PONTIFICIA **UNIVERSIDAD CATÓLICA** DEL PERÚ

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

MEDICIÓN Y ANÁLISIS DE TRÁFICO EN REDES MPLS

TESIS PARA OPTAR EL TÍTULO DE

INGENIERO DE LAS TELECOMUNICACIONES

PRESENTADA POR

Javier Igor Doménico Luna Victoria García

RESUMEN

La presente tesis se inicia con el estudio de los parámetros más resaltantes de MPLS (Multi Protocol Label Switching) tecnología que nos permite afrontar los múltiples requerimientos que las nuevas aplicaciones necesitan, en especial, las aplicaciones en línea denominadas en tiempo real bajo los esquemas que la ITU-T recomienda.

Durante el desarrollo de esta tesis se definen escenarios de redes IP; a los cuales serán sometidos a diversos tráficos; se evaluaran los comportamientos resultantes de la interacción con estos tráficos y se comprobará la mejor alternativa tecnológica para proporcionar QoS, Ingeniería de Tráfico, transmisión óptima de información, uso de recursos de red, entre otras características que son de interés.

INDICE

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO 1

INTRODUCCIÓN

1.1. Planteamiento de la tesis

En los últimos años, las Telecomunicaciones, en especial las tecnologías vinculadas a la Internet, han alcanzado un crecimiento y auge mayor al que se hubiese podido esperar en sus principios. En la actualidad, la Internet se caracteriza por un raudo crecimiento y una gran demanda de nuevos servicios interactivos en tiempo real mucho más sofisticados; prestando mayor atención, aquellos involucrados con transmisión de voz y video (VoIP y VTC), los que, a diferencia de los servicios clásicos, requieren alto grado de uso de recursos de la red.

La Internet actual funciona bajo el protocolo IPv4 el cual fue diseñado bajo el esquema Best Effort, en el cual se proporciona una mayor valorización por el servicio de acceso y distribución de contenidos más que por el servicio de transporte de datos [10].

Este esquema Best Effort se caracteriza por presentar un bajo nivel de rendimiento, el cual se refleja en la lentitud de las trasmisiones, pérdidas de información, pérdidas de conexión y graves casos de congestión. A pesar de que se proporciona a las aplicaciones y servicios clásicos (Telnet, FTP, Correo Electrónico, entre otros) un esquema en el que pueden funcionar de manera adecuada, es perjudicial para las nuevas aplicaciones ya que no permite proporcionar calidad de servicio necesario para su funcionamiento.

Ante estos inconvenientes, la Internet ha sufrido cambios para poder adaptarse a los nuevos requerimientos de los servicios por parte de los clientes, como soporte a aplicaciones de gran ancho de banda con el uso de tecnologías de capa de enlace tales como Frame Relay y ATM (Asynchronous Transfer Mode), sesiones seguras en la red pública, entre otros. Todos estos y otros factores, dependiendo de la aplicación, conforman un escenario en el cual IPv4 no fue concebido.

Esto tuvo como consecuencia una sobrecarga de procesamiento, mal uso de recursos, problemas de incompatibilidad con tecnologías de capa de enlace, necesidad de ingeniería de tráfico, redefinición de algoritmos de enrutamiento, entre otros, lo cual es percibido por el cliente como un rendimiento pobre de la red.

Otros problemas que se presentaron fueron:

- Falta de compatibilidad y adaptación con los diferentes protocolos de capa de enlace ya que estos últimos no fueron concebidos bajo el esquema de IP.
- El crecimiento desmesurado de usuarios. Las direcciones IP que nos facilita la comunicación a través de Internet se agotan rápidamente. El uso de un Server NAT (Network Address Translation) como un intérprete y/o traductor de direcciones privadas-públicas y viceversa trajo muchos más problemas que beneficios [15].
- Ingeniería de Tráfico, concepto el cual los protocolos de enrutamiento de la Internet no implementan; por lo que contribuyen a agravar el problema de congestión.
- Necesidad de Calidad de Servicio (QoS) por parte de las distintas aplicaciones que empezaron a surgir en los últimos años.

Como una solución a estos problemas se crea el protocolo IPv6, evolución del protocolo IPv4, un protocolo perfectamente capaz de proporcionar mayores facilidades y/o funcionalidades requeridas por las aplicaciones. Además, provee de nuevas herramientas, lo que proporciona simplicidad al modelo de red, mayor número de direcciones IP, compatibilidad con nuevos protocolos que podían mejorarlo, arquitecturas que proveen Calidad de Servicio (QoS), cabecera fija, entre otros; a todas estas nuevas funcionalidades se suma el hecho de que proporciona simplicidad, dado que las herramientas anteriormente mencionadas no implican que el modelo se vuelva más complejo.

A pesar de lo antes citado, muchas entidades no han adoptado todavía el nuevo protocolo de Internet IPv6 y algunas otras están en un proceso lento de adopción de este protocolo a pesar que entró en vigencia hace aproximadamente una década (IPv6 fue propuesto en 1998 como solución al problema de la escasez de direcciones del protocolo IPv4). La experiencia indica que las empresas proveedoras de Internet poseen muchos equipos e infraestructura diseñada para soportar IPv4, por lo cual son pocos los equipos capaces de soportar el cambio a IPv6, ya que la gran mayoría no fueron concebidos para este nuevo protocolo. Sólo los equipos más recientes están disponibles para este cambio de protocolo.

En el año 2000 se propone una nueva arquitectura de red llamada MPLS (Multi Protocol Label Switching), pensada para solucionar muchos problemas de la Internet actual: calidad de servicio, ingeniería de tráfico, entre otros. Sus características se discutirán en el capítulo correspondiente.

Gracias a MPLS se pueden afrontar los siguientes problemas [14]:

- La migración de IPv4 a IPv6 de manera progresiva sin los problemas citados.
- Los proveedores pueden adoptar este protocolo de manera más sencilla.
- Alta compatibilidad con tecnologías de capa 2 como ATM y Frame Relay.
- Ingeniería de tráfico y QoS.
- Capacidad de proporcionar un envío rápido de la información.
- Capacidad de definir VPN (Virtual Private Network).
- Compatibilidad con los protocolos de capa de enlace.

De los argumentos anteriores surge la necesidad de disponer de una evaluación de las redes que soportan la tecnología MPLS y los protocolos que esta tecnología usa para poder resolver los citados problemas.

1.2. Objetivos de la tesis

- **Simular una topología que soporte MPLS VPN**

  Se establecen diferentes escenarios MPLS utilizando el simulador OMNET++, en el cual se medirá el rendimiento de esta red, así también su comportamiento con diferentes tipos de tráfico relacionado con UDP y TCP. Los parámetros de medición con los cuales se contrastarán los resultados son los propuestos en las recomendaciones de la ITU-T. Resulta de interés la comparativa de resultados con una red real de ser posible.

- **Medir y analizar los parámetros que definen la QoS en una red MPLS**

  Las conclusiones de la presente tesis ofrecen recomendaciones para los usuarios y los proveedores de Internet para ofrecer mayor QoS a las recientes aplicaciones en tiempo real utilizando redes MPLS.

- **Proporcionar un estudio base para futuros trabajos**

  Se desea que esta tesis sea el punto de partida para futuras tesis que se relacionen con MPLS, Calidad de Servicio (QoS), así como nuevos servicios que puedan ofrecerse en redes que utilicen esta tecnología para el envío de los paquetes y que necesiten un grado de QoS determinado.

1.3. Justificación e importancia de la tesis

La importancia de la presente tesis se hace evidente cuando se empieza a comparar las diferentes soluciones para redes IP. Aunque el protocolo IPv6 nos da muchas funcionalidades y características que favorecen al envío del tráfico, estos no son suficientes para poder hacer un correcto uso de los recursos de la red.

A pesar de haberse definido dos arquitecturas basadas en IPv6, ninguna de estas llega a satisfacer las necesidades actuales de las aplicaciones. En la primera arquitectura, Servicios Integrados, se reservan recursos en los nodos de la red usando de señalización, lo cual deteriora el rendimiento de la red. En la segunda arquitectura, Servicios Diferenciados, se tiene un comportamiento en base a los bits DS pero no de forma garantizada ya que no se especifican las necesidades del tráfico a cursar sin mencionar que los routers no intercambian información sobre los tráficos (se asume

que todos están configurados de la misma manera y que presentarán el mismo comportamiento). Además ninguno de los anteriores modelos ofrece Ingeniería de Tráfico por lo que, en presencia de alto tráfico, habría sub-utilización de enlaces.

MPLS surge como la mejor opción que se puede encontrar actualmente para el manejo del backbone de Internet por sus diferentes características: ingeniería de tráfico, bajo costo de implementación, así como adaptación con tecnologías de capa de Enlace y de Red.

Uno de los objetivos de la tesis es respaldar estos conocimientos con datos obtenidos en simulaciones de redes MPLS, que permitan cuantificar la calidad de la red ante cierta topología y eventos que puedan ocurrir en ella: su manejo del tráfico, su capacidad de crear nuevos caminos o VPN en caso de fallas, entre otros. Como se indicó, esto puede hacerse tanto con routers reales cuyos sistemas operativos tengan la capacidad de manejar MPLS o a través de herramientas de simulación tales como NS2, OpensimMPLS, OMNET entre otras herramientas basadas en software libre.

En la presente tesis se empleará el simulador OMNET, simulador de eventos discretos orientado a objetos y ampliamente utilizado por universidades europeas por su flexibilidad, modularidad jerárquica, capacidad de simular arquitecturas complejas, interfaz gráfica de simulación intuitiva, capacidad multiplataforma y de fácil modificación. OMNET está programado bajo librerías de simulación basadas en C++.

CAPÍTULO 2

DESCRIPCIÓN Y OPERACIÓN DE LAS TECNOLOGÍAS

En este capítulo se explica de forma detallada las tecnologías que serán usadas en la ejecución de esta tesis. De esta forma, se detallan las características de las tecnologías, los beneficios que posee cada una, así como los problemas que se pretenden superar con el uso de estas tecnologías en la Internet actual.

2.1. Internet Protocol version 6 (IPv6)

2.1.1. Protocolo de Internet: Escenario Actual

En la Internet actual se presentan muchos problemas con el actual protocolo IPv4 a nivel de capa de red. Si bien en sus comienzos resultó ser un esquema adecuado y robusto dado que las aplicaciones no necesitaban un gran ancho de banda, no requerían una ingeniería de tráfico, solo se tenía pocos nodos capaces de generar tráfico en la red; pero con el pasar del tiempo, el desarrollo y la convergencia a la que tienden los múltiples servicios clásicos y los nuevos servicios que han

aparecido, en su mayoría servicios interactivos, hace que el escenario inicial en el que se planteó IPv4 no sea suficiente para satisfacer la demanda actual de QoS.

Actualmente para las aplicaciones y servicios interactivos, en especial aquellas que son en línea como voz y video, no encuentran la calidad de servicio necesaria para poder funcionar adecuadamente. El resultado se aprecia en un bajo rendimiento de la red y una calidad en las transmisiones de los servicios en línea muy baja. Los problemas que se presenta en la actual Internet con el protocolo IPv4 son:

- El escenario de Internet no es adecuado para las aplicaciones en tiempo real. El escenario en el cual trabaja IPv4, Best Effort, el cual se caracteriza por dar una mayor valorización por el servicio de acceso y distribución de contenidos que por el servicio de transporte de datos [6]. El modelo de servicio para el IPTC (IP transfer capability) de tipo Best Effort requiere que se utilicen los recursos disponibles para el reenvío de los paquetes de los flujos, en lo cual no hay ningún compromiso de QoS especificado ni garantías, sólo se tiene la expectativa de que se entregarán los paquetes siempre que estén disponibles suficientes recursos; un escenario que con poca frecuencia se da en las backbones de Internet [9].

- El tamaño variable de la cabecera de IPv4 afecta la performance de la red. El tamaño de la cabecera del protocolo IP, al ser variable, origina un procesamiento adicional, contribuyendo con retardos en cada routers. Tanto la congestión como el retardo son factores que afectan al tráfico, en especial al multimedia tiempo real [6].

- Número de direcciones IPv4 mal distribuido y escasos. Otro problema que surgió con el protocolo IPv4 fue su limitado número de direcciones. Si bien cuando recién se creo el número de direcciones que se tenía parecía ser adecuado (4294967296 direcciones), con el gran crecimiento de hosts (figura 2.1) y la mala administración de las direcciones trajo como consecuencia que el número de direcciones IP resultada insuficiente.

- La fragmentación de los paquetes se realiza en los Routers. Las redes IP se caracterizan por estar conformadas por enlaces con diferentes

capacidades de transferencia de datos llamado MTU, por lo que los routers tienen la necesidad de fragmentar los paquetes, creando un procesamiento adicional, lo que contribuye con el retardo extremo-extremo.

- Las soluciones de carácter transitorio y no toman en cuenta a las aplicaciones interactivas. Se hace uso de dispositivos que nos proporcionan soluciones de conectividad, seguridad, escalabilidad entre otros. Un ejemplo de lo antes mencionado es el uso de direcciones públicas y direcciones privadas para solucionar el agotamiento del número de direcciones; el uso de un NAT (Network Address Translator). Con estas soluciones, las aplicaciones se ven perjudicadas, se cambian los puertos y se manipula las direcciones de red o direcciones IP, lo cual a su vez causa que se rompa el esquema end-to-end en el que se basan las aplicaciones, un tiempo de procesamiento indeseado, retardo en las aplicaciones en tiempo real, entre otros.

- No se proporciona a las aplicaciones la Calidad de Servicio necesaria. El comportamiento Hop-by-Hop del protocolo IP conlleva a una asignación de recursos no fiable y sin garantía. A esto que soluciones transitorias como el NAT, la fragmentación de los paquetes, la falta de ingeniería de tráfico, producen que la Calidad de Servicio a las aplicaciones decaiga aún más.

2.1.2. Protocolo de Internet de Nueva Generación: IPng o IPv6

A pesar de las debilidades antes vistas, el protocolo de Internet IPv4 tiene buenas características; gracias a su robustez, flexibilidad y su compatibilidad con los protocolos de capa de transporte: UDP y TCP. En 1994 el Internet Enginnering Task Force decide una solución a las limitaciones de direcciones IPv4.

Así se crea la evolución del protocolo IP, IPng (the IP Next Generation) o IPv6, el cual se especifica en el RFC2460. Este nuevo protocolo no solo es pensado como una solución al actual protocolo, sino se introducen nuevas cabeceras, opciones de Calidad de Servicio (QoS), entre otros. Actualmente se siguen investigaciones para poder hallar nuevas utilidades para este nuevo protocolo así como deficiencias o "vacíos" [16]. Los cambios que se efectuaron IPv6, se muestran en la figura 2.1.:

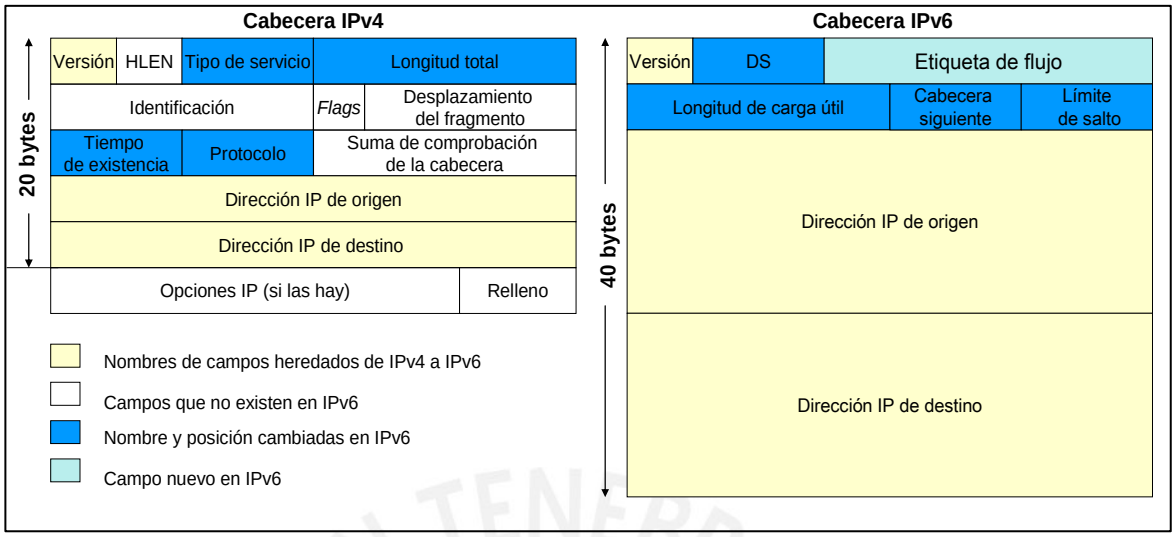

Figura 2.1: Evolución de IPv4 a IPv6

Existen campos que se han mantenido pero actualizados debido a su utilidad en el protocolo (version, source address, destination address), otros campos se les cambio de nombre y de posición (type of service a traffic class, total length a payload lengh, time to live a hop limit, protocol a next header) y los demás fueron suprimidos. El protocolo IPv6 nos provee de las siguientes facilidades:

- Amplia Capacidad de direcciones IP. La nueva longitud en bits de las direcciones IP provee de direcciones en un gran número, de múltiples usos y con nuevas funciones. Para ser más exactos, la nueva longitud de 128 bits del protocolo IPv6 nos provee de $3.4x10^{38}$ direcciones.

- Autoconfiguración de Equipos: Plug and Play. Gracias a los nuevos protocolos compatibles con IPv6 como Neighbor Discovery, DHCPv6, la configuración es transparente al usuario, rápida y confiable; lo cual ahorra tiempo tanto al administrador de red como al usuario ya que las aplicaciones no son seriamente afectadas por cambios en los parámetros de red.

- Seguridad y Privacidad en la transmisión de datos. A diferencia de su antecesor, IPv6 si fue diseñado para incorporar seguridad en transmisión de datos gracias a su compatibilidad con IPSec así con los nuevos protocolos de seguridad y/o encriptación de datos. Presenta cabeceras adicionales como

ESP (Encapsulating Security Payload), provee autenticación, confidencialidad e integridad; y AH (Authentication Header), que provee de autenticación e integridad.

- Modelos de Arquitectura de Qos: IntServ y DiffServ. IntServ y DiffServ son modelos que surgen por los campos etiqueta de flujo Flow Label y Differented Service, respectivamente. Actualmente la arquitectura que ha imperado es DiffServ por ser más flexible a los cambios tanto internos como externos y por usar menor cantidad de recursos que IntServ.

- No se admite la fragmentación en los routers. La funcionalidad de fragmentación ya no será realizada por los routers. El host del usuario será el encargado de realizar esta tarea a través del uso de protocolos de descubrimiento de la unidad máxima de transferencia MTU Maximum Transfer Unit de la red y/o de su conexión. Así el tiempo destinado al procesamiento y comprobación de tamaño de paquetes es eliminado de la backbone de Internet.

- Movilidad IP. Necesidad que empieza a exigir equipos móviles; tales como celulares, laptop's, pad's, entre otros.; los cuales tienen la necesidad de cambiar de manera dinámica los parámetros de capa de red. En la figura 2.2 se ilustra un escenario de Movilidad IP.

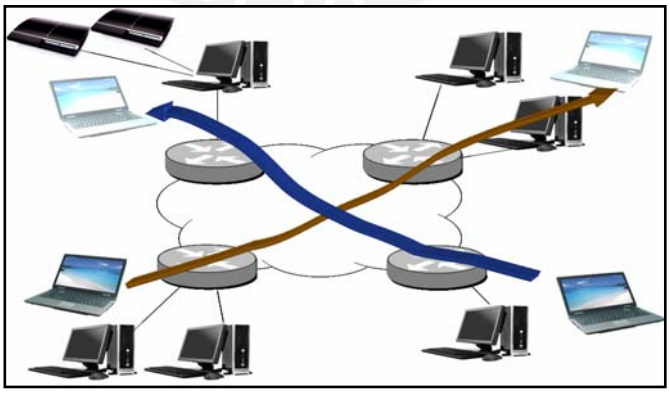

Figura 2.2: Idea Original del Proyecto Movilidad IP usando IPv6

2.2. Multi Protocol Label Switching MPLS

2.2.1. Protocolo de Internet: IP over ATM Escenario Actual

MPLS surge a partir de los esfuerzos de la IETF y otros fabricantes como Cisco System, Toshiba, Ipsilon, IBM; para simplificar el problema de compatibilidad del modelo IPoverATM, en el que se manejan redes completamente diferentes, una red ATM de capa 2 que da el soporte de sincronización, enlace, ingeniería de tráfico; y la otra, una red IP la cual proporciona el ruteo de paquetes bajo el esquema Best Effort. Esta situación de tener redes separadas ocasiona a los ISP's problemas de mantenimiento, velocidad, escalabilidad y sincronización.

MPLS Multi Protocol Label Switching, es creado con el fin de mejorar la compatibilidad entre la Capa de Red, protocolo IP, y la capa de enlace, tecnologías como ATM, Frame Relay, PPP, entre otros. Posee nuevas características tanto de capa de red como de capa de Enlace, lo cual lo hace atractivo para la Internet de la Nueva generación. Además de estas facilidades, nos provee de Calidad de Servicio (QoS) y de Ingeniería de Tráfico tanto para la generación del camino como para la restauración de este.

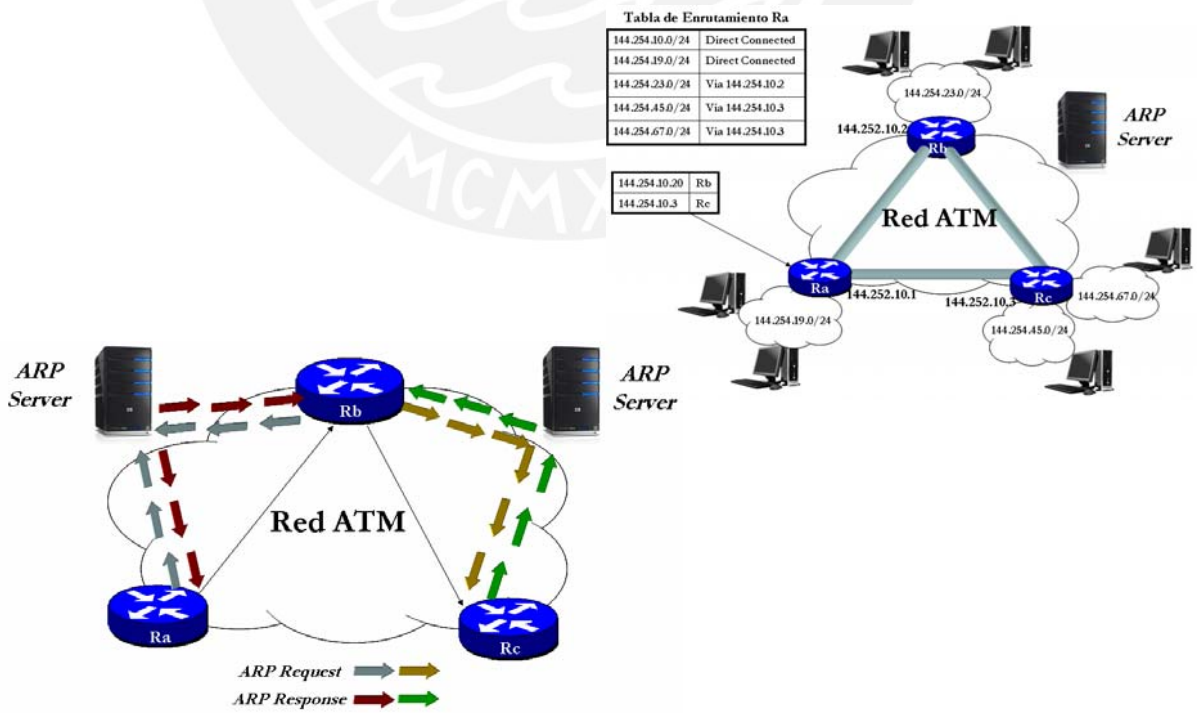

Figura 2.3: Redes IP y ATM

2.2.2. Terminología MPLS

- FEC Forwarding Equivalence Class

Conjunto de paquetes pertenecientes a determinado flujo que ingresan en la red MPLS a través de un mismo Router Ingress LER, a los cuales se les asigna la misma etiqueta y por tanto circulan por un mismo camino a través de la backbone hasta su destino. Normalmente se trata de paquetes que pertenecen a un mismo flujo correspondiente a una aplicación los que reciben la misma etiqueta. La FEC para la red especifica los recursos que se asignarán al tráfico que lleve la etiqueta a lo largo del camino LSP cuando este circule en la red MPLS. Cabe resaltar que un FEC puede agrupar varios flujos de diferentes aplicaciones según lo crea conveniente el administrador de la red y/o según lo estipulado en el contrato de arrendado con el ISP, pero un mismo flujo no puede pertenecer a más de una FEC al mismo tiempo ya que no se pueden asignar diferentes tipos de recursos a un mismo tráfico.

- LSP Label Switched Path

Camino por el cual el tráfico con FEC asignada será enviado a través de la red MPLS. Este camino es equivalente a un circuito virtual en la tecnología. En este trayecto que seguirá el tráfico, los recursos son asignados según el LSA. Además, si hubiese alguna falla de algún nodo intermedio en la red MPLS, el camino LSP puede conmutar de manera estática o dinámica según lo crea más conveniente el administrador de la red MPLS y/o lo estipulado en el contrato con el ISP [1] [3].

- LSR Label Switching Router

El router que puede conmutar paquetes en función a la etiqueta asignada MPLS según los diferentes parámetros del tráfico y por lo estipulado en el SLA. Estos routers se encuentran en el interior de la red MPLS y solo se encargan de la tarea de conmutar los paquetes etiquetados. Los protocolos de enrutamiento con los cuales los LSP's se formarán serán tarea del administrador de red, así también los mecanismos de recuperación o contingencia según lo que se indique en el LSA.

- LIB Label Information Base

Así como a nivel de capa de Red se tiene una tabla de ruteo con la cual el router puede tomar una decisión de envío para con el tráfico entrante según a donde el

paquete se dirija, existe una tabla de etiquetas que manejan los LSR muy semejante a la que existe a la de capa de red. Esta tabla relaciona interfaz de entrada - etiqueta de entrada con interfaz de salida - etiqueta de salida, es decir, si se recibe un paquete en un LSR, este para su reenvío solo cambiará la etiqueta y se conmutará a la interfaz correspondiente. Nótese que a pesar de que esta tabla LIB se forma en base a los protocolos de enrutamiento de la capa de red, funciona con una tabla de conmutación; Véase que a diferenta del modelo IPoverATM la capa de enlace tiene conocimiento de los sucesos que ocurren en capa de red [12].

- LSR Frontera de ingreso o Ingress LER Label Edge Router

Estos routers son los que se encuentran en la entrada de los flujos a la red MPLS. Se encargan de clasificar los paquetes en FEC y colocar las etiquetas correspondientes a los tráficos que se enviarán a la red según los parámetros acordados con el ISP a través de un contrato. Estos routers, al tener la tarea de clasificación, tienen que tener un muy alto poder de procesamiento para poder hacer esta tarea de manera muy rápida, eficiente y sin afectar al tráfico sensible a los retardos y al jitter [10].

- LSR Interior o LSR Label Switching Router

LSR es el router encargado de conmutar paquetes dentro de la red MPLS según la etiqueta que reciban estos por el Ingress LER. Como se mencionó anteriormente, la función de este router es únicamente intercambiar las etiquetas para cada FEC en base a su tabla LIB y efectuar el envío respectivo al router vecino para que este efectúe la misma acción, por lo cual no necesitan muchos recursos de procesamiento pero si de memoria RAM. Este router no puede agregar etiquetas ya que su función es netamente la conmutación de etiquetas, y en caso que reciba tráfico con etiqueta desconocida, este será descartado inmediatamente [10]. Además tienen la función de evitar loops en la red MPLS a través del uso del campo TTL de la Cabecera MPLS.

- LSR Frontera de egreso o Egrees LER Label Edge Router

Estos routers son los que se encuentran en la salida de los flujos a la red MPLS. Se encargan de extraer las etiquetas correspondientes a los tráficos y/o paquetes que se enviaron a través de la red, así se obtiene el paquete en su forma original antes de su clasificación en el Ingress LER. Estos routers, al tener la tarea de

eliminar la etiqueta y actualizar los campos TTL en la cabecera de capa de red, deben que tener un muy alto poder de procesamiento para poder hacer esta tarea de manera muy rápida, eficiente y sin afectar al tráfico sensible a los retardos y al jitter [6] [7] [9].

- LDP Label Distribution Protocol

Para el correcto funcionamiento de la arquitectura MPLS se necesita que el camino o LSP, así como los requerimientos del tráfico, sean anunciados a los nodos por los cuales este tráfico será enviado. Este protocolo recibe el nombre de LDP y es usado por los routers LSR y/o del tipo LER (Ingress o Egress). Estos protocolos pueden ser:

- CR-LDP Constraint Shortest Path First based LSP

Hacen uso los protocolos de enrutamiento tradicionales para hacer la reserva de etiquetas así como su anuncio a los vecinos. Cabe decir que con la implementación y/o ejecución de estos protocolos no se puede tener una protección ante fallas del camino principal ya que no contempla la formación de rutas alternativas hasta que una fallo hasta que este ocurra pero si se cumple con el hecho de proporcionar QoS a los tráficos que lo requiriesen [12] [15].

- MIRA Minimum Interference Routing Algorithm

Se toma en cuenta una ruta con mínimas interferencias y que además tiene en cuenta las características de debe de cumplir una red MPLS. Este tipo de algoritmos, realiza una fase de preproceso para crear un grafo con pesos en el que, posteriormente, se aplica un SPR shortest path routing. Si en el MIRA se aplicase el preproceso de cálculo de máximos flujos y enlaces críticos, hay propuestas donde el preproceso usa otro tipo de cálculo. El establecimiento de rutas de contingencia se reduce al hecho de que el algoritmo permite un mejor aprovechamiento de los recursos de la red, lo cual permite hacer un reenrutamiento en caso de fallo de eficiente y rápido [1][3].

- Redes Justas o FAIR NETWORKS

Dado que la capacidad de determinar la cantidad de tráfico de cada flujo deberá ser admitida por la red y el camino que se escogerá para el enrutamiento respectivo, y que el camino o LSP satisfaga los requerimientos de utilización

altos de la red y que así también garantice un trato justo a los usuarios de pendiendo de sus contratos LSA son conceptos que el enrutamiento tradicional no maneja, se desperdician recursos y los más afectados son las aplicaciones de usuario. La respuesta es el esquema de asignación de recursos MMF Max-Min Fairness, cuya idea básica es incrementar lo máximo posible el ancho de banda de una demanda sin que sea a expensas de otra. Hay cuatro variantes de este algoritmo: Max-Min Fairness basic para caminos fijos, Max-Min Fairness para caminos fijos acotados, Max-Min Fairness MultiPath y Max-Min Fairness MultiPath acotados [1] [3].

- RSVP-TE Resource ReSerVation Protocol for Traffic Engineering

El protocolo de reserva de recursos usado en un primer momento para la arquitectura IntServ (ver capítulo 3) se hicieron modificaciones para que sea más ligero, de fácil procesamiento. Así este protocolo sigue con la capacidad de reservar recursos (garantiza QoS) y establecer las etiquetas en los nodos MPLS, sea de manera explícita o dinámica. Como se crea un camino a través de los nodos, este camino se estable en los nodos activos en ese momento, así se permite la capacidad de reenrutamiento de los túneles LSP, recuperarse de caídas de red, evitar la cogestión entre otras facilidades.

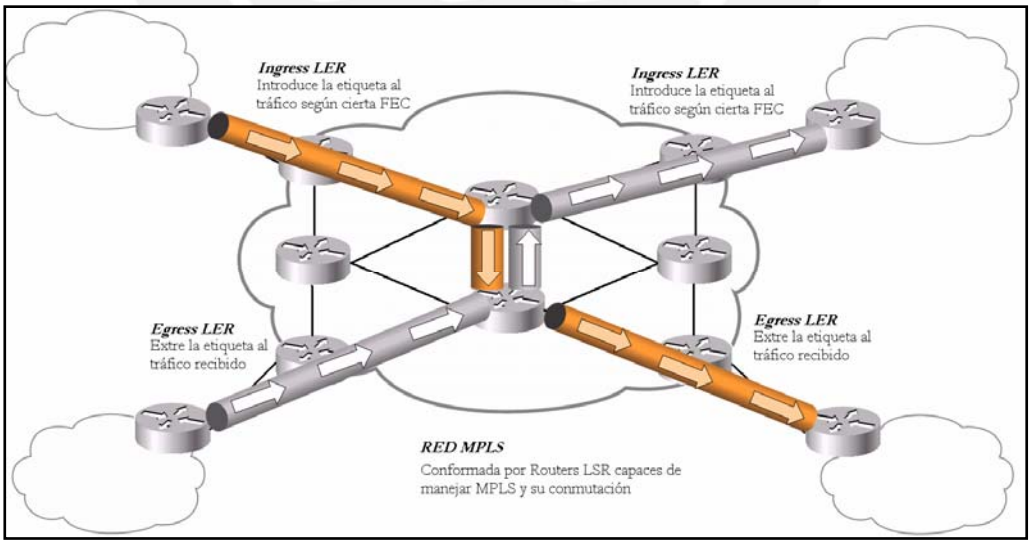

Figura 2.4: Dispositivos o entidades participantes en una arquitectura MPLS

2.2.3. Estructura de MPLS

La cabecera MPLS posee 32 bits de longitud, distribuidos en cuatro campos, cada uno con una función específica.

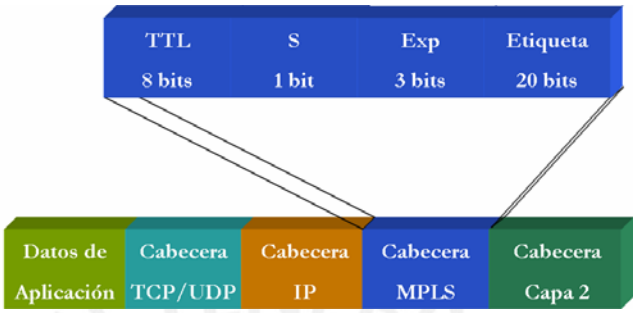

Figura 2.5: Cabecera MPLS

- Campo Label o Etiqueta. En base a este campo, los LSR pueden efectuar la conmutación. Esta etiqueta es asignada por el Ingress LER según parámetros descritos en el LSA. Como se indicó antes, los LSP son los que cambian la etiqueta a lo largo de su recorrido para poder formar un túnel LSP y la última etiqueta es extraída por el Egress LER.

- Campo Experimental EXP. Campo para uso experimental, pero actualmente se utiliza para transmitir información DiffServ por la creciente demanda de prioridades en el protocolo IP con lo que se tendrían ocho niveles de prioridad incluyendo el esquema de Best Effort.

- Campo Stacking. Gracias a este campo, se tienen jerarquías de etiquetas. MPLS tiene la capacidad de etiquetar tráfico MPLS de una red vecina con lo que se forma una pila o stack. Toma el valor 1 para la primera entrada en la pila, y cero para el resto.

- Campo TTL Time to Live. Al igual que en el protocolo IP, este campo sirve como un contador del número de saltos para poder evitar la creación de bucles o loops que se puedan generar en el envío de los paquetes etiquetados. Este campo reemplaza al TTL de la cabecera IP durante el viaje del datagrama por la red MPLS y es disminuido en una unidad por cada nodo por el que pasa; si llegase a cero en algún LSP, será descartado.

2.2.4. Funcionamiento de MPLS

La arquitectura MPLS, sigue parámetros para cumplir con la finalidad de asignar etiquetas al tráfico proveniente de una red en el Ingress LER. Este proceso, si bien se da en los routers de ingreso, las etiquetas deben ser comunicadas a los Routers LSR para que efectúen la conmutación respectiva a los paquetes según lo establecido en las tablas de conmutación. El proceso que se da en los routers, en forma general, se encuentra ilustrado en la figura 2.6. Cabe decir que el establecimiento de los LSP estará sujeto también a los cambios suscitados en la tabla de enrutamiento respectivo y si el protocolo de enrutamiento o el protocolo LDP contemplan alguna ruta alterna en caso de falla de algún nodo intermedio.

El primer paso en el funcionamiento de MPLS, una vez asignadas las etiquetas en el Ingress LER, es el envío de paquetes HELLO a los routers vecinos. Estos paquetes son del tipo UDP (User Datagram Protocol) con lo que se da prepara a establecer una sesión entre routers colindantes, verifica que el router vecino esta activo y habilitado para funcionar correctamente. Creada la condición para la poder establecer sesión entre routers vecinos, se procede al envío del paquete de inicio de sesión bajo TCP (Transmission Control Protocol), se crea la correspondencia entre dirección de capa de red, y la etiqueta correspondiente. Este proceso se llama Mapping.

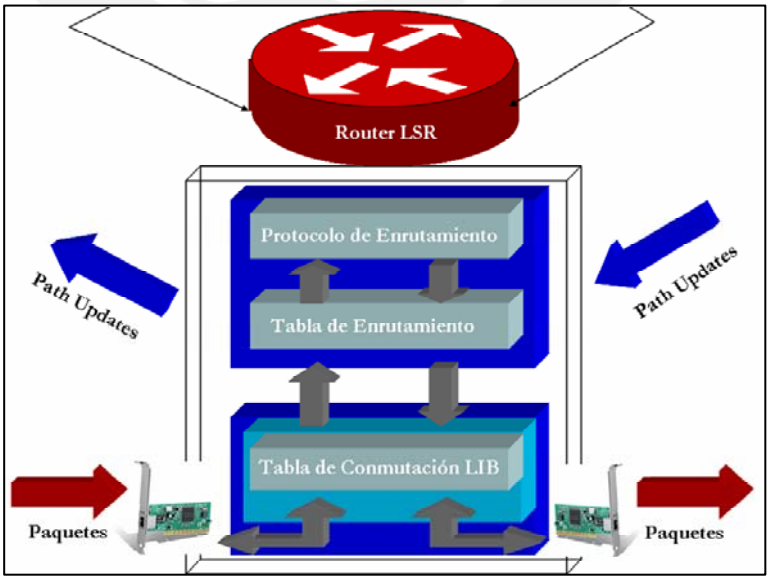

Figura 2.6: Funcionamiento de Tablas de conmutación MPLS y de Enrutamiento

Cabe decir que si bien el Ingress LER que efectúa el requerimiento es el Egress LER el que al final coloca la etiqueta a utilizar en el empaquetado; o en el caso que la etiqueta sea fija en el Ingress LER por configuración, el Egress LER crea el LSP o túnel por el cual atravesarán los paquetes la red MPLS hasta llegar a su destino. Después del establecimiento de las etiquetas y el circuito virtual LSP, se procede con el envío del tráfico. Véase que las etiquetas se cambian en cada nodo.

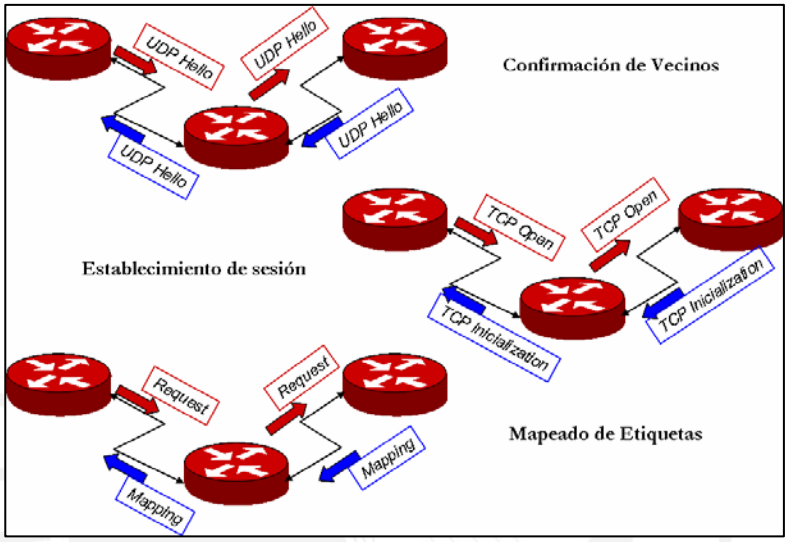

Figura 2.7: Intercambio y Mapeado de Etiquetas

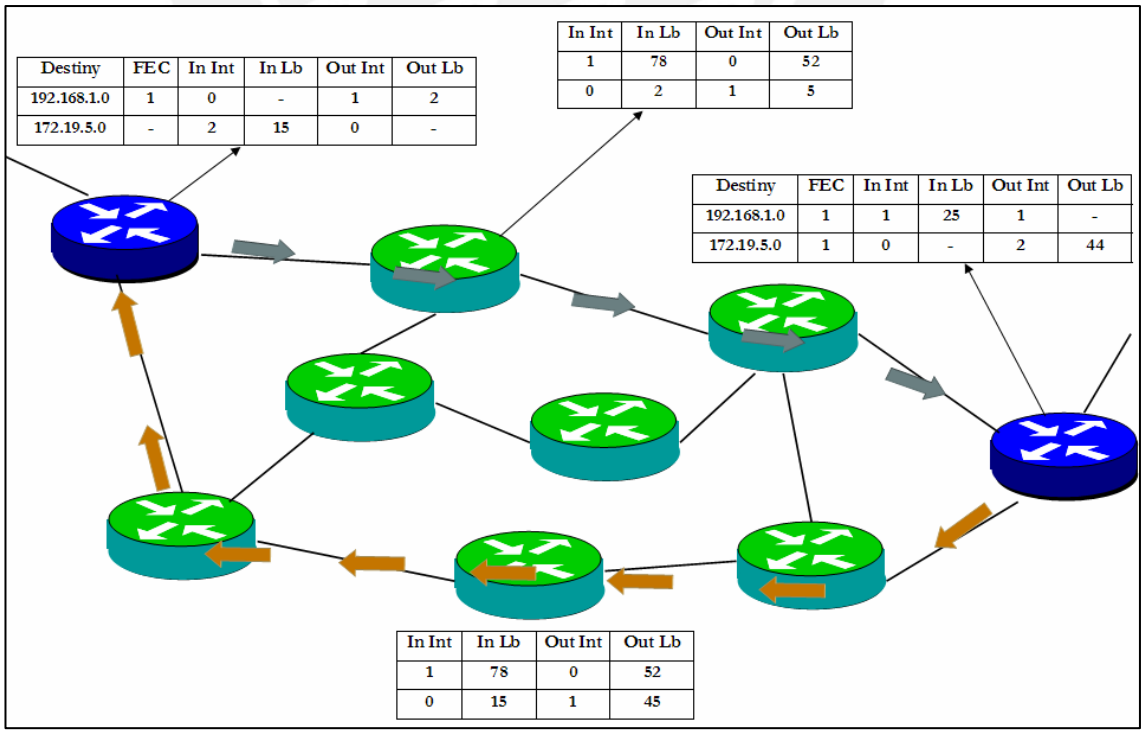

Figura 2.8: Envío de tráfico con MPLS

CAPÍTULO 3

PARÁMETROS DE CALIDAD DE SERVICIO QoS

En este capítulo se detallan los diferentes conceptos que definen la Calidad de Servicio (QoS) en la transmisión de datos en una red. También se explicarán los parámetros de Calidad de Servicio para los diferentes tipos de tráfico que se pueden enviar así como las arquitecturas QoS que se tienen actualmente. Finalmente, se explicará como la arquitectura MPLS encaja en estas arquitecturas QoS así también como efectúa el requerimiento y como maneja los requerimientos de recursos para la Calidad de Servicio (QoS) requerida.

