una configuración estática y no se adapta automáticamente a los cambios de los requerimientos de la red..

C.4 CQ (Class Queuing)

CQ fue diseñada para permitir que varias aplicaciones en organizaciones compartan la red entre aplicaciones con un mínimo de Ancho de Banda o requerimientos de Retardo. En estos ambientes, el Ancho de Banda puede ser compartido proporcionalmente entre aplicaciones y usuarios. CQ se puede usar para proveer garantía de Ancho de Banda en un punto potencial de congestión de la red, asegurando el tráfico especificado (una porción de Ancho de Banda disponible), y dejando el resto de Ancho de banda para otro tráfico. El encolamiento CQ maneja el tráfico asignando una específica cantidad de espacio de cola para cada clase de paquete y entonces servirá las colas en la forma de Round Robin.

Ejemplo, el encapsulamiento de un SNA (Systems Network Architecture) requiere un mínimo nivel de servicio garantizado. Se puede reservar la mitad de Ancho de Banda disponible para la data del SNA y permitir que el remanente pueda ser usado por otros protocolos como IP e IPX (Internetwork Packet Exchange).

Otro ejemplo: el algoritmo de encolamiento ubica los mensajes en una de las 17 colas ver figura 2-10 (donde la cola está destinada para los mensajes de señalización entre otros) y es atendido según la prioridad que le corresponde. El router atiende desde la cola 1 hasta la cola 16 según (Round Robin) orden, disminuyendo el contador establecido en un byte en cada ciclo, es decir, por cada cola que es atendida (según ponderación) respectiva. Esta característica asegura que muchas aplicaciones consigan más de una porción determinada de la capacidad total de enrolamiento, por ejemplo cuando la línea está bajo fuerte presión por el tráfico.

Así como PQ; CQ, es estadísticamente configurado y no se adapta automáticamente a los cambios de las condiciones de la red.

C.5 FLOW – BASED WFQ (Flow Based -Weighted Fair Queuing).

Flow Based WFQ más comúnmente se refiere a WFQ que fue diseñado para situaciones en las cuales es deseable proveer una respuesta consistente en el tiempo es decir sin mucho jitter (variaciones de retardo) y sin añadir en forma excesiva Ancho de Banda a los flujos. WFQ es un algoritmo de encolamiento basado en flujos que emplea un concepto de equidad, el cual permite que cada flujo sea atendido equitativamente con respecto al número de bytes que contienen los paquetes de los flujos, en la figura 2.12 se ilustra el funcionamiento de WFQ.

WFQ asegura que las colas tengan un Ancho de Banda asignado y que el tráfico sea predecible al tratar de estabilizar los retardos.

El concepto de equidad que usa WFQ esta basado en los bits de IP Precedence para proveer servicio a las colas. WFQ trabaja con IP Precedence y RSVP para proveer: QOS diferenciados además de Servicios Garantizados, es decir, (Ancho de Banda y otros recursos en forma garantizada), la ponderación de las colas se basan en el IP – Precedence.

El algoritmo de WFQ aborda el problema del retardo en las aplicaciones interactivas, lo que hace de estas aplicaciones un tráfico más predecible. WFQ mejora los algoritmos tales como SNA Logical Link Control (LLC), la congestión del tráfico TCP y las características de lento inicio del tráfico. Todo esto trae como consecuencia un predecible Throughput y respuesta de tiempo con menos variaciones.

WFQ actúa con RSVP para asignar un espacio de buffer, planificación de paquetes y garantía de Ancho de Banda para la reserva de flujos).

WFQ fue diseñado para minimizar el esfuerzo de configuración, y adaptarse automáticamente al tráfico para cambiar las condiciones de tráfico.

C.6 CLASS BASED -WFQ (Class Based Weighted Fair Queuing)

CBWFQ permite a un administrador crear clases de mínimo de Ancho de Banda garantizadas. En lugar de proveer una cola para cada flujo individual, se define una clase, que consiste de uno o más flujos y donde cada clase puede ser garantizada con una mínima cantidad de Ancho de Banda.

Como ejemplo podríamos tener el caso de un video stream que requiere la mitad de un T1; el CBWFQ le asigna lo requerido y el resto, es decir, T1/ 2 es para los demás tipos de tráfico.

