TESIS FINAL
INGENIERÍA EN SISTEMAS

REDES MPLS

(Parte teórica)

DATOS PERSONALES:

NOMBRE COMPLETO: Matias Eric Grassi

DNI: 31.192.578

DIRECTOR: Abel Crespo

1- Introducción

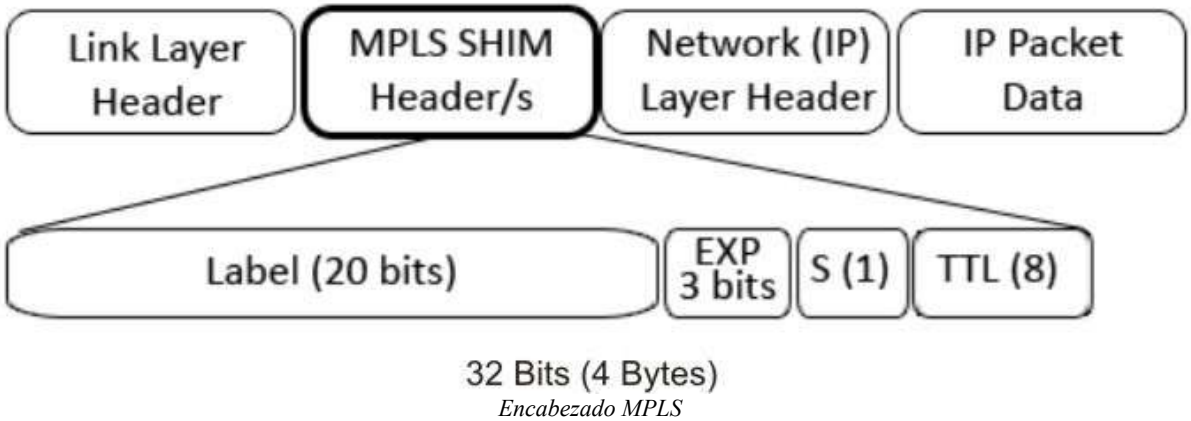
En el siguiente informe, se presentara de manera teórica el funcionamiento y las características de las redes MPLS (Multiprotocol Label Switching), este mecanismo de transporte de datos fue creado por la IETF (Internet Engineering Task Force), y definido en el RFC 3031. Mostrando desde su composición hasta las ventajas y desventajas que esta nueva tecnología brinda, siempre comparándola con tecnologías actualmente en uso, como Frame Relay, ATM, PPP y IEEE 802.2 (LAN), y como estas redes MPLS se adaptan para brindar soporte a estas tecnologías, y mejorarlas en aspectos de calidad y rendimiento, unificando las tecnologías actuales en una sola.
En una red, se puede pensar a los routers como “oficinas postales”, sin una forma de marcar, clasificar y monitorear el correo, no es posible procesar diferentes clases de correos, por lo tanto todos los mensajes que viajan por una red, independientemente de la tecnología que se utilice, siempre tendrá un mecanismo por el cual el mensaje indicara hacia donde ira y desde donde viene, como información principal. Luego dependiendo de la tecnología que se utilice, puede incluir información del tipo de la calidad de servicio para ese mensaje, la prioridad de ese mensaje sobre el resto en la red, el tipo de información que lleva ese mensaje, esta información siempre se encuentra en el encabezado del mensaje. Luego viene la información que se intenta enviar de un punto al otro, y finalmente una cola del mensaje indicando el final del mismo.
En una red MPLS, para poder designar diferentes clases de servicios y prioridades de servicios, el tráfico debe ser marcado con etiquetas especiales cuando entra en la red. Esta funcionalidad de etiquetado es llevada a cabo por un router llamado “Label Edge Router” (LER). El LER se encarga de convertir tanto los paquetes de Frame Relay, ATM, PPP y IEEE 802.2 (LAN) en paquetes MPLS y viceversa, donde dependiendo de la tecnología que se utilice como base se agregará en la cabecera información de la red MPLS para que luego este mensaje viaje en esta red, por medio de los “Label Switch Router” (LSR), que son los encargados de transportar cada mensaje hasta su destino dentro de la red MPLS.
Para entender mejor el encabezado MPLS, es conveniente mirar el modelo OSI (Open Systems Interconnection). Cuando un paquete es presentado al LER, éste se encarga de ubicar el encabezado MPLS. Éste no es un protocolo de Capa 3, ya que este es independiente del protocolo utilizado en esta capa, como por ejemplo el protocolo IP. MPLS tampoco es un protocolo de Capa 2, para brindar independencia del protocolo utilizado antes de ingresar a la red MPLS, como por ejemplo Frame Relay, ATM, PPP Y IEE 802.2 (LAN). Por lo tanto, en el modelo OSI, MPLS no se ubica ni en una capa, ni en otra. Quizás lo más fácil de hacer, sea ver a MPLS como una Capa 2,5. Es conveniente notar que si bien el encabezado MPLS no forma parte ni de la capa 2 ni de la capa 3, provee un medio por el cual relaciona la información de estas dos capas.