3.1. Concepto de Calidad de Servicio QoS

Se conoce como Calidad de Servicio (QoS) los efectos colectivos y/o globales de las prestaciones de uno o múltiples servicios, en estos casos aplicaciones diversas, los cuales determinan el grado de satisfacción de un usuario con respecto al servicio o servicios contratados por una o varias entidades.

Puede entenderse además que es el conjunto de requisitos del servicio que debe cumplir la red en el transporte de un flujo.

Cuando la Internet surgió no se necesitaba gran demanda de velocidad, Ingeniería de Tráfico, prioridades, diferenciación del tráfico, entre otro; solo se requería aplicaciones en las que solo importaba la información, en forma de paquetes, llegase a su destino de manera segura y fiable. El stack TCP/IP cubrió perfectamente las demandas que se necesitaban de envío de paquetes así como el control de flujo necesario.

Con el primer crecimiento de la Internet, se necesito en un aplicar Ingeniería de Red, es decir, los enlaces más usados debían ser mejorados e incrementar su capacidad de transferencia y así poder adecuarlos a la nueva demanda. Arquitecturas como IP over Frame Relay y IPoverATM, de las cuales la más usada es la última por poseer la capacidad de transferencia que puede llegar a un Gigabit de velocidad y fueron usadas para incrementar las capacidades los requerimientos de aquel entonces.

Gracias a la convergencia de los servicios en tiempo real y al creciente número de usuarios, los ISPs no podían seguir aplicando Ingeniería de Red ya que no resultaba eficiente y menos conveniente invertir grandes cantidades para incrementar la capacidad de un solo enlace mientras otros de menor capacidad eran subutilizados. La vía para poder utilizar de forma óptima la red era aplicando Ingeniería de Tráfico [30].

Las nuevas aplicaciones no requieren solamente que el tráfico llegue a su destino; dependiendo de la aplicación se necesitará retardo asegurado, ancho de banda asegurado, un jitter mínimo, probabilidad de pérdida determinada, entre otros. Los protocolos de enrutamiento tradicionales tales como RIP, OSPFv2, IS-IS no son capaces de detectar los picos de tráfico que se dan en las redes, la gestión de colas no beneficia a los tráficos sensibles a los retardos y a su variabilidad. Arquitecturas que sean capaces de proporcionar la Calidad de Servicio necesaria para las aplicaciones así como lo requerido en el LSA son propuestos en base a nuevos protocolos de Internet de la Nueva Generación como IPv6, MPLS, RSVP, entre otros.

La ITU-T propone arquitecturas en las cuales se cumplan los requerimientos de QoS: Capacidad de Transferencia con Anchura de banda Dedicada DBW (Dedicated Bandwidth), Capacidad de Transferencia con anchura de banda Estadística SBW (statistical bandwidth), y finalmente Capacidad de transferencia de tipo mejor esfuerzo Best Effort [6] y las recomendaciones dependiendo del tipo de tráfico que se quiere cursar en la red y los parámetros que se deberían cumplir.

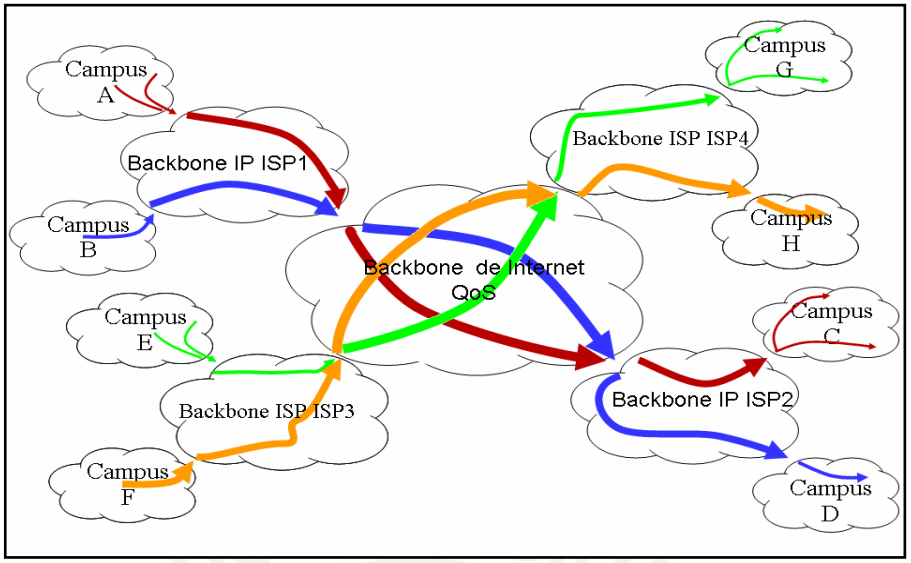

Figura 3.1: Esquema de Calidad de Servicio QoS

Por otro lado la IETF (Internet Engineering Task Force) ha propuesto nuevas arquitecturas que podrían dar solución a los requerimientos de Calidad de Servicios actuales gracias al uso de nuevas tecnologías tales como IPv6. Estas arquitecturas: Arquitectura de Servicios Diferenciados DiffServ, Arquitectura de Servicios Integrados IntServ y además se encuentra la aquitectura actual bajo el esquema Best Effort.

3.2. Parámetros de Calidad de Servicio

La ITU-T define parámetros de calidad de servicio con los cuales se basa para definir los diferentes requerimientos de las aplicaciones así como de los clientes hacen a la red del proveedor a través del LSA. Estos parámetros varían de tráfico en tráfico y de cliente en cliente, según los requerimientos y los aspectos técnicos de la red. Los parámetros que se mencionan se pueden utilizar para los diferentes tipos de especificaciones y/o para la evaluación de la calidad de funcionamiento de la red en lo referente a rendimiento de velocidad, exactitud del envío, seguridad en el funcionamiento y disponibilidad de la transmisión de los diferentes paquetes IP a nivel mundial ya sea de extremo a extremo, punto a punto y a tramos de la red [7].

- Retardo de transferencia de paquetes IP (IPTD). El retardo que sufre un paquete IP cuando es transmitido entre dos puntos de referencia cualesquiera. Se entiende que los puntos de referencia son puntos de extremo a extremo o puntos que se encuentran dentro de una red.

- Retardo medio en la transferencia de los paquetes IP. Este valor representa a la media aritmética de los diferentes retrasos que pueden sufrir los paquetes IP al ser transmitidos en la red. Recuérdese que los valores de retardo no son fijos ya que estos varían por lo que la media nos puede dar una idea cercana en algunos casos a lo que se podría esperar en el rendimiento de la red.

- Varianza del retardo de los paquetes de información (IPDV). Este parámetro se refiere al jitter o variación del retardo. Esta variación difícilmente sigue algún comportamiento fijo o que se pueda predecir, es decir, es aleatorio el valor que se tiene de muestra en muestra. Tráficos sensibles al retardo y a la variación de este dependen mucho de este parámetro para su funcionamiento.

- Tasa de errores en los paquetes IP (IPER). Se refiere a los paquetes erróneos que se obtiene en una transmisión total de paquetes. Las posibles fuentes de errores pueden venir desde la codificación en el trasmisor o fuente de tráfico hasta en la decodificación de la información en el receptor.

- Tasa de pérdida de paquetes IP (IPLR). Porcentaje de paquetes descartados de un total que han sido trasmitidos. Estas pérdidas se pueden dar por diversos motivos tales como congestión en las colas de los nodos, tiempo de vida (TTL en IPv4, HL en IPv6) prefijado ha expirado, algún nodo que ha fallado en la trasmisión, entre otros.

- Tasa de paquetes IP espurios (SPR). cuantifica el total de paquetes espurios detectados en el punto de medición de egreso en un intervalo de tiempo dividido por la duración del intervalo. Estos paquetes se generan por errores en la capa física como por errores en los nodos intermedios.

- Porcentaje de indisponibilidad del servicio IP (PIU). Este factor indica el porcentaje del tiempo del servicio programado total que se clasifica como período indisponible utilizando la función de disponibilidad del servicio. La función de disponibilidad de un servicio IP se basa en un umbral de la característica IPLR.

3.3. Requerimientos de las Clases de Servicio QoS

La ITU-T define 6 clases de servicio, en las que resume los requerimientos de los diferentes tráficos así como el rendimiento que el cliente debería percibir en los extremos de la red del ISP. Cabe mencionar que estos parámetros se han pensado para conexiones del tipo T1 1.544Mbps y/o E1 2.048Mbps.

Las clases así como los requerimientos de las diferentes clases de servicio que se explicarán se basan en mediciones en los extremos de las redes de los usuraos que han contratado el servicio, esto quiere decir que se puede atravesar la red así como los nodos de un solo ISP como también se puede atravesar las redes así como los nodos de más de un ISP. El proceso de envío y el tratamiento que será aplicado al tráfico de las aplicaciones a través de la red debe ser transparente al usuario que ha solicitado el servicio.

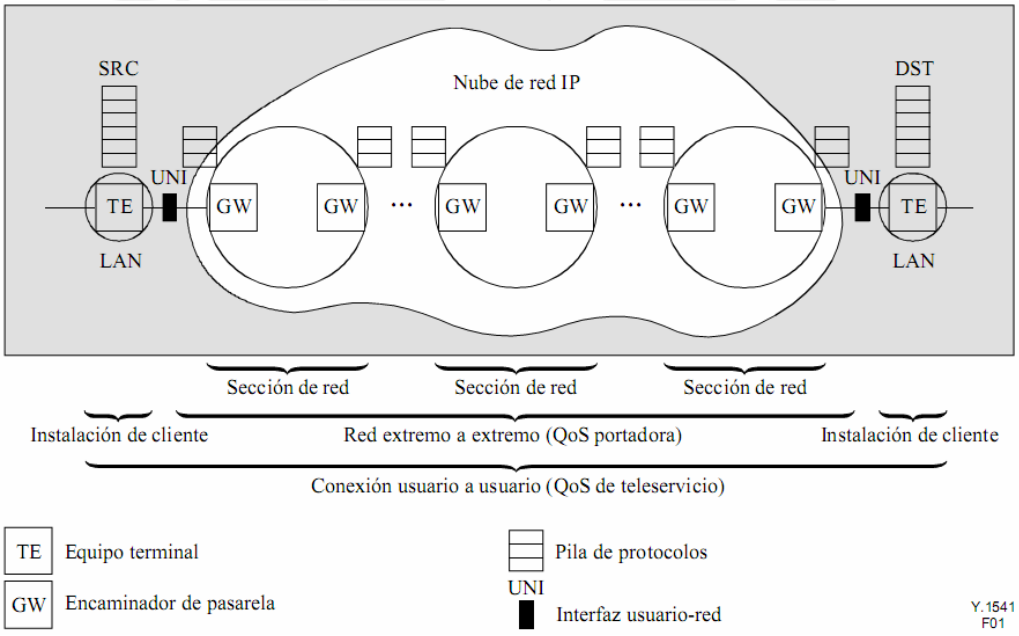

Figura 3.2: Camino de Referencia (Path Referente) para la Calidad de Servicio Punto - Punto entre las interfaces UNI (User Network Interface) [7]

Cabe mencionar que estas recomendaciones se cumplen para tipos de conexiones básicas proporcionadas por los ISPs y las conexiones de orden superior que se derivan de estas, también deberán cumplirlas, es decir, T2 6.312Mbps, T3 44.736Mbps, T4 274.176Mbps y E2 8.448Mbps, E3

34.368Mbps, E4 139.264Mbps, deberán cumplir con los requerimientos establecidos [7].

La tecnología de capa de enlace así como la del medio de transmisión serán provistos por el ISP/los IPS's y son considerados parte de la red IP, es decir, las clases son independientes de las diferentes tecnologías de acceso al medio que se dispongan/implementen dentro de la red del ISP o las diferentes tecnologías de cada ISP, según el caso respectivamente.

Parámetro de Rendimiento	Objetivo de Calidad de Rendimiento	Clase de Calidad de Servicio QoS					
		Clase 0	Clase 1	Clase 2	Clase 3	Clase 4	Clase 5
IPTD[1]	Límite superior en el IPTD medio	100 ms	400 ms	100 ms	400 ms	1000 ms	-
IPDV	Límite superior en el cuantil 1 – 1e-3 de IPTD menos IPTD mínimo	50 ms	50 ms	-	-	-	-
IPLR	Límite superior en probabilidad de pérdida de paquetes	1E-03	1E-03	1E-03	1E-03	1E-03	-
IPER	Límite Superior	1E-04					-

Tabla 3.1: Clases de Calidad de Servicio QoS y sus Requerimientos [7]

- **Clase 0: Aplicaciones en tiempo real de alta Interacción.**
  Estas aplicaciones se caracterizan por ser altamente sensibles no solo al retardo sino a la variación de este, el jitter. Poseen una alta interacción entre ambos puntos extremos de la red. El retardo promedio IPTD que se requiere como máximo es 100ms, la variación del retardo debe estar por debajo de 50ms, se requiere una tasa de pérdida IPLR menor a 10^{-3} y que la tasa de errores IPER no supere 10^{-4}. En este grupo de aplicaciones encontramos Voz sobre IP VoIP, Video Teleconferencia VTC.

[1] Cuando los tiempos de propagación sean muy largos no se cumplirán objetivos de bajo retardo extremo a extremo. En éstas y algunas otras circunstancias, que todo proveedor experimentará, no siempre se podrán cumplir los objetivos de IPTD en las clases 0 y 2. Los objetivos de retardo no impiden que un proveedor de servicios ofrezca servicios con compromisos de retardo más cortos. Se incluye el tiempo de inserción del paquete; en esta Recomendación se sugiere un campo de información de paquetes máximo de 1500 octetos para la evaluación [7] [8].

- **Clase 1: Aplicaciones en tiempo real**

  Estas aplicaciones se caracterizan por ser sensibles no sólo al retardo sino a la variación de este, es decir, el jitter pero no requieren parámetros tan rígidos como la Clase 0 descrita anteriormente. El retardo promedio IPTD que se requiere como máximo es de 400ms, la variación del retardo debe estar por debajo de 50ms, se requiere que la tasa de pérdida IPLR sea menor a 10^{-3} y que la tasa de errores IPER no supere 10^{-4}. En este grupo de aplicaciones encontramos Voz sobre IP VoIP, Video Teleconferencia VTC pero, dependiendo de la localización de los puntos extremos, con una interacción menor y una calidad cualitativa menor percibida en los extremos.

- **Clase 2: Transacciones de datos con alta interacción**

  Estas aplicaciones se caracterizan por ser la alta interacción a pesar de nos ser multimedia y/o de tiempo real. El retardo promedio IPTD que se requiere en el límite como máximo es de 100ms, la variación del retardo no se especifica dado que tiene prioridad de descarte, pero se requiere que la tasa de pérdida sea menor a 10^{-3} IPLR y que la tasa de errores IPER no supere 10^{-4}. En este grupo de aplicaciones encontramos tráfico de señalización.

- **Clase 3: Transacciones de datos**

  Estas aplicaciones se caracterizan por ser la interacción a pesar de nos ser multimedia y/o de tiempo real pero con menores requerimientos que la clase anterior. El retardo promedio IPTD que se requiere en el límite como máximo es de 400ms, la variación del retardo no se especifica dado que tiene prioridad de descarte, pero se requiere que la tasa de pérdida sea menor a 10^{-3} IPLR y que la tasa de errores IPER no supere 10^{-4}. En este grupo de aplicaciones encontramos tráfico de señalización menos rígido respecto a los parámetros.

- **Clase 4: Exclusivo para aplicaciones de bajas pérdidas.**

  Estas aplicaciones se caracterizan por tener como principal requerimiento una baja pérdida de paquetes correspondientes al tráfico marcado. El

retardo promedio IPTD que se requiere como máximo es de 1000ms, la variación del retardo no se especifica dado que tiene prioridad de descarte frente a otros tráficos de mayores requerimientos, pero se requiere que la tasa de pérdida IPLR sea menor a 10^{-3} y que la tasa de errores IPER no supere 10^{-4}. En este grupo de aplicaciones encontramos tráfico de señalización y/o de transacción de corta duración, flujo de video o videostreaming.

- **Clase 5: Aplicaciones Tradicionales de Redes IP**
  Estas aplicaciones tradicionales de las redes IP sin calidad de servicio, es decir, sus requerimientos no son rígidos en muchos aspectos. El retardo promedio IPTD, la variación del retardo, la tasa de pérdida IPLR y la tasa de errores IPER no se encuentran especificados dado que trabajan bajo el esquema de Best Effort en el cual solo se especifica y se espera que el tráfico llegue a los diferentes destinos. En este grupo de aplicaciones encontramos las aplicaciones tradicionales como correo electrónico email, FTP File Transfer Protocol, HTTP Hyper Text Transfer Protocol, entre otros.

Indicar además que la ITU-T especifica que los requerimientos de los tráficos que no han sido especificados en las recomendaciones, el ISP debe ofrecer una calidad mínima a los usuarios pero no se especifica cual debe ser esta pero recomienda que el Retardo de transferencia de paquetes IP IPTD no se deba exceder los 1000ms.

Las Clases de Calidad de Servicio especificadas anteriormente están sujetas a facilidades técnicas de la red del ISP. La ITU-T en esta recomendación indica que los requerimientos de Retardo de transferencia de paquetes IP IPTD de 100ms no siempre se pueden alcanzar así como otros parámetros de estas mismas clases como de las otras clases explicadas tales como la presencia de paquetes espurios. El ISP debe de especificar los parámetros que se proporcionarán como Calidad de Servicio (QoS) a los diferentes clientes, tratando de acercarse a las expectativas según las facilidades de red que se tenga.

Parámetro de Rendimiento	Aplicaciones	Mecanismos de nodo	Técnicas de Red
Clase 0	Tiempo real, sensibles a la fluctuación de fase, alta interacción (VoIP, VTC)	Cola separada con servicio preferencial, preparación del tráfico	Encaminamiento y distancia limitados
Clase 1	Tiempo real, sensibles a la fluctuación de fase, alta interacción (VoIP, VTC)		Encaminamiento y distancia menos limitados
Clase 2	Datos transaccionales, altamente interactivas (señalización)	Cola separada, prioridad por supresión	Encaminamiento y distancia limitados
Clase 3	Datos transaccionales, interactivas		Encaminamiento y distancia menos limitados
Clase 4	Sólo pérdida baja (transacciones cortas, datos en grandes cantidades, flujo continuo de vídeo)	Cola larga, prioridad por supresión	Cualquier ruta/trayecto

Tabla 3.2: Clases de Calidad de Servicio, Mecanismos de los nodos intermedios recomendados así como Las técnicas recomendadas para la red del ISP

Además de las clases citadas anteriormente, hay dos clases provisionales que están bajo estudio, ningún proveedor esta obligado a guiarse de estas clases hasta que se termine el estudio respectivo a menos de ser necesario. Estas clases estaría pensadas para soportar aplicaciones de altos flujos de usuario más allá de los que pudieran ser soportados por la clase 2 y la clase 0.

Parámetro de Rendimiento	Objetivo de Calidad de Rendimiento	Clase de Calidad de Servicio QoS	
		Clase 6	Clase 7
IPTD	Límite superior en el IPTD medio	100 ms	100 ms
IPDV	Límite superior en el cuantil 1 – 1e-5 de IPTD menos IPTD mínimo	50 ms	
IPLR	Límite superior en probabilidad de pérdida de paquetes	1E-05	
IPER	Límite Superior	1E-06	
IPRR	Límite Superior	1E-06	

Tabla 3.3: Clases de Calidad de Servicio Provisionales [7]

- **Clase 6: Emulación de circuitos TDM con alta interacción**
  Estas aplicaciones se caracterizan por emular circuitos TDM Time (Division Multiplexing) con alta interacción. El retardo promedio IPTD que se requiere como máximo es de 100ms, la variación del retardo es de 50ms, y se requiere que la tasa de pérdida IPLR sea menor a 10^{-5} y que la tasa de errores IPER no supere 10^{-6}. En este grupo de aplicaciones entra a tallar un parámetro de Radio de Reordenamiento IPRR (Packet Reordering Ratio), un parámetro que se aplicaría a tráficos TCP o para tráficos semejantes a UDP en el envío de los datagramas pero con un campo destinado para el reordenamiento en el destino. En este grupo de aplicaciones encontramos a la transferencia de televisión de alta calidad por Internet aún bajo estudio, transferencias TCP de alta capacidad y aplicaciones que se basan en emulación de circuitos TDM.

- **Clase 7: Emulación de circuitos TDM**
  Estas aplicaciones se caracterizan por emular circuitos TDM (Time Division Multiplexing). El retardo promedio IPTD que se requiere en el límite como máximo es de 400ms, la variación del retardo es de 50ms, y se requiere que la tasa de pérdida IPLR sea menor a 10^{-5} y que la tasa de errores IPER no supere 10^{-6}. En este grupo de aplicaciones también entra a tallar un parámetro de Radio de Reordenamiento IPRR (IP Packet Reordering Ratio), un parámetro que se aplicaría a tráficos TCP o para tráficos semejantes a UDP en el envío pero con un campo destinado para el reordenamiento en el destino. En este grupo de aplicaciones encontramos a la transferencia de televisión por Internet aún bajo estudio, transferencias TCP y aquellas aplicaciones que se basan en emulación de circuitos TDM pero con una interacción y sensibilidad menor a la clase antes explicada.

Estas clases están aún bajo un estudio muy minucioso por parte de la ITU-T, y los valores que se mencionan en estas clases pueden variar según los requerimientos de las aplicaciones que serán soportadas por estas.

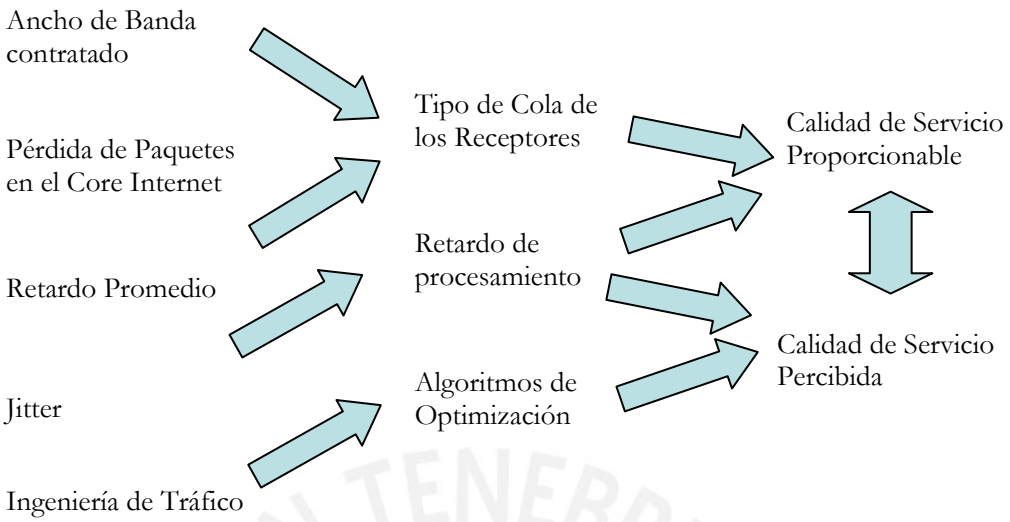

Figura 3.3: Parámetros de Calidad de Servicio QoS

3.4. Arquitecturas de Calidad de Servicio sobre IP

3.4.1. Arquitectura IntServ

En este tipo de arquitectura se usa el campo Flow Label de IPv6 para la identificación de los flujos que se enviarán o se envían a la red IP. En base a este campo se requerirá a la red y a los nodos que la conforman una asignación de recursos correspondiente. Para el funcionamiento de esta arquitectura se hace uso de un protocolo de reserva de recursos RSVP. La petición se hace de origen a destino, en los cuales los routers intermedios entrarán a un estado PATH STATE lo que significa que los routers certifican que tengan los recursos necesarios para la transferencia de datos. El receptor envía por estos mismos routers un mensaje de confirmación con los que los routers entrarán a un estado RESERVATION STATE con lo cual se confirma que la red esta preparada para el envío de información a la red. El estado en el cual el router ha aceptado la transferencia del tráfico correspondiente se llama SOFT STATE.

Dentro de esta arquitectura se tienen dos tipos de servicio según los recursos que se necesiten: Guaranteed Rate Service o Controled Load Service según lo que se especifique en el LSA con el ISP [37]. En el primero los recursos se comportan como un Circuito Virtual, es decir, los recursos se encuentran dedicados al cliente; en la segunda, los recursos se solicitan a la red y si esta los posee se los asigna de manera dinámica, es decir, cada vez que los necesite. Gracias a la

señalización en esta arquitectura antes del envío de información, se puede garantizar los recursos que se asignarán al tráfico y si la red esta en capacidad de poder ofrecer el servicio requerido, según el servicio y el tipo dentro de esta arquitectura. Esta arquitectura garantiza además que los tráficos que necesiten cierta cantidad de recursos asignados los obtengan, y no sobrepasen estos recursos. El inconveniente es la excesiva señalización que debe de efectuarse por cada flujo, así mismo la señalización para poder llegar al SOFT STATE se debe de efectuar por cada flujo que se quiera enviar a la red lo cual lo hace poco escalable y desperdicia muchos recursos para unos pocos flujos. Nótese que RSVP hace una actualización de estados cada 30segundos aproximadamente, así se detecta cualquier falla que hubiese en los nodos de la red; aunque ha habido estudios evitar la sobrecarga en la red, no es significativa la mejora ya que el proceso consume recursos y ancho de banda por cada flujo existente.

Actualmente no se usa este tipo de arquitectura por ser poco conveniente para cualquier tipo de red, en especial para la backbone de IP ya que esta debe controlar muchos flujos de muchos usuarios lo cual resultaría muy costoso de implementar por los recursos que se destinarían para el manejo de estos flujos.La ITU-T se refiere a esta arquitectura como arquitectura con Capacidad de Transferencia con Anchura de banda Dedicada DBW Dedicated Bandwidth, destinada al soporte de aplicaciones con un requerimiento de retardo riguroso específico, sin fragmentación de paquetes, con entrega garantizada y oportuna de los paquetes IP extremo a extremo. Los paquetes que se envíen por encima del límite establecido serán descartados por los nodos intermedios. También se menciona que se puede implementar este tipo de arquitectura bajo DiffServ siguiendo los mismos parámetros [6].

3.4.2. Arquitectura DiffServ

En este tipo de arquitectura se usa el campo DSCP Differentiated Service Code Point el cual se encuentra dentro del campo DS Differentiated Services también llamado Traffic Class. Esta clase de arquitectura es muy usada actualmente ya que es capaz de poder aceptar diferentes requerimientos de las aplicaciones y por asignarles prioridades a los tráficos.

Para el funcionamiento de esta arquitectura no se necesita ningún protocolo de reserva de recursos como RSVP ya que no es necesario efectuar ninguna clase de petición a los nodos que conforman la red. El procesamiento y el trato que se le dará a los tráficos dependerán únicamente de los valores que se tengan en el campo DSCP ya que cada valor tiene un significado distinto con respecto a requerimiento de recursos de red. Nótese que al no reservar recursos en la red para el envío de los diferentes paquetes, no se tiene una garantía de QoS como se tenía en IntServ por lo que el tráfico de baja prioridad puede verse afectado si la red se sobrecarga con tráfico de alta prioridad. Para evitar esta situación se le asigna a cada categoría un SLA.

El SLA es negociado con el ISP previamente y generalmente posee carácter estático. Los clientes pueden solicitar un determinado caudal en la categoría que necesiten según las necesidades de la red. Los routers de entrada de la red del proveedor son los responsables del control de admisión o Policy Control, así podrán colocar el valor correspondiente en los paquetes salientes para que sean tratados cada según su categoría dentro de la red del ISP.

El problema de una la arquitectura DiffServ es el procesamiento que se debe de hacer a los diferentes tipos de tráfico ya que se tienen prioridades diferentes, por ejemplo, que hacer si un paquete de menor prioridad es procesado cuando uno con mayor prioridad llega a la cola del router. Véase que el manejo de colas conocidas como RED (Random Early Detection), WRED (Weighted Random Early Detection) y DWRED (Distributed WRED) y otras más recientes como PQ (Priority Queuing), CQ (Custom Queuing), WFQ (Weighted Fair Queueing), CBWFQ (Class Based WFQ) requieren de un procesamiento extra de los routers que no siempre se puede conseguir de manera económica.

Actualmente esta arquitectura es usada ya que ofrece escalabilidad, no sobrecarga la red y si bien no ofrece una garantía de QoS se acepta muchas veces lo que se ofrece con esta arquitectura. Cabe resaltar que las políticas deben de ser muy bien especificadas en el SLA con el ISP ya que, como se mencionó antes, esta arquitectura se basa principalmente en el etiquetado o valor del campo DS de IPv6 o ToS de IPv4.

La ITU-T se refiere a esta arquitectura como una arquitectura con Capacidad de Transferencia con anchura de banda Estadística SBW (statistical bandwidth), la cual tiene por objetivo soportar aplicaciones que no requieran requisitos de retardo rigurosos. Está destinada al soporte de entrega garantizada de paquetes IP a lo largo del trayecto extremo a extremo de la red. Si la velocidad se envía por debajo o al tope de lo acordado en SLA, el tráfico es tratado como según se acordó en el respectivo contrato; caso contrario, se puede enviar los paquetes infractores bajo el esquema de Best Effort, es decir, sin ninguna garantía o prioridad con respecto a los demás paquetes así también puede descartarse [6].

3.4.3. Arquitectura Best Effort

En este tipo de arquitectura no se ofrece ninguna clase de garantía o de prioridad. Todos los paquetes son mandados a al red y no se asume ningún trato preferencial al tráfico que se envía bajo este tipo de arquitectura.
Si bien en esta arquitectura no se obtiene ningún beneficio, es la arquitectura que se tiene actualmente en la backbone de Internet. La cola que se implementa en esta arquitectura es la cola FIFO Fisrt In Fist Out, es decir, el paquete que llaga primero a la cola del router es el que será atendido primero por el router. Este escenario perjudica a los tráficos del tipo CBR Constant Bit Rate ya que estos son los que usan las aplicaciones multimedia sensibles a los retardos así como al jitter.

La ITU-T se refiere a esta arquitectura como una arquitectura Capacidad de transferencia de tipo mejor esfuerzo BE en la cual se requiere que se utilicen los recursos disponibles para el reenvío de los paquetes de los flujos de tipo mejor esfuerzo. Aún cuando no hay compromisos de QoS especificados, la expectativa es que se entregarán los paquetes siempre que estén disponibles suficientes recursos, un escenario que no suele darse en la Internet. Esta arquitectura resulta útil para aplicaciones que no tienen requisitos de pérdida o retardo rigurosos tales como email, FTP, WEB [6].

3.5. Relación entre las Clases de Servicio ITU-T y los Modelos de Arquitectura de Servicios Diferenciados de la IETF

Anteriormente se explicaron las clases de servicio que la ITU-T propone en la recomendación Y.1541 para los diferentes requerimientos que se especifican en el LSA así como los parámetros correspondientes que se deben de respetar según las clases solicitadas. Se explicó además que la IETF propone modelos de arquitecturas destinadas a ofrecer de cierta manera Calidad de Servicio (QoS). La relación entre estas proposiciones se detalla en el siguiente cuadro.

Servicio de Transferencia IP	Clases de Calidad de Servicio QoS IP	Observaciones
PDB con mejor servicio posible (BE)	Clase 5 de QoS no especificada	Un servicio IP heredado, que cuando funciona en una red ligeramente cargada puede alcanzar un buen nivel de QoS IP
PDB basados en reenvío asegurado (DSBW)	Clases 2, 3, 4 de QoS	Un servicio IP heredado, que cuando funciona en una red ligeramente cargada puede alcanzar un buen nivel de QoS IP
PDB basados en reenvío acelerado (DBW)	Clases 0 y 1 de QoS	-

Tabla 3.3: Asociación entre las Arquitecturas de Calidad de Servicio QoS de la IETF y las Clases de Calidades de Servicio de la ITU-T [7]

3.6. Arquitectura MPLS y Calidad de Servicio QoS en una red IP

La arquitectura MPLS nos provee de un circuito virtual o LSP a través de los diferentes nodos que conforman la red MPLS. Gracias a este tipo de funcionamiento, el circuito virtual creado provee de un trato igualitario a los diferentes tráficos que se envían bajo a un mismo túnel LSP bajo una etiqueta FEC en particular. Estudios relacionados a la Calidad de Servicio en diferentes escenarios son de interés actualmente dado que, a comparación con las demás arquitecturas, MPLS ofrece escalabilidad, simplicidad, velocidad, entre otros. Las facilidades que ofrece esta arquitectura para la implementación de Calidad de Servicio son las que se explicarán a continuación:

- Velocidad frente al esquema de enrutamiento IP. En efecto, la conmutación de etiquetas es más rápida y eficiente en primer lugar porque se produce en la capa inmediatamente anterior a la capa de red. En segundo lugar, el envío o forwarding se hace en base a un campo específico de la cabecera de la tecnología de conmutación; véase el caso de Frame Relay cuyo campo de identificación de camino es el DLCI Data Link Connection Identifier, en el caso de ATM el campo de identificación es el VCI Virtual Circuit Identifiers. En MPLS, el campo de identificación de camino es el campo Label el cual representa una FEC y un camino en la red. En tercer lugar, el proceso de clasificación del tráfico solo se da en los routers de entrada y el proceso inverso en el router de salida [12].

- Procesamiento más rápido de la cabecera MPLS. En el caso particular de MPLS solo se necesita tener en cuenta los campos Label, TTL para el envío del tráfico y en ciertos casos dependiendo del tipo de SLA que se haga, los campos correspondientes a Stacking para envío interdominio y el campo Experimental EXP para implementar prioridades. En el caso de IP, se deben de procesar muchos campos como dirección origen, destino, TTL (IPv4), opciones (IPv4), Hop Limit (IPv6), Payload, entre otros campos lo cual retrasa el envío y procesamiento de los paquetes en los nodos de la red, además de que este proceso se repite en cada nodo de la red [14][15].

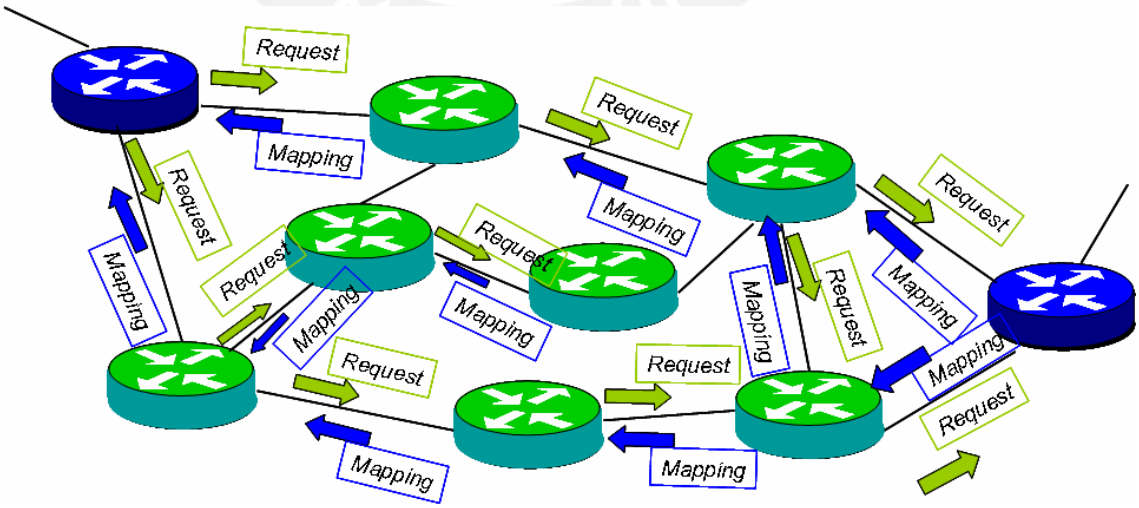

Figura 3.4: Esquema de envío en la Arquitectura MPLS. Uso de etiquetas [27]

- Facilidad de Implementación. Cualquiera de las formas que existe para el correcto funcionamiento de la arquitectura MPLS requiere de poca señalización entre los nodos. Las contramedidas que se pueden tomar en caso de fallas, protocolos de enrutamiento, entre otras tecnologías de capas superiores y/o inferiores no afecta el funcionamiento de la arquitectura, es decir, para cualquier cambio en cualquier capa la arquitectura se amolda a los posibles cambios sin la intervención del administrador de la red MPLS [25].

- Adaptabilidad frente a la Capa de Red como a la Capa de Enlace. La arquitectura MPLS se localiza entre la capa de red y la capa de enlace, se vale de la conmutación para el envío del tráfico y de los protocolos de enrutamiento para la creación de las tablas de conmutación y de rutas alternas para diferentes fines. Como se puede observar, MPLS utiliza la capa superior inmediata así como la inferior pero su funcionamiento no depende de estas. Además, MPLS se adapta perfectamente a las tecnologías de capa de enlace; tales como ATM, PPP, Frame Relay, la Familia Ethernet; así como a cualquier tecnología de capa de red como IPv4, IPv6.

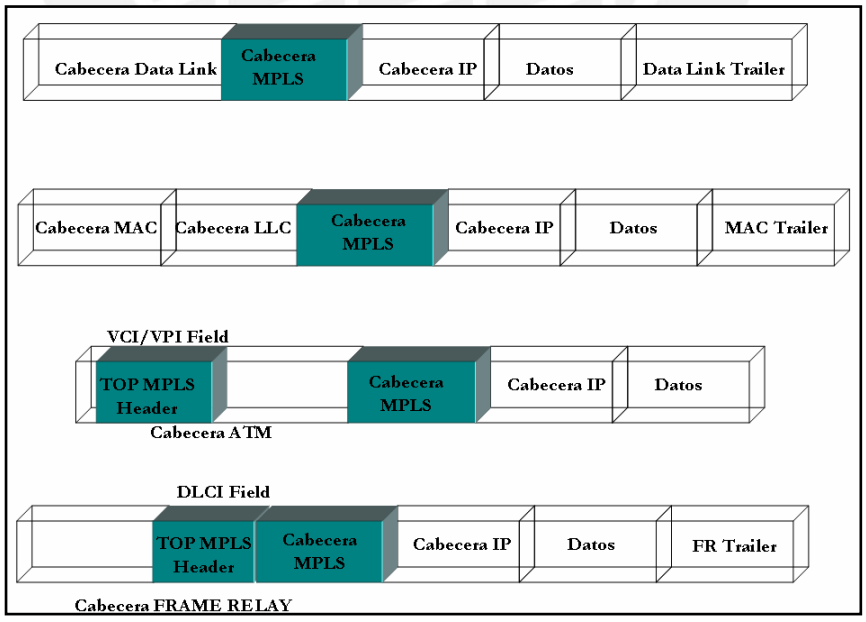

Figura 3.5: Compatibilidad de MPLS con tecnologías de capa de enlace

- El cambio se da en el Software y no en el Hardware. Los nodos que conformarían la red MPLS necesitan solo los procesos correspondientes al manejo de MPLS independientemente de las tecnologías y funcionamiento de la capa de enlace y la capa de red. Para equipos actuales basta con una actualización al sistema operativo de los routers que conforman la Backbone.