C.7 ADMINISTRACIÓN DE COLAS (Queue Management)

La administración de colas es necesaria porque la capacidad de las colas tiene un límite, ya que si se sobrepasa este límite se produce el OVERFLOW. Cuando una cola esta llena, un paquete más, ya no puede ingresar, entonces este paquete se perderá. Esta pérdida es previsible. Los routers no pueden evitarlo aún si se trata de paquetes considerados de alta prioridad. Entonces un mecanismo es necesario para hacer lo siguiente:

C.8 MECANISMOS DE ADMINISTRACIÓN DE COLAS

Los mecanismos para evitar la congestión son formas de administración de colas. Estas técnicas lo que hacen son monitorizar la carga de tráfico de la red en un esfuerzo para anticipar y evitar la congestión en los "cuellos de botellas", como oposición a las técnicas de Administración de Congestión que opera para controlar después que se ha producido la congestión.

Entre las principales técnicas para evitar la congestión se tiene el RED (Weighted Random Early Detection).

C.9 RED (Random Early Detection)

Es un algoritmo que fue diseñado para evitar la congestión antes que se convierta en un problema. RED trabaja monitorizando la carga de tráfico en los puntos de la red en los cuales estadísticamente se descartan paquetes cuando empieza a producirse la congestión. El resultado es que la pérdida de paquetes, es detectada por la fuente de tráfico, el cual empieza a disminuir la velocidad de transmisión de los paquetes.

RED fue diseñada principalmente para trabajar con TCP en IP y a partir de RED se han diseñado otras variantes de RED entre las cuales tenemos: WRED (Weighted Random Early Detection) y FRED (Flow Random Early Detection).

C.10 WRED (Weighted Random Early Detection)

Este algoritmo combina las capacidades de RED con IP Precedence. Esta combinación provee un tratamiento preferencial a los tráficos de paquetes de alta prioridad. Puede selectivamente descartar los paquetes correspondientes a tráficos de baja prioridad, cuando empieza la congestión, provee un tratamiento diferenciado a los tráficos, según las diferentes Clases de Servicio. WRED puede proveer servicios integrados de carga controlada..

C.11 FRED (Flow Random Early Detection)

WRED es principalmente usado para tráfico TCP en los cuales la velocidad de transmisión disminuirá si empieza a producirse una congestión. Pero cuando se trata de un tráfico que no es TCP, en este caso WRED no se puede aplicar, es en esta situación que FRED es usado.

FRED clasifica los flujos en función de las direcciones de origen y destino, mantiene los flujos activos, y asegura que cada flujo no consuma recursos del buffer más allá de los que puedan compartir. FRED determina que flujo monopoliza los recursos de buffer y lo penaliza.

APENDICE D

D.1 MIB (Management Information Base) Para MPLS

Actualmente se viene implementando los módulos respectivos para la MIB MPLS que permitirá realizarse por medio del protocolo SNMP (Simple Network Management Protocol), la gestión respectiva de las redes MPLS hasta el momento se tienen propuestos los módulos para la ingeniería de tráfico de MPLS (MPLS-TE), LSRs y para el soporte de gestión para el protocolo LDP según **RFC3812, RFC3813, RFC3815**.

D.2 BUCLES

Prácticamente en todos los protocolos existentes se pueden dar casos de bucles; por ejemplo en el periodo siguiente al fallo de un enlace [ROVI01]; dependiendo de la existencia del campo TTL (Time-To-Live: tiempo de vida) se actuará de distinta forma.

En el reenvío IP convencional, los paquetes tienen un campo TTL en la cabecera. Cuando el paquete atraviesa un router se decrementa dicho campo. Si el campo TTL llegara a valer cero se descarta el paquete. De esta forma tenemos un grado de protección contra los bucles que puedan existir debidos a fallos en enlaces, convergencias lentas de los algoritmos de encaminamiento (ejemplo RIP), etc.