Modelo OSI

MPLS en el Modelo OSI

2- Composición de la etiqueta MPLS

En un encabezado normal de una red MPLS, la etiqueta se encuentra, entre la cabecera de la capa de datos, y la cabecera de la capa de red, de la siguiente forma:



Encabezado MPLS

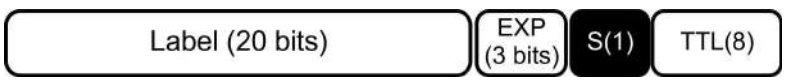
El encabezado MPLS consiste de 32 bits (4 Bytes) con una estructura determinada desde un principio.
Los primeros 20 bits son los valores de la etiqueta. Este valor puede ser entre 0 y 2^{20}-1 (o 1048575). Sin embargo, los primeros 16 valores están exentos para uso normal. Este valor solo tiene un sentido local entre los routers punto a punto.

Campo Label

Los bits 20 al 22 son los tres bits para funciones experimentales (EXP). Estos bits son usados solamente para calidad de servicio (QoS).

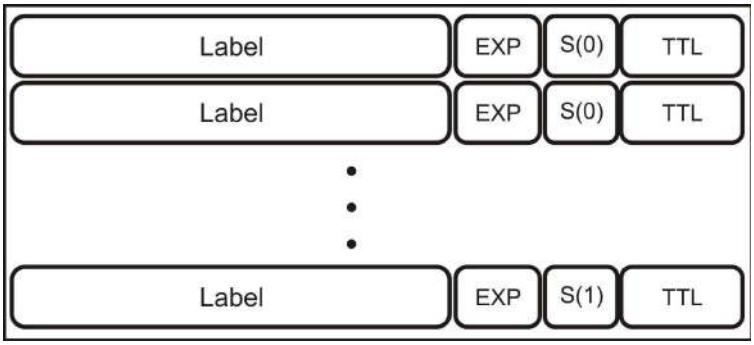
Campo EXP

Los router que soportan MPLS pueden necesitar más de una etiqueta en el tope del paquete para encaminarlo a través de la red MPLS. Esto se logra empaquetando las etiquetas en una pila. La primera etiqueta de la pila le la llama "Etiqueta Tope" (top label), y la última es llamada "Etiqueta Base" (Bottom label). El bit 23 es el "Bottom of Stack" (S – Base de la pila) y sirve para la función de apilado de etiquetas. Este es siempre cero (0), excepto si ésta es la base de la pila de etiquetas.



Campo Pila (S)

La pila es una colección de etiquetas que se encuentran en lo alto del paquete. La pila puede consistir de solo una etiqueta o puede contener más. El número de etiquetas (el campo de 32 bits) que se pueden encontrar en la pila es ilimitado, aunque raramente se debería encontrar una pila que consiste de cuatro o más etiquetas, como se puede ver en la imagen *"Apilamiento de etiquetas"*.