- Se acomoda a los modelos de Calidad de Servicio (QoS) de la ITU-T. Gracias al campo experimental EXP el cual cuenta con 3 bits, se puede priorizar los diferentes tipos de tráficos cursados en el mismo túnel LSP. Nótese que con 3 bits podemos obtener 8 tipos de prioridades, lo cual coincide con el número de clases que ha sido propuesta por la ITU-T [7] [9]. Esta característica se suma al hecho de que MPLS es capaz de reservar recursos a través de un mismo así como de diferentes dominios. Puede entenderse que una Clase de Servicio pueda ser implementada bajo una reserva de recursos para ciertos tipos de tráficos provenientes de un cliente y dentro de esta reserva de recursos se daría prioridad a los tráficos que la necesiten.

- Permite la implementación de Ingeniería de Tráfico. Gracias a nuevos protocolos de enrutamiento como a mejoras a otros protocolos de capas superiores, MPLS tiene la capacidad de cambiar dinámicamente de ruta. Las nuevas rutas pueden ser generadas por protocolos de capa de red destinados a crear la tabla de enrutamiento bajo ciertas métricas, así también se aplicarán ciertas políticas en estos mismos protocolos para una mejor evaluación de los recursos de la red [25]. Además estudios sobre posibles alternativas, impacto en la Calidad de Servicio (QoS) se llevan acabo actualmente como la utilización del protocolo RSVP-TE RSVP Traffic Engieering [1] [3].

- Reserva de Recursos Intradominio MPLS. Gracias al túnel LSP que se crea para el envío de los tráficos correspondientes, se asegura que la red los pueda soportar los requerimientos solicitados dado que si fuese el caso contrario, el túnel no puede establecerse. Así mismo, con el uso de algoritmos de enrutamiento como de protocolos de reserva de recursos

RSVP, se puede asegurar una correcta asignación de recursos a los tráficos correspondientes según dirección de destino, origen, puertos, entre otros. A diferencia de la arquitectura IntServ, el protocolo RSVP utilizado en la reserva de recursos de MPLS no requiere el procesamiento y la asignación de recursos que se necesitaba en IntServ por lo que lo convierte en muy adecuado para los fines antes mencionados.

- Garantía de Calidad de servicio sobre el esquema IP. A diferencia del esquema actual de Internet Best Effort y de DiffServ, los cuales poseen un comportamiento salto por salto o hop by hop, es decir, no dan una garantía total sobre el envío del tráfico que se inserta ala red IP. MPLS, por su parte, antes del envío construye un túnel LSP, donde el comportamiento es igual en todos los nodos que constituyen este túnel LSP, es decir, los recursos que se destinan para este tráfico FEC serán destinados para este tráfico exclusivamente hasta que el tráfico acabe y se liberen los recursos asignados y sean tomados por otro requerimiento. Aunque IntServ tiene un comportamiento muy parecido en lo que respecta a asignación de recursos, MPLS lo hace de red a red, es decir, crea un túnel LSP desde el router origen al router destino pero no de hots a host como lo hace IntServ; otra diferencia entre estas arquitecturas es el hecho que IntServ crea un comportamiento de recursos dedicados por cada flujo en la red, MPLS crea el mismo comportamiento de recursos dedicados pero con la gran diferencia que los mismos recursos pueden ser usados por diferentes tráficos según los requerimientos especificados en el LSA [1] [3].

- Reserva de Recursos Interdominio MPLS. Así como MPLS está habilitado para el envío como también para la reserva de recursos en las redes MPLS en una misma red de un solo proveedor; gracias al campo Stacking de MPLS se puede extender el túnel creado dentro de una sola red a las redes MPLS que se necesitarán atravesar hasta llegar al destino. En cada dominio MPLS externo se coloca una cabecera adicional al flujo de tráfico, por lo que, el tráfico que poseía una etiqueta será nuevamente etiquetado cuantas veces sea necesario; y estos paquetes se comportarán como paquetes convencionales en esta arquitectura, es decir, al egreso de cada red se les removerá la etiqueta que se les asignó inicialmente. La cabecera

MPLS original, es decir, con la que salió de la red MPLS de nuestro proveedor tendrá marcado el campo Stacking en 1. Gracias a esta facilidad de MPLS, la reserva de recursos se extiende cuanto sea necesario de una forma muy sencilla en comparación a otras tecnologías.

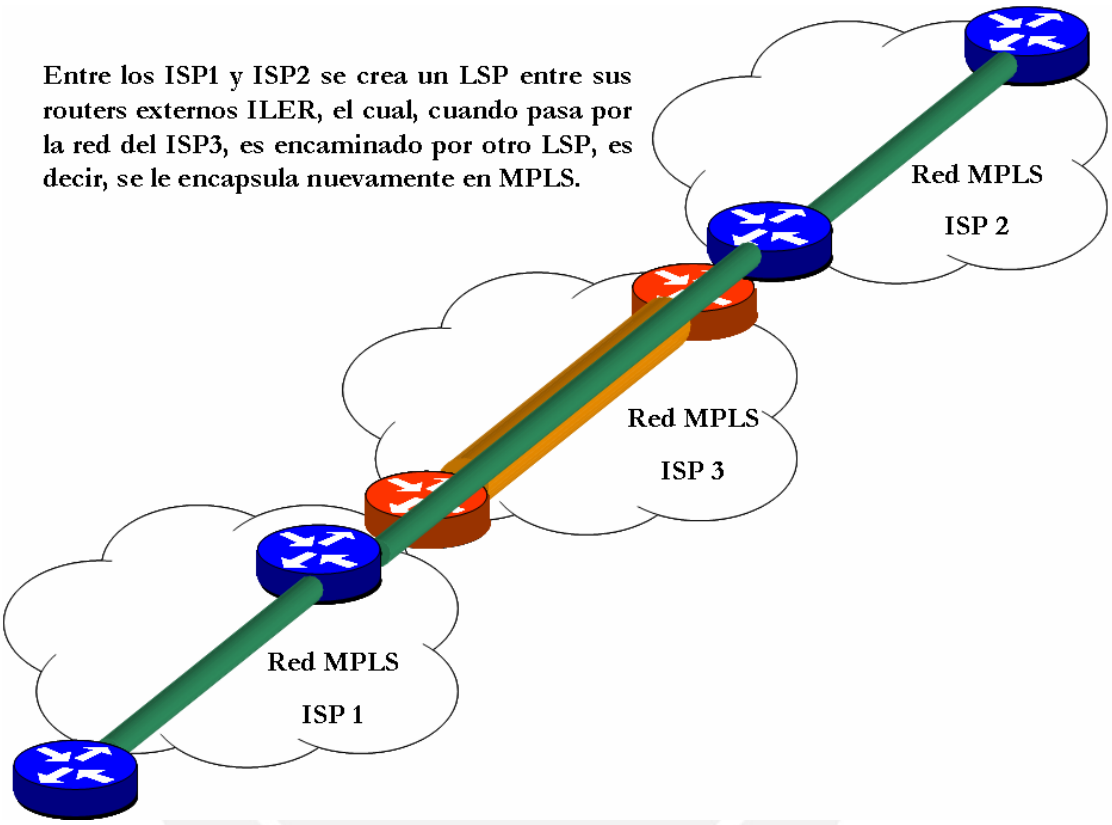

Figura 3.6: Reserva de recursos Interdominio

- Permite la implementación de Balanceo de Carga. Al igual que la posibilidad de Ingeniería de Tráfico se puede implementar en los nodos que manejan MPLS es proporcionada por los protocolos de capas superiores, así también el balanceo de carga se puede proporcionar a la red usando estos mismo protocolos según lo crea lo más conveniente el administrador de red. Esta funcionalidad se suele combinar con otras características de MPLS como ingeniería de Tráfico ya que así se evitaría la creación de congestión como la subutilización tanto de los enlaces como de los recursos de los nodos con menores capacidades existentes en la red pero que en un instante de tiempo con media a alta congestión podrían convertirse en la mejor ruta de cierto tipo de tráfico con lo que se podría evitar pérdidas y retardos en el tráfico que cursa la red [1] [3].

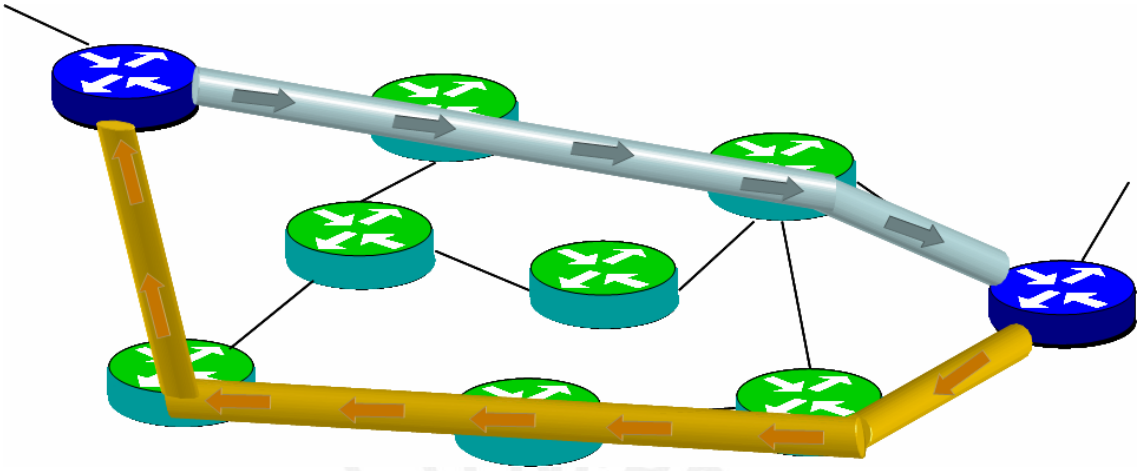

Figura 3.7: Balanceo de carga en MPLS. Envío de tráfico por diferentes caminos para evitar congestiones

CAPÍTULO 4

PROPUESTA DE TESIS

En los capítulos anteriores se explicó lo referente a las arquitecturas MPLS, las cuales, son capaces de proporcionar la Calidad de Servicio (QoS) a los diferentes tipos de tráfico que pueden cursar la red provenientes de uno o varios clientes que hallan suscrito un LSA con el ISP. En este capítulo se explicará los parámetros a elegir en la medición así como las justificaciones por las que se eligieron estos parámetros. Así también se definen las topologías a emplear en las pruebas. Se tomará la decisión de implementar o simular la red así como las razones que llevaron a esta decisión.

4.1. Propuesta de Tesis

La tesis propone una comparación entre las Arquitecturas MPLS LDP y MPLS RSVP, ya que ambas arquitecturas presentan facilidades para poder proporcionar Calidad de Servicio (QoS). Además, estas arquitecturas se encargan del envío de los paquetes, dejando de lado el esquema de Hop-by-Hop con prioridades en su modelo DiffServ y MPLS que usa túneles LSP en los que se envía el tráfico ya que se forma una VPN.

Los estudios [1] [3] [12] muestran que ambas arquitecturas son provechosas para el fin antes mencionado, nuevas propuestas están bajo una rigurosa investigación en especial para los estudios referentes a MPLS e ingeniería de tráfico. Con los resultados a obtenerse se recomendará la tecnología más favorable y bajo que casos.

4.2. Justificación de la Propuesta

Las Arquitecturas IPv6 y MPLS surgieron casi contemporáneamente ante los requerimientos cada vez más exigentes de las nuevas aplicaciones. Se menciona como capacidad principal de ambas su poco procesamiento en los routers de la Backbone, capacidad de proporcionar Calidad de Servicio, capacidad de aplicar Ingeniería de Tráfico, recuperación rápida ante fallas de la red, entre otros. Si bien MPLS no es capa de red, tampoco pertenece es capa de enlace; MPLS forma sus LSP Label Switching Path en base a la información que obtiene de la capa de red, donde se localiza generalmente el protocolo de Internet en cualquiera de sus versiones; entonces se deduce que si en la capa de red se implementa un protocolo de enrutamiento, este también influirá en el LIB Label Information Base que se seguirá para el envío de los paquetes.

Nótese que MPLS introduce una cabecera adicional al paquete IP, es decir, antes de la introducción de la cabecera correspondiente a la capa de enlace, MPLS introduce su cabecera inmediatamente después de la que el corresponde a la cabecera de capa de red. Este proceso toma procesamiento y esto a su vez un tiempo en el proceso de etiquetado de los paquetes. Se desea saber si este esquema justifica este proceso en el hecho de que gracias a la etiqueta introducida en el Ingress LER, se puede hacer un envío más rápido de los flujos ya que este envío se basa en la conmutación y no en el enrutamiento tradicional; es realmente justificable y para que casos podría recomendarse.

Finalmente; se desea saber, en base a los parámetros que se medirán en las diferentes topologías; cual es la Calidad de Servicio (QoS) que se podría esperar en redes bajo estas arquitecturas, y cual puede soportar mejor los diferentes requerimientos que las aplicaciones pueden necesitar para su correcto funcionamiento y bajo que casos.

4.3. Parámetros de Medición

Los parámetros que se tendrán en cuenta serán los parámetros que propone la ITU-T en las recomendaciones Y.1540 Y.1541. Los datos que se obtendrán con las pruebas serán comparados con las clases que en estas recomendaciones se explican así como los parámetros a cumplir en cada una de estas clases. Se podrá dar como conclusión a que aplicaciones se le puede dotar de los tiempos y/o retardos necesarios para su correcta transmisión, las topologías en las que las implementaciones de las arquitecturas son apropiadas así como que parámetros se pueden cumplir en las mismas. Con el resultado de las experiencias, se pretende proporcionar una base para estudios de arquitecturas que tengan la capacidad de proporcionar Calidad de Servicio (QoS).

4.4. Simulación e Implementación

Las pruebas a realizarse bajo simulaciones de red, el simulador que se eligió es el OMNet++. Nótese que a pesar que las topologías serán simuladas, la tesis no pierde valor ya que muchos estudios de diferentes centros de investigación, universidades, entre otros. Se basan en diferentes simuladores, lo cual hace entender que se tienen buenos resultados, confiables con respecto a los que se obtendrían con los equipos verdaderos y aplicables a los casos de la realidad. Finalmente indicar que la implementación en equipos reales usando equipos que pueden soportar ambos protocolos, es decir, que pueden soportar MPLS LDP y MPLS RSVP, de las topologías de red que se proponen en este proyecto de tesis se incluirán en la dentro del capítulo 6: Trabajos a Futuros, así también cualquier posible trabajo que se vea conveniente que se desprenda de esta tesis.

4.5. Escenarios de Simulación

Se propone una evaluación en escenarios de tamaño mediano como de un gran tamaño. Las razones de esta proposición son las que se exponen a continuación:

- La Calidad de Servicio (QoS) es medida por parámetros que utilizan el tiempo. A excepción de pérdida de paquetes, los otros parámetros que se

proponen dependen del retardo así como la variación de este los cuales se dan a escalas de milisegundos.

- La Calidad de Servicio (QoS) no siempre es alcanzada en las Backbone ISP. En las recomendaciones de la ITU-T Y.1540 y Y.1541, se menciona que no siempre pueden satisfacerse estos parámetros dado que las dificultades técnicas, geográficas, climáticas y de otra índole afectan en el desempeño de la red y la Calidad de Servicio que se puede proporcionar.

- El número de nodos en la red afectará el tránsito de los tráficos en la red. El envío de los paquetes requiere un tiempo de procesamiento en los nodos para el envío de los flujos de tráfico y la asignación de recursos, ya que el procesamiento de los paquetes se hace de forma diferente; si bien MPLS que se basa en conmutación de etiquetas, LDP tiene un protocolo de envío de información diferente al de RSVP.

- Los límites de cada arquitectura para satisfacer la Calidad de Servicio (QoS). Las arquitecturas MPLS si bien son capaces de proporcionar un cierto nivel de Calidad de Servicio requerido, son deseables el conocimiento de sus límites de funcionamiento según el tamaño de la red así como los recursos que se tienen disponibles en la red.

4.6. Definición de las Topologías de Simulación

4.6.1. Topología Mediana: MAN

Número de Routers: 14 Routers

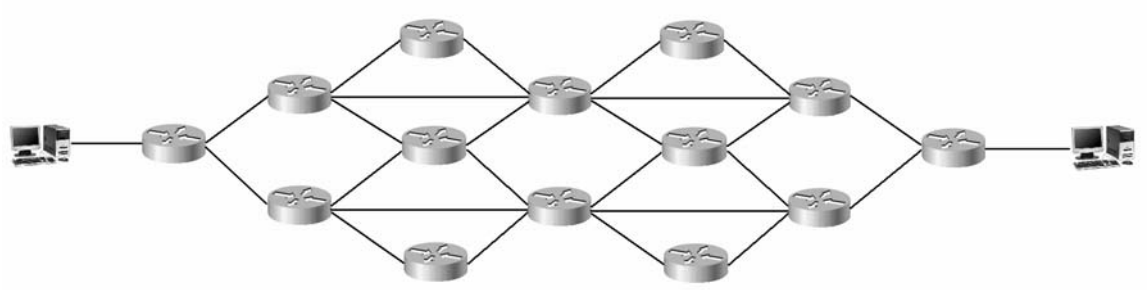

Figura 4.1: Topología de nivel Mediano

- **Objetivos y Alcances**

En la figura 4.1 se presenta una topología de mediano tamaño en la cual puede obtenerse una gama de rutas en las que se experimentará un considerable en la transmisión de datos hacia cada nodo de la red. Este tipo de redes son usadas por los ISP pequeños que usualmente operan en una sola ciudad. En este escenario se desea experimentar la capacidad de proporcionar Calidad de Servicio (QoS) así como los parámetros que deben de cumplirse para las clases de Calidad de Servicio (QoS) propuestas por la ITU-T [7] [9].

Las pruebas consistirán en la transmisión de paquetes IP entre extremos de la red. Los paquetes de video streaming y de videoconferencia serán considerados un flujo de tráfico al cual se le brindará la máxima Calidad de Servicio (QoS) frente a otros tipos de tráfico que pudiesen cursar la red. Nótese que al tener un número de routers considerable así como la existencia de tráfico ajeno al de interés, influirá en el rendimiento de las arquitecturas a implementarse, de manera diferente en cada caso. Con los resultados que se obtendrán de la experiencia, podrá determinarse cual es la mejor arquitectura bajo las condiciones que se presentan así como las clases de Calidad de Servicio (QoS) que puede proporcionar.

4.6.2. Topología Muy Grande: Nivel Backbone

Número de Routers: 40 Routers

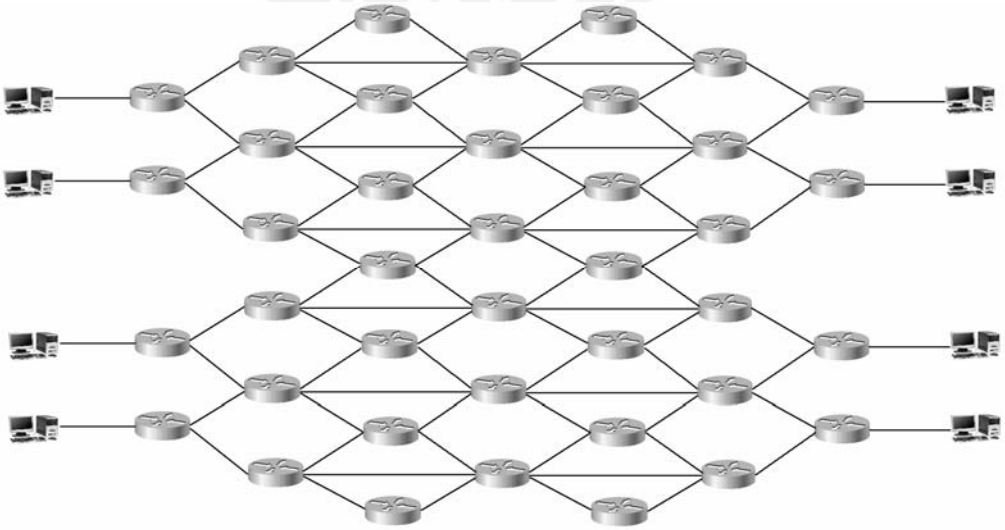

Figura 4.2: Topología de nivel Backbone

- **Objetivos y Alcances**

En la figura 4.2 se presenta una topología backbone equivalente a las grandes ISP con conexiones a otros ISP, las cuales experimentan una enorme carga de tráfico en la transmisión de los paquetes IP en las rutas que comunican a los nodos; por este motivo, es de interés experimentar la capacidad de proporcionar Calidad de Servicio (QoS) así como los parámetros que deben de cumplirse según las recomendaciones de la ITU-T [7] [9].

Las pruebas consistirán en la transmisión de paquetes IP entre extremos de la red. Los paquetes de video streaming y de videoconferencia serán considerados un flujo de tráfico al cual se le brindará la máxima Calidad de Servicio (QoS) frente a otros tipos de tráfico que pudiesen cursar la red. Nótese que al tener un número de routers considerable así como la existencia de tráfico ajeno al de interés, influirá en el rendimiento de las arquitecturas a implementarse, de manera diferente en cada caso. Con los resultados que se obtendrán, podrá determinarse la mejor arquitectura bajo las condiciones que se presentan en esta topología así como las clases de Calidad de Servicio (QoS) que puede proporcionar para que así seleccionar la mejor opción.

CAPÍTULO 5

SIMULACIONES Y RESULTADOS

En este capítulo se explicarán los resultados obtenidos al simular las topologías propuestas en el capítulo anterior. Asimismo, se detallan los parámetros de simulación utilizados y se explican los criterios utilizados en el desarrollo de las experiencias.

5.1. Topologías Simuladas.

De las topologías propuestas en el capítulo 4 se eligieron la MAN y la BACKBONE, dado que son las más representativas de las topologías propuestas. Se justifica la selección dado en la simulación de las topologías, las que se señalan tienen comportamientos que aportan más al estudio realizado. Las pruebas se realizan entre arquitecturas MPLS que usan tanto LDP (Label Distribution Protocol) y RSVP-TE (Resource ReSerVation Protocol for Traffic Engineering) para poder recomendar el uso de una de estas tecnologías MPLS en escenarios con tráfico idóneo, es decir, en el cual el tráfico de interés el único en la red; así como en escenarios con tráfico real, es decir, escenarios donde nuestro tráfico disputa recursos de red con otros tipos de tráfico.

5.2. Evaluación de Arquitecturas MPLS LDP y MPLS RSVP-TE.

Se justifica la evaluación de estas arquitecturas MPLS ya que si bien ambas proporcionan una Calidad de Servicio con el uso de VPN, pero lo hacen de formas diferentes tal como se explicó en el capítulo 2.

5.2.1. Topología MAN.

La topología MAN, como se explicó en el capítulo 4, tiene dos hosts capaces de crear tráficos de interés y 14 routers LSR que manejan MPLS.

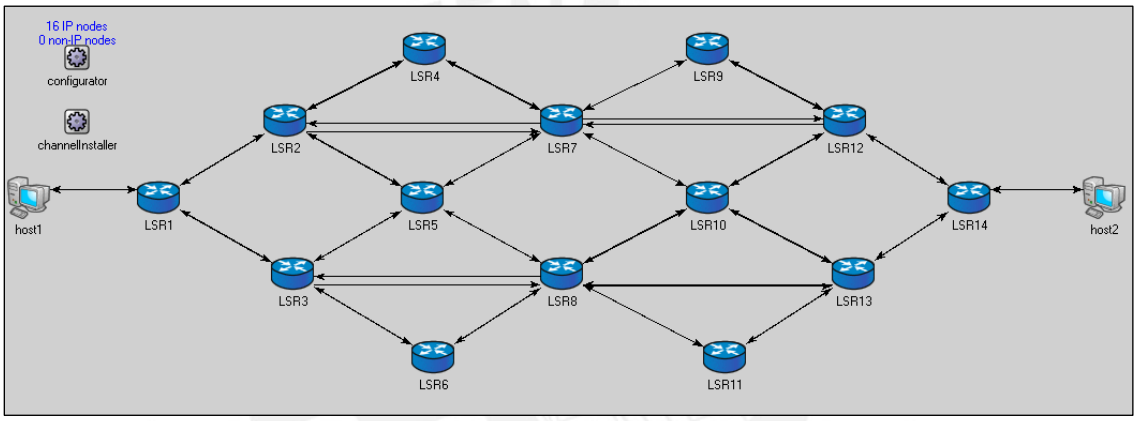

Figura 5.1: Topología MAN. La transmisión de los datos es del host2 a host 1

Esta topología tiene los parámetros que se muestran en el siguiente cuadro:

Tabla 5.1: Parámetros de Simulación MAN simple

	Cantidad	Retardo (ms)	Velocidad de Transmisión (Mbps)	Tipo de Cola	Duración (s)
Host	2	1.00E-04	-	FIFO	-
Routers	14	1.00E-04	-	FIFO	-
Enlace	-	1.00E+01	2.00	-	-
Aplicación	1	-	1.00	-	-
Simulación	-	-	-	-	900

Con estos parámetros configurados, el canal se ajusta a los requerimientos de ancho de banda descritos en la ITU-T explicada en el capítulo 3. En esta experiencia se aplican parámetros de simulación a nivel de aplicación, los cuales se especifican en el archivo de configuración omnetpp.ini que se encuentra en el anexo 4.

La aplicación se refiere a un tráfico de video streaming el cual se encapsula en UDP para el transporte el cual es transmitido a una velocidad de aproximadamente 1Mbps repartidos en 1000 paquetes de 1Kb los cuales se envían en 1 seg. Las colas que se implementan en los nodos son del tipo FIFO, es decir, First In First Out. Finalmente indicar que la simulación tiene una duración máxima de 15 minutos para ambos casos.

Caso 1: Arquitectura MPLS LDP

Al comenzar la simulación, existe un intercambio de información de LIB así como los paquetes Hello con los cuales descubre a sus peers para intercambiar paquetes etiquetados MPLS. Este intercambio de información sigue los parámetros especificados en el archivo omnetpp.ini, en los que se especifica un intervalo entre mensajes de 2 segundos y un tiempo de espera de paquetes hello de 6 segundos. Nótese que la señalización y el intercambio de información, llenan las colas de los routers parcialmente al iniciarse la simulación. Esta ocupación se da en porcentajes los cuales se indican a los respectivos lados de cada interfaz.

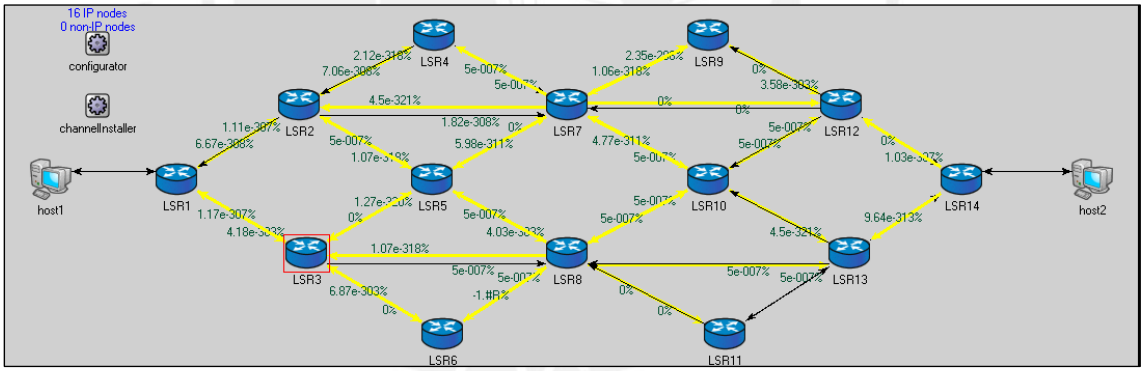

Figura 5.2: Topología MAN. Intercambio de información de los routers LSR

El intercambio finaliza aproximadamente a los 5 segundos de iniciada la simulación. E el tráfico de video streaming se envía desde el host2, (servidor de video streaming), con destino al host1 (cliente de video streaming). Los paquetes que conforman a este tráfico serán enviados a través de una VPN LSR14-LSR1. La VPN que se ha creado sigue el camino LSR14- LSR13- LSR8- LSR3- LSR1 que se caracteriza por una carga de 1MBps y como se puede percibir en la figura 5.3, las colas de los routers son llenadas casi en su totalidad. Finalmente, a los 56seg. de iniciada la prueba, se deja de transmitir el tráfico de video streaming, se despejan las colas y sólo se encuentra tráfico de paquetes Hello.

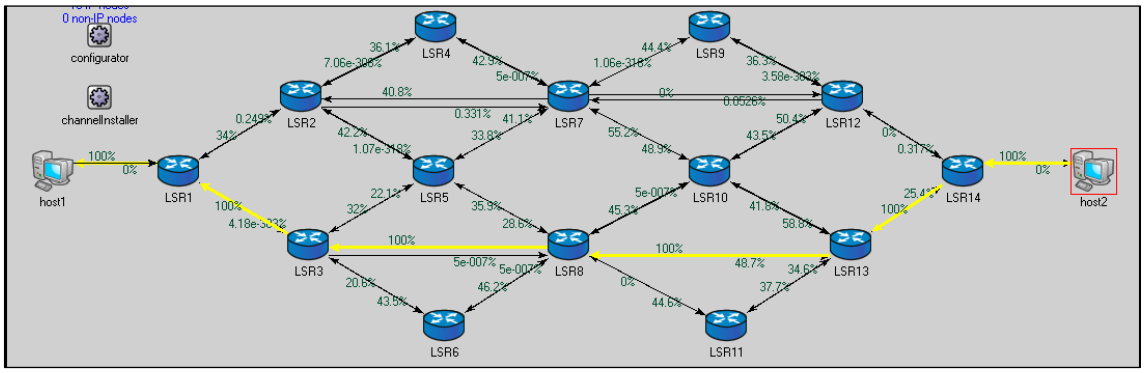

Figura 5.3: Topología MAN. Creación de la VPN LSR14-LSR1

A continuación se presentan los resultados que se obtuvieron después de la simulación de la topología con los parámetros señalados anteriormente:

Tabla 5.2: Resultados de Simulación MAN/LDP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	165.27	164.71	166.73

Puede observarse que conforme pasa el tiempo, los paquetes tardan más en llegar a su destino. Esto se explica por el hecho que las colas se van llenando conforme se envía el tráfico, y al ser UDP, tiende a copar el canal o en este caso la VPN.

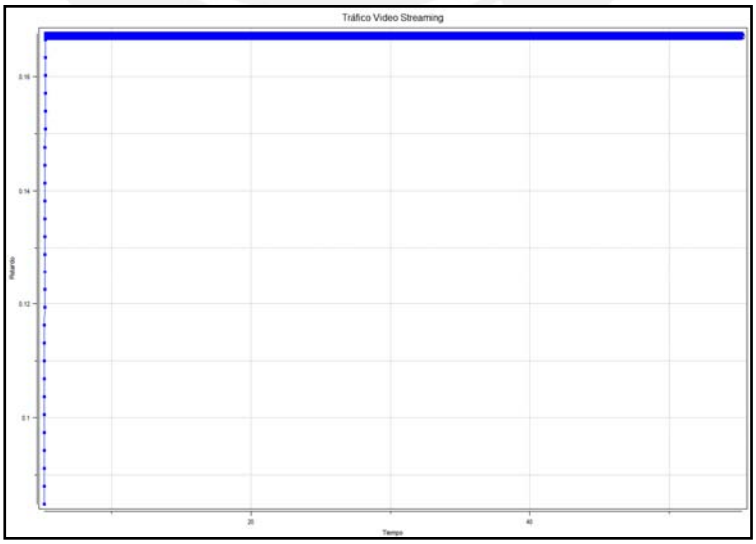

Figura 5.4: Retardo del tráfico video streaming VPN LSR14-LSR1

Vemos que este tipo de tráfico cumple con los requerimientos de las clases 2, 3, 4, 5 y 7 propuestas por la ITU-T y.1541 explicadas en el capítulo 4 ya que su media se encuentra debajo de los 400ms pero arriba de los 100ms de retardo.

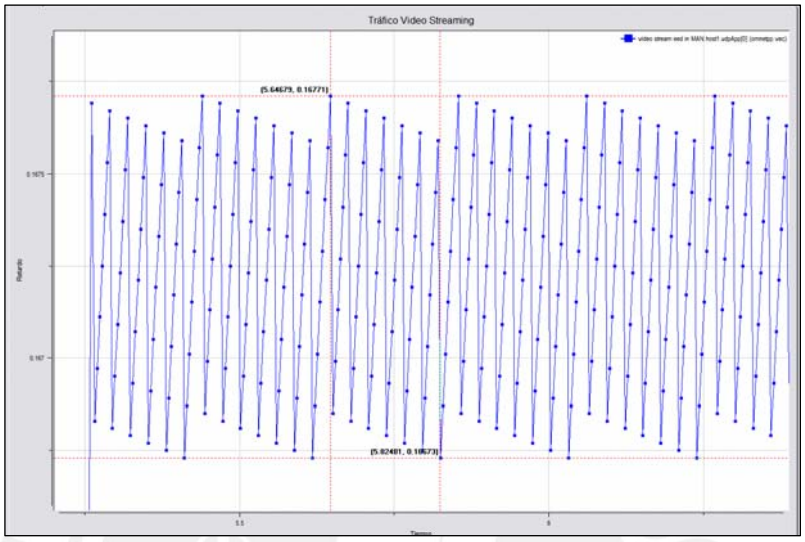

Figura 5.5: Retardo del tráfico video streaming VPN LSR14-LSR1

Caso 2: Arquitectura MPLS RSVP-TE

Al comenzar la simulación, se tiene un intercambio de información de LIB, de paquetes Hello y señalización para la creación del nuevo túnel entre el router LSR14 y LSR1. Este intercambio sigue los parámetros especificados en el archivo omnetpp.ini que se encuentra en el anexo 4. La configuración del túnel VPN LSR14-LSR1 sigue la configuración de los archivos XML, los cuales indican una VPN basada en protocolo de enrutamiento en la capa de red.

Este intercambio de información finaliza a los pocos segundos pero toma más tiempo que MPLS LDP. Aproximadamente a los 5 segundos de iniciada la simulación, el tráfico de video streaming se envía desde el host2, el cual funciona como servidor de video streaming, con destino al host1, el cual funciona como cliente de video streaming. Los paquetes que conforman a este tráfico serán enviados a través de la VPN LSR14-LSR1 que fue creada por LSR14 hace unos instantes gracias a RSVP.

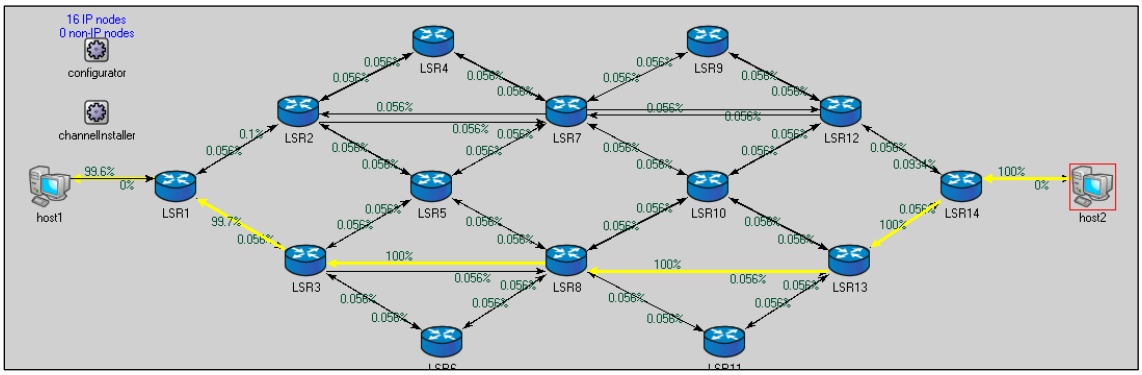

Figura 5.6: Topología MAN. Intercambio de información de los routers LSR

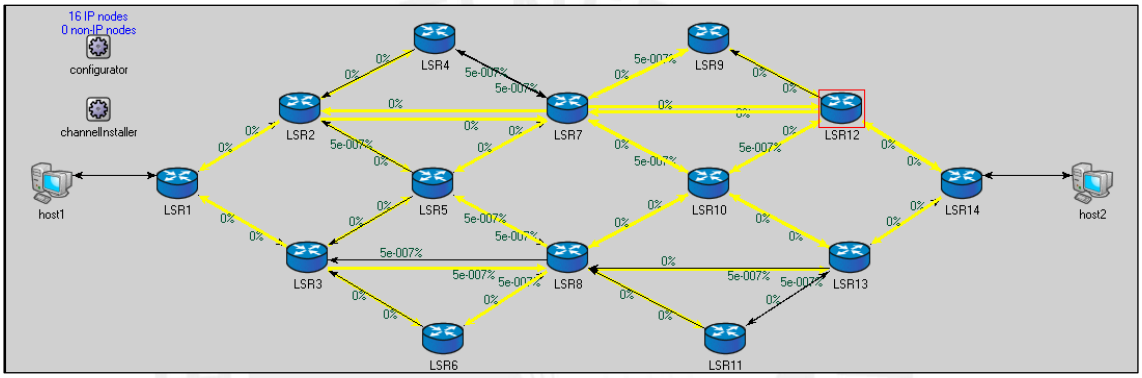

Figura 5.7: Topología MAN. Creación de la VPN LSR14-LSR1

La VPN creada sigue el camino LSR14- LSR13- LSR8- LSR3- LSR1 el cual se caracterizará por ser un túnel dinámico, es decir, se basa en información de la capa superior para su formación (OSPF como protocolo de enrutamiento). Las colas de los routers son llenadas casi en su totalidad debido a la señalización RSVP así como los paquetes Hello. Finalmente, a los 57seg., se deja de transmitir el tráfico de video streaming, se despejan las colas y solo se encuentra tráfico de paquetes Hello.

Los resultados obtenidos de esta simulación son los que se muestran en el cuadro:

Tabla 5.3: Resultados de Simulación MAN/RSVP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	245.955	247.888	242.668

La figura 5.7 corresponde al tráfico de video streaming recibido en host 1 luego de se enviado por la VPN RSVP. Puede observarse que conforme pasa el tiempo, los paquetes tardan más en llegar a su destino. Esto se explica por el hecho de que las

colas se van llenando conforme se envía el tráfico, y al ser UDP, tiende a copar el canal o en este caso la VPN.

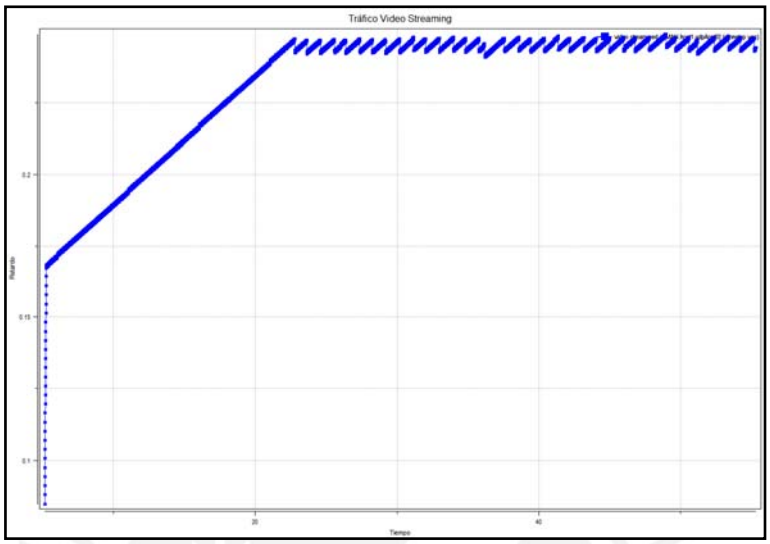

Figura 5.7: Retardo del tráfico video streaming VPN LSR14-LSR1

Nótese que hay una gran diferencia con respecto a MPLS LDP, entre 5.26seg. y los 22.8seg. de la simulación se tiene un crecimiento del tipo lineal del retardo del tráfico desde los 167.204ms de retardo hasta los 247.434ms. Después del crecimiento lineal, el retado llega a estabilizarse entre. Vemos que este tipo de tráfico puede cumplir con los requerimientos de las clases 2, 3, 4, 5 y 7 propuestas por la ITU-T Y.1541.