En muchos casos MPLS actúa de la misma forma. Esto dependerá exclusivamente de cómo sea la cabecera MPLS. En el caso de que exista la cabecera genérica (denominada también cabecera shim), ésta tendrá el campo TTL. Si los valores de las etiquetas están codificados en una cabecera del nivel de enlace (por ejemplo, en el campo VPI/VCI de la cabecera ATM), los paquetes son reenviados por un

conmutador del nivel de enlace y si el nivel de enlace no tiene un campo TTL, entonces no se podrá actuar de esta forma.

A un segmento de un LSP que contenga una secuencia de LSRs que no puedan decrementar el campo TTL de un paquete, se le llama **segmento LSP no TTL**.

Cuando un paquete sale de un segmento LSP no TTL, el campo TTL deberá reflejar el número de saltos (LSRs) que ha atravesado. Para el caso unicast, se puede propagar la longitud a los LSRs de entrada, de tal forma que estos decrementen el valor TTL antes de reenviarlos al **segmento LSP no TTL**.

Si se dispone del campo TTL en la cabecera shim, se deberá inicializar éste copiándolo del campo TTL de la cabecera del nivel de red. Del mismo modo que en el reenvío IP convencional, se deberá decrementar en cada salto (LSR). Cuando el paquete salga del LSP se copiará el campo TTL de la cabecera shim al campo TTL de la cabecera del nivel de red.

En principio, el hardware de los componentes ATM no les permite decrementar el campo TTL, por lo que no habrá protección contra los bucles. Una posible solución a esto es la siguiente: muchos conmutadores ATM pueden limitar la cantidad de espacio del buffer del conmutador que puede consumir un circuito virtual. El objetivo durante los bucles transitorios es permitir que el encaminamiento reconverja, es decir, que las tablas de encaminamiento se estabilicen y el bucle desaparezca. Para conseguir esto hay que asegurarse que los routers no se saturen reenviando paquetes que estén metidos en un bucle. En un LSR ATM si los paquetes que pertenecen al bucle consumen sólo una pequeña cantidad del espacio del buffer del conmutador, entonces los conmutadores todavía podrán reenviar paquetes de actualización de encaminamiento, lo que garantizará una convergencia del encaminamiento.

Otra solución sería usar una técnica de detección de bucles. La técnica de detección de bucles es opcional y está especificada en MPLS-ATM y MPLS-LDP

D.3 AGREGACIÓN

La arquitectura MPLS define la agregación como el procedimiento mediante el que se asocia un única etiqueta a un unión de FEC's, que será a su vez una FEC en algún dominio una FEC en algún dominio y que aplica dicha etiqueta a todo el tráfico de la unión. La ventaja de la Agregación es que puede reducir la cantidad de etiquetas que se necesitan para manejar un conjunto particular de paquetes, y también para reducir la cantidad necesaria de tráfico de control de distribución de etiquetas, esto con referencia a los protocolos de señalización TE-RSVP y CR-LDP.

Un conjunto de FEC's pueden ser agregadas en cualquiera de las siguientes formas:

- Agregarlas en única FEC.
- Agregarlas en un conjunto de FEC's.
- No agregarlas.

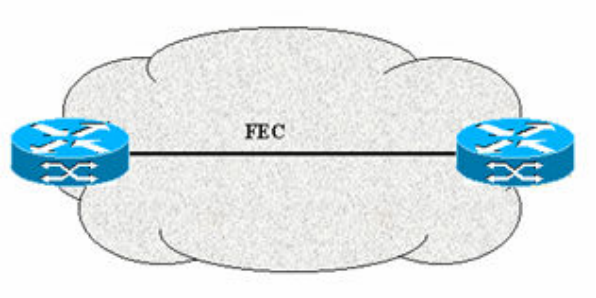

Figura D.1

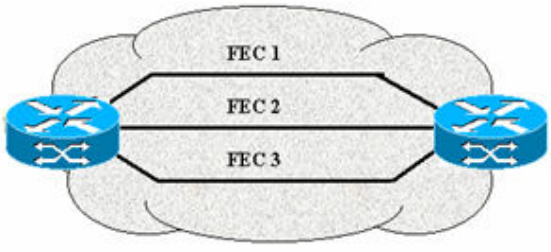

Figura D.2

Por tanto se puede hablar de un grado de granulado de la agregación o Granularidad de una Agregación.