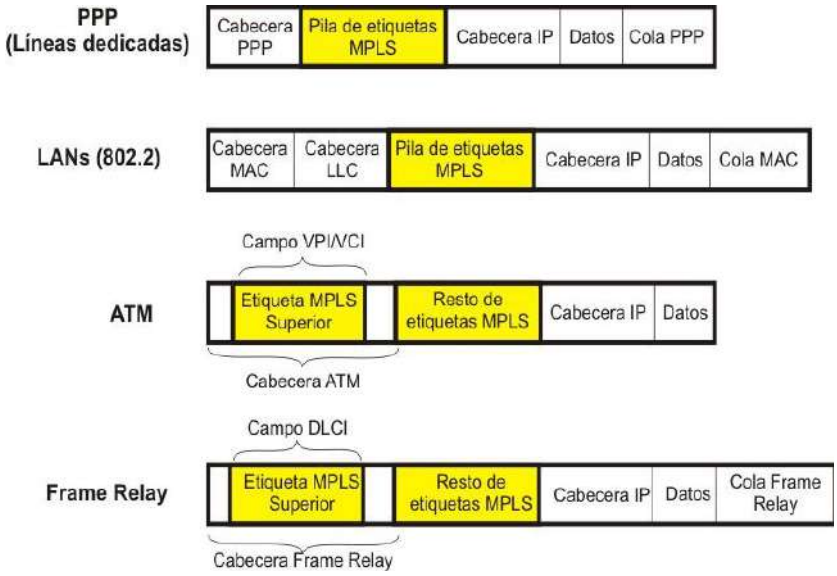
Apilamiento de etiquetas

Los bits 24 al 31 son los ocho bits usados para "Tiempo de Vida" ("Time To Live", TTL). Este TTL tiene la misma función que el TTL encontrado en el encabezado IP. Éste simplemente disminuye en 1 en cada salto su valor, y su principal función es evitar que un paquete quede atrapado en un bucle de enrutamiento. Si ocurre un bucle de enrutamiento y ningún TTL está presente, el paquete permanecería por siempre saltando entre un router y otro. Si el TTL de un paquete llega a cero, el paquete es descartado.

Campo TTL

2.1- Ubicación de etiquetas

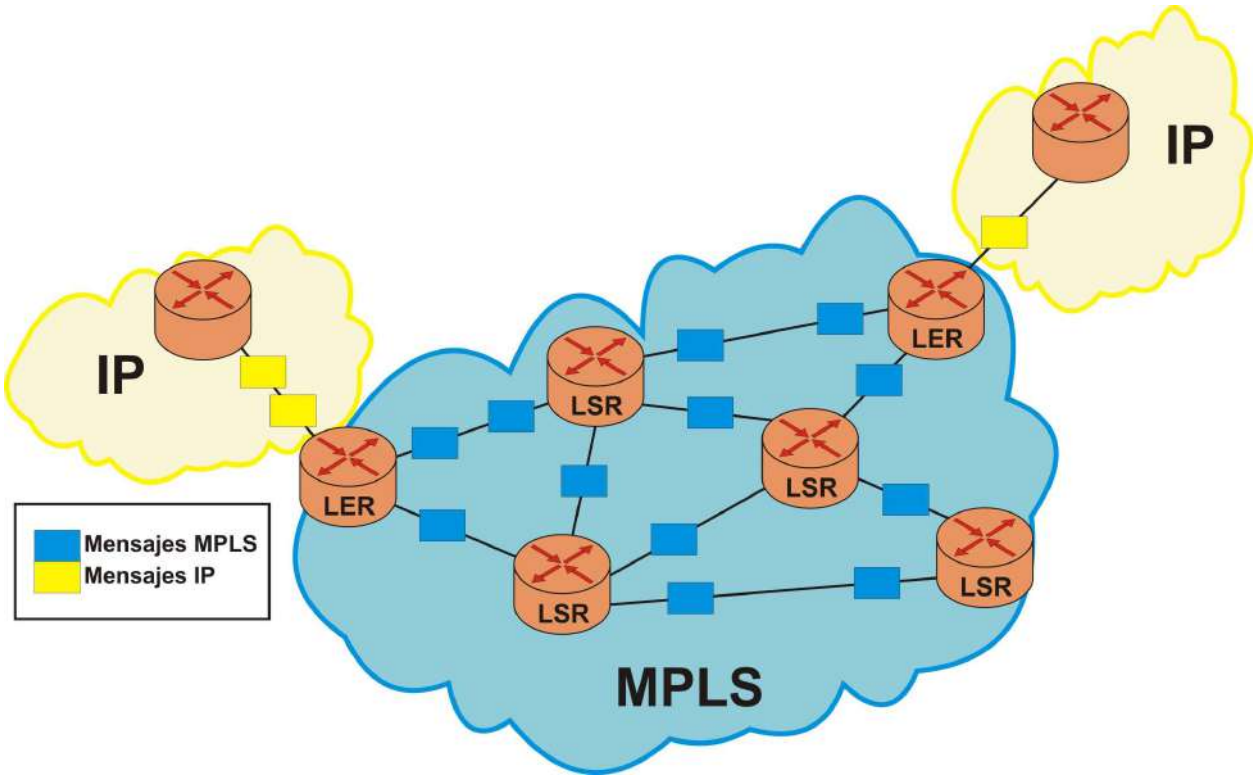
MPLS funciona sobre una gran variedad de tecnologías de nivel de enlaces. En ATM y en Frame Relay la etiqueta MPLS, ocupa el lugar del campo VPI/VCI o el DLCI, ambos utilizados en sus respectivas tecnologías (VPI/VCI en ATM y DLCI en Frame Relay) para identificar el próximo destino de una celda a medida que pasa sobre una serie de switches hasta su destino final, para aprovechar el mecanismo de conmutación inherente. Esta etiqueta identifica a la FEC asociada a ese paquete.
En la siguiente figura se puede ver la ubicación de la etiqueta MPLS según sea la tecnología usada.