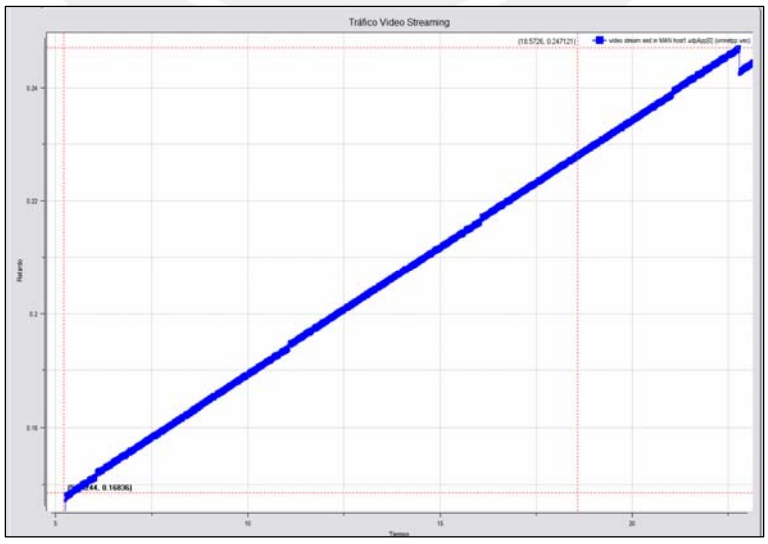

Figura 5.8: Retardo del tráfico video streaming VPN RSVP LSR14-LSR1

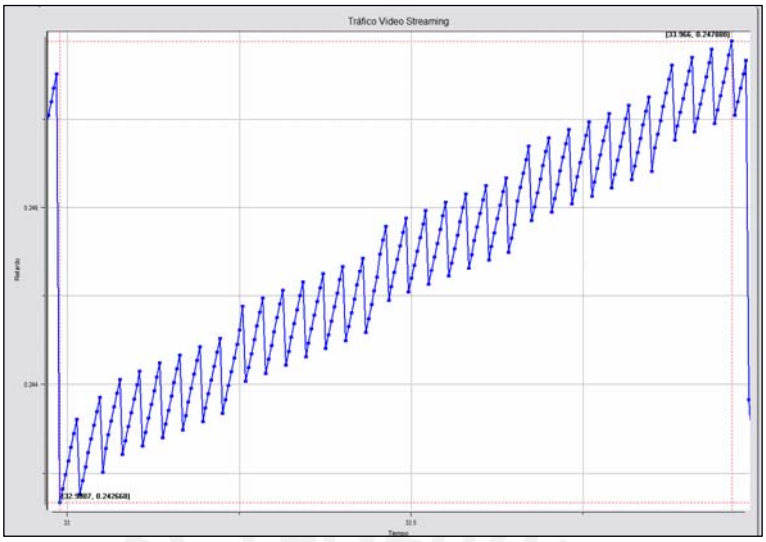

Figura 5.9: Retardo del tráfico video streaming VPN LSR14-LSR1

5.2.2. Topología BACKBONE.

La topología BACKBONE tiene ocho hosts capaces de crear tráficos de interés y 40 routers LSR capaces de manejar MPLS. Esta topología tiene los parámetros que se muestran en el siguiente cuadro:

Tabla 5.4: Parámetros de Simulación BACKBONE simple

	Cantidad	Retardo (ms)	Velocidad de Transmisión (Mbps)	Tipo de Cola	Duración (s)
Host	8	1.00E-04	-	FIFO	-
Routers	40	1.00E-04	-	FIFO	-
Enlace	-	1.00E+01	2.00	-	-
Aplicación	4	-	1.00	-	-
Simulación	-	-	-	-	900

Con estos parámetros configurados, el canal se ajusta a los requerimientos de ancho de banda descritos en la ITU-T explicada en el capítulo 3. En esta experiencia se aplican de simulación a nivel de aplicación, los cuales se especifican en el archivo de configuración omnetpp.ini que se encuentra en el anexo 4.

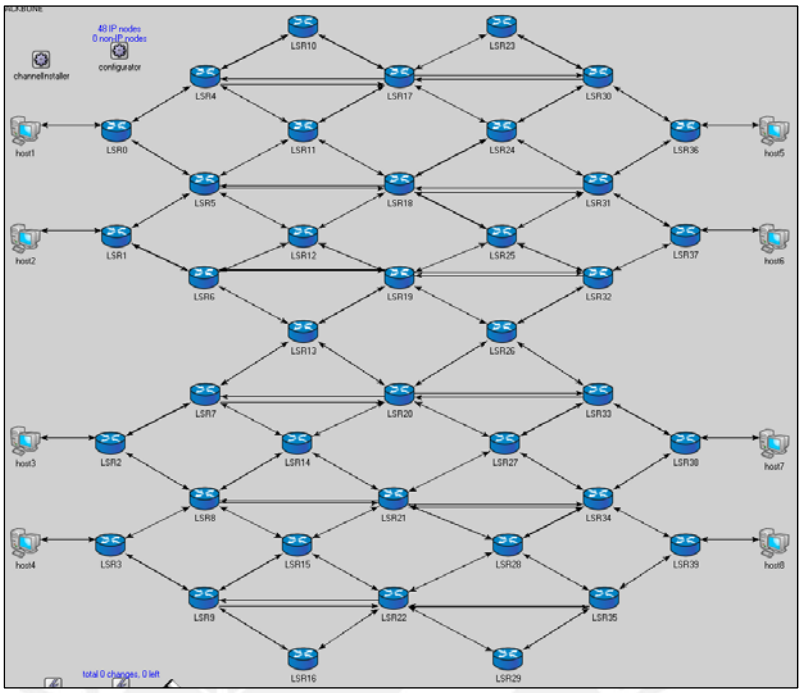

Figura 5.10: Topología BACKBONE

Las aplicaciones que se mencionan son tráficos de video streaming encapsulado en UDP y transmitido a una velocidad de aproximadamente 1Mbps repartidos en 1000 paquetes de 1Kb los cuales se envían en 1 seg. Las colas que se implementan en los nodos son del tipo FIFO, es decir, First In First Out.

Caso 1: Arquitectura MPLS LDP

Al comenzar la simulación, se tiene un intercambio de información de LIB así como los paquetes Hello con los cuales descubre a sus peers con los cuales intercambia paquetes etiquetados MPLS. Este intercambio de información sigue los parámetros especificados en el archivo omnetpp.ini, en los que se especifica un intervalo entre mensajes de 2 segundos y un tiempo de espera de paquetes hello de 6 segundos.

Nótese que la señalización y el intercambio de información llenan las colas de los routers parcialmente al iniciarse la simulación la cual se distingue en porcentajes indicados en los respectivos lados de cada interfaz. El tráfico de señalización es superior al generado en la red MAN por el número de Routers LSR.

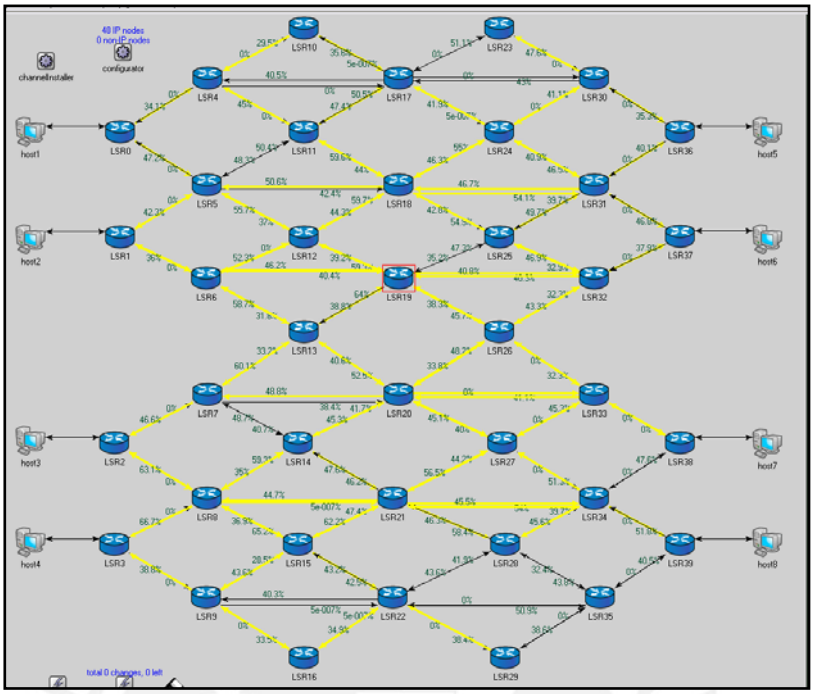

Figura 5.11: Intercambio de información de los routers LSR

El intercambio de información finaliza a los 4 segundos, un segundo después el tráfico de video streaming se envía desde los host3, host4, host7, host8 (servidores de video streaming) cuyos destinos son host5, host6, host1, host2 (clientes).

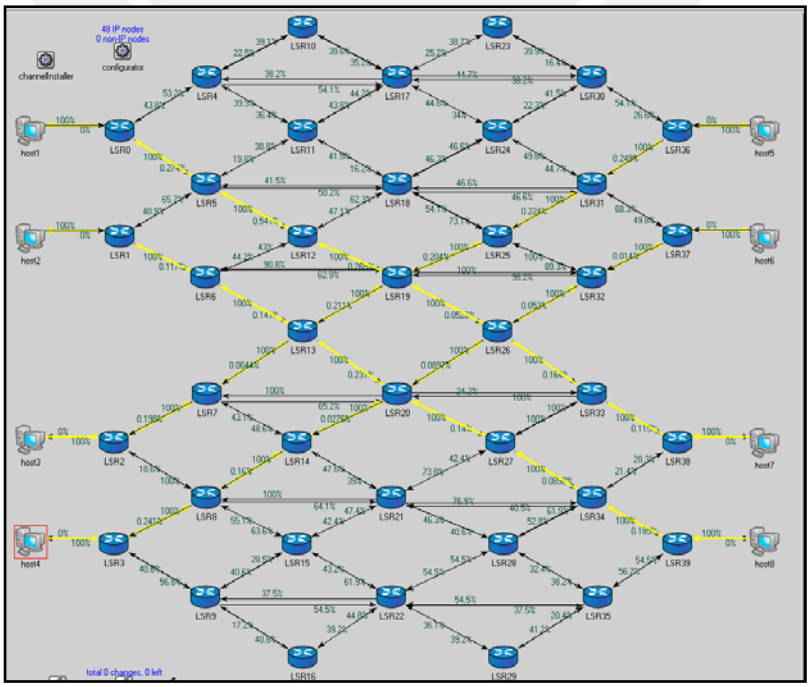

Figura 5.12: Creación y uso de las VPNs en BACKBONE

Los resultados que se obtienen de la simulación e presentan a continuación:

Tabla 5.5. Resultados de Simulación BACKBONE/LDP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	154.17	154.53	153.67
Host2	154.17	154.53	153.67
Host5	154.17	154.53	153.67
Host6	154.17	154.53	153.67

A continuación se presenta los gráficos correspondientes a los tráficos de video streaming recibido en los hosts clientes.

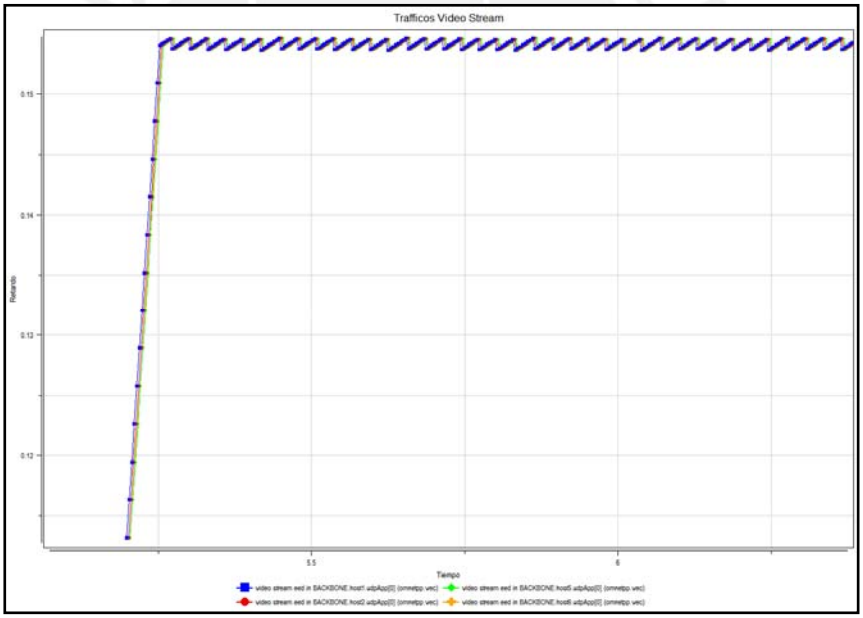

Figura 5.13: Retardos de los tráficos video streaming de las VPNs

Puede distinguirse que conforme pasa el tiempo, los paquetes tardan más en llegar a su destino. Esto se explica por el hecho que las colas se van llenando conforme se envía el tráfico, y al ser UDP, tiende a copar el canal o en este caso la VPN. Debe observarse además que los tráficos tienen un comportamiento con respecto a sus retardos muy similar, casi están superpuestas las gráficas que se han generado. Esto quiere decir que cada VPN tuvo un entorno similar de funcionamiento.

Nótese que las curvas están distanciadas un pequeño diferencial antes de volverse estables, el cual llega a estabilizarse a los pocos segundos.

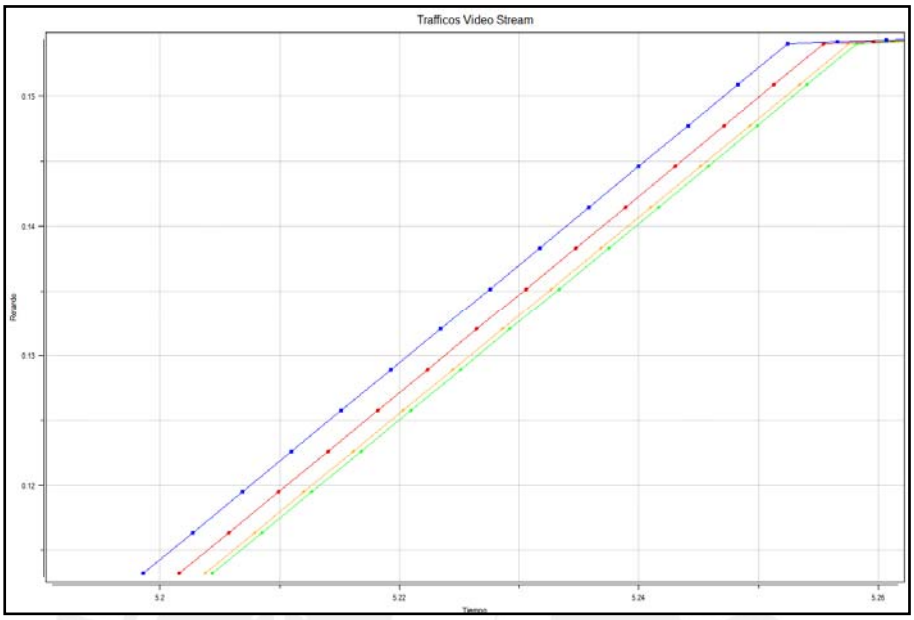

Figura 5.14: Retardos de los tráficos video streaming

Vemos que este tipo de tráfico puede cumplir con los requerimientos de las clases 2, 3, 4, 5 y 7 propuestas por la ITU-T Y.1541 ya que su media se encuentra debajo de los 400ms pero arriba de los 100ms de retardo. Indicar además que estas curvas presentan un desfase una con respecto de las otras.

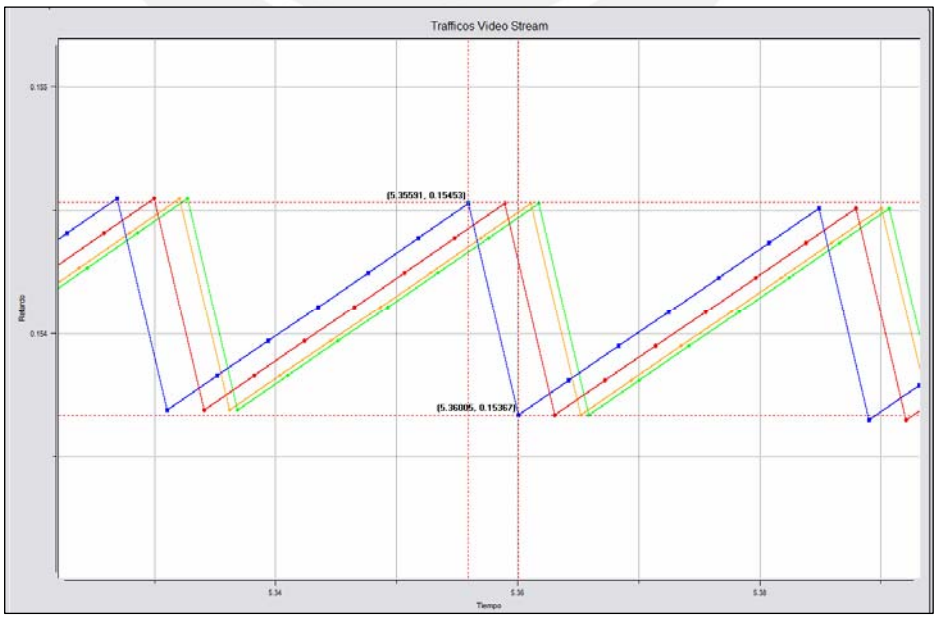

Figura 5.15: Retardo de los tráficos video streaming

Caso 2: Arquitectura MPLS RSVP-TE

Al comenzar la simulación, se tiene un intercambio de información de LIB así como los paquetes Hello; por otro lado la señalización de los routers LSR2, LSR3, LSR38 y finalmente LSR39 con la cual cada LSR será capaz de crear su propio túnel con su destino respectivo. Este intercambio sigue los parámetros especificados en el archivo de configuración omnetpp.ini que se encuentra en el anexo 4 del presente documento.

Análogamente a lo visto en la topología MAN, la configuración de los túneles VPN sigue la configuración de los archivos xml, los cuales indican el uso del protocolo de enrutamiento para la creación de los túneles RSVP. Nótese que la señalización y el intercambio de información, someten a la red a un tráfico constante.

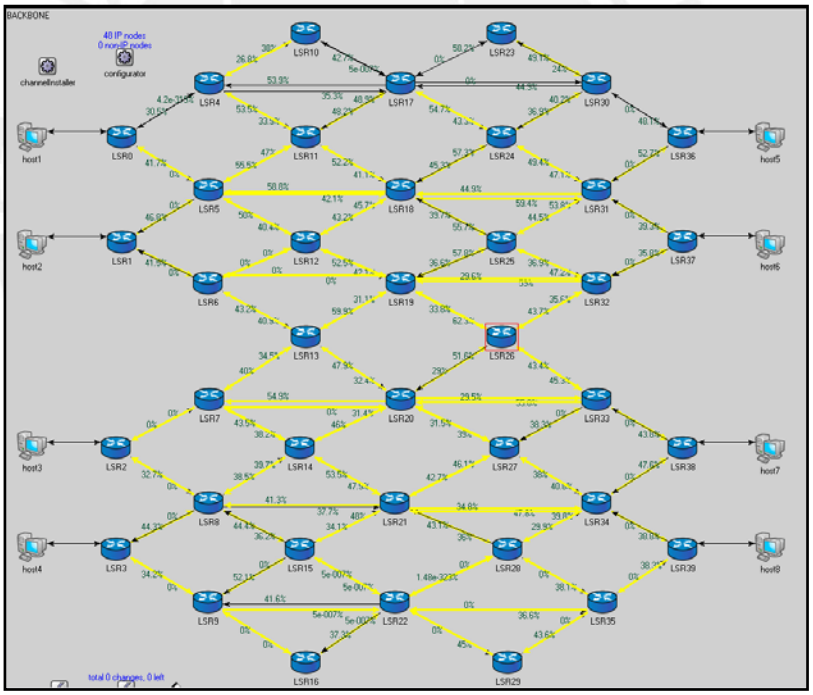

Figura 5.16: Topología BACKBONE RSVP-TE

Este intercambio de información finaliza a los 4 segundos. Aproximadamente a los 5 segundos de iniciada la simulación, el tráfico de video streaming se envía desde los host3, host4, host7, host8, (servidores de video streaming) cuyos destinos son host5, host6, host1, host2 (clientes).

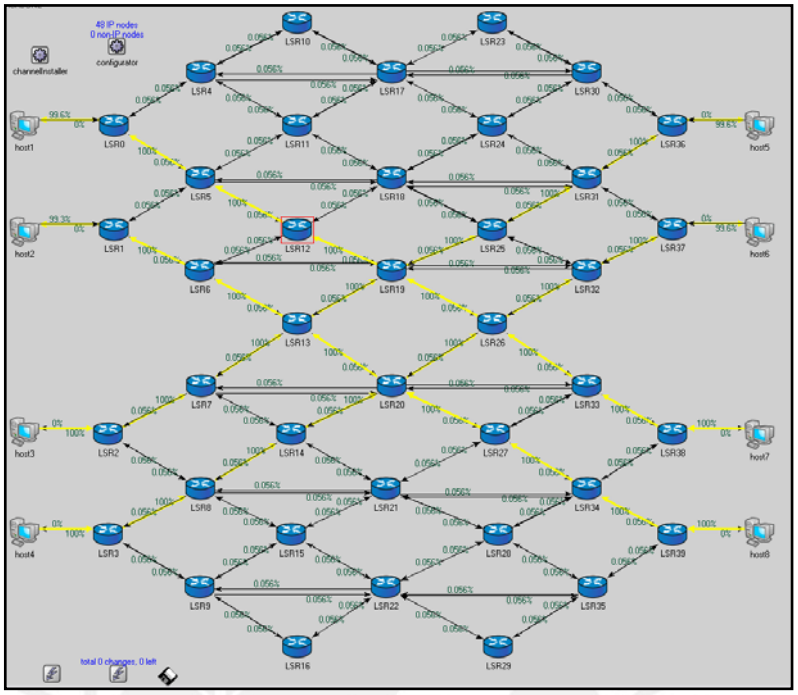

Figura 5.17: Topología BACKBONE. Creación y uso de las VPNs

Las VPNs se caracterizaran por un tráfico de datos de 1MBps, las colas de los routers son llenadas casi en su totalidad en especial los del core no solo por el tráfico de video streaming sino por la señalización RSVP para mantener los túneles. Finalmente, a los 58seg de iniciada la prueba, se deja de transmitir el tráfico de video streaming.

A continuación se presentan los resultados obtenidos de lo descrito anteriormente:

Tabla 5.6: Parámetros de Simulación BACKBONE/RSVP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	191.888	193.804	188.328
Host2	191.888	193.804	188.328
Host5	191.888	193.804	188.328
Host6	191.888	193.804	188.328

Puede observarse que conforme pasa el tiempo, los paquetes tardan más en llegar a su destino dado que las colas se van llenando conforme se envía el tráfico, lo cual produce que se sature el canal o VPN. Debe observarse además que los tráficos tienen un comportamiento con respecto a sus retardos muy similar, casi están superpuestas; cada VPN tuvo un entorno similar de funcionamiento.

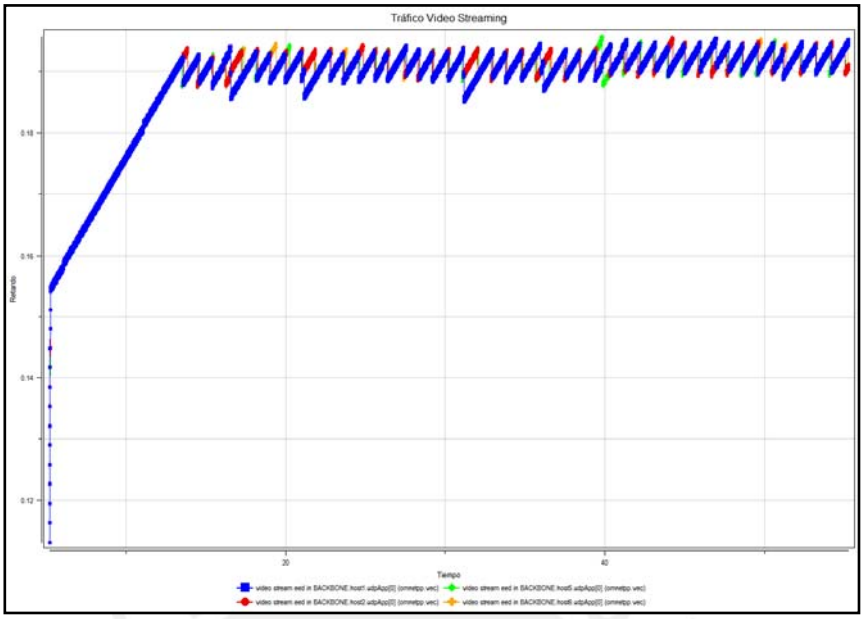

Figura 5.18: Retardos de los tráficos video streaming de las VPN

Nótese que las curvas están distanciadas un intervalo pequeño antes de volverse estables (énfasis en la sección lineal); pero, a diferencia del caso anterior, la diferencial es variable. Nótese que en el gráfico que la curva naranja a veces supera a las demás curvas, la curva verde supera a la roja en ciertos tramos, los picos de las curvas tienden a juntarse y luego separarse.

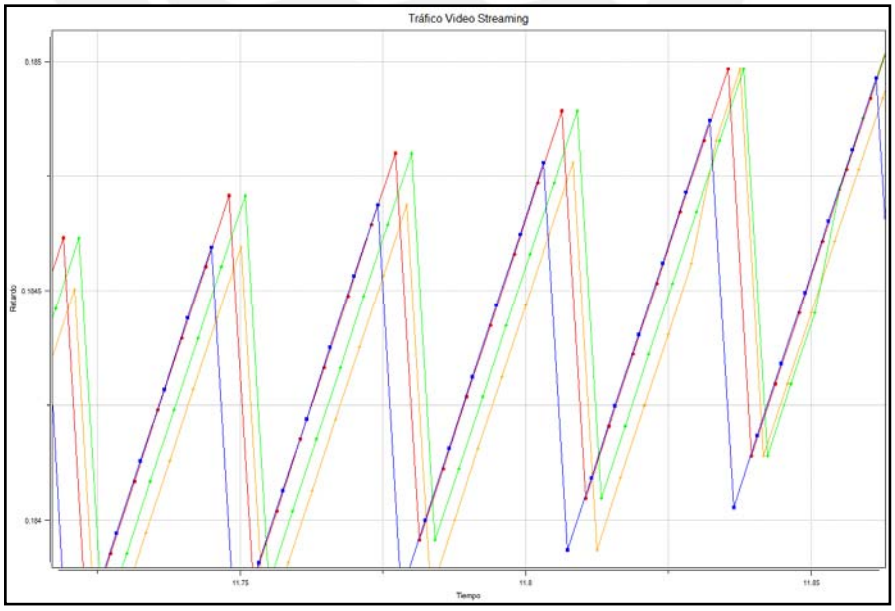

Figura 5.19: Retardos de los tráficos video streaming

La curva azul tiene un comportamiento atípico a los 31.1794seg ya que su retardo disminuye a 0.185032seg, probablemente debido a una pérdida en una cola. Este tipo de tráfico cumple con los requerimientos de las clases 2, 3, 4, 5 y 7 propuestas por la ITU-T Y.1541 ya que esta por debajo de los límites solicitados por la recomendación.

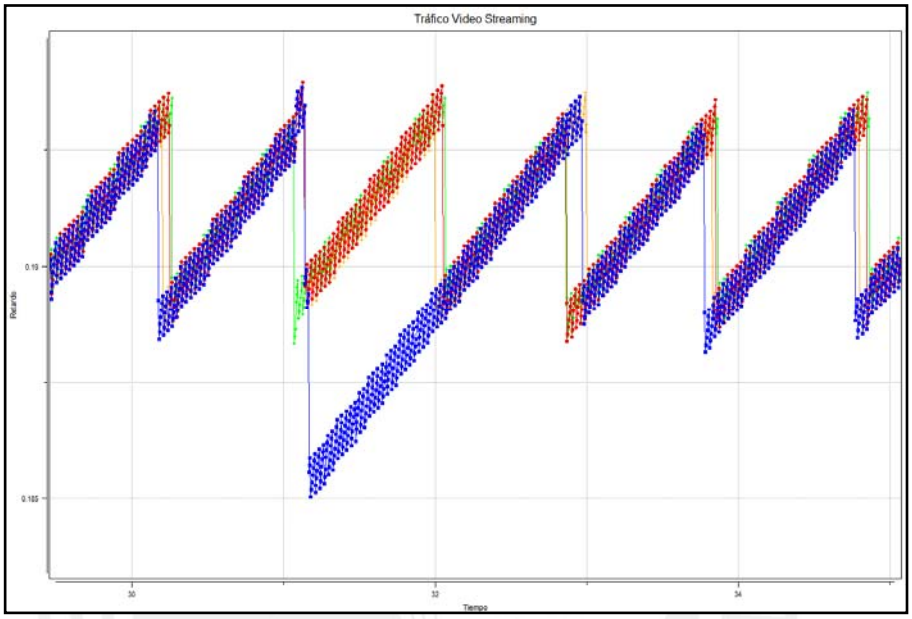

Figura 5.20: Retardo y desfase de los tráficos video streaming

5.3. Evaluación de Capacidad de Garantizar Calidad de Servicio en las Arquitecturas MPLS LDP y Arquitecturas MPLS RSVP-TE.

En el apartado anterior se hizo el estudio de Calidad de Servicio en redes de diferentes tamaños así como de diferentes capacidades de recursos. Ahora se altera el escenario agregando más recursos a la red, principalmente ancho de banda; así mismo también se ha introducido más carga de tráfico en la red, es decir, tipos de tráfico que no es de interés para nuestras redes terminales pero que si consumirán diferentes tipos de recursos de red e impactará en la performance de la topología.

En este apartado se tiene como objetivo poner a prueba la capacidad que nos ofrecen las arquitecturas MPLS, tanto las basadas en LDP como las basadas en RSVP, de poder garantizar la Calidad de Servicio (QoS) a los tráficos de interés en escenarios en los cuales se pueden encontrar tráficos ajenos a los que deseamos transmitir pero que consumirán recursos de red ya que pertenecen a la misma y que, por lo tanto, afectarán al envío y recepción de nuestro tráfico de interés. Con el resultado de las

simulaciones podrá recomendarse incluso una arquitectura para llevar los tráficos más importantes a los destinos que se han elegido; así también podemos observar los recursos que se tienen que destinar para poder asegurar recursos en la red dependiendo del protocolo y la señalización que este último este utilizando para poder hacer la reserva en los nodos de la red y asegurar.

Cabe resaltar que estos escenarios que se proponen en las simulaciones tratan de acercarse a lo que convencionalmente se tiene en una red real como la Internet comercial, es decir, dentro de todos los tráficos existentes que pueden estar cursando en una red, solo se desea asegurar ciertos tipos de tráfico a través de una VPN Virtual Private Network ya que solo algunos cliente lo requieren, para lo cual se debe de utilizar políticas de asignación de recursos de red a los mismos, de tal modo que en los instantes en que se tiene un gran demanda de recursos por parte de todos los nodos y/o clientes de la red, en caso disminuya los recursos disponibles de la red por causas internas y/o externas al ISP, el tráfico de importancia pueda ser enviado a los destinos que se han elegido siguiendo las políticas antes mencionadas.

5.3.1. Topología MAN

La topología MAN que se empleará en esta simulación, a diferencia de la usada anteriormente, cuenta con mayor numero de nodos, y además dos Servers, uno de tráfico FTP y otro de tráfico Web los cuales interactuarán en la simulación con los clientes FTP y los clientes Web respectivamente.

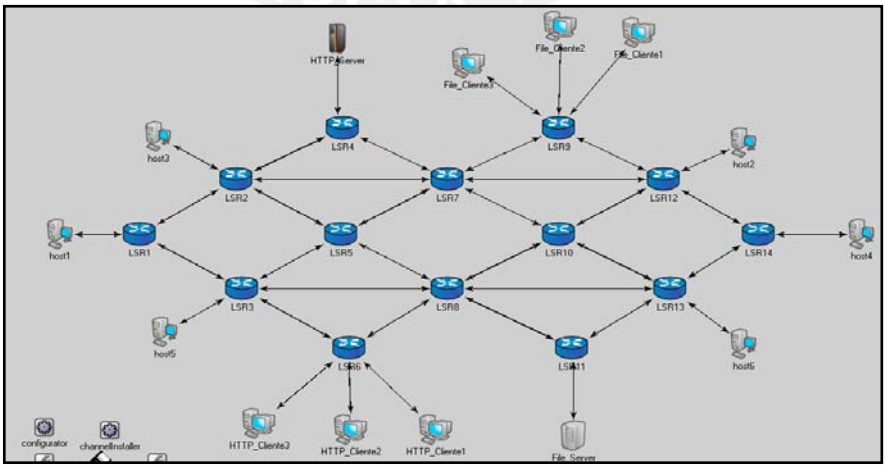

Figura 5.21: Topología MAN

Los parámetros que se utilizaran son los mostrados en el siguiente cuadro:

Tabla 5.7: Parámetros de Simulación MAN Real

	Cantidad	Retardo (ms)	Velocidad de Transmisión (Mbps)	Tipo de Cola	Duración (s)
Host	6	1.00E-04	-	FIFO	-
Routers	14	1.00E-04	-	FIFO	-
Enlace	-	1.00E+01	8.00	-	-
Aplicación	1	-	3.36	-	-
File Service*	1	-	-	-	-
Web Sevice*	1	-	-	-	-
Simulación	-	-	-	-	900

En la experiencia se aplican parámetros de simulación a nivel de aplicación, los cuales se especifican en el archivo de configuración el cual se presenta en el anexo 4. La aplicación que se utilizará para la experimentación es videoconferencia se trasmite por a una velocidad de aproximadamente 3.36Mbps repartidos en 1000 paquetes de 33.6Kp los cuales se envían un intervalo de 1 seg. Las colas son del tipo FIFO, es decir, Fist In First Out. Finalmente indicar que la simulación tiene una duración máxima de 15 minutos para ambos casos.

Caso 1: Arquitectura MPLS LDP

La característica de una topología configurada como MPLS LDP es el intercambio de información de LIB y de Hello al comenzar la simulación, ya que gracias a este proceso de descubrimiento los routers pueden reconocer a sus peers con los cuales intercambia paquetes MPLS. Este intercambio de información sigue los parámetros especificados en el archivo omnetpp.ini, en los que se especifica un intervalo entre mensajes de 2 segundos y un tiempo de espera de paquetes hello de 6 segundos. El intercambio de información entre los LSR es el primer tráfico que presenta la red, el cual llena las colas de los routers parcialmente al iniciarse la simulación.

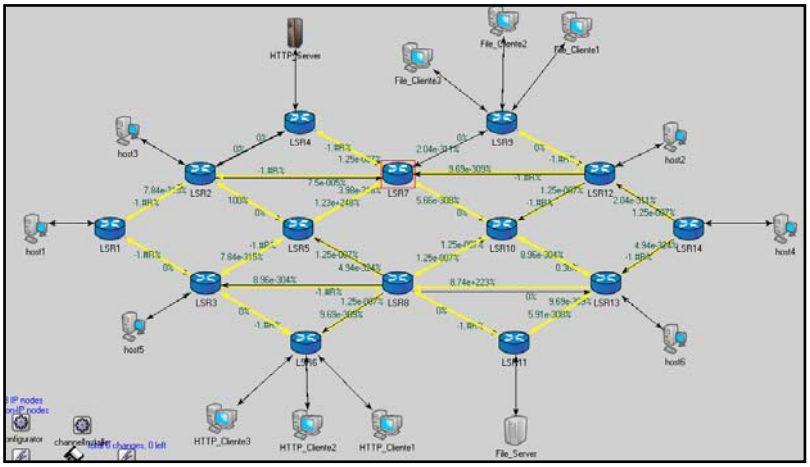

Figura 5.22: Topología MAN. Intercambio de información de los routers LSR

Este intercambio de información finaliza a los pocos segundos. Aproximadamente a los 5 segundos de iniciada la simulación, el tráfico de videoconferencia entre los host que se encuentran en las redes de la derecha con los de la izquierda.

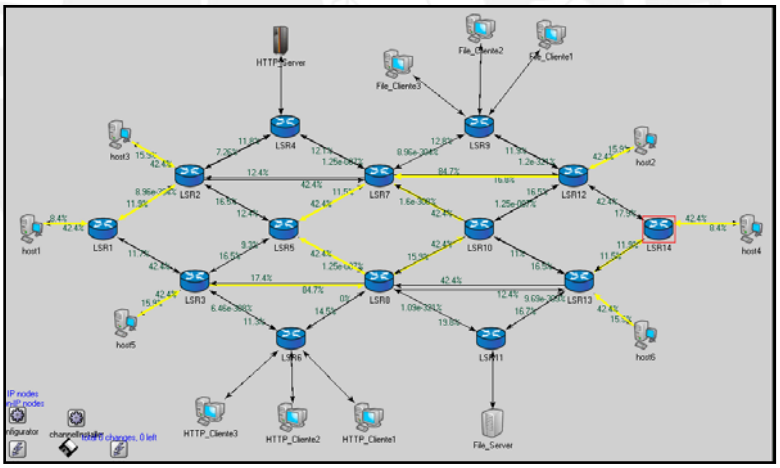

Figura 5.23: Creación de las VPN en la topología WAN

Las VPN que se crean serán usadas para las transmisiones de los paquetes correspondientes a videoconferencia, cuyo tráfico tiene como característica 3.36Mbps de consumo de ancho de banda. Después de unos instantes comenzados la videoconferencia, se presenta un tipo de tráfico que para los host de videoconferencia no es de interés pero que afectará a los LSR ya que consumirá recursos de la red tales como ancho de banda, tiempo de procesamiento, espacio en las colas, entre otros. Estos tráficos que van a afectar el rendimiento de la videoconferencia son del tipo ráfaga, es decir, no es un tráfico continuo como es el tráfico de interés (el tráfico

de videoconferencia se aproxima a un tráfico CBR (Constant Bit Rate) sino que son tráficos que tienen picos muy altos de consumo de recursos en ciertos instantes de tiempo, después de lo cuales no consumen recursos o los consumen pero de forma muy mínima además de no poder acomodarse a ningún modelo estadístico.

Los tráficos que se tienen en la simulación son los tráficos correspondientes a transferencia de archivos FTP File Transfer Protocol y tráfico web HTTP HyperText Transfer Protocol, el cual es muy usado en la Internet además de diferentes aplicativos. Las VPN deben tener prioridad para los routers LSR por lo que se espera una afectación mínima de tal forma que no afecte la Calidad de Servicio (QoS) que se requiere para la videoconferencia entre los host.

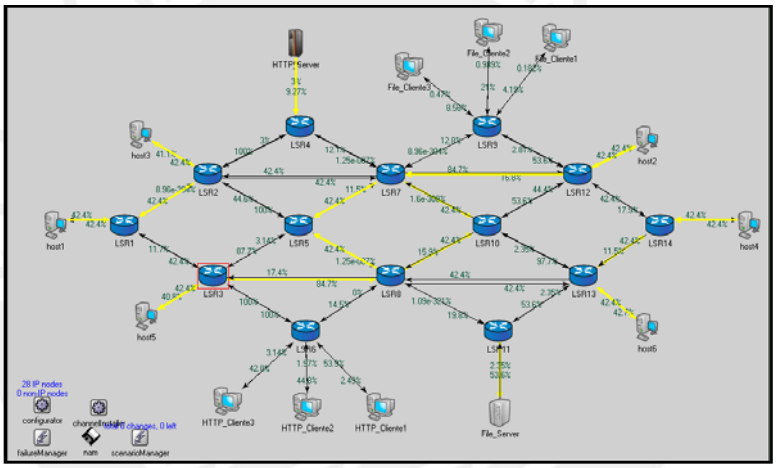

Figura 5.24: Transmisión de tráficos FTP y HTTP

En el siguiente cuadro se muestran los resultados obtenidos de la simulación:

Tabla 5.8: Resultados de Simulación MAN Real/LDP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	85.417	85.418	85.416
Host2	71.182	81.573	71.180
Host3	71.181	87.748	71.105
Host4	85.416	85.417	71.850
Host5	71.181	83.921	71.100
Host6	71.181	87.748	71.430

En el cuadro mostrado así como en la figura 5.25 puede observarse el comportamiento del retardo para los tráficos son impactados de formas diferentes por la performance de la red. El tráfico más impactado es del host1. Los tráficos correspondientes al host3 y al host5 presentan valores más bajos de retardo en el tiempo de simulación que el host 1, pero no poseen un comportamiento tan estable.