D.4 GRANULARIDAD

Mencionaremos según el grado de granulado:

- Granulado grueso.
- Granulado fino.

D.4.1 GRANULADO GRUESO

Se presenta cuando un conjunto de FEC's son agregadas en una única FEC. Esta FEC puede incluir todos los paquetes en los que la dirección de destino del nivel de red coincidiera con un determinado prefijo de dirección. La ventaja del granulado grueso es que el sistema sería muy escalable. Por otro lado la desventaja sería que no se pude diferenciar diferentes tipos de tráfico y por tanto no permite clases de servicio ni operaciones de QoS.

D.4.2 GRANULADO FINO

Cuando un conjunto de FEC's no son agregadas, se presenta el granulado fino. En este caso una FEC podría incluir los paquetes pertenecientes a una aplicación entre dos ordenadores, es decir paquetes que tengan las mismas direcciones de origen y destino, los mismos puertos e incluso la misma clase de servicio.

En este caso tendríamos más clasificaciones de tráfico, mas FEC's más etiquetas, y una tabla de encaminamiento más grande. Por otro lado el sistema sería menos escalable, por que hay mayor cantidad de etiquetas a procesar en cada nodo.

En consecuencia una red de conmutación de etiquetas permite distintos grados de granularidad de la FEC.

APENDICE E

1. ESTADÍSTICAS COMPLEMENTARIAS A LAS SIMULACIONES PARA EL CASO DE UNA RED DUMMBELL

ESTADISTICAS DEL RENDIMIENTO DE TRAFICO EN DIFFSERV-MPLS CON MULTIPLES LSPs

	TRAFICO BAJO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.000000	0.000000	0.000000	0.000000	0.000000
TCP Fuente	0.939084	0.872727	0.205699	0.000000	1.745455
TCP Destino	0.933467	1.018182	0.205779	0.000000	1.018182
	TRAFICO MEDIO DE UDP				

TABLA E.1

En la tabla E.1 observamos el resultado de MPLS con DiffServ para LSPs múltiples LSPs cuya topología se muestra en la figura E.1