Ubicación de la etiqueta MPLS

3- Elementos de una red MPLS

Para poder encaminar el trafico a través de la red, una vez que las etiquetas han sido aplicadas, los router intermedios sirven como "Label Switch Routers" (LSRs). Es conveniente notar que estos son todavía routers, un análisis de paquetes permite determinar cuando éstos sirven como switches MPLS o routers.
La función del LSR es examinar los paquetes entrantes. Siempre que una etiqueta está presente, el LSR buscará y llevará a cabo las instrucciones provistas en la etiqueta y luego redireccionará el paquete de acuerdo a estas instrucciones. En general, el LSR lleva a cabo la función de conmutación de etiquetas (label-swapping function).



Red MPLS

Los caminos son establecidos entre los LER y los LSR. Estos caminos son llamados "Label Switch Paths" (LSPs).

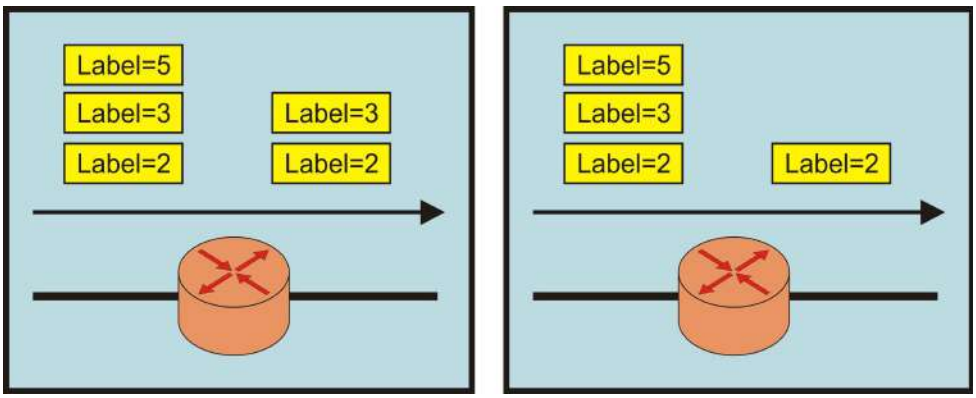
3.1- Label Switch Router

Un LSR es un router que soporta MPLS. Éste es capás de entender etiquetas MPLS y de recibir y transmitir un paquete etiquetado. Existen tres tipos de LSR en una red MPLS:

- **Ingress LSRs:** Reciben un paquete que no está etiquetado todavía, inserta una etiqueta (pila) en el paquete y lo envía. También se lo conoce como LER ("Label Edge Router").
- **Egress LSRs:** Recibe paquetes etiquetados, remueve las etiquetas y los envía. También se lo conoce como LER ("Label Edge Router").
- **Intermediate LSRs:** recibe un paquete etiquetado entrante, hace switch sobre el valor de la etiqueta del paquete y lo envía por el enlace correcto.