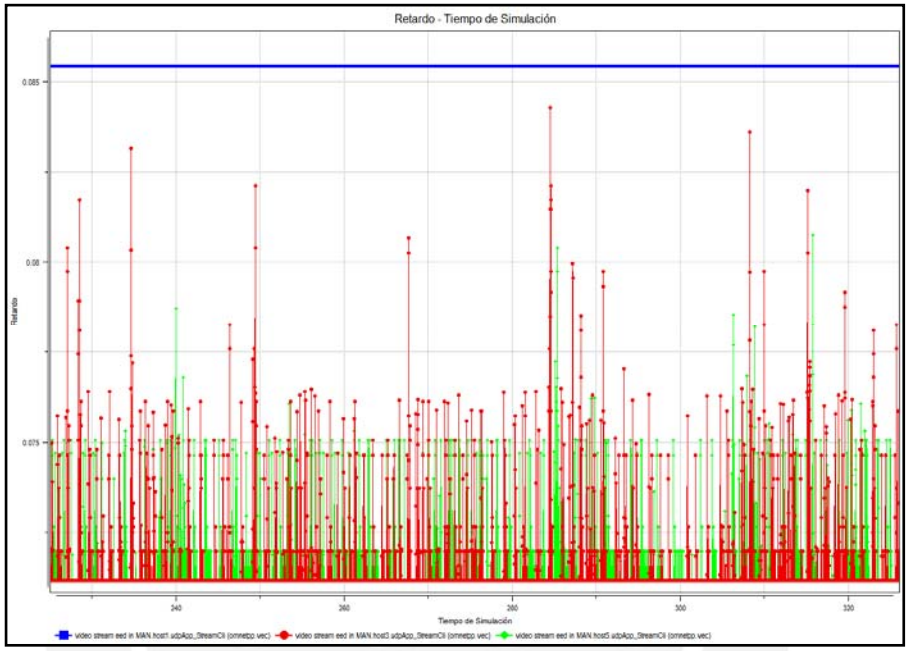

Figura 5.25: Retardos de los tráficos videoconferencia de los Host 1, 3 y 5

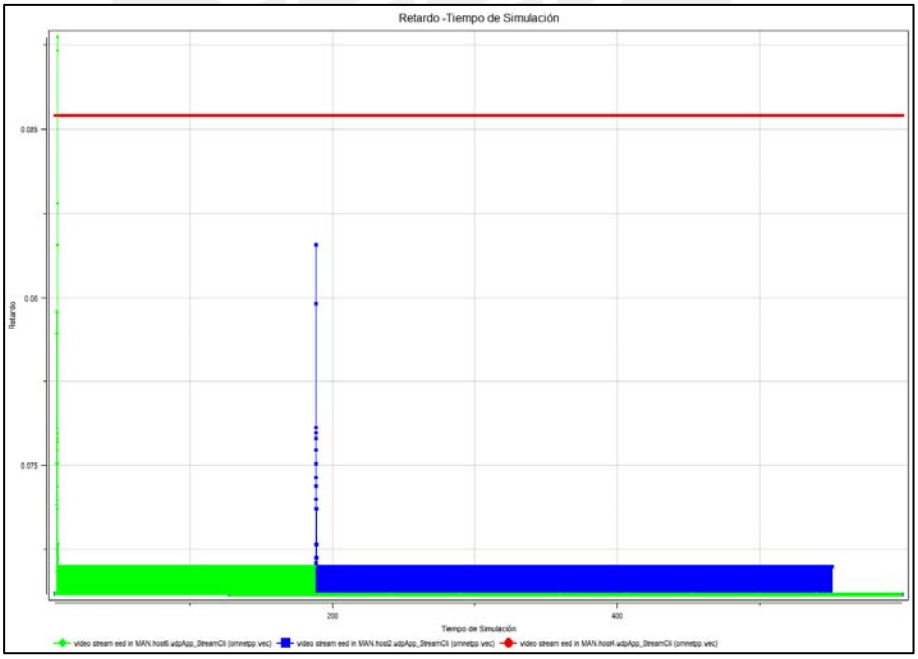

Figura 5.26: Retardos de los tráficos videoconferencia de los Host 2, 4 y 6

Así también, puede concluirse de los resultados así como de la figura 5.26 que el retardo para los tráficos en los host 2, 4, y 6 son impactados de formas diferentes por la performance de la red, de los cuales el más impactado es del host4.

Se observa además que a lo largo de la simulación, los host6 y host2 tienen comportamientos inversos de sus retardos como se muestra en la figura 5.27.

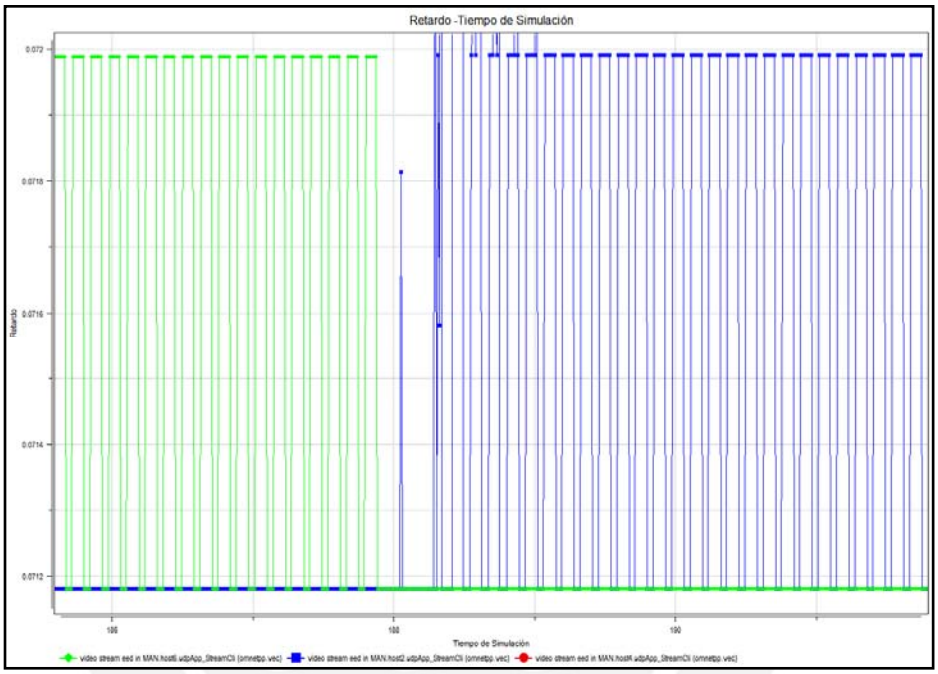

Figura 5.27: Cambio de comportamiento del Host6 y del Host2

Caso 2: Arquitectura MPLS RSVP-TE

Como se explicó en los apartados anteriores, la característica de una topología configurada como MPLS RSVP-TE es el intercambio de información de LIB, asignación de recursos, así como los paquetes Hello. La señalización que se usa en este tipo de topologías es crítica y de intercambio constante, diferencia principal que lo diferencia de LDP, ya que la señalización consumirá recursos de los routers LSR para poder configurar los túneles, asignarles los recursos requeridos, y mantenerlos activos. Los parámetros y las configuraciones para que se propicie este intercambio de información se especifican en el anexo 4.

Los archivos XML contienen la información de os túneles que se crearán en la topología, es decir, estos indican a los ILER los parámetros para los túneles que se crearán en base al destino del tráfico. Nótese que la señalización y el intercambio de información, llenan las colas de los routers parcialmente al iniciarse la simulación como puede apreciarse en la figura. Esta ocupación se da en porcentajes los cuales se indican a los respectivos lados de cada interfaz.

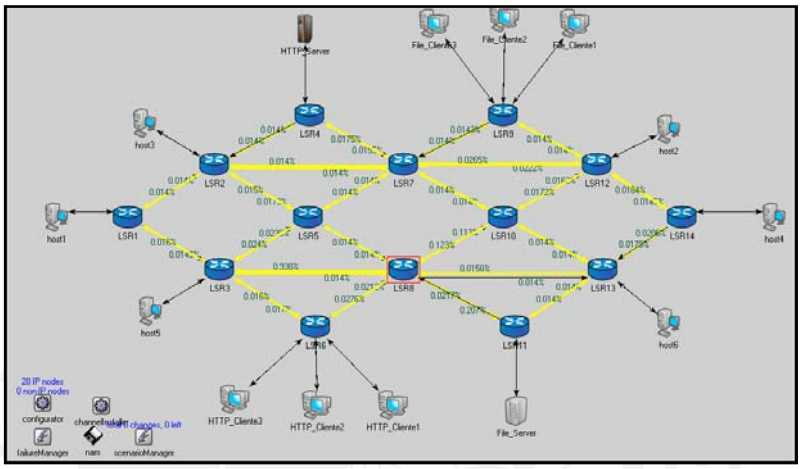

Figura 5.28: Topología MAN. Intercambio de información de los routers LSR

Este intercambio de información finaliza a los pocos segundos. Aproximadamente a los 5 segundos de haber comenzado la simulación comienza la transmisión de paquetes de videoconferencia entre los nodos extremos los cuales usarán los túneles que se han creado bajo RSVP.

Las VPN's que se han creado llevan el tráfico de los host extremos tienen la característica de ser túneles dinámicos, es decir, se basa en la información de la capa superior para transmitir un tráfico de 3.36MBps. Las colas de los routers son llenadas casi en su totalidad debido a la señalización de RSVP así como los paquetes Hello que se necesitan para que los túneles creados puedan seguir funcionando, en este caso, con una asignación de recursos de 5.5Mbps para el tráfico de videoconferencia.

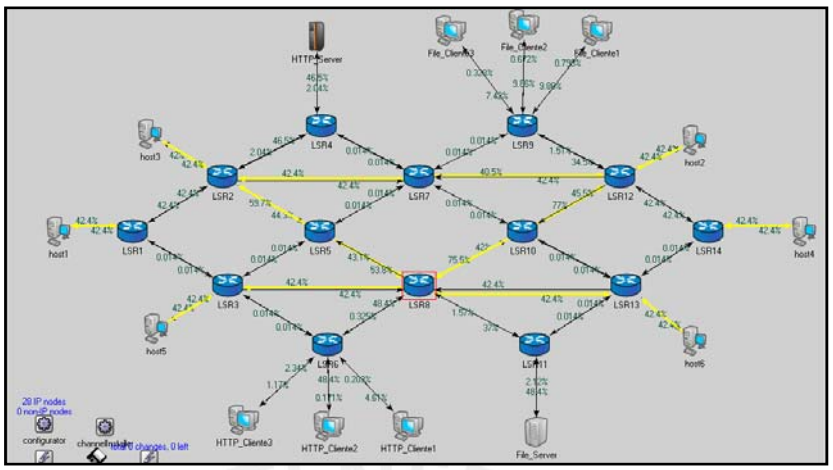

Figura 5.29: Topología MAN. Creación de la VPN LSR14-LSR1

Al igual que en la simulación de MPLS LDP, se tiene un tráfico que no es de interés para los nodos extremos que afectará a la performance de la red dado que consume recursos. Estos tráficos son del tipo ráfaga, tráfico no continuo que presenta picos muy altos de consumo de recursos en ciertos instantes de tiempo, después de lo cuales no consumen recursos o los consumen de forma muy mínima. Los tráficos que se tienen en la simulación son los tráficos correspondientes a transferencia de archivos FTP File Transfer Protocol y tráfico web HTTP HyperText Transfer Protocol, el cual es muy usado en la Internet además de diferentes aplicativos.

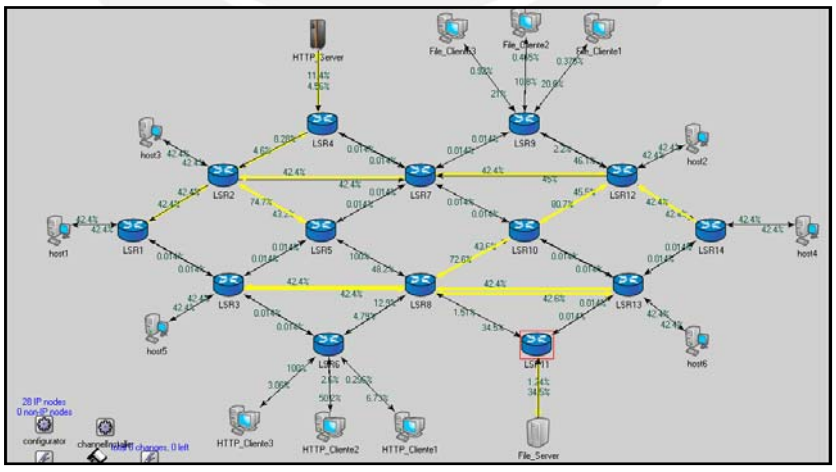

Figura 5.30: Topología MAN. Transmisión de tráficos FTP y HTTP

Los resultados que se obtuvieron de la simulación son:

Tabla 5.9: Resultados de Simulación MAN Real/RSVP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	85.446	85.449	85.441
Host2	71.121	71.180	71.111
Host3	71.221	85.484	71.121
Host4	85.446	85.447	85.446
Host5	71.188	83.949	71.207
Host6	71.121	76.121	71.103

En la figura 5.31 puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 1, 3, y 5; las VPN's que se formaron para cursar los tráficos son impactados de formas diferentes por la performance de la red.

En la figura 5.32 puede observarse el los comportamientos de los tráficos de los host 2, 4, y 6. El tráfico más impactado es del host4, el cual tiene un mayor retardo; los host2 y al host6 presentan valores más bajos de retardo en el tiempo de simulación pero con una estabilidad menor que la que se puede observar en el caso del host4.

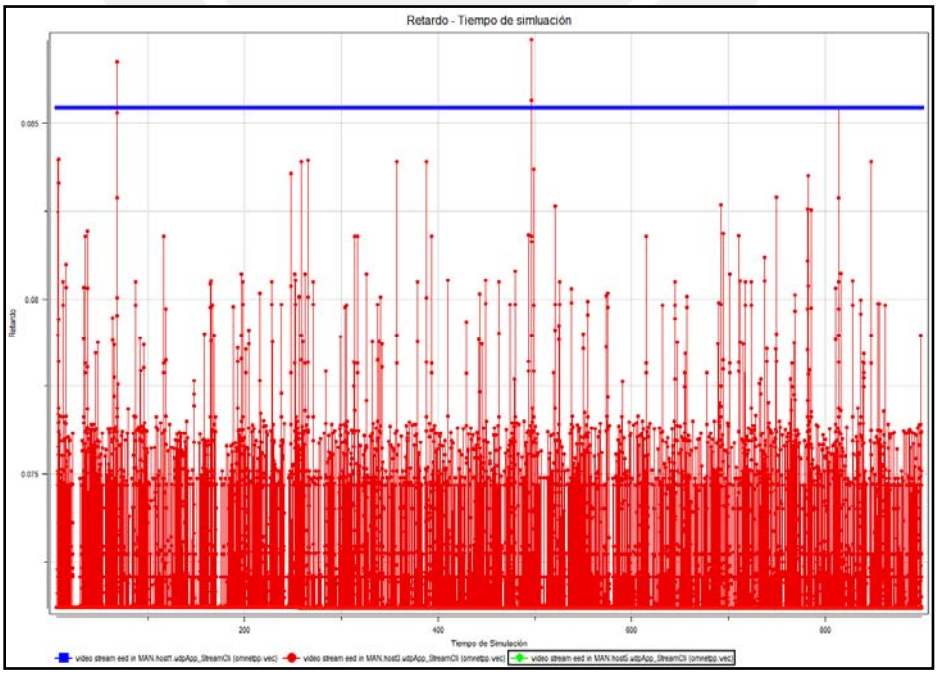

Figura 5.31: Retardos de los tráficos videoconferencia de los Host 1, 3 y 5.

El tráfico más impactado es del host1 al presentar un retardo más acentuado. Los tráficos correspondientes al host3 y al host5 presentan valores más bajos de retardo en el tiempo de simulación, pero no poseen un comportamiento estable como el host1.

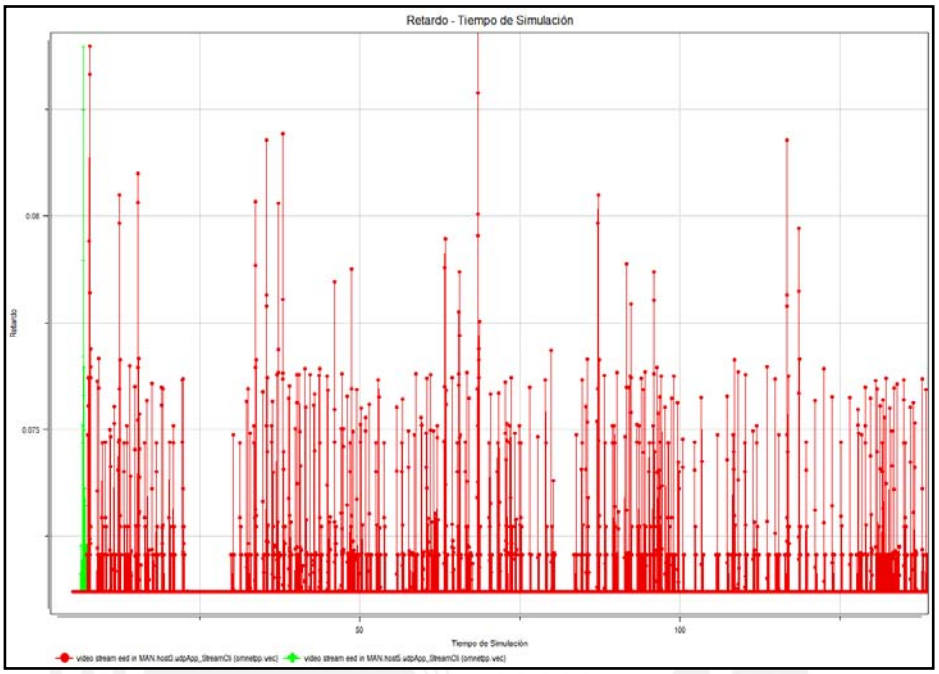

Figura 5.32: Retardos de los tráficos videoconferencia de los Host 1 y 5

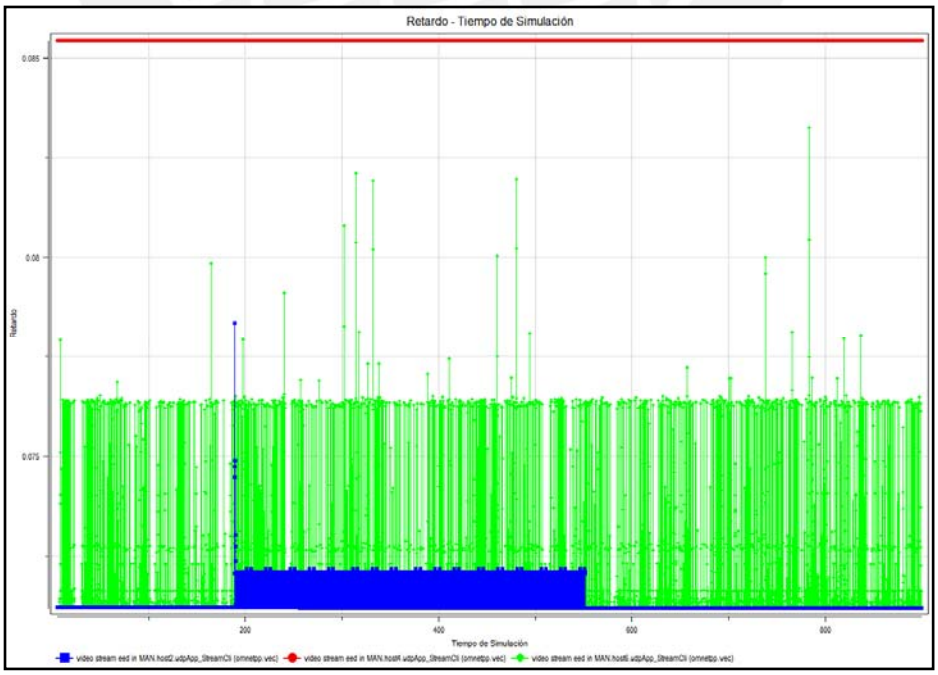

Figura 5.33: Retardos de los tráficos videoconferencia de los Host 2, 4 y 6

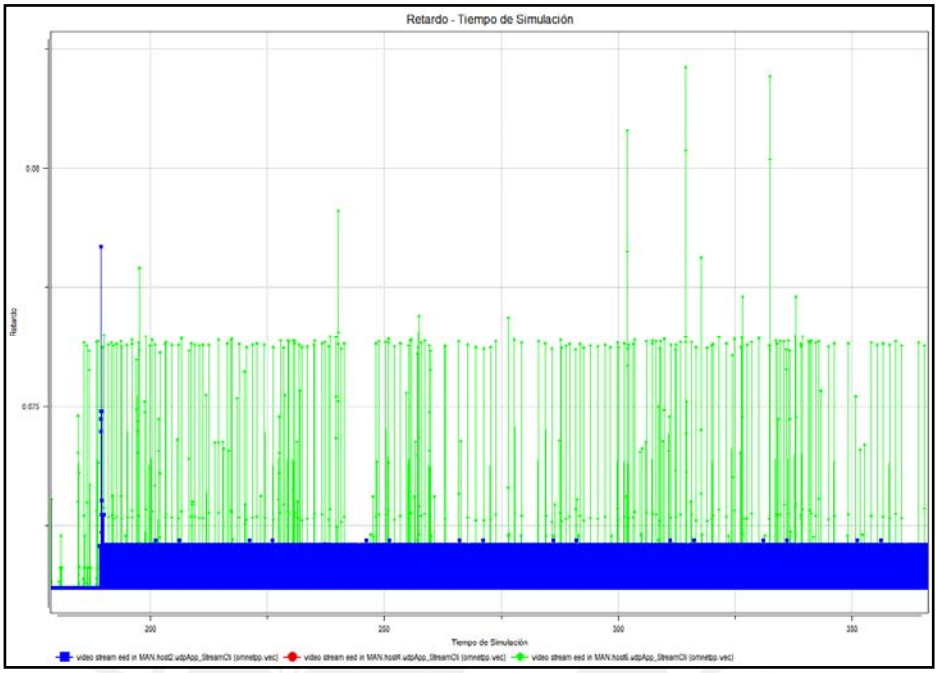

Figura 5.34: Retardo y desfase de los tráficos video streaming

5.3.2. Topología Backbone

La topología BACKBONE que se presenta a continuación, a diferencia de la usada anteriormente, cuenta con mayor numero de host de videoconferencia, y además de presentar dos Servers, uno de tráfico FTP y otro de tráfico Web los cuales interactuarán en la simulación con los clientes FTP y los clientes Web respectivamente.

La configuración a nivel físico de los nodos correspondientes a esta topología así como la interconexión entre ellos se especifica en el archivo BACKBONE_Real.ned. Con estos parámetros, similar al caso de la red MAN descrito, simulamos cuatro tipos de tráfico de video streaming los cuales se encapsulan en UDP, es decir, es una aplicación que usa a UDP para el transporte. Las colas que se implementan en los nodos son del tipo FIFO, es decir, First In First Out con una capacidad máxima de 30 tramas por interfaz. Finalmente indicar que la simulación tiene una duración máxima de 15 minutos para ambos casos.

Los parámetros de simulación son los que se muestran a continuación:

Tabla 5.9: Parámetros de Simulación BACKBONE Real

	Cantidad	Retardo (ms)	Velocidad de Transmisión (Mbps)	Tipo de Cola	Duración (s)
Host	16	1.00E-04	-	FIFO	-
Routers	40	1.00E-04	-	FIFO	-
Enlace	-	1.00E+01	8.00	-	-
Aplicación	4	-	3.36	-	-
File Service[2]	1	1.00E-04	-	-	-
Web Service[1]	1	1.00E-04	-	-	-
Simulación	-	-	-	-	900

La diferencia de retardo se al procesamiento de los paquetes en los nodos (host, routers, Server) y otro retardo muy diferente que depende del medio de transmisión (enlace). Al ser de diferente origen no se les puede relacionar directamente pero si sus efectos serán reflejados en la performance de la red

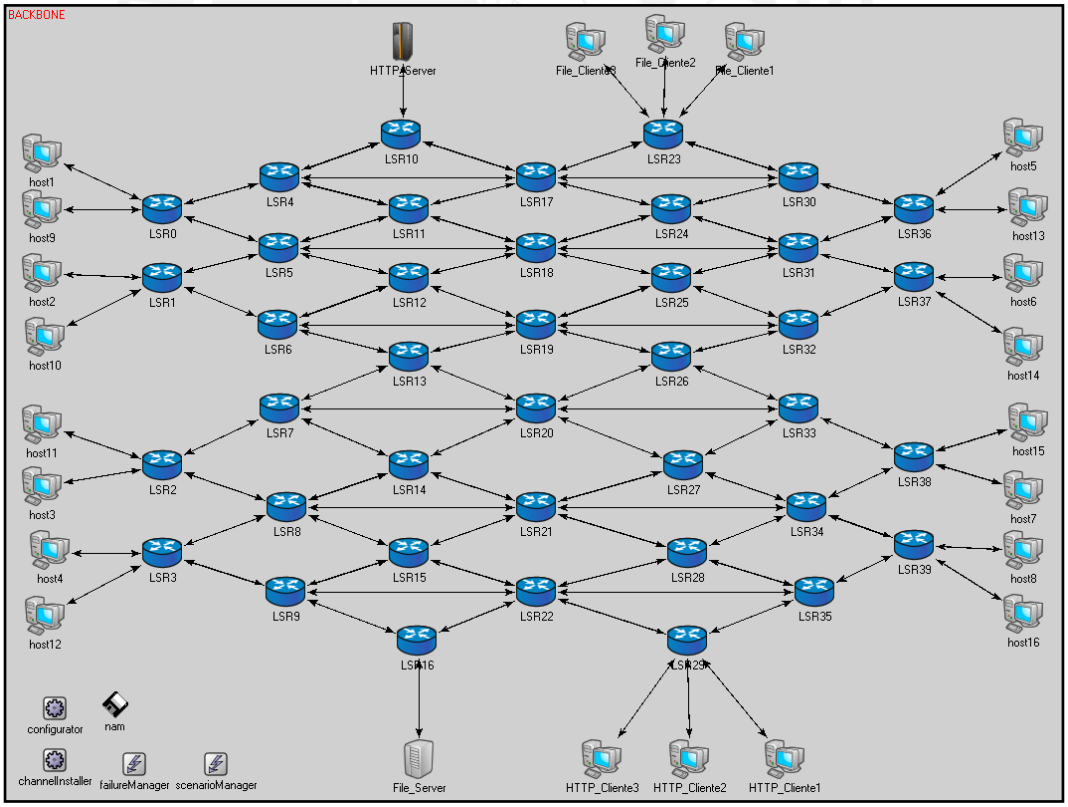

Figura 5.35: Topología BACKBONE

[2] Son tráficos TCP por lo que se adaptan al tráfico cursado de la red, tratando de utilizar el mayor ancho de banda posible del enlace

Caso 1: Arquitectura MPLS LDP

Se observa un intercambio de información LIB y paquetes Hello con los que descubre a sus peers. La señalización y el intercambio de información necesaria para el funcionamiento llenan las colas de los routers parcialmente al iniciarse la simulación. El intercambio de información finaliza a los 5 segundos, tiempo después del cual los host de videoconferencia comienzan a intercambiar tráfico el cual utilizará las VPN's creadas obligadas a compartir recursos de red en los casos que estas pasen por los mimos LSR, lo cual afecta su propio rendimiento si no se tienen los recursos suficientes para poder satisfacer las necesidades de los tráficos.

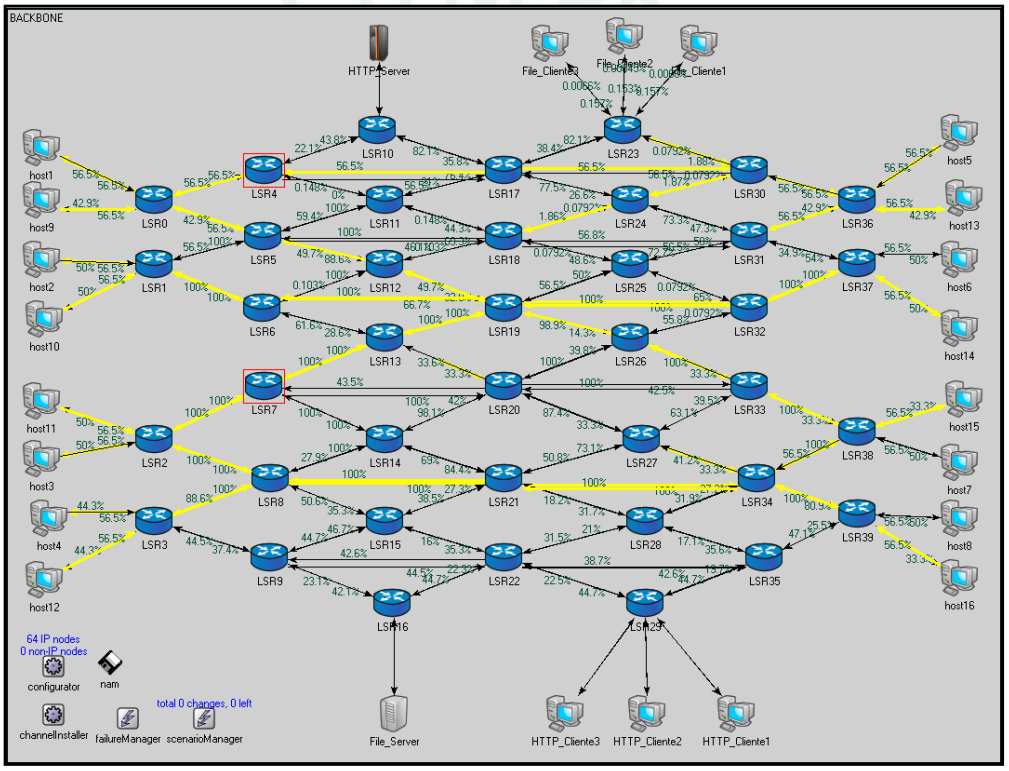

Figura 5.36: Topología MAN. Intercambio de información de los routers LSR

Una vez establecido el tráfico entre los host de videoconferencia, se empieza a trasmitir tráficos que no son de interés que afectará la performance debido al consumo de recursos de estos tráficos. Los tráficos que se tienen en la simulación son los tráficos de transferencia de archivos FTP (File Transfer Protocol) y tráfico Web. Estos tráficos son del tipo ráfaga los cuales presentan picos muy altos de consumo de recursos en ciertos instantes de tiempo, después de lo cuales no consumen recursos o los consumen pero de forma muy mínima.

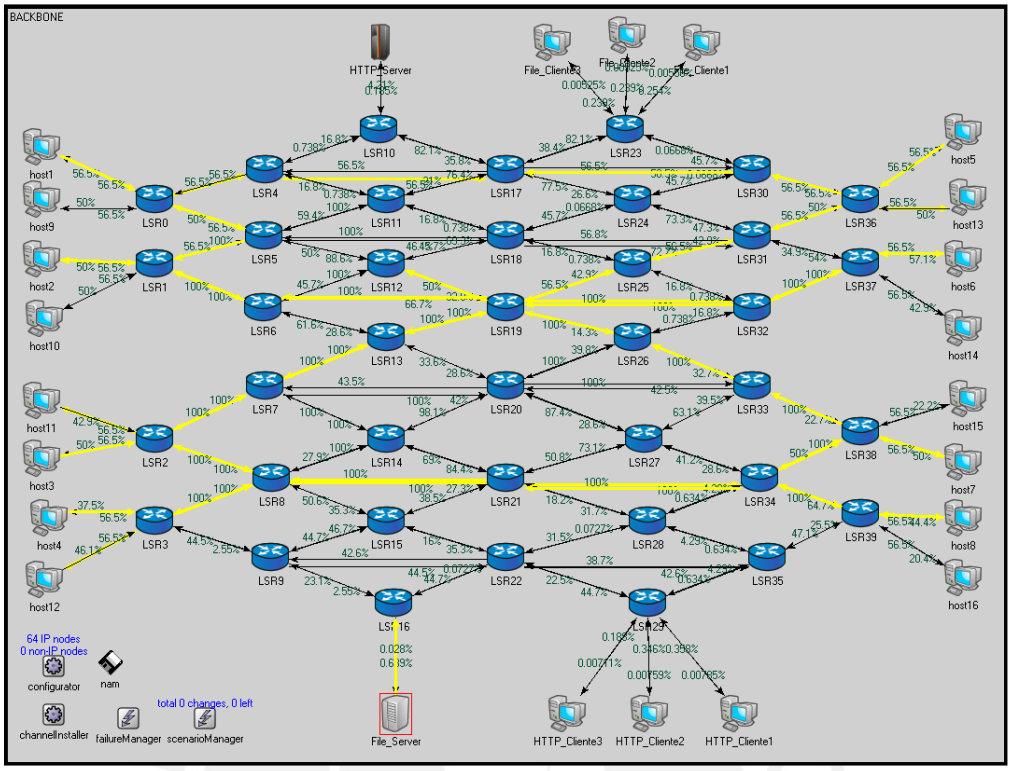

Figura 5.37: Creación y uso de las VPN

Figura 5.38: Transmisión de tráfico FTP y Web

Los resultados que se obtuvieron con la simulación son:

Tabla 5.11: Resultados de Simulación BACKBONE Real/LDP

	Retardo Promedio (ms)	Retardo Máximo (ms)	Retardo Mínimo (ms)
Host1	85.417	85.417	85.417
Host2	87.741	87.321	85.417
Host3	85.47	86.67	85.35

Host4	85.417	85.419	85.415
Host5	85.417	85.417	85.417
Host6	86.739	89.739	86.739
Host7	89.852	98.7898	88.752
Host8	85.418	85.745	85.4175
Host9	142.371	158.64	141.89
Host10	102.561	103.534	99.871
Host11	99.653	100.23	99.649
Host12	142.361	144.393	142.36
Host13	142.361	142.361	142.361
Host14	102.433	102.433	99.978
Host15	103.825	104.813	102.466
Host16	142.461	149.834	142.473

En la figura 5.39 puede observarse el retardo para los tráficos de videoconferencia en los host 1, 2, 9, y 10. El tráfico más impactado es del host9 no solo en los valores de retardo que se obtienen sino también en los valores de jitter

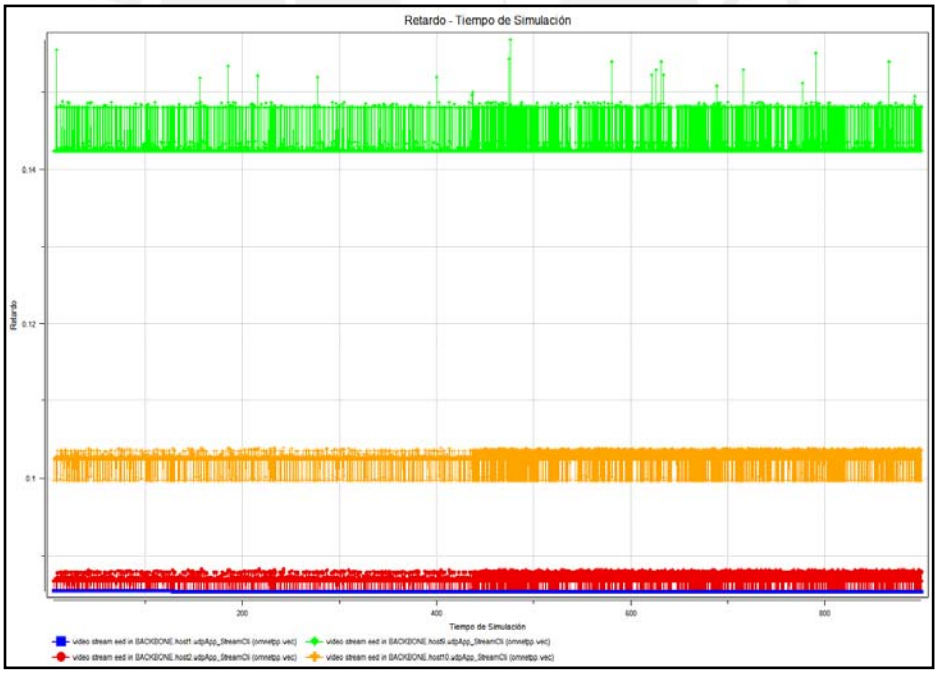

Figura 5.39: Retardos de los tráficos video streaming

En el caso del host10, tiene un comportamiento más estable que el host9 además de tener valores de retardo más bajos que este último. El tráfico de host1, por su parte, se presenta estable y con muy pequeñas varianzas cuyas influencias en el rendimiento del tráfico no es percibido fácilmente por los nodos.

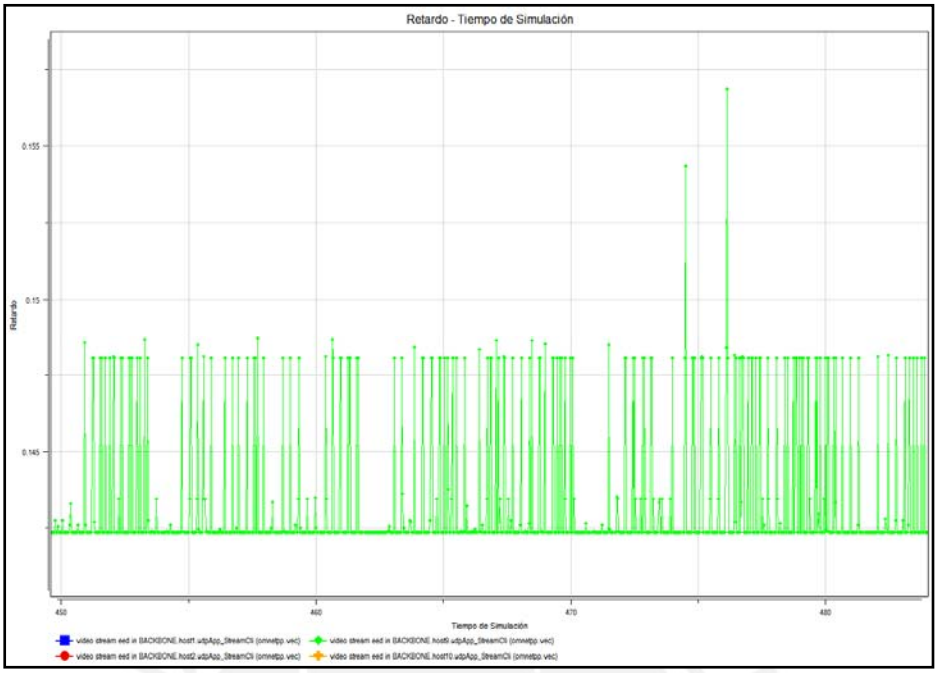

Figura 5.40: Comportamiento del Retardo en la VPN de host9

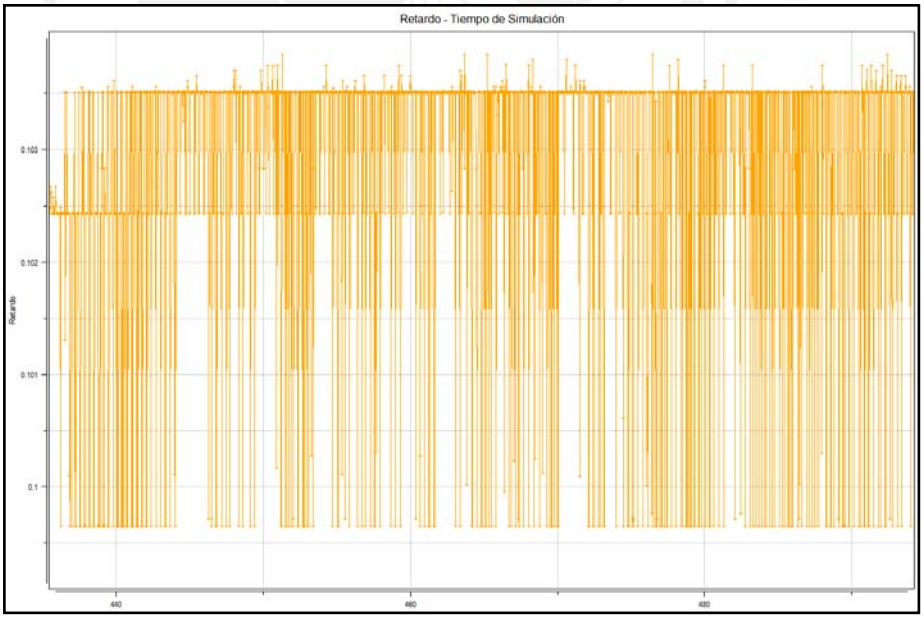

Figura 5.41: Comportamiento del Retardo en la VPN de host10

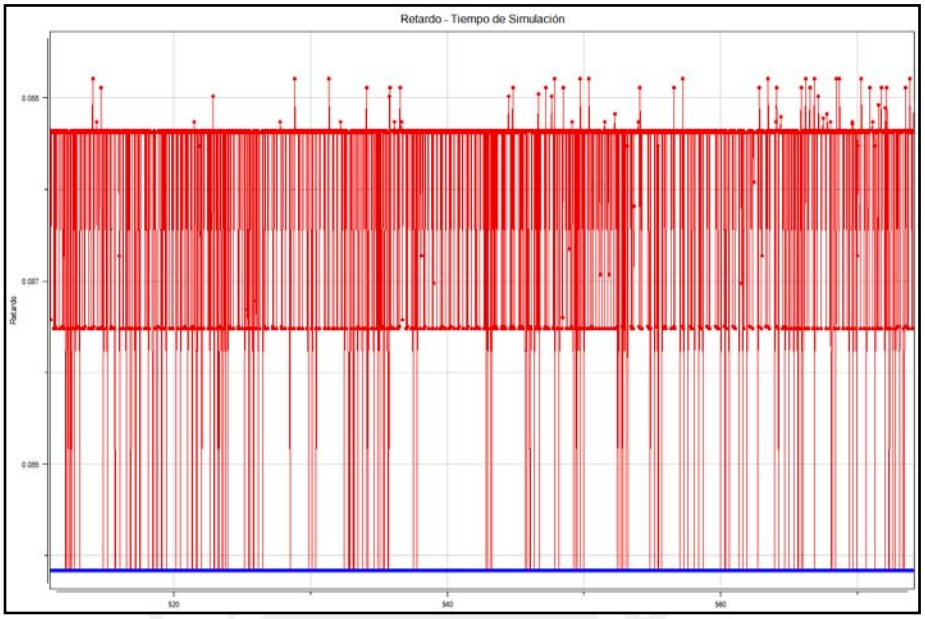

Figura 5.42: Comportamiento del retardo para lo host1 y host2

En la gráfica que se muestra a continuación puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 3, 4, 11, y 12 gracias los túneles VPN que se formaron para cursar los tráficos.