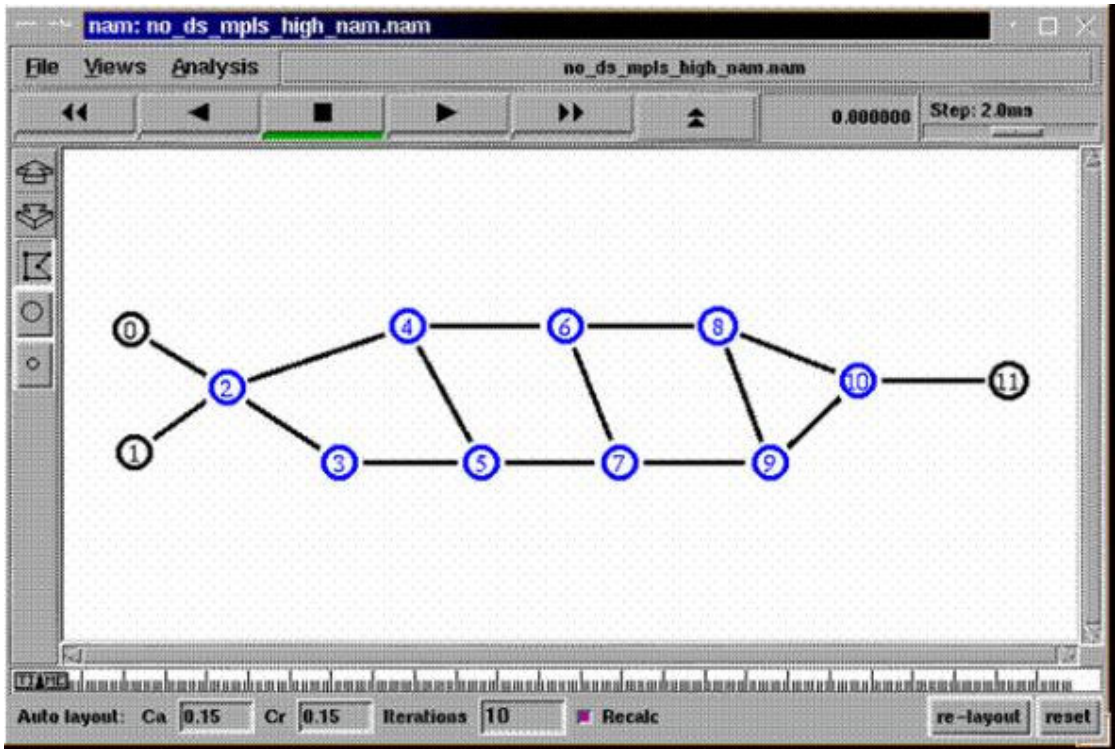

Figura E.1 topología alternativa para LSP's alternativas de la red MPS con DiffServ.

En este caso se ha usado la red basada en la topología de Dumbbell con múltiples rutas posibles y a pesar de ello los resultados de la simulación no variaron por tanto; se pude seguir afirmando que con MPLS y DiffServ con múltiples rutas es la mejor alternativa con referencia al rendimiento.

2. RESULTADOS DE LA SIMULACIÓN DE UNA RED CON SOLO LA ARQUITECTURA MPLS, BAJO LAS CONDICIONES PROPUESTA EN LA TESIS PARA UNA RED EN ANILLO

ESTADISTICAS DEL RENDIMIENTO DE TRAFICO EN NO DIFFSERV Y MPLS					
	TRAFICO BAJO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.000000	0.000000	0.000000	0.000000	0.000000
TCP Fuente	0.939084	0.872727	0.205699	0.000000	1.745455
TCP Destino	0.933467	1.018182	0.205779	0.000000	1.018182
	TRAFICO MEDIO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.158301	0.130909	0.024685	0.000000	0.261818
TCP Fuente	0.809025	0.727273	0.174618	0.000000	1.745455
TCP Destino	0.802397	0.872727	0.174210	0.000000	0.872727
	TRAFICO ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.315541	0.290909	0.049410	0.000000	0.581818
TCP Fuente	0.660385	0.581818	0.145203	0.000000	1.454545
TCP Destino	0.653214	0.727273	0.142818	0.000000	0.872727
	TRAFICO MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.779827	0.581818	0.181854	0.000000	1.018182
TCP Fuente	0.222960	0.000000	0.235923	0.000000	1.163636
TCP Destino	0.207259	0.290909	0.171631	0.000000	0.581818
	TRAFICO EXTRA MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.708162	0.581818	0.190703	0.000000	1.018182
TCP Fuente	0.290378	0.000000	0.204001	0.000000	1.163636
TCP Destino	0.281319	0.290909	0.180920	0.000000	0.581818

Tabla E 2

Comentario:

Se observa que los valores obtenidos son similares a los valores obtenidos a los resultados de la simulación de una red sin la arquitectura MPLS ni DiffServ. Excepto en caso del nivel de trafico Muy Alto de UDP en cual si se observa un cambio, con un incremento del trafico UDP y también del tráfico TCP que como se puede observar con respecto a la tabla correspondiente al caso donde MPLS y Diffserv son deshabilitados, en este caso el tráfico TCP presenta un incremento de 0.3 Mbps de trafico permanente.