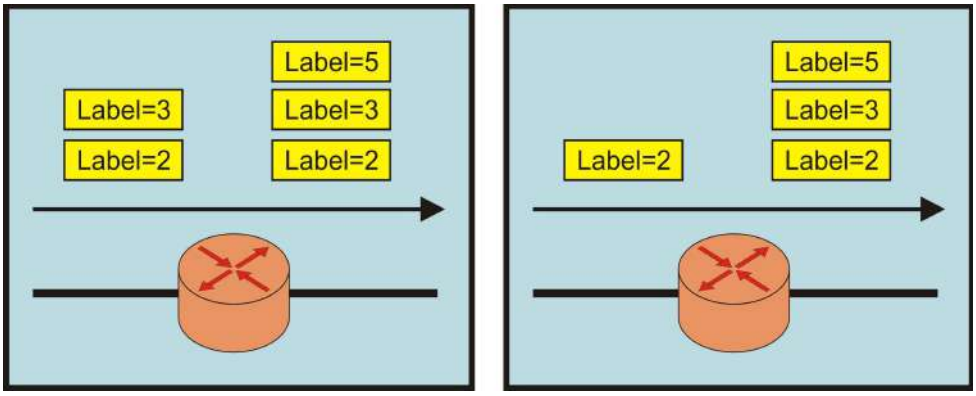
Un LSR puede hacer tres operaciones (pop, push o swap), las cuales se aplican en cada salto, y

dependiendo de si es un LSR o un LER de entrada o salida, se va a realizar una de las tres operaciones mencionadas anteriormente.
Tiene que poder hacer pop en una o más etiquetas (remover una o más del tope de la pila de etiquetas) antes de encaminar el paquete.



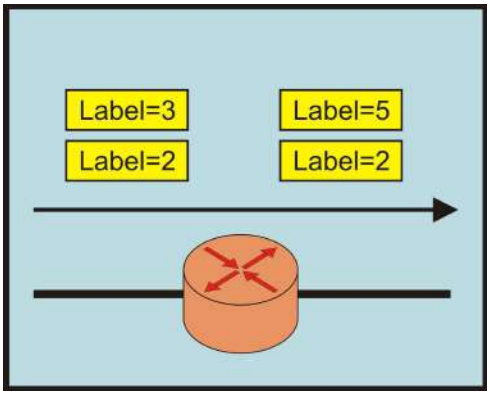
Operación POP

Un LSR tiene que poder además hacer push en una o más etiquetas (agregar una o más al tope de la pila de etiquetas) del paquete recibido. Si el paquete recibido no está todavía etiquetado, el LSR crea una pila de etiquetas nueva y la agrega al paquete.



Operación PUSH

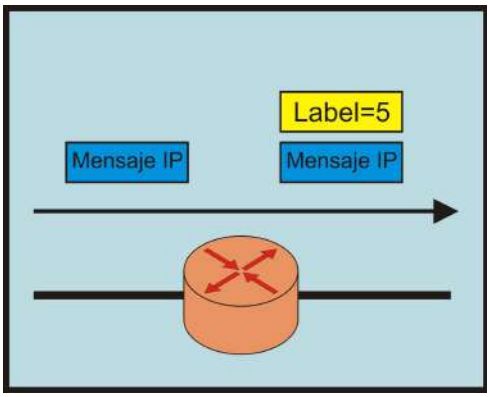
Un LSR tiene que poder hacer swap sobre una etiqueta. Esto simplemente es que cuando un paquete etiquetado es recibido, la etiqueta tope de la pila de etiquetas es intercambiada por una nueva etiqueta y el paquete es encaminado.



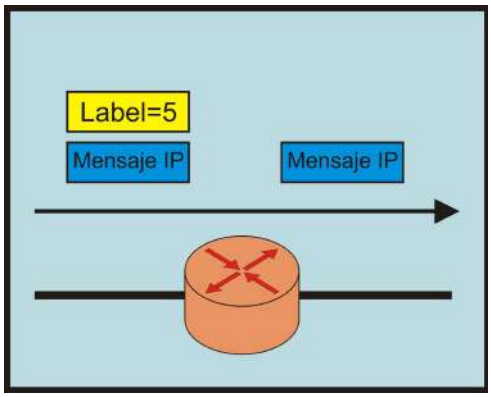
Operación SWAP

Un LSR que hace push en un paquete que no estaba etiquetado se lo llama “imposing LSR” porque

es el primero en imponer etiquetas en el paquete (ej. ingress LSR). Por el otro lado, un LSR que remueve todas las etiquetas de un paquete etiquetado antes de encaminarlo es un “disposing LSR” (ej. egress LSR).