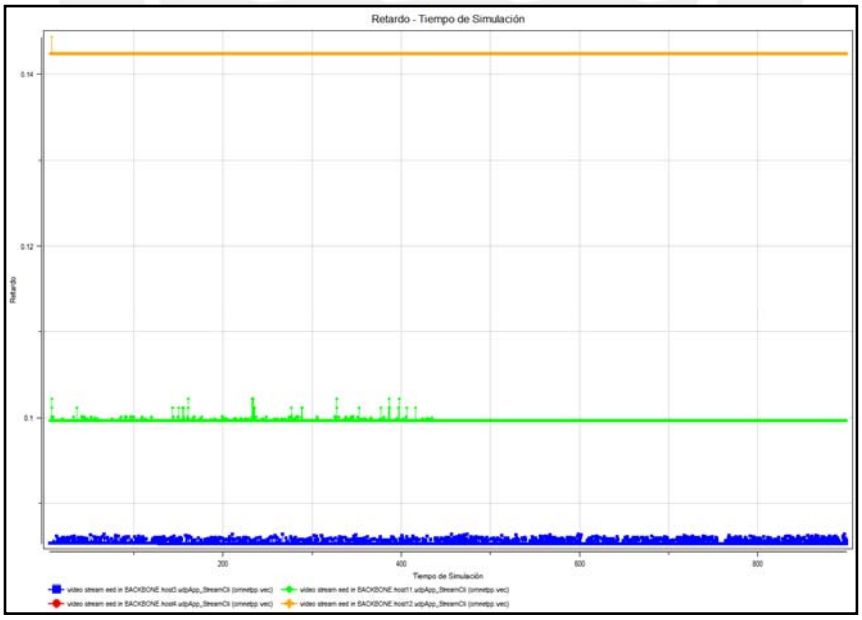

Figura 5.43: Comportamiento del retardo para lo host3, host4, host11 y host12

El tráfico más impactado es del host12. Este tráfico si bien es el más impactado, tiene un comportamiento muy estable lo cual es deseable en tráficos en tiempo real, en nuestro caso, tráfico de videoconferencia.

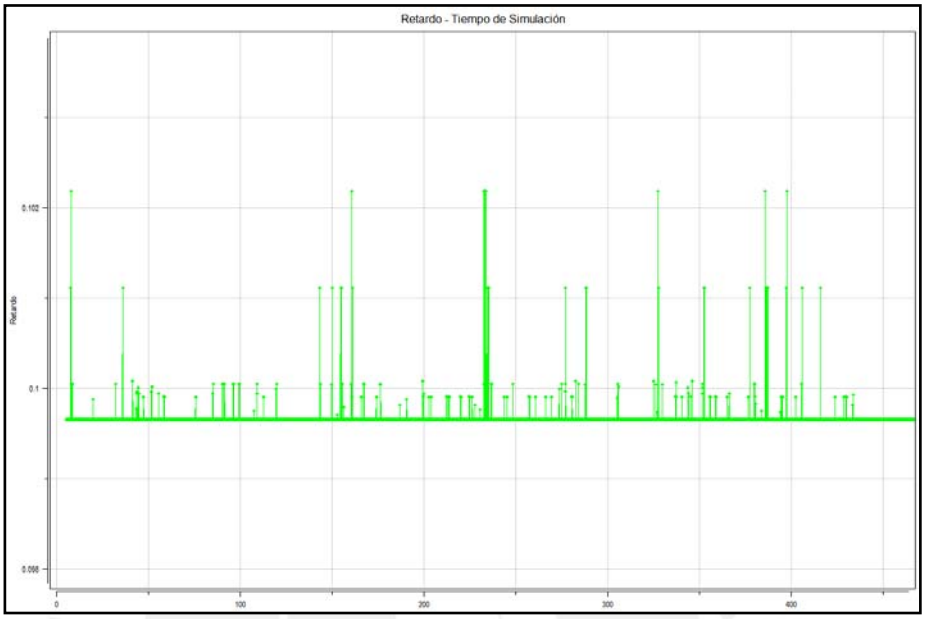

Figura 5.44: Comportamiento del retardo para el host11

El host4 tiene el comportamiento uno de los más estables y con menor retardo.

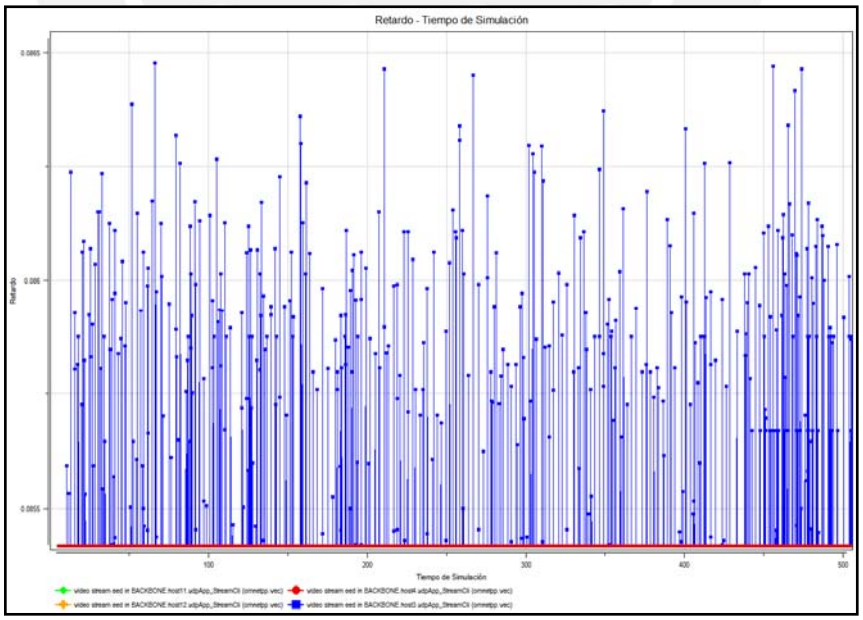

Figura 5.45: Comportamiento del retardo para el host3 y host4

En la gráfica que se muestra a continuación puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 5, 6, 13, y 14.

Podemos observar que los comportamientos de estos tráficos son estables, es decir, el jitter que presentan se asemeja a una intermitencia en los primeros segundos de la simulación, luego del cual se estabiliza en un valor nominal.

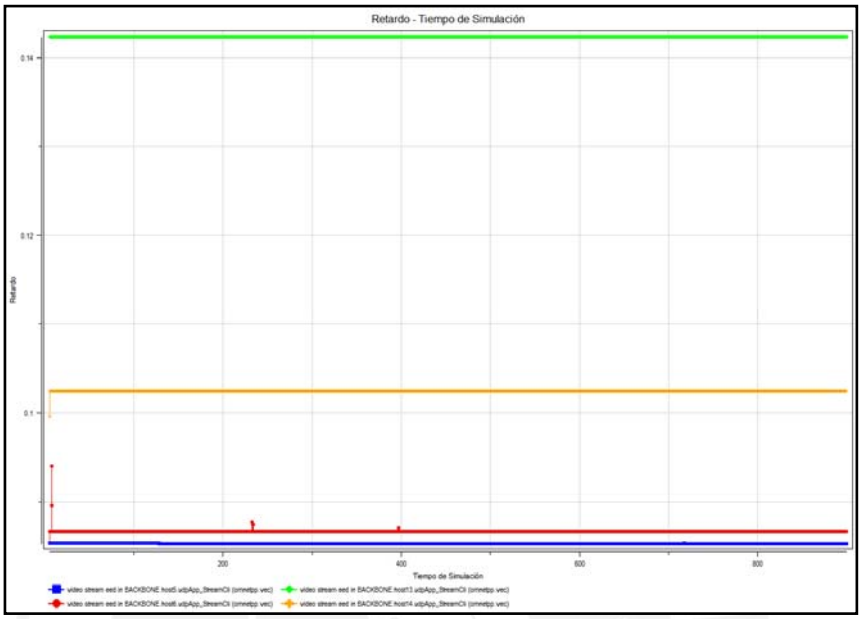

Figura 5.46: Comportamiento del retardo para lo host5, host6, host13 y host14

En la gráfica que se muestra a continuación puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 7, 8, 15, y 16.

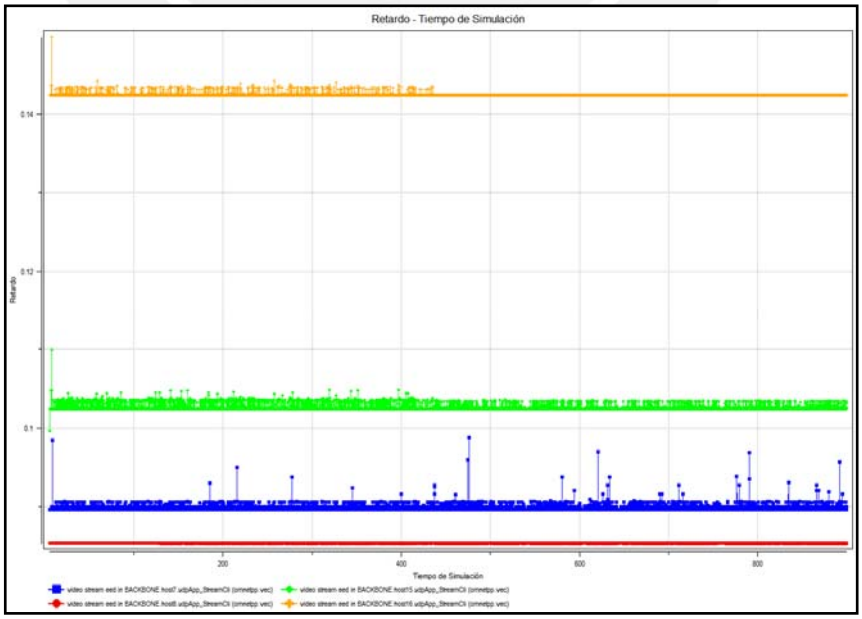

Figura 5.47: Comportamiento del retardo para lo host7, host8, host15 y host16

El host 16 presenta un comportamiento estable y hasta un punto adecuado para tráficos en tiempo real a pesar de ser el tráfico más impactado es del host16.

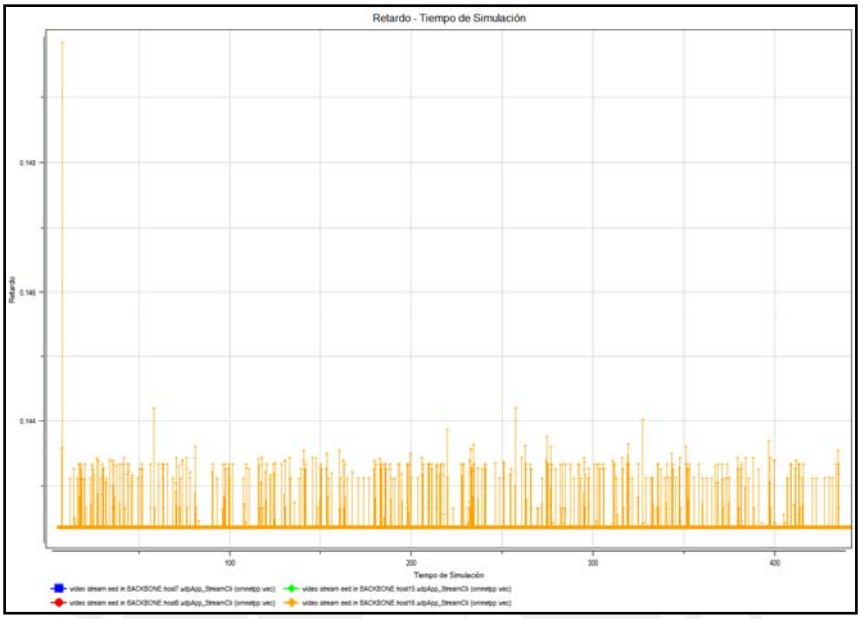

Figura 5.48: Comportamiento del retardo para lo host16

El host8 presenta un comportamiento de tráfico el cual alcanza ser el más estable entre los tráficos mostrados en la figura anterior

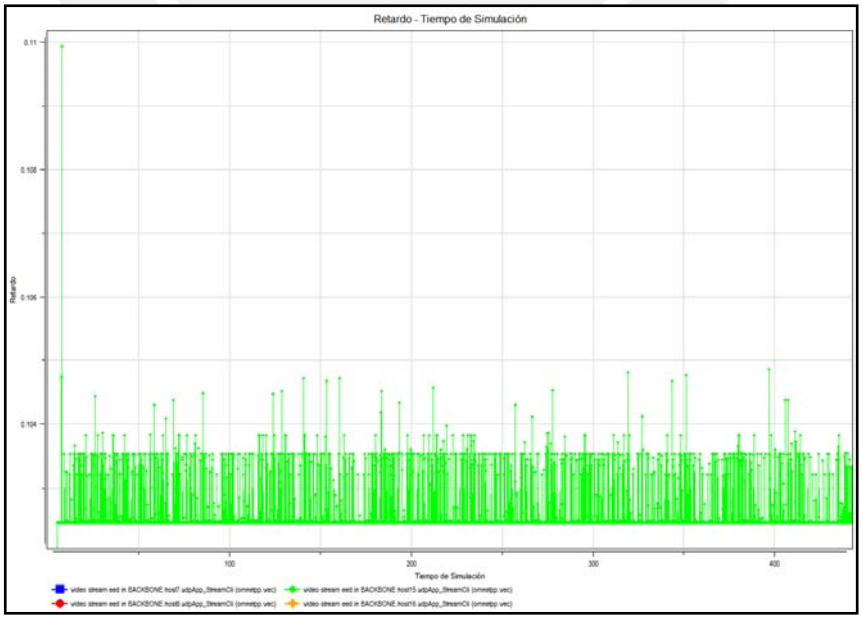

Figura 5.49: Comportamiento del retardo para lo host15

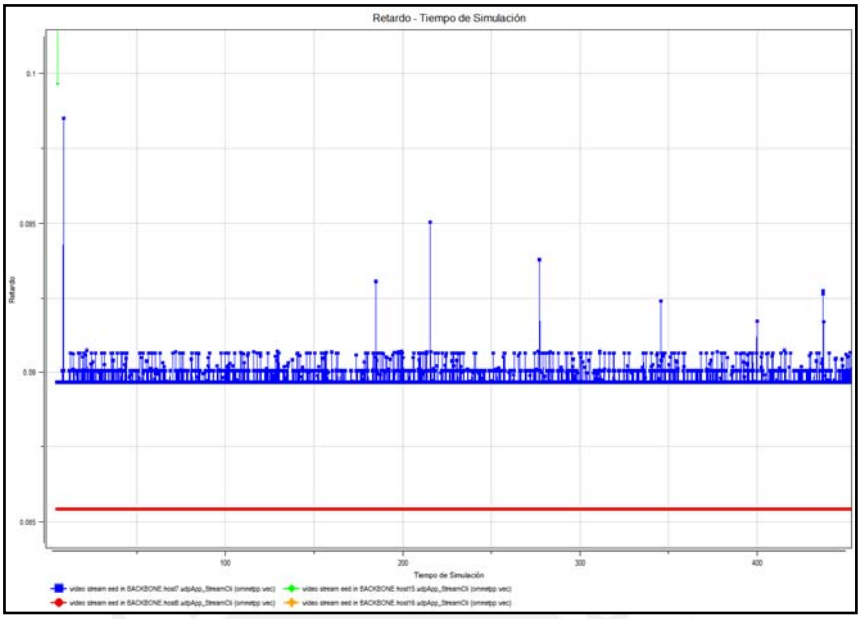

Figura 5.50: Comportamiento del retardo para los host7 y host8

Caso 2: Arquitectura MPLS RSVP-TE

A al comenzar la simulación, se observa un intercambio de información de LIB así como los paquetes Hello, señalización con la cual será capaz de crear su propio túnel con su destino respectivo. La señalización y el intercambio de información que se da en los primeros segundos llenan las colas de los routers parcialmente.

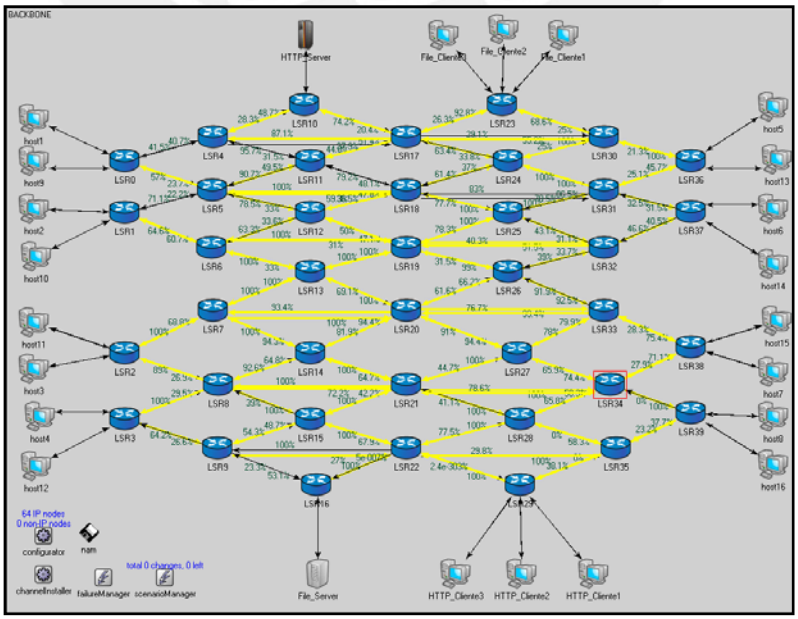

Figura 5.51: Intercambio de información y envío del protocolo RSVP-TE

Este intercambio de información finaliza a los 5 segundos de iniciada la simulación en los cuales se intercambia la información necesaria para poder establecer los túneles. Las colas de los routers son llenadas casi en su totalidad debido a la señalización de RSVP así como los paquetes Hello que se necesitan para que los túneles creados puedan seguir funcionando, en este caso, con una asignación de recursos de 5.5Mbps.

Una vez establecidos los túneles bajo el protocolo RSVP los nodos comienzan a transmitir el tráfico de interés así como la señalización que se necesita para que puedan seguir comunicándose entre los Peers y seguir con la asignación de recursos respectiva para los túneles RSVP ya establecidos.

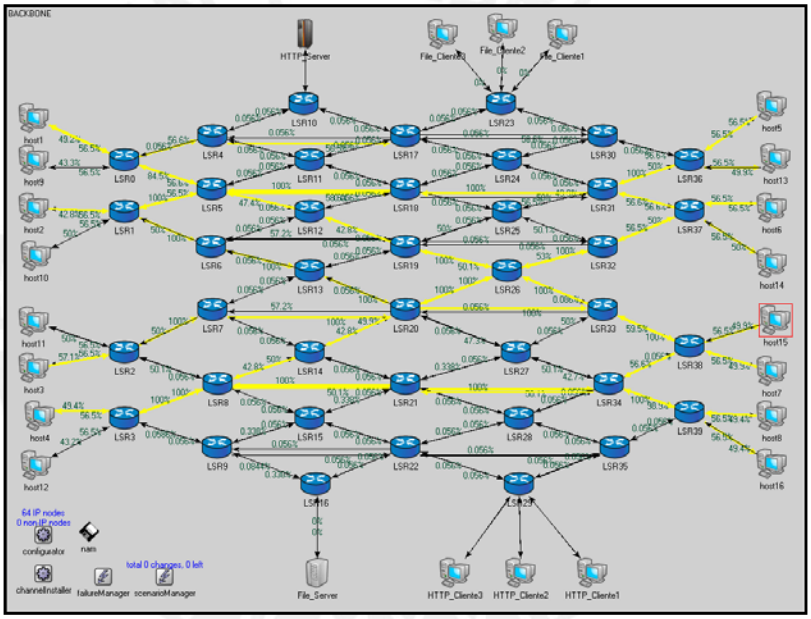

Figura 5.52: Intercambio de tráfico de Videoconferencia

Al igual que en la simulación de MPLS LDP, aquí también se tiene un tráfico que no es de interés para los nodos extremos que afectará a la performance. Estos tráficos son del tipo ráfaga. Los tráficos que se tienen en la simulación son los tráficos correspondientes a transferencia de archivos FTP File Transfer Protocol y tráfico web HTTP HyperText Transfer Protocol, el cual es muy usado en la Internet además de diferentes aplicativos.

Las VPNs se caracterizaran por un tráfico de 3.36MBps. Las colas de los routers llegan a sus límites, en especial los routers del núcleo del core.

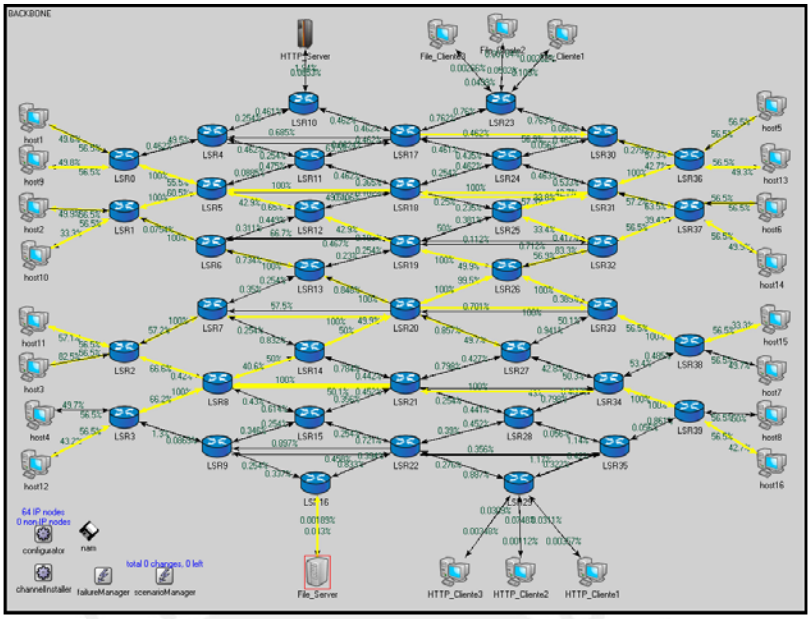

Figura 5.53: Topología BACKBONE. Transmisión de tráfico FTP y Web

Los resultados que se obtienen a partir de la topología propuesta son:

Tabla 5.12: Resultados de Simulación BACKBONE Real/RSVP

	Retardo Prom. (ms)	Retardo Máx. (ms)	Retardo Mín. (ms)
Host1	85.428	85.431	85.422
Host2	89.867	94.527	89.525
Host3	89.667	89.67	89.665
Host4	85.428	85.429	85.426
Host5	85.428	85.428	85.428
Host6	85.428	85.428	85.428
Host7	86.712	88.251	85.449
Host8	85.594	86.501	85.543
Host9	142.384	155.089	142.216
Host10	102.729	107.987	102.412
Host11	102.501	102.501	99.899
Host12	143.711	143.711	143.711
Host13	142.384	142.684	142.354
Host14	99.667	110.825	98.957
Host15	102.634	103.755	99.203
Host16	142.984	148.764	142.352

En la figura 5.54 puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 1, 2, 9, y 10 gracias las VPN's creadas.

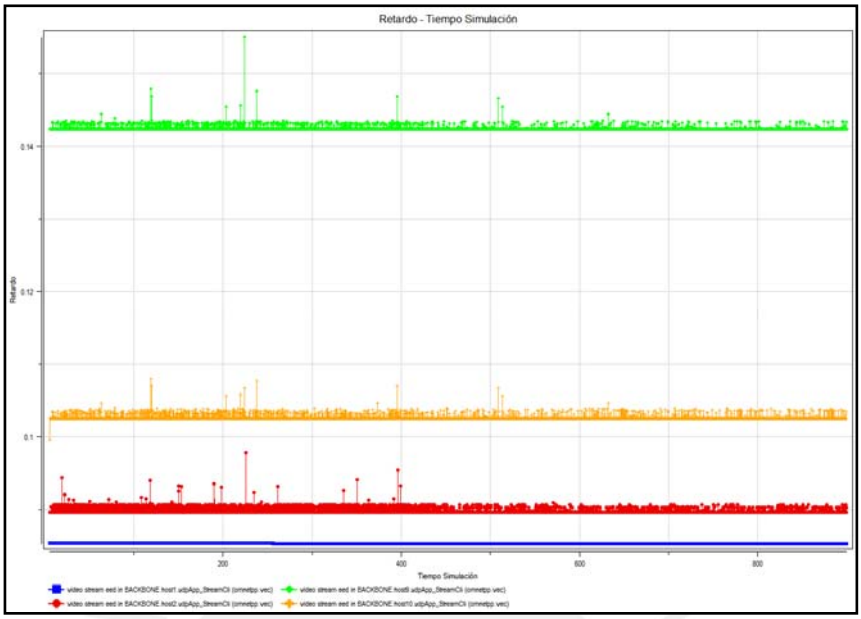

Figura 5.54: Retardos de los tráficos video streaming de las VPN

El tráfico más impactado es del host9, el cual presenta retardos altos al comienzo de la simulación pero conforme pasa el tiempo de simulación, disminuyen los valores de retardo. En el caso del host10, tiene un comportamiento ligeramente más estable que el host9 además de tener valores de retardo y de jitter más bajos.

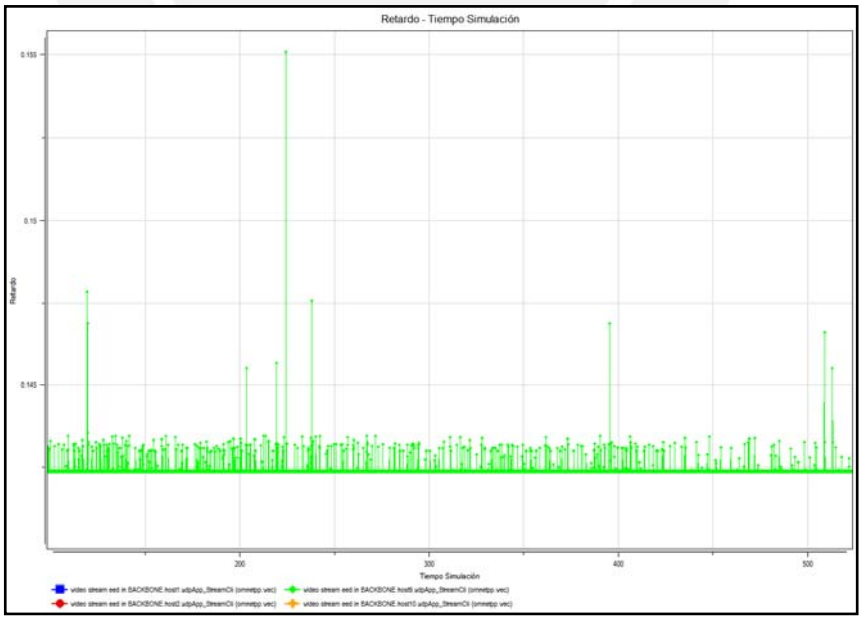

Figura 5.55: Comportamiento del Retardo en la VPN de host9

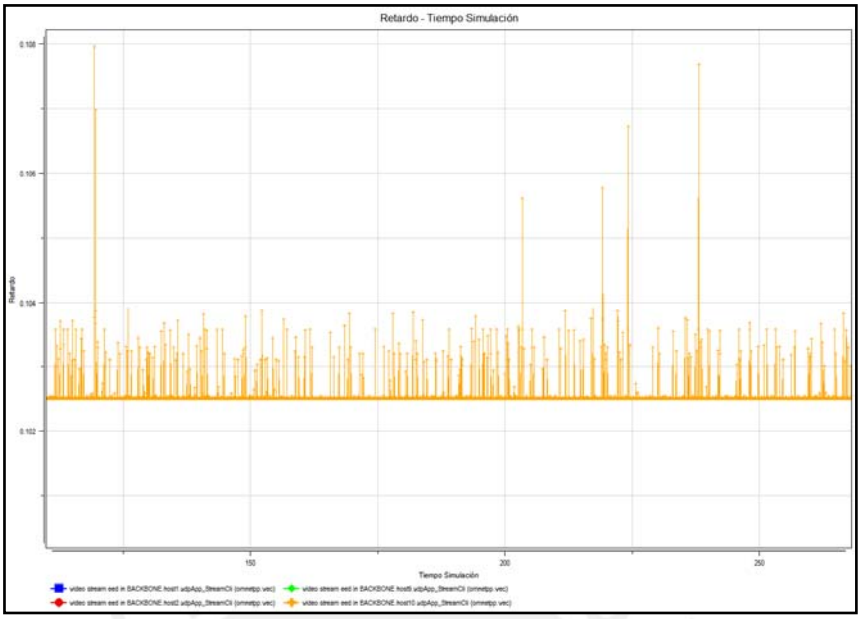

Figura 5.56: Comportamiento del Retardo en la VPN de host10

Para los casos de los host1 y el host2, se tienen valores menores que los obtenidos con el host9 y el host10. El comportamiento del retardo del host2 tiene un retardo el cual disminuye su frecuencia de aparición según el tiempo de simulación continúa. El tráfico del host1, por su parte, se presenta como un tráfico muy estable con variaciones que no son percibidos fácilmente por los nodos terminales.

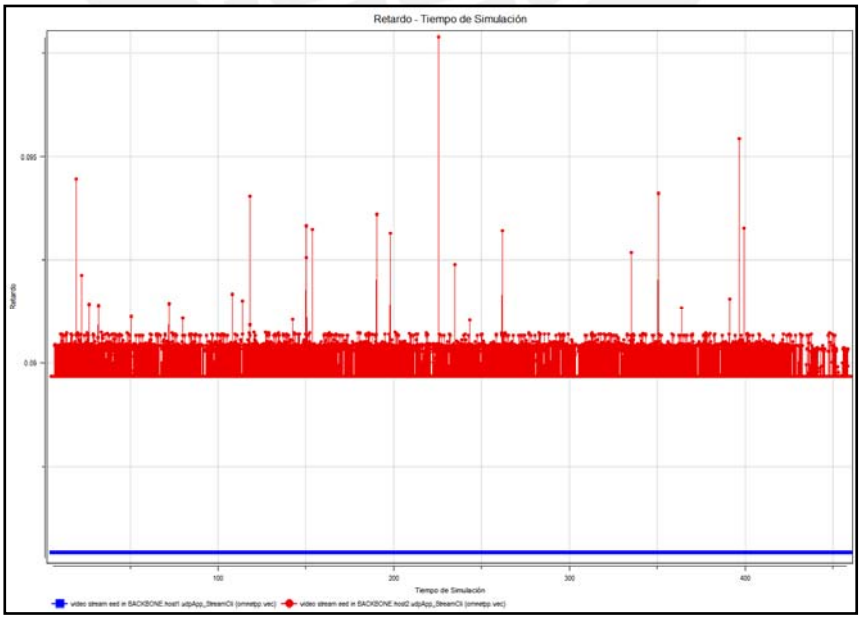

Figura 5.57: Comportamiento del retardo para lo host1 y host2

En la figura 5.58 se observa el comportamiento del retardo para los tráficos de videoconferencia recibidos en los host 3, 4, 11, y 12. Como se observa en la figura, los tráficos de estos host poseen un retardo muy estable, es decir, tienen un jitter muy pequeño incluso que no es perceptible por los nodos terminales.

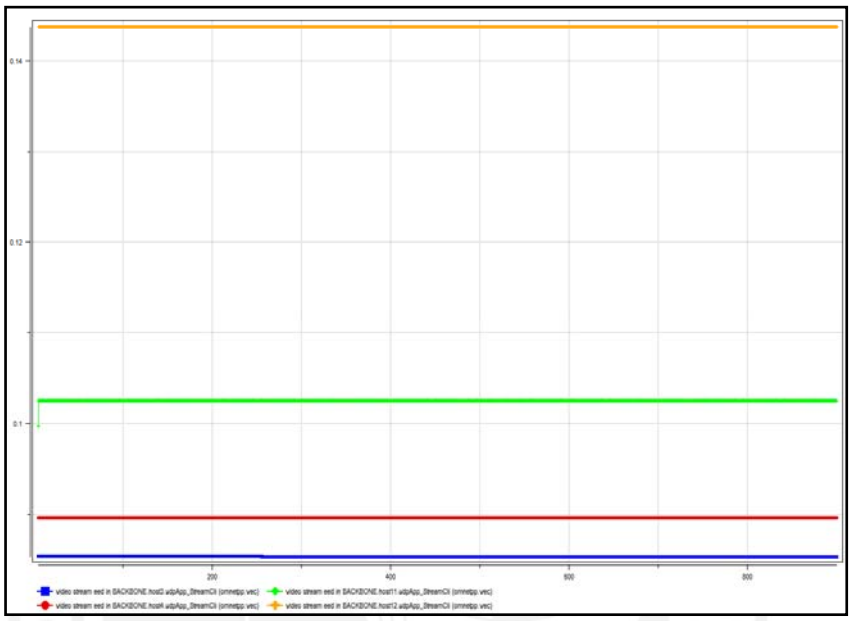

Figura 5.58: Comportamiento del retardo para lo host3, host4, host11 y host12

En la gráfica que se muestra a continuación puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 5, 6, 13, y 14.

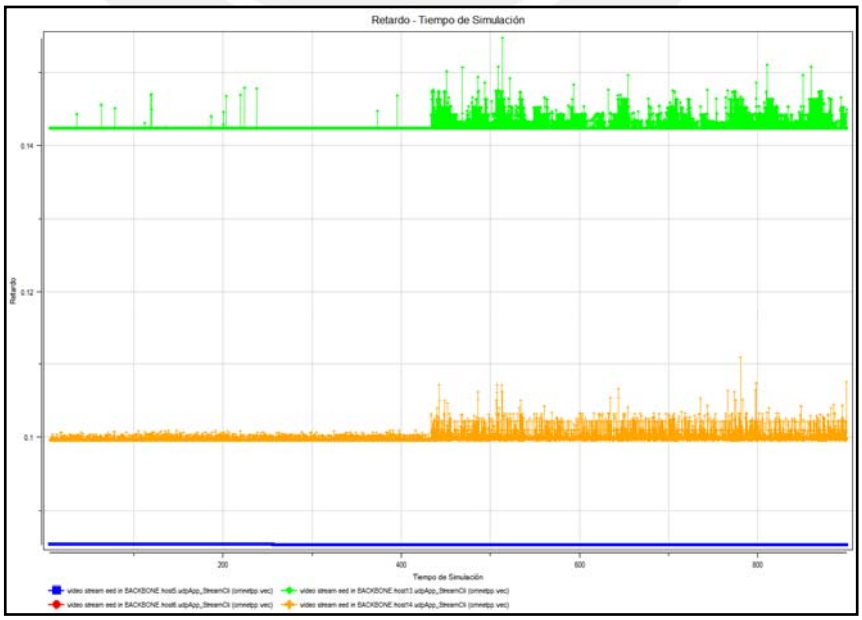

Figura 5.59: Comportamiento del retardo para lo host5, host6, host13 y host14

Podemos observar que los comportamientos de estos tráficos son estables hasta el tiempo de simulación 433.453seg después del cual se presenta un jitter en los tráficos pertenecientes al host13 y host14.

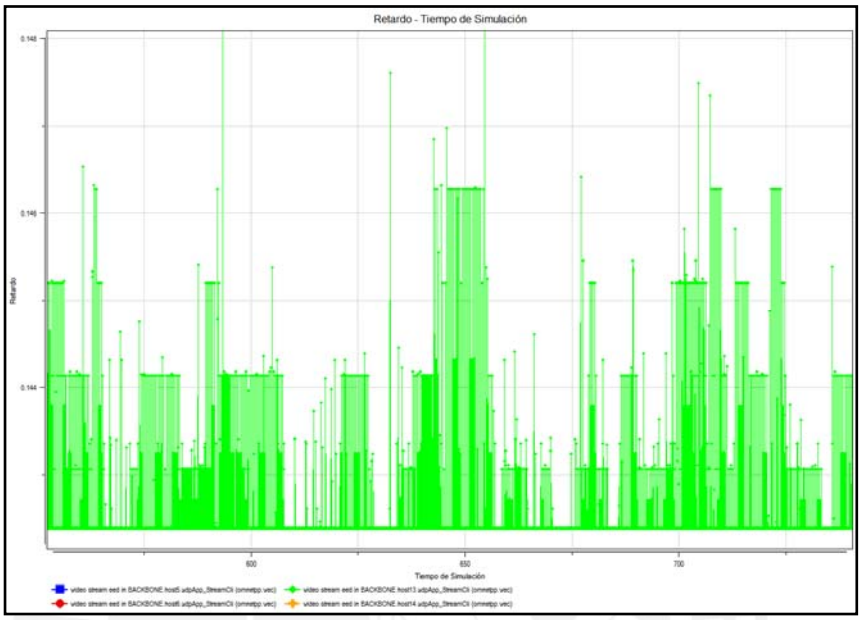

Figura 5.60: Comportamiento del retardo para el host13

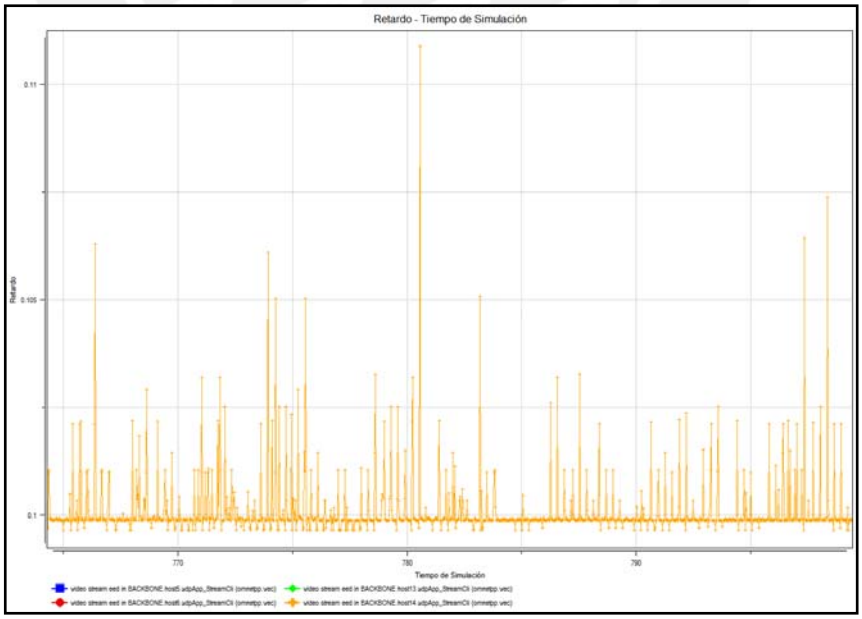

Figura 5.61: Comportamiento del retardo para el host14

En la figura 5.62 puede observarse el comportamiento el retardo para los tráficos de videoconferencia recibidos en los host 7, 8, 15, y 16.

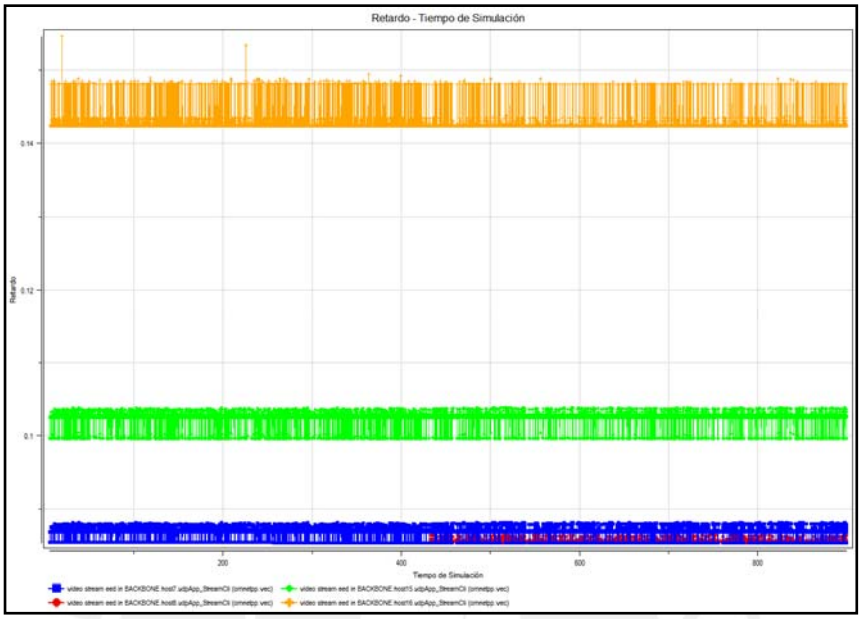

Figura 5.62: Comportamiento del retardo para los host7, host8, host15 y host16

Podemos observar que los tráficos cursados presentan un tráfico muy estable, con buen comportamiento de retardo y jitter. El tráfico más impactado es del host16, el cual a pesar de ser uno de los más impactado, tiene un comportamiento muy estable.