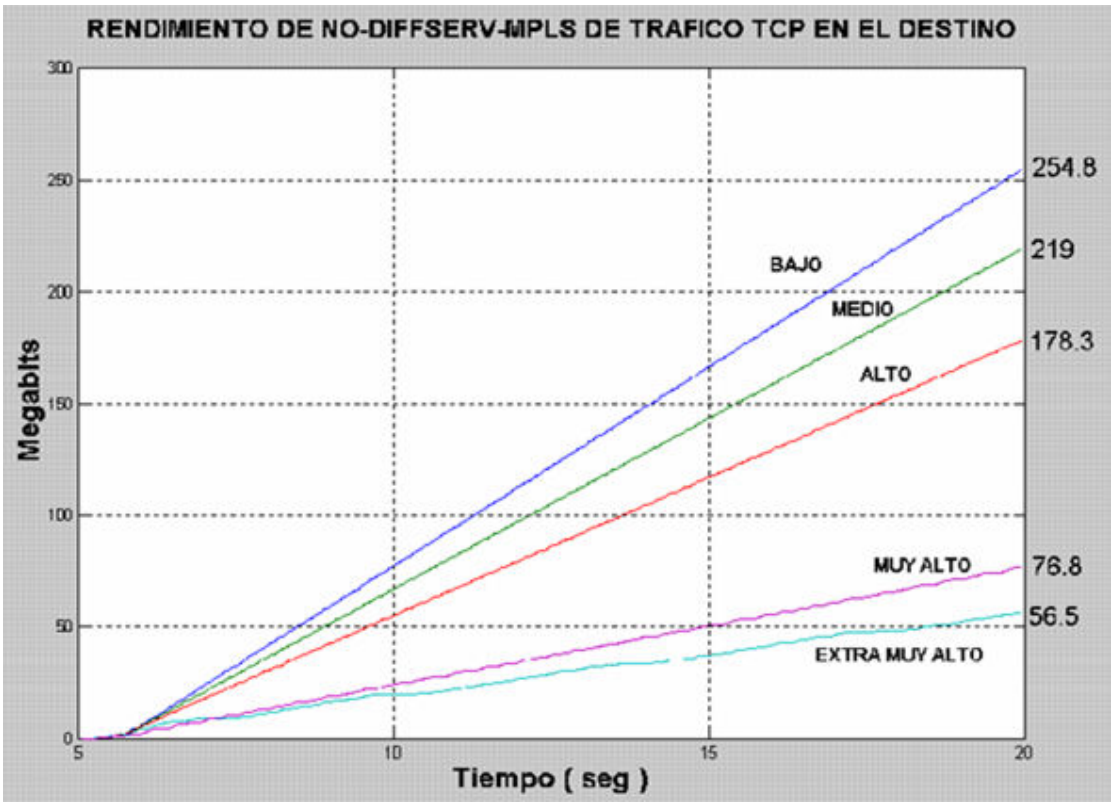

Figura E 2

Se muestra los resultados para un throughput del trafico TCP obtenido en el nodo destino. Se espera, que no haya diferencia significativa como en el caso de la figura 5.3.6 que corresponde a MPLS y DiffServ deshabilitados. Los incrementos en el ancho de banda para el nivel Extra Muy Alto de tráfico UDP puede ser atribuido al uso de MPLS basado en la conmutación de la capa 2 el cual es mas eficiente que el encaminamiento (routing) en la capa 3, ya que solo se analizan de las etiquetas en las cabeceras de los paquetes que circulan en una red con arquitectura MPLS. El cual se puede ver con el incremento de la transferencia de información mediante TCP para el caso de tráfico extra muy alto.

3. RESULTADOS DE LA SIMULACIÓN DE UNA RED CON SOLO LA ARQUITECTURA DIFFSERV, BAJO LAS CONDICIONES PROPUESTA EN LA TESIS PARA UNA RED EN ANILLO

ESTADISTICAS DEL RENDIMIENTO DE TRAFICO EN DIFFSERV Y NO MPLS					
	TRAFICO BAJO DE UDP				
UDP CBR	0.000000	0.000000	0.000000	0.000000	0.000000
TCP Fuente	0.939084	0.872727	0.206699	0.000000	1.745466
TCP Destino	0.933467	1.018182	0.205779	0.000000	1.018182
	TRAFICO MEDIO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.158514	0.145455	0.023901	0.000000	0.247273
TCP Fuente	0.809025	0.727273	0.175944	0.000000	1.745455
TCP Destino	0.802397	0.872727	0.174210	0.000000	0.872727
	TRAFICO ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.475541	0.436364	0.074190	0.000000	0.741818
TCP Fuente	0.508029	0.436364	0.124649	0.000000	1.163636
TCP Destino	0.500300	0.581818	0.118080	0.000000	0.727273
	TRAFICO MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.505640	0.436364	0.097473	0.000000	1.018182
TCP Fuente	0.489980	0.436364	0.126893	0.000000	1.163636
TCP Destino	0.481652	0.436364	0.112439	0.000000	0.581818
	TRAFICO EXTRA MUY ALTO DE UDP				
	Prom.	Mediana	Desv.Estd.	Minimo	Maximo
UDP CBR	0.509091	0.436364	0.117616	0.000000	1.018182
TCP Fuente	0.488918	0.436364	0.128549	0.000000	1.163636
TCP Destino	0.480586	0.436364	0.117589	0.000000	0.581818