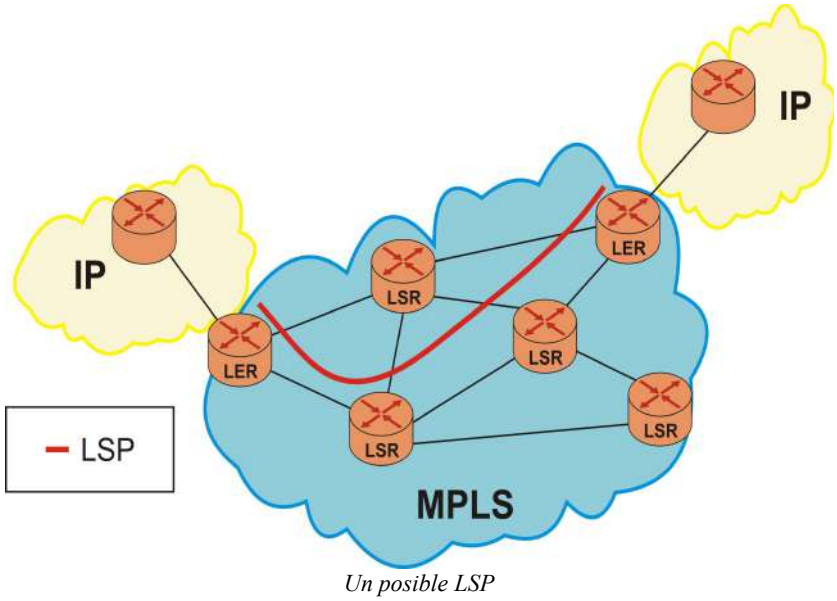
Imposing LSR



Disposing LSR

3.2- Label Switched Path

Un “label switched path” (LSP) es una secuencia de LSRs que encaminan un paquete etiquetado a través de una red MPLS o una parte de ésta. Básicamente, el LSP es el camino de la red MPLS.
El primer LSR de un LSP es el “ingress LSR”, mientras que el último LSR del LSP es el “egress LSR”. Todos los LSRs entre el ingress LSR y el egress LSRs con los “intermediate LSRs”.
El “ingress LSR” de un LSP no necesariamente es el primer router en etiquetar un paquete. El paquete podría haber sido etiquetado por algún LSR anterior. Tal caso sería un “nested LSP” (LSP anidado), esto es, un LSP dentro de otro LSP.



Un posible LSP

3.3- Forwarding Equivalence Class

Para cualquier protocolo de enrutamiento que tenga alguna oportunidad de sobrevivir, la escalabilidad es un problema que se debe de resolver desde el principio. Con el fin de garantizar esto, se debe administrar en la agregación y no sobre un flujo individual. MPLS garantiza el apoyo

a la agregación con lo que llama Forwarding Equivalence Class (FEC).
Una FEC es un grupo o una corriente de paquetes que son encaminados a lo largo del mismo camino y son tratados de la misma forma.
Todos los paquetes pertenecientes a la misma FEC tienen la misma etiqueta. Sin embargo, no todos los paquetes que tienen igual etiqueta pertenecen al mismo FEC, porque su valor de EXP podría diferir, el tratamiento de encaminado podría ser diferente, y podrían pertenecer a diferentes FEC.
El router decide que paquetes pertenecen a que FEC, esto depende de su configuración, pero normalmente esto es al menos la dirección IP de destino, otro criterio es tomar la calidad de servicio.

3.4- Label Information Base (LIB)

La LIB, es una tabla utilizada por un router MPLS que se encarga de gestionar por medio de la dirección IP, los detalles del puerto y de la correspondiente etiqueta MPLS de entrada, la cual luego se realizará el cambio por el valor de la etiqueta saliente.

3.5- Label Forwarding Information Base (LFIB)

La LFIB, es un sub-conjunto de la FIB (Forwarding Information Base), esta se genera, cuando se elige un camino más corto a un destino real. Por lo tanto, en el LFIB habrá un par de etiqueta para un destino (o FEC), mientras que la LIB tiene todos los caminos posibles para el destino.

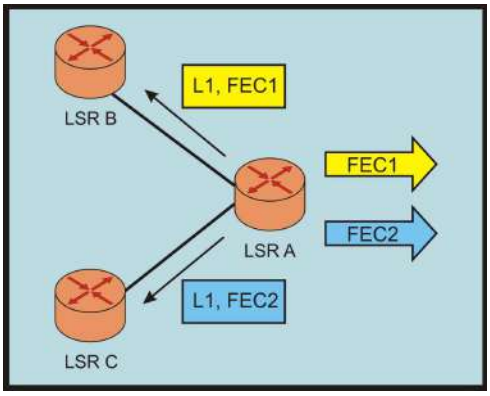
4- Espacios de etiquetas MPLS

El espacio de etiquetas determina el ámbito en el que se pueden clasificar los paquetes dentro de una red MPLS, estos pueden ser:

- Por interfaz.
- Por plataforma.