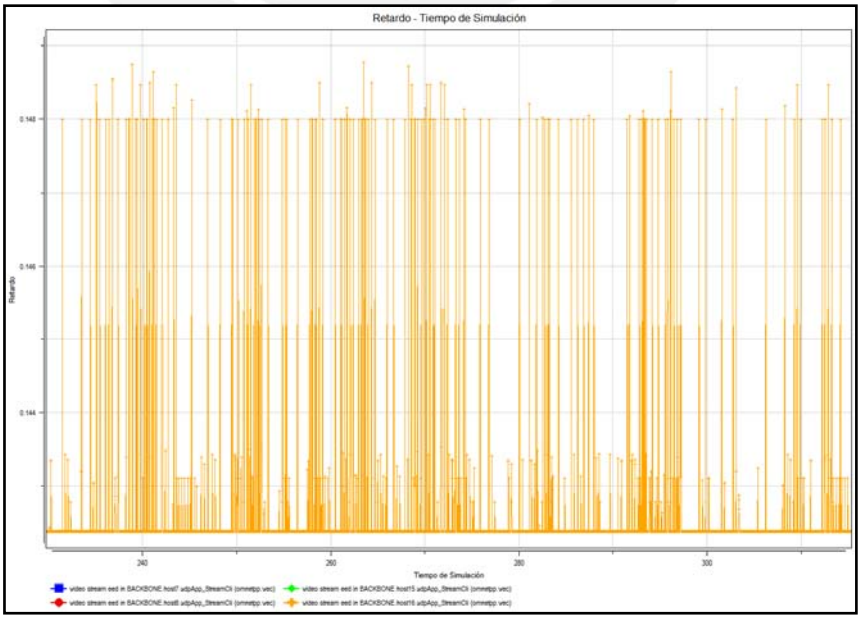

Figura 5.63: Comportamiento del retardo para lo host16

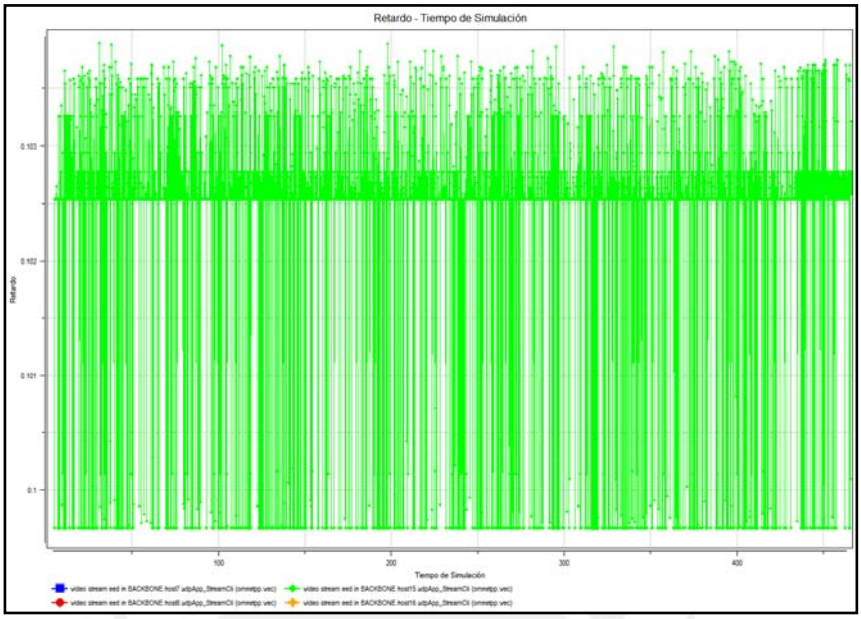

Figura 5.64: Comportamiento del retardo para lo host15

Los host 7 y 15 tienen variaciones de retardo pequeños, los cuales le permiten ser aptos para cumplir con los requerimientos de Calidad de Servicio a los tráficos de interés que las aplicaciones requieren.

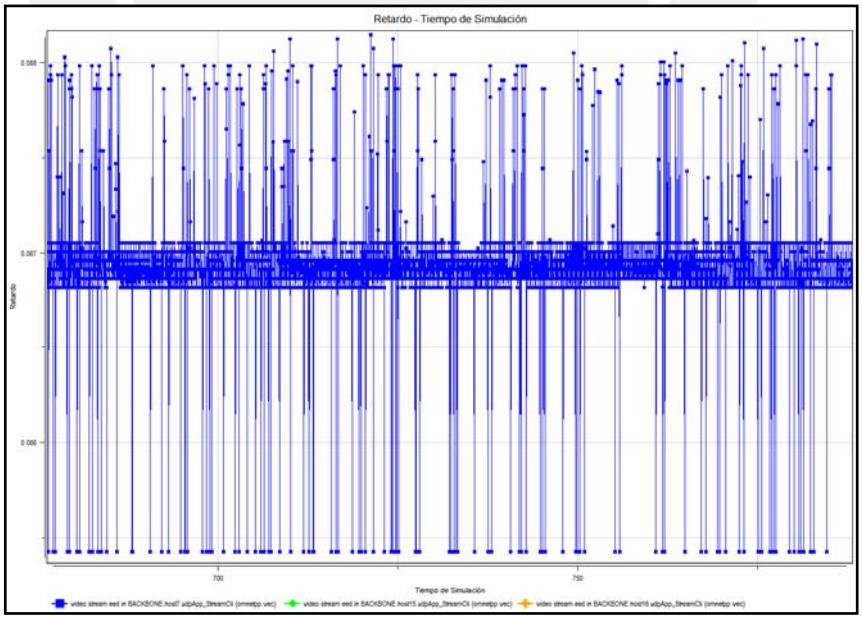

Figura 5.65: Comportamiento del retardo para lo host7

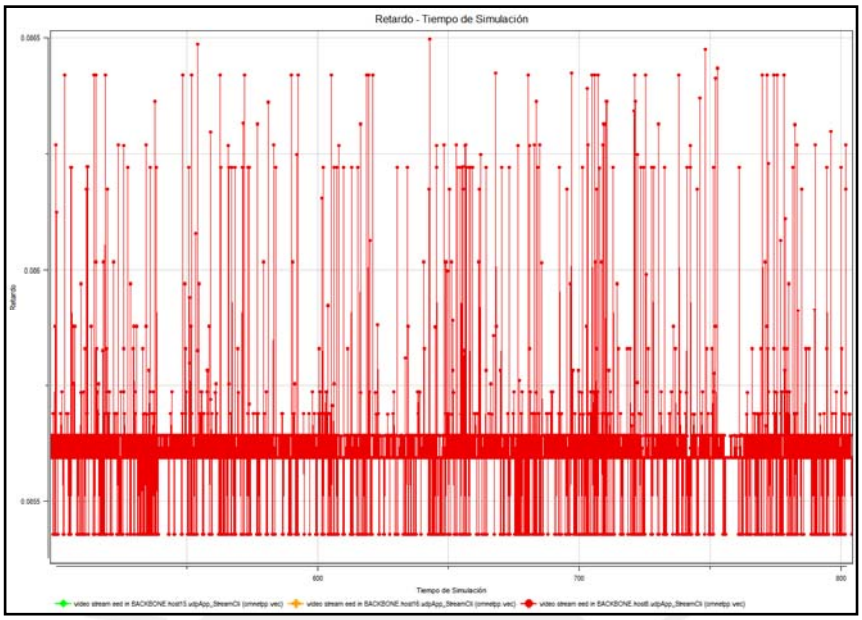

Figura 5.66: Comportamiento del retardo para host8.

CONCLUSIONES Y RECOMENDACIONES

En este capítulo se indican las conclusiones a las que la tesis ha llevado a lo largo de su desarrollo. Según los datos recolectados como los comportamientos observados a través de las simulaciones, se procede a señalar la tecnología que nos ofrece el escenario más idóneo para la implantación de servicios de tiempo real. También se presentan diferentes propuestas de trabajo tesis basadas en la experiencia adquirida e inquietudes producto del trabajo involucrado.

1. Conclusiones.

- Se ha experimentado el rendimiento de MPLS LDP y RSVP, de los cuales el protocolo que ha proporcionado mejores resultados es LDP con respecto al retardo y jitter. En efecto, LDP es un protocolo nativo de MPLS, por lo que tiene mayor compatibilidad que cualquier adaptación; RSVP-TE, por su parte, es un protocolo que fue adaptado para utilizarse con MPLS y con Ingeniería de Trafico.

- Existe una marcada diferencia en los valores mínimos y máximos obtenidos de MPLS LDP y RSVP. La razón radica en que RSVP necesita mayor señalización

(RSVP se trata de una adaptación) la cual no solo satura las colas de los routers y posibles pérdidas de paquetes, sino tiempo de procesamiento en los nodos intermedios del camino creado.

- Aplicaciones en tiempo real, en este caso Video Streaming, requieren el mayor ancho de banda posible para que su transmisión sea adecuada según lo indica la ITU-T en sus recomendaciones Y.1540 e Y1541. Con MPLS LDP se pudo comprobar esta afirmación dado que al utilizar menos señalización, proporciona mayor ancho de banda y el retardo estará dentro de los límites que estas recomendaciones indican lo cual puede ser apreciado en las tablas de resultados de simulaciones con LDP (las tablas 5.11 y 5.12).

- Se demostró que las topologías de gran tamaño (caso BACKBONE) favorecen a MPLS, es decir, se presenta un mejor comportamiento con respecto al retardo de los tráficos así como del jitter que puede generarse. Gracias a esto, muchas clases de servicio propuestas por la ITU-T se pueden cumplir.

- Los comportamientos de los tráficos mostraron una necesidad de protocolos de enrutamiento que puedan discernir en base a la ocupación, tamaño de la cola y tipo de servicio ya que se encontraron escenarios en los cuales a pesar de tener enlaces libres, estos no eran usados. Con la utilización de protocolos de enrutamiento con mayor inteligencia los paquetes de los tráficos importantes no circularán por enlaces congestionados.

- En base a las simulaciones, se desprende que RSVP necesita una cantidad de recursos de procesamiento alto frente a LDP, por lo que utilizar sólo este protocolo para todos los tráficos cursantes podría resultar contraproducente debido a los recursos necesitados.

2. Recomendaciones y Trabajos Sucesivos.

- Evaluar MPLS sobre Sistemas inalámbricos. Dado que MPLS puede adecuarse a cualquier capa de enlace, resulta de interés saber el comportamiento de este protocolo en ambientes IEEE802.11 o en redes celulares; en especial, las redes de tercera generación o superior, de tal forma que puedan soportar nuevos

servicios con Calidad de Servicio (QoS), adecuada utilizando los recursos de la red al máximo. Nótese que en este tipo de redes, el ambiente influye mucho en el rendimiento de a red, diferencia crucial entre las redes propuestas.

- Realizar una actualización para INET Framework. El INET Framework ha resultado ser un paquete de mucha utilidad para la presente tesis, existen módulos que aún están a prueba por lo que no tienen compatibilidad con todos los protocolos de capas de aplicación, se presenta inestabilidad en ciertos escenarios, entre otros inconvenientes. Para futuros estudios se desea que soporte mayores cantidades y tipos de tráfico de forma adecuada, estable y que se optimice los tiempos de análisis en los nodos que conforman la red.

- Evaluar MPLS en diferentes tipos de colas y aplicando Políticas de Justicia. En la tesis se evaluó MPLS con cola FIFO, cola sin ningún tipo de consideración sobre el tipo de tráfico y/o prioridad, bajo políticas de redistribución de recursos igualitaria sin corroborar la existencia de recursos suficientes. Una evaluación de diversos escenarios con diferentes tipos de cola, control de cogestión y políticas de redistribución basada en el tipo de tráfico cursado resulta de interés para poder dimensionar una red en base a los tipos de tráficos a cursar.

- Se desea hacer un estudio de escenarios que puedan soportar diversas aplicaciones aplicando gestión de colas, control de congestión y políticas de redistribución adecuadas según los tráficos cursados lo requieran.

- Comparar una arquitectura MPLS con IPv6 para poder contrastar el desempeño. Como se mencionó en apartados anteriores, ambas arquitecturas pueden ofrecer Calidad de Servicio a los tráficos de interés de diferentes formas por lo cual resulta de interés comparar el grado de QoS que estas tecnologías pueden ofrecer a los tráficos de interés.

- Contrastar los resultados obtenidos con los que se pueden obtener con otros simuladores, tales como KivaNS, Open SimMPLS están destinados a simular topologías MPLS con los cuales se puede complementar y comparar los comportamientos experimentados en esta tesis.

ANEXO 1
INGENIERÍA DE TRÁFICO EN REDES MPLS

1. Definición de Ingeniería de Tráfico

La Ingeniería de Tráfico Traffic Engineering es la disciplina encargada del estudio de la utilización de los diferentes tipos de recursos de la red de tal forma que pueda utilizar al máximo los recursos disponibles y dar proporcionar ciertas políticas de contingencia en caso de fallas y así proporcionar la Calidad de Servicio QoS necesaria a las aplicaciones y a los clientes con el respectivo LSA [4]. Esta disciplina pretende mejorar la performance o rendimiento de la red sin agregar algún recurso extra a la red.

El problema que presenta la Internet actualmente es la incapacidad de proporcionar Calidad de Servicio QoS necesaria a los diferentes tipos de aplicaciones, en especial a las aplicaciones de tiempo real. El esquema Best Effort proporciona un comportamiento igualitario a todos los tráficos que cursan la red, es decir, los tráficos que son sensibles a los retardos son tratados de igual forma que los tráficos inmunes al retardo sin ninguna clase de prioridad. El comportamiento salto a salto hop-by-hop no proporciona ninguna garantía al envío de los tráficos ya que un router no conoce la situación de sus vecinos próximos y mucho menos los de los demás routers. La reserva de recursos no se puede proporcionar bajo este esquema de trabajo y la congestión puede suceder en cualquier enlace o router.

El problema con los recursos no solo se reduce al esquema de Best Effort, sino también a los protocolos de enrutamiento contribuyen a intensificar problema ya que estos protocolos no tienen una métrica que refleje la verdadera situación de los recursos de la red. Protocolos más usados en las backbones de la Internet y de los Sistemas Autónomos se basan en algoritmos SPF (Shortes Path First) tales como RIP (Routing Information Protocol), OSPF (Open Shortest Path First), IS-IS (Intermediate System to Intermediate System), etc. cuyas métricas usan como base parámetros estacionarios: ancho de banda, número de saltos, etc. los cuales no dependen del tiempo sino del parámetro físico de conexión. El comportamiento salto a salto hop-by-hop resulta evidente en esta clase de protocolos ya que solo se evalúa las conexione a los vecinos más no como se usan los recursos en las rutas hacía ellos.

Actualmente, La arquitectura que se ofrece en Internet es IP over ATM la cual ofrece muchos problemas ya que se tratan de dos redes diferentes. La arquitectura ATM ofrece un escenario propicio para poder ofrecer la Calidad de Servicio necesaria así como la Ingeniería de Tráfico; pero como su contraparte, la red IP, no puede ofrecer estas características; tiene como consecuencia que no se puedan ofrecer estas facilidades o que se ofrezcan de manera limitada y/o muy complicada.

2. Objetivos de la Ingeniería de Tráfico

La Ingeniería de Tráfico tiene como objetivo la maximización de los recursos de la red en su totalidad, es decir, que ningún recurso sea desperdiciado o subutilizado. Para este fin se necesitan realizar de manera muy eficiente las siguientes tareas:

- **La Gestión de Recursos de Red disponibles**

La gestión de los recursos de red con un plano de gestión de recursos o en base a protocolos permite un mejor uso del ancho de banda requerido por cada cliente, los equipos que se disponen así como sus capacidades y facilidades de red tales como enlaces redundantes en la capa física. Gracias a la gestión de Recursos podemos asegurar un correcto servicio además de aprovechar al máximo los recursos que se disponen así también permite aplicar Ingeniería de Red Network Engineering de tal forma que la red esté capacitada de proveer los recursos que se ofrecen [4].

- **Caracterización y medida de tráfico.**

En ciertos casos se puede caracterizar el tráfico de las redes de los ISP por separado. A diferencia del tráfico telefónico, el tráfico en la Internet no puede ser modelado ya que se trata de un tráfico tipo ráfagas, es decir, en un instante de tiempo se pasa de un enlace y colas vacíos a un enlace muy usado con una cola muy copada. A esto se suma el hecho de que en la Internet la comunicación no es en un circuito establecido dedicado, sino que uno puede ser fuente de múltiples comunicaciones a muchos destinos. A pesar de este problema, los picos de tráfico en horas se pueden detectar y en esos instantes aplicar una gestión de recursos más dedicada en especial a los tráficos que tienen un SLA que les asegure ciertos parámetros de Calidad de Servicio.

- **Gestión de tráfico (por ejemplo, mecanismos de encolado, programación).**

Este objetivo se refiere a tener caminos y/o rutas de contingencia para los tráficos que cursan la red, en especial aquellos que son parte de una VPN o de algún servicio

diferenciado. Nótese que estos caminos deben establecerse lo más rápido posible ya que durante el tiempo que se realiza el cambio, los flujos de tráfico van a experimentar alguna clase de retardo o, en el peor de los casos, pueden experimentar pérdidas considerables de información ya que las colas en los routers comienzan a llenarse de los paquetes que han sido enviados y serán descartados en caso de llenarse la cola de los mismos [3] [4].

- **Conformación del tráfico**

Esto se refiere a que el cliente cumpla con el LSA que ha suscrito con el ISP. El tráfico que es enviado por el cliente al los nodos de la red, debe de ser clasificado para que pueda recibir un trato diferenciado en comparación a los demás tráficos de la red. Si el tráfico esta por debajo de los límites o al tope de estos mismos, el tráfico será marcado apropiadamente para su envío; en el caso contrario, habrán paquetes que se podrán etiquetar bajo un esquema de Best Effort, una categoría menor a lo establecido o simplemente descartar el tráfico excedente. Estos términos de uso, conformación y de límites debe de ser especificado en el LSA entre cliente e ISP [6] [7] [8].

- **Control de Congestión**

La ingeniería de Tráfico no solo debe limitarse a un camino alternativo. El control de la congestión se realiza en las colas de los routers que conforman el backbone del ISP, en los cuales, según la prioridad o al túnel al que se pertenezca, el router tomará la decisión de descartar el paquete, de retrasar un tiempo definido su envío, procesarlo inmediatamente para que sufra solo un retardo mínimo [6] [7].

- **Planeamiento de capacidad de red (capacity planning)**

La capacidad que poseen los múltiples enlaces dentro de una red no puede ser excedida, lo que si se pude hacer es una correcta distribución y uso de estos de tal forma que se tenga la mayor eficiencia posible de todos los enlaces. La ingeniería de red Network Engineering nos permite saber que partes de nuestra red necesitan una mayor cantidad de recursos para poder satisfacer el tráfico cursado [6] [7].

- **Enrutamiento dinámico y adaptado a necesidades del servicio.**

Si bien las rutas por defecto se deben de activar para los flujos cuando los caminos sufren alguna caída de sus componentes, esta ruta no debe de utilizarse por todos los flujos; debe usarse por los flujos que realmente lo necesitan así como los flujos que

tienen que cumplir con una clase de servicio en especial la cual se especifica en los LSA de los clientes con el ISP. Así también, los congestionamientos pueden ser evitados por este tipo de comportamiento, es decir, los tráficos que siguen el comportamiento Best Effort los cuales no requieren ninguna clase de servicio adicional pueden ser enviados por un camino en el cual pueden experimentar congestión, los tráficos con una calidad de servicio especificada media por una camino en el cual experimentarán una congestión no severa media, y los tráficos con una calidad de servio QoS se puede enviar por un camino diferente con el que puedan cumplir los parámetros especificados en el LSA de los clientes.

3. Ingeniería de Tráfico en MPLS

La ingeniería de tráfico es una cualidad nativa en MPLS. Como se explicó antes, MLSP posee la capacidad de definir múltiples VPN Virtual Private Network, en este proceso los nodos que se encargarán de este túnel se especifican según las tablas de enrutamiento de capa de red formadas a su vez por los protocolos de enrutamiento clásicos como otros protocolos de la nueva generación; pero las etiquetas correspondientes al túnel se informarán a la red con un protocolo de distribución como LDP, RSVP-TE, etc.

El concepto de creación de túneles en una red, permite que los tráficos de un cliente se envíen por un túnel por medio de la Backbone del ISP en el cual se le proporcionará un acceso a recursos según lo requiera el tráfico o según lo especificado por el LSA. Este concepto de túnel permite que frente a congestión, el tráfico tenga una cantidad de recursos mínimos necesarios para el enrutamiento de sus flujos por medio de la red. Además, permite el manejo de ingeniería de tráfico al poder establecer LSP basados en las tablas de enrutamiento de la capa de red, es decir, según el protocolo de enrutamiento que se esté usando en la capa correspondiente se puede cambiar de manera dinámica los nodos LSR que conforman el túnel frente a alguna caída de a algún nodo intermedio, congestión, agotamiento de recursos, balanceo de carga, etc.

Otra de las facilidades que tiene la arquitectura MPLS es la capacidad de ofrecer Calidad de Servicio QoS Interdominio. Los ISP pueden establecer políticas según el comportamiento del tráfico pero solo en su propia red, es decir, dentro de su propia red pueden emplear los diferentes túneles LSP de contingencia que se requieran,

balanceo de carga, entre otras facilidades y/o necesidades. La Calidad de Servicio Intradominio es catalogado muchas veces como un servicio utópico muy difícil de alcanzar en la vida real. MPLS da una arquitectura en la cual solo se debe de asignar una etiqueta temporal, y según lo establecido entre los proveedores, se aplicaran las políticas necesarias para que el tráfico llegue a su destino cumpliendo los parámetros que necesite [12].

Véase además que puede tenerse en la red un agregado de tráfico. Este proceso consiste en que el tráfico que pertenece a una clase en especial con parámetros y/o Clase de Servicio similares puede ir por un mismo túnel ay que así podría ahorrarse recursos y los tráficos que pertenecieran a este túnel cumplirían con los parámetros requeridos para su envío. De esta misma capacidad se desprende que puede establecerse una VPN para cada demanda y/o Clase de Servicio específica de tal manera que un túnel no sea afectado por otro, sino que tengan un comportamiento independiente.

4. Conclusiones

- Gracias a la capacidad de MPLS de crear túneles de manera dinámica, puede recuperarse de eventuales fallas en los nodos de la red, esto quiere decir que si se creasen nuevos protocolos de enrutamiento para la capa de red, MPLS podría usarlos para el establecimiento de sus túneles LSP en su tabla LIB. Estas rutas de contingencia también pueden ser definidas de manera estática según los criterios que el administrador de la red Backbone del ISP lo vea más conveniente.

- El campo experimental contiene 3 bits con los cuales podemos representar perfectamente las ocho tipos de Clase Servicio QoS propuestas por la ITU-T en la recomendación Y.1541 con lo cual podemos dar prioridad en un mismo túnel a los diferentes tráficos sean encaminados por este camino. Esto da la facilidad tanto al cliente como al ISP de poder enviar diferentes tipos de tráfico por un mismo túnel que les proporcione los recursos básicos y que solo se distinguen en el retardo de atención.

- Gracias a la capacidad de MPLS de poder crear túneles LSP de manera tanto dinámica como estática, podemos manejar y evitar la congestión. Nótese que puede cambiarse los nodos que conforman el túnel LSP si se sobrepasa el uso de

los recursos. La capacidad de creación de túneles bajo demanda, es decir, cuando se necesite resulta ser muy conveniente y muy útil para los fines de balanceo de carga.

- La ingeniería de tráfico podría extenderse fuera del dominio MPLS del ISP. Las características antes vistas son nativas de MPLS, si los dominios colindantes con los que el ISP intercambia información así como los tráficos que lo requieren establecen una política o un convenio por el cual ciertos tipos de tráfico serán enviados siguiendo ciertos parámetros, el túnel establecido no solo podría dar Calidad de Servicio QoS a un solo tráfico sino a muchos tipos así también a los que dentro de los túneles a establecerse necesiten alguna clase de prioridad.

- En lo referente a políticas Interdominio, MPLS provee las facilidades necesarias para que entre proveedores pueda implementarse Calidad de Servicio QoS, balanceo de carga, reserva de recursos, establecimiento de túneles LSP de manera estática y/o dinámica, lo cual hace que esta arquitectura sea idónea para la backbone de Intenet en lo referente a asegurar la Calidad de Servicio solicitada.

ANEXO 2
GESTIÓN DE COLAS Y CONTROL DE CONGESTIÓN

1. Definición de Gestión de Colas

La Gestión de Colas se refiere al manejo que se tendrá con respecto a los tráficos de las diferentes aplicaciones y/o servicios entrantes a los nodos de la red. Generalmente los nodos que aplica esta gestión de Colas con mayor énfasis son los routers de la red. La gestión de colas permite aplicar diferentes políticas a los diferentes tipos de paquetes que provienen de los flujos que los usuarios envían a al red. La gestión de colas asigna ciertos recursos, prioridades, etc. a los tráficos que recibe y/o que envía de tal forma que se cumplan los requerimientos que se especifican para su correcto envío.

En las recomendaciones de la ITU-T Y.1541 se especifican Clases de Servicio para diferentes necesidades que las aplicaciones y/o servicios pertenecientes a estos necesitan al ser transmitidos por la red. Nótese que los flujos de cualquier tipo de tráfico se reciben en los nodos de la backbone del ISP, en los cuales los flujos deben ser enviados según los requerimientos solicitados en el LSA por lo que los routers no podrán aplicar un mismo comportamiento a cada paquete ya que aquí no se aseguraría ningún tipo de servicio y dado que no todos los tráficos requieren que se cumplan todos los parámetros por igual, es decir, necesitan que algunos sigan un comportamiento en especial asegurado mientras los otros parámetros no se necesitará ser tan riguroso.

Los recursos de cualquier tipo de red son compartidos por todos los nodos pertenecientes a la red, en especial los recursos de la Backbone como los routers que la conforman. El tamaño de las colas por ende debe de ser compartido por muchos usuarios por lo que la tendencia a llenarse y desbordarse es alta lo cual afecta a la performance del tráfico ya que los paquetes comienzan a ser descartados según el esquema Tail Drop.

El mecanismo de Gestión de Colas debe buscar cumplir:

- Tratar que las Colas existentes no lleguen a su límite. De este modo los tráficos de mayor prioridad, generalmente los de tiempo real, no se ven afectados por congestiones no previstas o por perdidas por congestión.

- Poseer un criterio de envío de paquetes que se acomode a la performance que se necesita dependiendo de las aplicaciones que se tendrán en la red.
- Poseer un criterio de descarte de paquetes que se acomode a las necesidades en caso de tener las colas llenas sin capacidad de recepción de paquetes.
- Asignación de recursos de forma adecuada según políticas de redistribución de recursos.

La gestión de Colas se vale de algoritmos para que en base a políticas y de parámetros se busque llegar a los objetivos antes descritos según los requerimientos de los usuarios.

2. Definición de Control de Congestión

El tráfico de Internet se caracteriza no solo porque el tráfico no puede ser modelado sino que además tiene un comportamiento tipo ráfaga, es decir, tiene picos de congestión en los cuales excede las capacidades de transmisión del medio arrendado. La cola de los routers recepciona a los paquetes que no se han podido trasmitir, con lo cual el buffer de los mismos comienza a coparse hasta que llega al límite y comienza el descarte de los paquetes.

En este tipo de escenarios en donde entra a tallar lo referente a Control de Congestión, políticas, criterios junto a parámetros y algoritmos que se aplican para reducir el impacto sobre la performance de la red que la congestión representa.

Dado que no todas las aplicaciones y/o servicios no tienen la misma prioridad, necesidad de recursos, tiempo de retardo, en especial las aplicaciones de tiempo real como VoIP y Video conferencia; los criterios a aplicarse en caso de congestión deben priorizar a estas aplicaciones pero, por otra parte, no deben de perjudicar a las aplicaciones de menores requerimientos.

El control de congestión proporciona las siguientes características:

- Provee de menores retardos a los tráficos que lo requieren, no solo porque les otorga prioridad al momento de la congestión sino que al ayudar a la descongestión,

- Ayuda a mejorar el thoughput de la red. La congestión reduce la performance de la red por las pérdidas de paquetes, las retrasmisiones que los servicios están obligados a realizar por lo que si se presenta un estado de congestión lo mejor es amainar tanto su impacto en la red así como su tiempo de duración.
- Previene el descarte de los paquetes más importantes si el buffer de las colas se llena por la cantidad de tráfico entrante. Bajo el esquema actual que se trabaja, Tail Drop y FIFO, los paquetes de cualquier aplicación son tratados de igual manera por lo que, en caso de entrar a aun estado de congestión, los paquetes serán descartados de igual manera.
- Proporcionar una correcta distribución de recursos a los servicios. Los servicios que se pueden estar ejecutando en la red tienen diferentes necesidades de ancho de banda, retardo, pérdida de paquetes por lo que según estas necesidades se debe no solo dimensionar sino también priorizar la asignación de ciertos recursos a ciertos servicios.

3. Relación entre la Gestión de Colas y el Control de Congestión

Gestión de colas y control de congestión son conceptos muy allegados e interrelacionados, muchas veces confundidos a pesar que poseen marcadas diferencias las cuales radican desde su empleo hasta los objetivos que se desean alcanzar aplicando estos conceptos.

Las diferencias más resaltantes entre estos conceptos son:

- Gestión de Colas se aplica en todo instante de tiempo. En todo momento la red produce tráfico, ya sea para funciones básicas del sistema o para los diferentes tipos de servicios que deben operar mientras la red esté operativa. La forma en el que los paquetes serán procesados deberá aplicarse en todo momento para un correcto desempeño de las aplicaciones, en especial para las aplicaciones de tiempo real.
- Control de Congestión se aplica no solo a los tráficos sino a ciertas aplicaciones. Muchas aplicaciones ya tiene integrada una solución de control de congestión de manera nativa como el caso de las aplicaciones que usan el protocolo TCP o aplicaciones que usan el protocolo DCCP que se acomodan al control de congestión establecido.

- Gestión de colas se basa en parámetros de Capas inferiores. Para lo que es colas, los parámetros radican en prioridades de la cabecera IP, de la cabecera MPLS (el campo experimental), para el caso de Frame Relay en el DLCI, en ATM en lo referente a VCI o circuito virtual, el protocolo de transporte o en el puerto (socket).
- Control de Congestión se basa en parámetros de Capas superiores y de aplicación. Las aplicaciones que las implementan se basan en acuses de recibo para poder medir la performance de la red y así poder adecuarse a los recursos disponibles.
- Gestión de colas trata de evitar el descarte de paquetes mientras que control de congestión utiliza esta alternativa para cumplir su objetivo. En efecto, en lo referente a colas se trata de evitar el encolamiento masivo de los paquetes de las aplicaciones ya que son nocivas para estas últimas, es decir, su función es preventiva. Por su parte, control de congestión se manifiesta cuando los recursos se vuelven insuficientes para poder satisfacer las necesidades de los tráficos por lo tanto tienen que priorizar bajo ciertos parámetros los paquetes que recibirán recursos pero afectando, en la menor forma posible, a los tráficos de menor prioridad.

Las similitudes más cercanas son:

- Ambos conceptos son aplicados a los routers del Core de la red. En efecto, los dos conceptos son necesarios para un óptimo funcionamiento de la red, ya que los servicios principales deben tener una prioridad mayor que los servicios que no son tan primordiales para el desempeño de los usuarios. Los recursos del router son escasos y limitados pero que deben ser compartidos por los usuarios por lo que la utilización adecuada de cada tipo de recurso disponible es esencial.
- Redistribución de recursos. Los dos conceptos tienen como objetivo la redistribución de los recursos dado que estos mismos se deben de compartir por muchos servicios, aplicaciones y usuarios en la red. Los criterios que se necesiten o que sean los más convenientes para el funcionamiento de la red deberá tener criterios que se aplicarán en todo momento así también políticas en caso de congestión en la red.
- Eficiencia de las aplicaciones así como de los servicios de red. Como se dijo anteriormente, las aplicaciones que se cursan en la red no tienen los mismos requerimientos de recursos para con la red, en especial las aplicaciones de tiempo

real [18] mientras que las aplicaciones tradicionales tienen necesidades menos rigurosas.

- Utilización de Conformación de Tráfico y de Políticas de Red. En efecto, la conformación de tráfico permite que los tráficos se aproximen a una función uniforme en el tiempo, es decir, que se aproxime a un valor adecuado dependiendo del tipo de tráfico; la suma de todos los tráficos conformados no deben superar la máxima capacidad de transferencia del enlace arrendado por lo que la probabilidad de congestión se reduce. Por otro lado las políticas de red nos permiten aplicar parámetros de descarte, de envío y de seguridad con lo cual complementamos a los diferentes criterios que se aplican en la red para obtener una mayor performance de la misma.

4. Tipos de Algoritmos para Manejo de Colas

A continuación se explicaN los algoritmos más conocidos para gestión de colas.

4.1. FIFO: First In First Out

Este algoritmo es el más difundido y el que se utiliza por defecto en las redes mundiales en el cual el primer paquete en llegar a la cola del router es el primero también en ser atendido y por ende puede ser enviado con mayor rapidez que los que le siguen.

Los beneficios de este tipo de cola son:

- El algoritmo FIFO consume pocos recursos de los routers por lo que puede ejecutarse muy rápidamente en comparación a otros algoritmos más sofisticados.
- Dado que FIFO tiene un comportamiento simple, es sencillo predecir cual será el comportamiento de los paquetes en el tiempo ya que no son reordenados y el tamaño de recepción de los mismos dependerá del tamaño del buffer de la cola.
- Mientras la cola no alcance su tope, el algoritmo FIFO provee una asignación de recursos simple sin sacrificar recursos y/o afectando el retardo del tráfico.

Las limitaciones de este tipo de cola son:

- El algoritmo simple de FIFO no es capaz de clasificar por si solo los paquetes entrantes, es decir, los paquetes son tomados como iguales sin prioridad entre ellos.

- El algoritmo tiene un impacto igualitario en los flujos al no ser capaz de una diferenciación por lo que contribuye a la congestión lo cual impacta a parámetros como el retardo, jitter y pérdida de paquetes en especial para los tráficos de tiempo real.
- Cuando la cola llega a sus límites y causa congestión, el tráfico TCP es más afectado frente al tráfico UDP. Véase que el tráfico TCP al experimentar pérdidas de paquetes y ausencia de confirmaciones, reduce su velocidad para amainar la congestión; al contrario, UDP no deja de enviar paquetes por lo que termina copando el buffer de la cola.
- La naturaleza de Internet es de tráficos de tipo ráfaga, comportamiento que llena rápidamente los buffer de los routers por lo que otras aplicaciones diferentes a la que emitió el tráfico tipo ráfaga no podrán acceder a los recursos y sus paquetes sufrirán descarte de paquetes hasta que el estado de congestión se disipe.

4.2. Priority Queuing

Este tipo de algoritmo es el más simple y esencial entre los criterios de encolamiento para poder ofrecer una diferenciación entre los diferentes flujos que cursan a red y compiten por los recursos del router. Este algoritmo clasifica a los paquetes entrantes, generalmente en base al campo ToS o DS, después de lo cual los paquetes son enviados a colas FIFO de diferente prioridad, en las que para pasar de cola en cola, la cola de mayo prioridad debe estar vacía.

Los beneficios de este tipo de cola son:

- El algoritmo PQ consume pocos recursos de los routers por lo que puede ejecutarse muy rápidamente en comparación a otros algoritmos más sofisticados de clasificación.
- La clasificación que realiza el algoritmo permite aplicar un trato diferente a los flujos, de tal modo que se aplica una redistribución de recursos básica.

Las limitaciones de este tipo de cola son:

- El algoritmo beneficia a los flujos de alta prioridad frente a los de menor prioridad ya que si las colas de altas prioridades no son atendidas en su totalidad las de menor

prioridad no pueden ser atendidas, lo cual contribuye al congestionamiento de estas colas y en casos extremos, el descarte de los paquetes.

- Un tráfico de alta prioridad tipo ráfaga puede consumir los recursos enteros de la cola correspondiente, lo cual lleva a una congestión que se da con el algoritmo FIFO, es decir, ningún paquete adicional puede ser recepcionado.
- La limitación de TCP vs. UDP no es solucionada por este algoritmo, el que tenga mayor prioridad será el menos afectado pero perjudica a los demás.

4.3. Fair Queuing

Tiene el mismo principio que su antecesor PQ pero con la diferencia de que si bien se tiene colas de mayor prioridad, estas solo serán atendidas una sola vez por ciclo con lo cual las de menor prioridad tienen oportunidad de enviar los paquetes encolados. A diferencia de FIFO y PQ, se atiende por flujos encolados y no por paquete por paquete.

Los beneficios de este tipo de cola son:

- Los tráficos tipo ráfaga no afectan a los demás tráficos que solicitan recursos dado que cada uno posee su propia cola de envío, es decir, que si el tráfico tiende a utilizar todos los recursos, solo se utilizarán los recursos propios y no de otros tráficos por lo que el impacto solo será para este tráfico tipo ráfaga.

Las limitaciones de este tipo de cola son:

- El procesamiento que se necesita así como la cantidad de recursos es considerable, por lo que los routers en los que se implementará este tipo de gestión deben estar preparados para las diferentes exigencias de los tráficos así como para su rápido procesamiento.
- Al atender a las colas de diferentes prioridades les otorga en ese instante le uso exclusivo del ancho de banda, discriminando su necesidad de este. Los flujos conformados por paquetes de mayor tamaño son los más beneficiados.
- Los flujos estar forzados a esperar el, turno por ciclo que les corresponde para ser atendidos. Si las colas que se tienen en el routers son variadas, el tiempo de espera se extiende y por ende los parámetros de los flujos serán impactados negativamente.

- La falta de escalabilidad es un punto crítico de este algoritmo. La clasificación de los flujos por parte del administrador de red son una parte esencial del funcionamiento ya que según los criterios a emplear, los tráficos serán impactados. Si se tiene demasiadas colas en los routers estos necesitarán mayor cantidad de recursos y los tráficos tendrán que esperar más tiempo para poder ser atendidos.

4.4. Weighted Fair Queuing WFQ

Este algoritmo WFQ provee mejoras frente a su antecesor FQ el cual poseía muchos supuestos que no necesariamente se deben de dar en una red, en especial en la red de Internet. Este algoritmo es capaz de proporcionar a los flujos definidos un ancho de banda de acuerdo a sus necesidades, es decir, un ancho de banda adecuado para su funcionamiento.

Este tipo de cola sugiere una aproximación del algoritmo GPS, el cual según el ancho de banda asignado, transmite bit-a-bit el paquete recibido para que luego sea reensamblado después de recibirse el último bit del mismo. Dado que en la realidad esta implementación no es posible, el algoritmo WFQ, el algoritmo se basa en un cálculo de tiempo límite en función del ancho de banda asignado; el paquete que posea un menor tiempo limite de envío será el que se trasmita.

Los beneficios de este tipo de cola son:

- Provee a los tráficos de una asignación mínima de ancho de banda independiente del comportamiento de los otros tráficos en las colas. El ancho de banda asignado a cada tráfico así como el tamaño de los paquetes determinarán el orden de transmisión.
- Los tráficos tienen un retardo limitado ya que, incluso en congestión de la cola, los tráficos de menor prioridad pueden trasmitir sus paquetes a las redes destino.

Las limitaciones de este tipo de cola son:

- El costo computacional de esta cola es muy alto, en especial por los constantes cálculos que se tiene que efectuar por cada paquete que arriba a las colas creadas.

- La performance de los tráficos puede ser afectado negativamente por tráficos de la misma prioridad con paquetes más grandes, ya que los paquetes grandes son transmitidos después de los pequeños por lo que si en una misma cola se presentan paquetes de diferente tamaño, los paquetes grandes incrementarán el retardo.
- Falta de escalabilidad de la cola, ya que depende mucho de los recursos computacionales del router así como de la clasificación de los flujos que curse la red.

4.5. Weighted Round Robin o Class-based Queuing

Este algoritmo es uno de los más sofisticados el cual recoge lo mejor de FQ y PQ además de superar las limitaciones que se tenían en estos esquemas. Gracias a este algoritmo se puede asignar diferentes anchos de banda a los tráficos, clasificación de paquetes en colas Round Robin y asegurando que los paquetes de menor prioridad no se vean afectados. Según el ancho de banda asignado, las colas podrán enviar más de un paquete por ciclo pero no inmediatamente sino que lo hace dentro de otro tiempo dentro del mismo ciclo.

Los beneficios de este tipo de cola son:

- Permite un manejo más rígido sobre el ancho de banda asignado a los tráficos que cursan la red. La cantidad de veces que se visita una cola para que se trasmita un paquete es proporcional al ancho de banda que se le asigna por lo que se asegura la trasmisión de por lo menos de un paquete por ciclo además de suprimir la apropiación de recursos.
- Brinda un escenario para el soporte de servicios diferenciados.

Las limitaciones de este tipo de cola son:

- Dado que el ancho de banda es el parámetro que define el número de veces que la cola a pueda trasmitir un paquete durante un ciclo, por lo que se intuye que si los paquetes son del mismo tamaño, la distribución de ancho de banda será la indicada, caso contrario, los paquetes de mayor tamaño excederán el ancho de banda asignado.

4.6. Deficit Weighted Round Robin

Este algoritmo es una mejora del WRR definido anteriormente ya que se acomoda a los tráficos con diferente tamaño de paquete además de proporcionar un algoritmo simple basado en el algoritmo antecesor WFF por lo que no consume grandes recursos de procesamiento.

En este algoritmo se utilizan tres parámetros:

Weight: Fracción del ancho de banda asignado a la cola.

DeficitCounter: Contador que almacena la cantidad de de bytes que la cola tiene la capacidad de trasmitir durante la visita a la cola. El valor inicial de la variable es cero.

Quantum: Valor proporcional al ancho de banda asignado el cual incrementará el DeficitCounter un momento antes de que se visite la cola.