Tabla E 3

El resultado mostrado en la tabla E 3 indica el beneficio del tráfico TCP en DiffServ. El valor de la promedio en la tabla E 3 especifica el valor del throughput. Comparando con el valor del promedio de la tabla E.2 (MPLS, no DiffServ) y la tabla E 3 (No MPLS, DiffServ), para los niveles de trafico de Muy Alto y Muy Extra Alto de UDP se observa que hay cerca de un 50 % que se incrementa en el throughput de TCP observandose un incremento del Throughput en la red Diffserv. Comparando el resultado de la tabla 7.3.1 (No MPLS y No Diffserv) con la tabla E 3, se observa cerca de un 170 % de incremento de throughput de TCP.

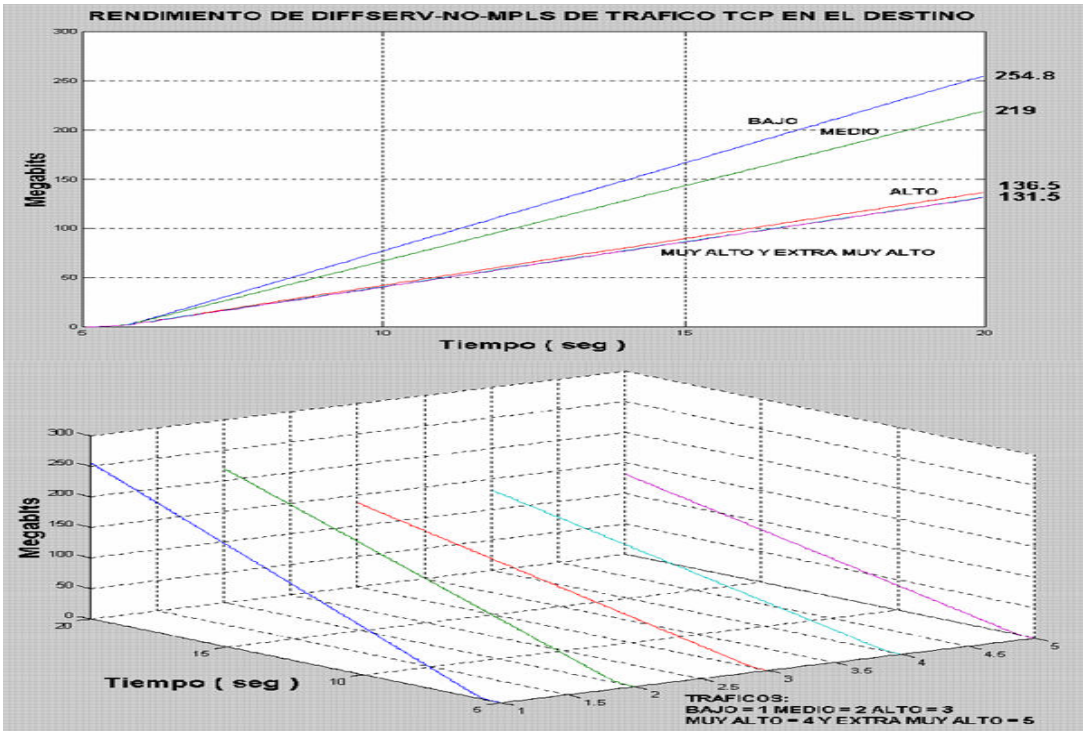

Figura E.3

Comentario:

La figura E.3 muestra el total de throughput del tráfico TCP que resulta de una red donde solamente tiene la arquitectura DiffServ. Se observa que el total de TCP throughput en el destino en el experimento con el tráfico UDP tipo Muy Alto y Extra Muy Alto es considerablemente incrementado comparado con los resultados de una red donde no existe una arquitectura DiffServ. La configuración de DiffServ en la simulación da igual ancho de banda tanto para TCP y UDP. La configuración de Diffserv puede ser alterado para dar preferencia al tráfico TCP, así se puede incrementar su throughput.

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.