4.1- Espacio de etiquetas por interfaz

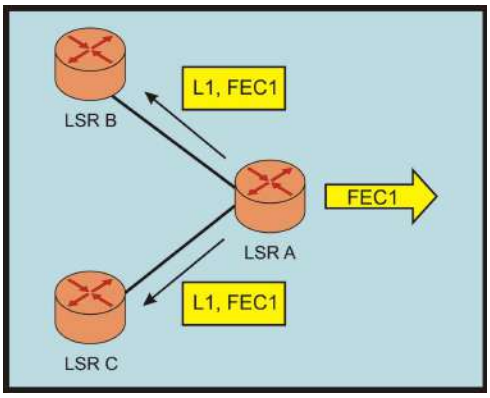
Con un espacio de etiquetas por interfaz el enrutamiento de los paquetes será en base a la etiqueta y a la interfaz por la que se recibe dicho paquete, esto se debe a que una misma etiqueta tiene distinto significado en cada interfaz del router LSR.



Espacio de etiquetas por interfaz

4.2- Espacio de etiquetas por plataforma

Con un espacio de etiquetas por plataforma el enrutamiento de los paquetes será únicamente en base a la etiqueta, esto se debe a que la etiqueta tiene el mismo significado para todas las interfaces del router LSR.



Espacio de etiquetas por plataforma

5- Modos en MPLS

Un LSR puede usar distintos modos cuando distribuye etiquetas a otros LSRs. Estos se pueden clasificar en:

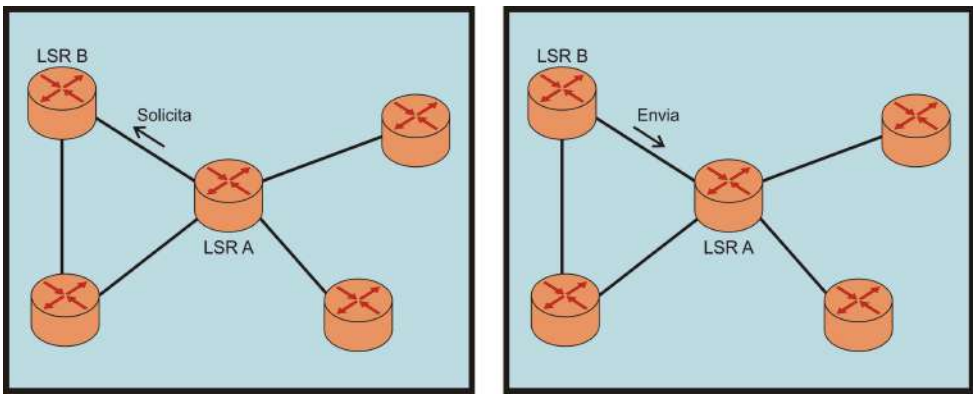
- Modo de distribución de etiquetas.
- Modo de retención de etiquetas.
- Modo de control de LSP.

5.1- Modo de distribución de etiquetas

La arquitectura MPLS tiene dos modos para distribuir etiquetas enlazadas:

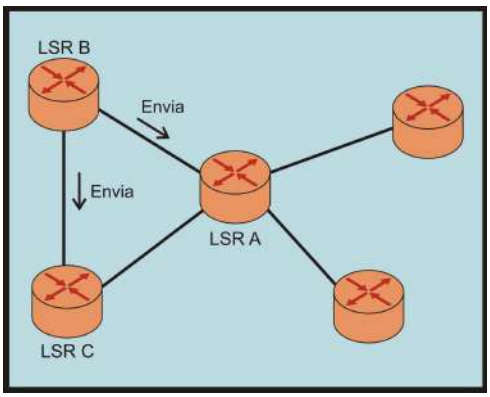
- Bajo Demanda (Downstream-on-Demand o DoD).
- Sin Solicitud (Unsolicited Downstream o UD).

En el modo **Bajo Demanda**, como se puede ver en la imagen *"Solicitud bajo demanda"*, cada LSR solicita a su próximo salto en un LSP, una etiqueta enlazada para una FEC determinada. Cada LSR recibe una etiqueta enlazada por FEC solamente de su próximo LSR en esa FEC. EL próximo LSR es el router del próximo salto indicado por la tabla de ruteo IP.