El funcionamiento de este algoritmo es muy parecido al WRR con la diferencia de que en este algoritmo se opera al déficit del tamaño del paquete. Cada vez que la cola se visita, el DeficitCounter es incrementado en un Quantum, al nuevo valor del DeficitCounter se le compara con el tamaño del paquete a trasmitir, si este es menor igual procede con la trasmisión y el DeficitCounter tomará el valor de la diferencia entre el tamaño del paquete y su antiguo valor para nuevamente compararlo con el siguiente paquete y verificar que se cumpla la condición de trasmisión, caso contrario, se pasa a la siguiente cola y se guarda el valor obtenido del DeficitCounter. El ciclo se repetirá hasta que la condición para la trasmisión se suscite.

Los beneficios de este tipo de cola son:

- Los comportamientos tipo ráfaga de ciertos flujos no tienen impacto sobre los otros flujos en otras colas, nótese que cada cola tiene independencia no solo en comportamiento sino también en lo referente a parámetros.
- Al tener operación por déficit, puede acomodarse a tamaño fijo o variable de los paquetes. El DeficitCounter es el encargado de brindar la capacidad de trasmisión, sino se cumple, se espera a que el DeficitCounter llegue al valor que se necesite para trasmitir.
- En cada ciclo se asegura una asignación de ancho de banda para las colas.

Las limitaciones de este tipo de cola son:

- La congestión por cola no se puede evitar. Si bien la congestión entre las colas configuradas puede ser manejada con la asignación de recursos y parámetros antes mencionada, los tráficos pertenecientes a una sola cola (tráficos agregados) si pueden tener conflicto entre ellos lo cual amaina su performance.
- El retardo que se tiene no es predecible. Cuando el paquete es más grande que el DeficitCounter se espera un ciclo para que este se incremente y verificar que se cumpla la condición de trasmisión y así se seguirá hasta que el paquete cumpla con la condición. En las redes IP los tamaños son muy variables y no hay garantía de su tamaño, por lo cual el retardo dependerá mucho de los parámetros asignados.

5. Conclusiones

- Necesidad de implantar en las redes de la Backbone de Internet una Gestión de Colas y Control de Congestión adecuada para las aplicaciones de tiempo real. El tipo de cola que más beneficie a los tráficos cursantes sin perjudicar a los que tienen menor prioridad.
- Las políticas a implantar en las redes internas deben tener en cuenta las aplicaciones más importantes y trascendentales para su funcionamiento.
- Las aplicaciones al tener diferentes necesidades de asignación de recursos reaccionan diferente ante los parámetros implantados en las redes.
- La utilización de conformado de trafico así como de Políticas de red. Así se evita congestiones indeseadas en el Core de la red el cual es el punto más vulnerable de cualquier red IP por la gran cantidad de tráfico que esta maneja.
- Necesidad combinar los conceptos con ingeniería de tráfico y con ingeniería de red. No solo dimensionar los enlaces y los nodos para que resistan la carga de tráfico, sino que también puedan redirigir el tráfico a los enlaces menos utilizados en la red para poder equilibrar la carga y evitar pérdida de paquetes.

ANEXO 3
INSTALACIÓN Y UTILIZACIÓN DEL SIMULADOR OMNET++ Y DEL INET FRAMEWORK EN WINDOWS

1. Instalación del Simulador OMNET++

El OMNET++ se puede instalar en cualquier sistema operativo, sea este Windows o Linux. Para la presente tesis, el sistema operativo elegido fue Windows XP por lo que en nuestro caso se utilizó la versión creada para este sistema operativo.

Para poder obtener el instalador basta con ir al OMNET++ Community Site: http://www.omnetpp.org. La versión más reciente del simulador es la que se encuentra disponible en la página: http://www.omnetpp.org/filemgmt/singlefile.php?lid=123, en la cual se explican además las ventajas y algunos consejos sobre el uso del simulador.

Antes de su instalación se requiere otros programas ara su funcionamiento óptimo. Los programas que se necesitan son:

- **Ghostscript:** Este programa, al igual que el OMNET++, es de distribución gratuita. Este programa nos ayudará para ciertas funciones de los gráficos a generarse así como en la presentación de archivos tipo PS y/o PDF si se necesitase. Se puede descargar el inhalador para Windows de la página: http://pages.cs.wisc.edu/~ghost/.
- **Visual C++ Express Edition:** Este IDE gratuito nos permitirá no solo compilar las simulaciones, sino que para futuros trabajos programar nuevos módulos de nuevos protocolos que interaccionen con el OMNET++ sin mayores complicaciones. La página de descarga: http://www.microsoft.com/express/download/
- **Platform SDK:** Este pull de librerías propias del Windows nos permitirán, junto con el Visual C++ Express Edition, programar los módulos así como acceder a las librerías necesarias como la “windows.h” y sus dependencias. La página de descarga gratuita es: http://www.microsoft.com/downloads/details.aspx?FamilyId=A55B6B43-E24F-4EA3-A93E-40C0EC4F68E5&displaylang=en

Nótese que los instaladores de los programas antes mencionados serán los encargados de configurar las variables de entorno del sistema operativo, de tal forma que solo se tendrá que corroborar el buen funcionamiento de las mismas.

Después de haber instalado el software antes indicado, se procede a la instalación del OMNET++. En el proceso de instalación, se nos solicitará la ruta del Ghostscript y la versión del Visual C++ Express Edition que hemos instalado, después de lo cual completará la instalación sin mayores inconvenientes.

2. Instalación del INET Framework

Luego de instalar el OMNET++, se procede con la instalación del INET Framework. Este software nos permite hacer uso de los protocolos tales como IEEE802.11, Ethernet, PPP, IPv6, OSPF, RIP, MPLS LDP y MPLS RSVP-TE signalling, entre otros protocolos.

La versión INET Framework Demo for Windows, la cual contiene los ejemplos de los módulos soportados ya compilados y listos para ejecutarse. La página de descarga del instalador es: http://www.omnetpp.org/filemgmt/visit.php?lid=121.

La versión tradicional tiene el código fuente de los módulos si se necesitase la modificación de estos. Después de modificaciones en el código fuente es necesario compilar nuevamente el código fuente como se indica en los archivos contenidos en el package. La página de descarga del instalador es: http://www.omnetpp.org/filemgmt/visit.php?lid=120.

ANEXO 4
ARCHIVOS DE SIMULACIÓN DE MPLS EN EL OMNET++/INET FRAMEWORK

En el presente anexo se presentan los archivos de configuración “omnetpp.ini” que se utilizaron para poder efectuar las simulaciones y obtener los resultados que se señalaron en el capítulo 5 del presente documento

1. Arquitectura MPLS LDP

En este apartado se muestran los archivos omnetpp.ini que se utilizaron para simular las topologías MAN y las topologías BACKBONE en las cuales se aplicaron el funcionamiento de la arquitectura MPLS LDP.

1.1. Topología MPLS LDP MAN

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = MAN
sim-time-limit = 15m
total-stack-kb =  65536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]

# Configuración de los Host Cliente y Servidor
**.numUdpApps=1
**.host1.udpAppType="UDPVideoStreamCli"
**.host1.udpApp[0].serverAddress = "host2"
**.host1.udpApp[0].localPort = 9999
**.host1.udpApp[0].serverPort = 3088
**.host1.udpApp[0].startTime = uniform(5, 5.01)

**.host2.udpAppType = "UDPVideoStreamSvr"
**.host2.udpApp[0].videoSize = 5e7
**.host2.udpApp[0].serverPort = 3088
**.host2.udpApp[0].waitInterval = .001
**.host2.udpApp[0].packetLen = 1000

**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
```

```
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.host*.IPForward=false
**.host*.routingFile = ""
**.ip.procDelay=10us

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# Configuración de los routers MPLS LSR
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")
**.LSR*.holdTime = 6s
**.LSR*.helloInterval = 2s

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 4 -n Brown;c
-t * -i 5 -n Magenta;c -t * -i 6 -n Orange;c -t * -i 100 -n Green"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto


# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 10  # en routers

# scenario
**.scenarioManager.script = xmldoc("scenario.xml")
```

1.2. Topología MPLS LDP MAN Real

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = MAN
sim-time-limit = 10m
total-stack-kb =  6665536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1
```

```
[Parameters]
# Configuración de los Host de videoconferencia
**.host1.udpApp_StreamCli.serverAddress = "host4"
**.host4.udpApp_StreamCli.serverAddress = "host1"
**.host3.udpApp_StreamCli.serverAddress = "host6"
**.host6.udpApp_StreamCli.serverAddress = "host3"
**.host5.udpApp_StreamCli.serverAddress = "host2"
**.host2.udpApp_StreamCli.serverAddress = "host5"

**.host*.udpApp_StreamCli.localPort = 9999
**.host*.udpApp_StreamCli.serverPort = 3088
**.host*.udpApp_StreamCli.startTime = 5

# Se envía video de 30fps, cada uno de 33.6Kb.
**.host*.udpApp_StreamSvr.videoSize = 50e7 #800Mb
**.host*.udpApp_StreamSvr.serverPort = 3088
**.host*.udpApp_StreamSvr.waitInterval = .01 # 30 fps
**.host*.udpApp_StreamSvr.packetLen =4.2e3 # 33.6Kb

#Se configuran las aplicaciones TCP
**.numTcpApps=1

# Se procede con la configuración FTP de clientes y servers correspondientes
**.File_Cliente*.tcpAppType="TCPSessionApp"
**.File_Cliente*.tcpApp[0].active=true
**.File_Cliente*.tcpApp[0].address=""
**.File_Cliente*.tcpApp[0].port=-1            # puerto predeterminado
**.File_Cliente*.tcpApp[0].connectAddress="File_Server"
**.File_Cliente*.tcpApp[0].connectPort=1000
**.File_Cliente*.tcpApp[0].tOpen=6.0          # Tiempo de apertura de sesión
**.File_Cliente*.tcpApp[0].tSend=1.1
**.File_Cliente*.tcpApp[0].sendBytes=20e6     # Se envían 20MB
**.File_Cliente*.tcpApp[0].sendScript=""
**.File_Cliente*.tcpApp[0].tClose=0

**.File_Server.tcpAppType="TCPEchoApp"
**.File_Server.tcpApp[0].address=""
**.File_Server.tcpApp[0].port=1000
**.File_Server.tcpApp[0].echoFactor=2.0
**.File_Server.tcpApp[0].echoDelay=0

# Se procede con la configuración HTTP de clientes y servers correspondientes
**.HTTP_Cliente*.tcpAppType="TCPBasicClientApp"
**.HTTP_Cliente*.tcpApp[0].address=""
**.HTTP_Cliente*.tcpApp[0].port=-1
**.HTTP_Cliente*.tcpApp[0].connectAddress="HTTP_Server"
**.HTTP_Cliente*.tcpApp[0].connectPort=80

# Se configuran parámetros del cliente tales como el número de solicitudes por
# sesión el tamaño de la respuesta a las solicitudes y el tiempo de respuesta.
**.HTTP_Cliente*.tcpApp[0].startTime=7
**.HTTP_Cliente*.tcpApp[0].numRequestsPerSession = exponential(10)
**.HTTP_Cliente*.tcpApp[0].requestLength = truncnormal(45000,20)
**.HTTP_Cliente*.tcpApp[0].replyLength = exponential(40000)
**.HTTP_Cliente*.tcpApp[0].thinkTime=truncnormal(0.1,0.3)
**.HTTP_Cliente*.tcpApp[0].idleInterval=truncnormal(2,5)
**.HTTP_Cliente*.tcpApp[0].reconnectInterval=2

**.HTTP_Server.tcpAppType="TCPGenericSrvApp"
**.HTTP_Server.tcpApp[0].address=""
**.HTTP_Server.tcpApp[0].port=80
**.HTTP_Server.tcpApp[0].replyDelay=0

**.HTTP**.numUdpApps=0
**.HTTP**.udpAppType="UDPBasicApp"
```

```
**.File**.numUdpApps=0
**.File**.udpAppType="UDPBasicApp"

# Parámetros de configuración a nivel IP
**.IPForward=false
**.routingFile = ""
**.ip.procDelay=1us

**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 4 -n Brown;c
-t * -i 5 -n Magenta;c -t * -i 6 -n Orange;c -t * -i 100 -n Green"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# Configuración de los routers MPLS LSR
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")
**.LSR*.holdTime = 6s
**.LSR*.helloInterval = 2s

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 30  # en routers

# parámetros propios del escenario
**.scenarioManager.script = xmldoc("scenario.xml")
```

1.3. Topología MPLS LDP BACKBONE

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = BACKBONE
sim-time-limit = 15m
total-stack-kb =  6665536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]

# Configuración de los Host Cliente y Servidor
**.numUdpApps=1
**.host1.udpAppType="UDPVideoStreamCli"
**.host1.udpApp[*].serverAddress = "host7"
**.host1.udpApp[*].localPort = 9999
**.host1.udpApp[*].serverPort = 3088
**.host1.udpApp[*].startTime = uniform(5, 5.01)

**.host7.udpAppType = "UDPVideoStreamSvr"
**.host7.udpApp[0].videoSize = 5e7
**.host7.udpApp[0].serverPort = 3088
**.host7.udpApp[0].waitInterval = .001
**.host7.udpApp[0].packetLen = 1000

**.host2.udpAppType="UDPVideoStreamCli"
**.host2.udpApp[*].serverAddress = "host8"
**.host2.udpApp[*].localPort = 9999
**.host2.udpApp[*].serverPort = 3088
**.host2.udpApp[*].startTime = uniform(5, 5.01)

**.host8.udpAppType = "UDPVideoStreamSvr"
**.host8.udpApp[*].videoSize = 5e7
**.host8.udpApp[*].serverPort = 3088
**.host8.udpApp[*].waitInterval = .001
**.host8.udpApp[*].packetLen = 1000

**.host5.udpAppType="UDPVideoStreamCli"
**.host5.udpApp[*].serverAddress = "host3"
**.host5.udpApp[*].localPort = 9999
**.host5.udpApp[*].serverPort = 3088
**.host5.udpApp[*].startTime = uniform(5, 5.01)

**.host3.udpAppType = "UDPVideoStreamSvr"
**.host3.udpApp[*].videoSize = 5e7
**.host3.udpApp[*].serverPort = 3088
**.host3.udpApp[*].waitInterval = .001
**.host3.udpApp[*].packetLen = 1000

**.host6.udpAppType="UDPVideoStreamCli"
**.host6.udpApp[*].serverAddress = "host4"
**.host6.udpApp[*].localPort = 9999
**.host6.udpApp[*].serverPort = 3088
**.host6.udpApp[*].startTime = uniform(5, 5.01)

**.host4.udpAppType = "UDPVideoStreamSvr"
**.host4.udpApp[*].videoSize = 5e7
**.host4.udpApp[*].serverPort = 3088
```

```
**.host4.udpApp[*].waitInterval = .001
**.host4.udpApp[*].packetLen = 1000

# tcp apps
**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

# ping app
**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.host*.IPForward=false
**.host*.routingFile = ""
**.ip.procDelay=10us

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# LSR configuration
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.holdTime = 6s
**.LSR*.helloInterval = 2s

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 100 -n
Green;c -t * -i 101 -n Magenta;c -t * -i 200 -n Orange;c -t * -i 300 -n Brown"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

**.LSR*.libTable.conf = xmldoc("_lib.xml")

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 10  # en routers

# scenario
**.scenarioManager.script = xmldoc("scenario.xml")
```

1.4. Topología MPLS LDP BACKBONE Real

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = BACKBONE
sim-time-limit = 15m
total-stack-kb =  6665536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]
# Configuración de los Host de videoconferencia
**.host1.udpApp_StreamCli.serverAddress = "host5"
**.host5.udpApp_StreamCli.serverAddress = "host1"
**.host9.udpApp_StreamCli.serverAddress = "host16"
**.host16.udpApp_StreamCli.serverAddress = "host9"
**.host2.udpApp_StreamCli.serverAddress = "host6"
**.host6.udpApp_StreamCli.serverAddress = "host2"
**.host15.udpApp_StreamCli.serverAddress = "host10"

**.host10.udpApp_StreamCli.serverAddress = "host15"
**.host3.udpApp_StreamCli.serverAddress = "host7"
**.host7.udpApp_StreamCli.serverAddress = "host3"
**.host11.udpApp_StreamCli.serverAddress = "host14"
**.host14.udpApp_StreamCli.serverAddress = "host11"
**.host8.udpApp_StreamCli.serverAddress = "host4"
**.host4.udpApp_StreamCli.serverAddress = "host8"
**.host12.udpApp_StreamCli.serverAddress = "host13"
**.host13.udpApp_StreamCli.serverAddress = "host12"

**.host*.udpApp_StreamCli.localPort = 9999
**.host*.udpApp_StreamCli.serverPort = 3088
**.host*.udpApp_StreamCli.startTime = 5

# Se envía video de 30fps, cada uno de 33.6Kb.
**.host*.udpApp_StreamSvr.videoSize = 50e7 #800Mb
**.host*.udpApp_StreamSvr.serverPort = 3088
**.host*.udpApp_StreamSvr.waitInterval = .03 # 30 fps
**.host*.udpApp_StreamSvr.packetLen = 4.2e3 # 33.6Kb

#este parámetro es para confirar TCP en los host que lo tendrán
**.numTcpApps=1

# Se procede con la configuración FTP de clientes y servers correspondientes
**.File_Cliente*.tcpAppType="TCPSessionApp"
**.File_Cliente*.tcpApp[0].active=true
**.File_Cliente*.tcpApp[0].address=""
**.File_Cliente*.tcpApp[0].port=-1
**.File_Cliente*.tcpApp[0].connectAddress="File_Server"
**.File_Cliente*.tcpApp[0].connectPort=1000
**.File_Cliente*.tcpApp[0].tOpen=6.0          # Tiempo de apertura de sesión
**.File_Cliente*.tcpApp[0].tSend=1.1
**.File_Cliente*.tcpApp[0].sendBytes=20e6  # 20MB
**.File_Cliente*.tcpApp[0].sendScript=""
**.File_Cliente*.tcpApp[0].tClose=0

**.File_Server.tcpAppType="TCPEchoApp"
**.File_Server.tcpApp[0].address=""
**.File_Server.tcpApp[0].port=1000
```

```
**.File_Server.tcpApp[0].echoFactor=2.0
**.File_Server.tcpApp[0].echoDelay=0

# Se procede con la configuración HTTP de clientes y servers correspondientes
**.HTTP_Cliente*.tcpAppType="TCPBasicClientApp"
**.HTTP_Cliente*.tcpApp[0].address=""
**.HTTP_Cliente*.tcpApp[0].port=-1
**.HTTP_Cliente*.tcpApp[0].connectAddress="HTTP_Server"
**.HTTP_Cliente*.tcpApp[0].connectPort=80

# Se configuran parámetros del cliente tales como el número de solicitudes por
# sesión el tamaño de la respuesta a las solicitudes y el tiempo de respuesta.
**.HTTP_Cliente*.tcpApp[0].startTime=7
**.HTTP_Cliente*.tcpApp[0].numRequestsPerSession = exponential(10)
**.HTTP_Cliente*.tcpApp[0].requestLength = truncnormal(45000,20)
**.HTTP_Cliente*.tcpApp[0].replyLength = exponential(40000)
**.HTTP_Cliente*.tcpApp[0].thinkTime=truncnormal(0.1,0.3)
**.HTTP_Cliente*.tcpApp[0].idleInterval=truncnormal(2,5)
**.HTTP_Cliente*.tcpApp[0].reconnectInterval=2

**.HTTP_Server.tcpAppType="TCPGenericSrvApp"
**.HTTP_Server.tcpApp[0].address=""
**.HTTP_Server.tcpApp[0].port=80
**.HTTP_Server.tcpApp[0].replyDelay=0

**.HTTP**.numUdpApps=0
**.HTTP**.udpAppType="UDPBasicApp"
**.File**.numUdpApps=0
**.File**.udpAppType="UDPBasicApp"

**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

# ping app
**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.IPForward=false
**.routingFile = ""
**.ip.procDelay=1us

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"
```

```
# Configuración de los routers MPLS LSR
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")
**.LSR*.holdTime = 6s
**.LSR*.helloInterval = 2s

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 100 -n
Green;c -t * -i 101 -n Magenta;c -t * -i 200 -n Orange;c -t * -i 300 -n Brown"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 30  # en routers

# parámetros propios del escenario
**.scenarioManager.script = xmldoc("scenario.xml")
```

2. Arquitectura MPLS RSVP

En este apartado se muestran los archivos omnetpp.ini que se utilizaron para simular las topologías MAN y las topologías BACKBONE en las cuales se aplicaron el funcionamiento de la arquitectura MPLS RSVP. Nótese que se presentan los archivos correspondientes a las simulaciones con tráfico ajeno al de interés dado que estas topologías son de mayor interés al estudio realizado.

2.1. Topología MPLS RSVP MAN

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = MAN
sim-time-limit = 15m
total-stack-kb =  65536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]

# Configuración de los Host Cliente y Servidor
**.numUdpApps=1
**.host1.udpAppType="UDPVideoStreamCli"
**.host1.udpApp[0].serverAddress = "host2"
**.host1.udpApp[0].localPort = 9999
**.host1.udpApp[0].serverPort = 3088
**.host1.udpApp[0].startTime = uniform(5, 5.01)
```

```
**.host2.udpAppType = "UDPVideoStreamSvr"
**.host2.udpApp[0].videoSize = 5e7
**.host2.udpApp[0].serverPort = 3088
**.host2.udpApp[0].waitInterval = .001
**.host2.udpApp[0].packetLen = 1000
#se envía 1MBps

# tcp apps
**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

# ping app
**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.host*.IPForward=false
**.host*.routingFile = ""
**.ip.procDelay=10us""

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# LSR configuration

**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 4 -n Brown;c
-t * -i 5 -n Magenta;c -t * -i 6 -n Orange;c -t * -i 100 -n Green"

**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

**.LSR*.libTable.conf = xmldoc("_lib.xml")


# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
```

```
**.ppp[*].queue.frameCapacity = 10  # en routers

# scenario
**.scenarioManager.script = xmldoc("scenario.xml")

# RSVP, MPLS settings
# Nótese que el tráfico de interés se da entre los host cliente y servidor,
# entonces los routers encargados de la formación de VPN's son los ILER y ELER
**.LSR14.classifier.conf = xmldoc("LSR14_fec.xml")
**.LSR14.rsvp.traffic = xmldoc("LSR14_rsvp.xml")
# Se da la opción de clasificación del tráfico para su respectiva
diferenciación.
**.LSR*.classifier.conf = xmldoc("_fec.xml")
**.LSR*.rsvp.traffic = xmldoc("_traffic.xml")
**.LSR*.rsvp.helloInterval = 0.2
**.LSR*.rsvp.helloTimeout = 0.5
**.LSR*.libTable.conf = xmldoc("_lib.xml")
```

2.2. Topología MPLS RSVP MAN Real

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = MAN
sim-time-limit = 15m
total-stack-kb =  65536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]
# Configuración de los Host de videoconferencia
**.host1.udpApp_StreamCli.serverAddress = "host4"
**.host4.udpApp_StreamCli.serverAddress = "host1"
**.host3.udpApp_StreamCli.serverAddress = "host6"
**.host6.udpApp_StreamCli.serverAddress = "host3"
**.host5.udpApp_StreamCli.serverAddress = "host2"
**.host2.udpApp_StreamCli.serverAddress = "host5"

**.host*.udpApp_StreamCli.localPort = 9999
**.host*.udpApp_StreamCli.serverPort = 3088
**.host*.udpApp_StreamCli.startTime = 5

# Se envía video de 30fps, cada uno de 33.6Kb.
**.host*.udpApp_StreamSvr.videoSize = 50e7 #800Mb
**.host*.udpApp_StreamSvr.serverPort = 3088
**.host*.udpApp_StreamSvr.waitInterval = .01 # 30 fps
**.host*.udpApp_StreamSvr.packetLen =4.2e3 # 33.6Kb

#Se configuran las aplicaciones TCP
**.numTcpApps=1

# Se procede con la configuración FTP de clientes y Servers correspondientes
**.File_Cliente*.tcpAppType="TCPSessionApp"
**.File_Cliente*.tcpApp[0].active=true
**.File_Cliente*.tcpApp[0].address=""
**.File_Cliente*.tcpApp[0].port=-1            # puerto predeterminado
**.File_Cliente*.tcpApp[0].connectAddress="File_Server"
**.File_Cliente*.tcpApp[0].connectPort=1000
**.File_Cliente*.tcpApp[0].tOpen=6.0          # Tiempo de apertura de sesión
```

```
**.File_Cliente*.tcpApp[0].tSend=1.1
**.File_Cliente*.tcpApp[0].sendBytes=20e6     # Se envían 20MB
**.File_Cliente*.tcpApp[0].sendScript=""
**.File_Cliente*.tcpApp[0].tClose=0

**.File_Server.tcpAppType="TCPEchoApp"
**.File_Server.tcpApp[0].address=""
**.File_Server.tcpApp[0].port=1000
**.File_Server.tcpApp[0].echoFactor=2.0
**.File_Server.tcpApp[0].echoDelay=0

# Se procede con la configuración HTTP de clientes y servers correspondientes
**.HTTP_Cliente*.tcpAppType="TCPBasicClientApp"
**.HTTP_Cliente*.tcpApp[0].address=""
**.HTTP_Cliente*.tcpApp[0].port=-1
**.HTTP_Cliente*.tcpApp[0].connectAddress="HTTP_Server"
**.HTTP_Cliente*.tcpApp[0].connectPort=80

# Se configuran parámetros del cliente tales como el número de solicitudes por
# sesión el tamaño de la respuesta a las solicitudes y el tiempo de respuesta.
**.HTTP_Cliente*.tcpApp[0].startTime=7
**.HTTP_Cliente*.tcpApp[0].numRequestsPerSession = exponential(10)
**.HTTP_Cliente*.tcpApp[0].requestLength = truncnormal(45000,20)
**.HTTP_Cliente*.tcpApp[0].replyLength = exponential(40000)
**.HTTP_Cliente*.tcpApp[0].thinkTime=truncnormal(0.1,0.3)
**.HTTP_Cliente*.tcpApp[0].idleInterval=truncnormal(2,5)
**.HTTP_Cliente*.tcpApp[0].reconnectInterval=2

**.HTTP_Server.tcpAppType="TCPGenericSrvApp"
**.HTTP_Server.tcpApp[0].address=""
**.HTTP_Server.tcpApp[0].port=80
**.HTTP_Server.tcpApp[0].replyDelay=0

**.HTTP**.numUdpApps=0
**.HTTP**.udpAppType="UDPBasicApp"
**.File**.numUdpApps=0
**.File**.udpAppType="UDPBasicApp"

# Parámetros de configuración a nivel IP
**.IPForward=false
**.routingFile = ""
**.ip.procDelay=1us

**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
```

```
**.tcp.recordStats=true


# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# Configuración de los routers MPLS RSVP
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 4 -n Brown;c
-t * -i 5 -n Magenta;c -t * -i 6 -n Orange;c -t * -i 100 -n Green"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 30  # en routers

# parámetros propios del escenario
**.scenarioManager.script = xmldoc("scenario.xml")

# RSVP, MPLS settings
# Nótese que el tráfico de interés se da entre los host de videoconferencia,
# entonces los routers encargados de la formación de VPN's son los ILER y ELER
**.LSR1.classifier.conf = xmldoc("LSR1_fec.xml")
**.LSR1.rsvp.traffic = xmldoc("LSR1_rsvp.xml")
**.LSR2.classifier.conf = xmldoc("LSR2_fec.xml")
**.LSR2.rsvp.traffic = xmldoc("LSR2_rsvp.xml")
**.LSR3.classifier.conf = xmldoc("LSR3_fec.xml")
**.LSR3.rsvp.traffic = xmldoc("LSR3_rsvp.xml")
**.LSR12.classifier.conf = xmldoc("LSR12_fec.xml")
**.LSR12.rsvp.traffic = xmldoc("LSR12_rsvp.xml")
**.LSR13.classifier.conf = xmldoc("LSR13_fec.xml")
**.LSR13.rsvp.traffic = xmldoc("LSR13_rsvp.xml")
**.LSR14.classifier.conf = xmldoc("LSR14_fec.xml")
**.LSR14.rsvp.traffic = xmldoc("LSR14_rsvp.xml")

# Se da la opción de clasificación del tráfico para su respectiva
diferenciación.
**.LSR*.classifier.conf = xmldoc("_fec.xml")
**.LSR*.rsvp.traffic = xmldoc("_traffic.xml")
**.LSR*.rsvp.helloInterval = 0.2              # Intevalo de Paquetes Hello
**.LSR*.rsvp.helloTimeout = 0.5               # Tiempo de espera de Hello
**.LSR*.libTable.conf = xmldoc("_lib.xml")
```

2.3. Topología MPLS RSVP BACKBONE

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = BACKBONE
sim-time-limit = 15m
total-stack-kb =  6665536
[Cmdenv]
express-mode = no
```

```
[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]

# Configuración de los Host Cliente y Servidor
**.numUdpApps=1
**.host1.udpAppType="UDPVideoStreamCli"
**.host1.udpApp[*].serverAddress = "host7"
**.host1.udpApp[*].localPort = 9999
**.host1.udpApp[*].serverPort = 3088
**.host1.udpApp[*].startTime = uniform(5, 5.01)

**.host7.udpAppType = "UDPVideoStreamSvr"
**.host7.udpApp[0].videoSize = 5e7
**.host7.udpApp[0].serverPort = 3088
**.host7.udpApp[0].waitInterval = .001
**.host7.udpApp[0].packetLen = 1000

**.host2.udpAppType="UDPVideoStreamCli"
**.host2.udpApp[*].serverAddress = "host8"
**.host2.udpApp[*].localPort = 9999
**.host2.udpApp[*].serverPort = 3088
**.host2.udpApp[*].startTime = uniform(5, 5.01)

**.host8.udpAppType = "UDPVideoStreamSvr"
**.host8.udpApp[*].videoSize = 5e7
**.host8.udpApp[*].serverPort = 3088
**.host8.udpApp[*].waitInterval = .001
**.host8.udpApp[*].packetLen = 1000

**.host5.udpAppType="UDPVideoStreamCli"
**.host5.udpApp[*].serverAddress = "host3"
**.host5.udpApp[*].localPort = 9999
**.host5.udpApp[*].serverPort = 3088
**.host5.udpApp[*].startTime = uniform(5, 5.01)

**.host3.udpAppType = "UDPVideoStreamSvr"
**.host3.udpApp[*].videoSize = 5e7
**.host3.udpApp[*].serverPort = 3088
**.host3.udpApp[*].waitInterval = .001
**.host3.udpApp[*].packetLen = 1000

**.host6.udpAppType="UDPVideoStreamCli"
**.host6.udpApp[*].serverAddress = "host4"
**.host6.udpApp[*].localPort = 9999
**.host6.udpApp[*].serverPort = 3088
**.host6.udpApp[*].startTime = uniform(5, 5.01)

**.host4.udpAppType = "UDPVideoStreamSvr"
**.host4.udpApp[*].videoSize = 5e7
**.host4.udpApp[*].serverPort = 3088
**.host4.udpApp[*].waitInterval = .001
**.host4.udpApp[*].packetLen = 1000

# tcp apps
**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0
# ping app
**.pingApp.destAddr=""
```

```
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.host*.IPForward=false
**.host*.routingFile = ""
**.ip.procDelay=10us""

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# LSR configuration
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 100 -n
Green;c -t * -i 101 -n Magenta;c -t * -i 200 -n Orange;c -t * -i 300 -n Brown"

**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 10  # en routers

# scenario
**.scenarioManager.script = xmldoc("scenario.xml")

# RSVP, MPLS settings
# Nótese que el tráfico de interés se da entre los host cliente y servidor,
# entonces los routers encargados de la formación de VPN's son los ILER y ELER
**.LSR2.classifier.conf = xmldoc("LSR2_fec.xml")
**.LSR2.rsvp.traffic = xmldoc("LSR2_rsvp.xml")
**.LSR3.classifier.conf = xmldoc("LSR3_fec.xml")
**.LSR3.rsvp.traffic = xmldoc("LSR3_rsvp.xml")
**.LSR38.classifier.conf = xmldoc("LSR38_fec.xml")
**.LSR38.rsvp.traffic = xmldoc("LSR38_rsvp.xml")
**.LSR39.classifier.conf = xmldoc("LSR39_fec.xml")
**.LSR39.rsvp.traffic = xmldoc("LSR39_rsvp.xml")
**.LSR*.classifier.conf = xmldoc("_fec.xml")
**.LSR*.rsvp.traffic = xmldoc("_traffic.xml")
```

```
# Se da la opción de clasificación del tráfico para su respectiva
diferenciación.
**.LSR*.rsvp.helloInterval = 0.2
**.LSR*.rsvp.helloTimeout = 0.5
**.LSR*.libTable.conf = xmldoc("_lib.xml")
```

2.4. Topología MPLS RSVP BACKBONE Real

```
[General]
preload-ned-files = *.ned ../../NED/*.ned
network = BACKBONE
sim-time-limit = 15m
total-stack-kb =  6665536

[Cmdenv]
express-mode = no

[Tkenv]
plugin-path=../../Etc/plugins
default-run = 1

[Parameters]
# Configuración de los Host de videoconferencia
**.host1.udpApp_StreamCli.serverAddress = "host5"
**.host5.udpApp_StreamCli.serverAddress = "host1"
**.host9.udpApp_StreamCli.serverAddress = "host16"
**.host16.udpApp_StreamCli.serverAddress = "host9"
**.host2.udpApp_StreamCli.serverAddress = "host6"
**.host6.udpApp_StreamCli.serverAddress = "host2"
**.host15.udpApp_StreamCli.serverAddress = "host10"
**.host10.udpApp_StreamCli.serverAddress = "host15"
**.host3.udpApp_StreamCli.serverAddress = "host7"
**.host7.udpApp_StreamCli.serverAddress = "host3"
**.host11.udpApp_StreamCli.serverAddress = "host14"
**.host14.udpApp_StreamCli.serverAddress = "host11"
**.host8.udpApp_StreamCli.serverAddress = "host4"
**.host4.udpApp_StreamCli.serverAddress = "host8"
**.host12.udpApp_StreamCli.serverAddress = "host13"
**.host13.udpApp_StreamCli.serverAddress = "host12"

**.host*.udpApp_StreamCli.localPort = 9999
**.host*.udpApp_StreamCli.serverPort = 3088
**.host*.udpApp_StreamCli.startTime = 5

# Se envía video de 30fps, cada uno de 33.6Kb.
**.host*.udpApp_StreamSvr.videoSize = 50e7 #800Mb
**.host*.udpApp_StreamSvr.serverPort = 3088
**.host*.udpApp_StreamSvr.waitInterval = .01 # 30 fps
**.host*.udpApp_StreamSvr.packetLen =4.2e3 # 33.6Kb

#Se configuran las aplicaciones TCP
**.numTcpApps=1

# Se procede con la configuración FTP de clientes y Servers correspondientes
**.File_Cliente*.tcpAppType="TCPSessionApp"
**.File_Cliente*.tcpApp[0].active=true
**.File_Cliente*.tcpApp[0].address=""
**.File_Cliente*.tcpApp[0].port=-1            # puerto predeterminado
**.File_Cliente*.tcpApp[0].connectAddress="File_Server"
**.File_Cliente*.tcpApp[0].connectPort=1000
**.File_Cliente*.tcpApp[0].tOpen=6.0          # Tiempo de apertura de sesión
```

```
**.File_Cliente*.tcpApp[0].tSend=1.1
**.File_Cliente*.tcpApp[0].sendBytes=20e6     # Se envían 20MB
**.File_Cliente*.tcpApp[0].sendScript=""
**.File_Cliente*.tcpApp[0].tClose=0

**.File_Server.tcpAppType="TCPEchoApp"
**.File_Server.tcpApp[0].address=""
**.File_Server.tcpApp[0].port=1000
**.File_Server.tcpApp[0].echoFactor=2.0
**.File_Server.tcpApp[0].echoDelay=0

# Se procede con la configuración HTTP de clientes y Servers correspondientes
**.HTTP_Cliente*.tcpAppType="TCPBasicClientApp"
**.HTTP_Cliente*.tcpApp[0].address=""
**.HTTP_Cliente*.tcpApp[0].port=-1
**.HTTP_Cliente*.tcpApp[0].connectAddress="HTTP_Server"
**.HTTP_Cliente*.tcpApp[0].connectPort=80

# Se configuran parámetros del cliente tales como el número de solicitudes por
# sesión el tamaño de la respuesta a las solicitudes y el tiempo de respuesta.
**.HTTP_Cliente*.tcpApp[0].startTime=7
**.HTTP_Cliente*.tcpApp[0].numRequestsPerSession = exponential(10)
**.HTTP_Cliente*.tcpApp[0].requestLength = truncnormal(45000,20)
**.HTTP_Cliente*.tcpApp[0].replyLength = exponential(40000)
**.HTTP_Cliente*.tcpApp[0].thinkTime=truncnormal(0.1,0.3)
**.HTTP_Cliente*.tcpApp[0].idleInterval=truncnormal(2,5)
**.HTTP_Cliente*.tcpApp[0].reconnectInterval=2

**.HTTP_Server.tcpAppType="TCPGenericSrvApp"
**.HTTP_Server.tcpApp[0].address=""
**.HTTP_Server.tcpApp[0].port=80
**.HTTP_Server.tcpApp[0].replyDelay=0


**.HTTP**.numUdpApps=0
**.HTTP**.udpAppType="UDPBasicApp"
**.File**.numUdpApps=0
**.File**.udpAppType="UDPBasicApp"

**.host*.numTcpApps=0
**.host*.tcpAppType="TCPGenericSrvApp"
**.host*.tcpApp[0].address=""
**.host*.tcpApp[0].port=1000
**.host*.tcpApp[0].replyDelay=0

**.pingApp.destAddr=""
**.pingApp.srcAddr=""
**.pingApp.packetSize=56
**.pingApp.interval=1
**.pingApp.hopLimit=32
**.pingApp.count=0
**.pingApp.startTime=1
**.pingApp.stopTime=0
**.pingApp.printPing=true

# Configuración de parámetros genéricos de TCP
**.tcp.mss = 1024
**.tcp.advertisedWindow = 14336  # 14*mss
**.tcp.sendQueueClass="TCPMsgBasedSendQueue"
**.tcp.receiveQueueClass="TCPMsgBasedRcvQueue"
**.tcp.tcpAlgorithmClass="TCPReno"
**.tcp.recordStats=true

# Parámetros de configuración a nivel IP
**.IPForward=false
```

```
**.routingFile = ""
**.ip.procDelay=1us

# ARP configuration
**.arp.retryTimeout = 1
**.arp.retryCount = 3
**.arp.cacheTimeout = 100
**.networkLayer.proxyARP = true  # Host's is hardwired "false"

# Configuración de los routers MPLS RSVP
**.LSR*.peers=""
**.LSR*.routerId="auto"
**.LSR*.routingFile=""
**.LSR*.namid=-1
**.LSR*.libTable.conf = xmldoc("_lib.xml")

**.nam.logfile = "trace.nam"
**.nam.prolog = "c -t * -i 1 -n Red;c -t * -i 2 -n Blue;c -t * -i 100 -n
Green;c -t * -i 101 -n Magenta;c -t * -i 200 -n Orange;c -t * -i 300 -n Brown"
**.host*.namprolog = ""
**.host*.namlog = "trace.nam"
**.namid = -1  # auto

# Configuración de las colas
**.ppp[*].queueType = "DropTailQueue" # en routers
**.ppp[*].queue.frameCapacity = 30  # en routers

# parámetros propios del escenario
**.scenarioManager.script = xmldoc("scenario.xml")


# RSVP, MPLS settings
# Nótese que el tráfico de interés se da entre los host de videoconferencia,
# entonces los routers encargados de la formación de VPN's son los ILER y ELER

**.LSR0.classifier.conf = xmldoc("LSR0_fec.xml")
**.LSR0.rsvp.traffic = xmldoc("LSR0_rsvp.xml")
**.LSR1.classifier.conf = xmldoc("LSR1_fec.xml")
**.LSR1.rsvp.traffic = xmldoc("LSR1_rsvp.xml")
**.LSR2.classifier.conf = xmldoc("LSR2_fec.xml")
**.LSR2.rsvp.traffic = xmldoc("LSR2_rsvp.xml")
**.LSR3.classifier.conf = xmldoc("LSR3_fec.xml")
**.LSR3.rsvp.traffic = xmldoc("LSR3_rsvp.xml")
**.LSR36.classifier.conf = xmldoc("LSR36_fec.xml")
**.LSR36.rsvp.traffic = xmldoc("LSR36_rsvp.xml")
**.LSR37.classifier.conf = xmldoc("LSR37_fec.xml")
**.LSR37.rsvp.traffic = xmldoc("LSR37_rsvp.xml")
**.LSR38.classifier.conf = xmldoc("LSR38_fec.xml")
**.LSR38.rsvp.traffic = xmldoc("LSR38_rsvp.xml")
**.LSR39.classifier.conf = xmldoc("LSR39_fec.xml")
**.LSR39.rsvp.traffic = xmldoc("LSR39_rsvp.xml")
# Se da la opción de clasificación del tráfico para su respectiva
# diferenciación en el envío.

**.LSR*.classifier.conf = xmldoc("_fec.xml")
**.LSR*.rsvp.traffic = xmldoc("_traffic.xml")
**.LSR*.rsvp.helloInterval = 0.2              # Intevalo de Paquetes Hello
**.LSR*.rsvp.helloTimeout = 0.5               # Tiempo de espera de Hello
**.LSR*.libTable.conf = xmldoc("_lib.xml")
```