Solicitud bajo demanda

En el modo **Sin Solicitud**, como se puede ver en la imagen *"Sin solicitud"*, cada LSR distribuye etiquetas enlazadas a sus LSRs adyacentes, sin que esos LSRs hayan solicitado una etiqueta. En este modo, un LSR recibe una etiqueta enlazada remota desde cada LSR adyacente.



Sin solicitud

En el caso de **Bajo Demanda**, la LIB muestra solo una etiqueta enlazada remota, mientras que en **Sin Solicitud**, la LIB mostrará más de una etiqueta remota. El modo de distribución de etiquetas utilizado depende de la interfaz y de la implementación.

5.2- Modo de retención de etiquetas

Hay dos posibles modos de retención de etiquetas:

- Modo liberal de retención de etiquetas (LLR).
- Modo conservador de retención de etiquetas (CLR).

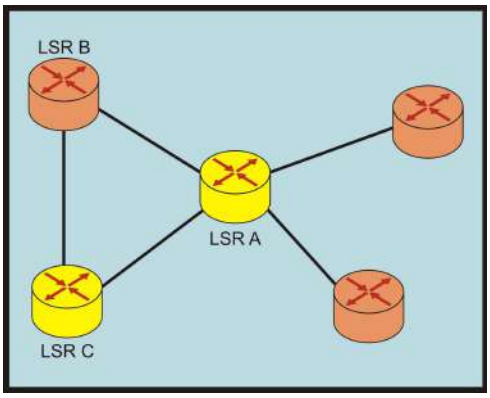
En el modo LLR, un LSR mantiene todas las etiquetas remotas enlazadas recibidas en la LIB. Una de esas etiquetas es la recibida desde el próximo salto para una FEC. La etiqueta desde ese enlace remoto es usada en la LFIB, pero ninguna de las otras etiquetas desde los otros enlaces remotos es puesta en la LFIB, por lo tanto no todas son usadas para enviar paquetes. Estas son mantenidas en la LIB ya que el enrutamiento en la red es dinámico. En cualquier momento la topología de

enrutamiento puede cambiar, por ejemplo, cuando cae un enlace, por esto el router de próximo salto para una FEC en particular cambia. En ese momento, la etiqueta para un nuevo router de próximo salto ya se encuentra en la LIB y la LFIB puede ser actualizada rápidamente.

En el modo CLR, un LSR no almacena todos los enlaces remotos en la LIB, solo almacena el enlace remoto que es asociado con el LSR de próximo salto para una FEC en particular.

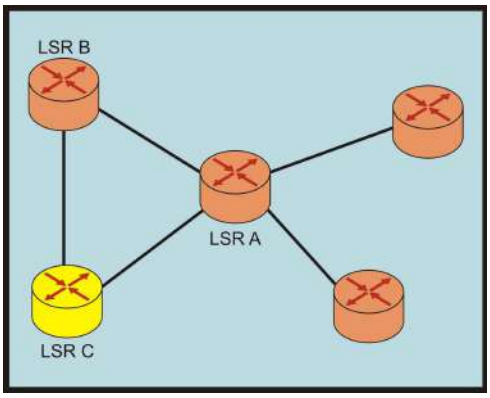
El modo LLR provee una rápida adaptación a los cambios de ruteo, mientras que el modo CLR provee un mejor uso de la memoria disponible en los routers.

En el siguiente ejemplo de LLR, el LSR B guarda la asociación de etiquetas a una FEC de los LSR A y LSR C, los cuales pueden o no ser su siguiente salto, pero manejan esa misma FEC.



Modo liberal de retención de etiquetas

Ahora en un ejemplo de CLR, el LSR B solo guarda la asociación de etiquetas a una FEC de los LSR que son su siguiente salto (LSR C).



Modo conservador de retención de etiquetas

5.3- Modo de control de LSP

Los LSRs pueden crear enlaces locales para una FEC de dos maneras:

- Modo de control de LSP independiente.
- Modo de control de LSP ordenado.

El LSR puede crear enlaces locales para una FEC independientemente de otros LSRs. Esto es llamado modo de control de LSP independiente. En este modo, cada LSR crea un enlace local para una FEC en particular tan pronto reconoce a la FEC. Normalmente, esto significa que el prefijo para una FEC esta en la tabla de ruteo.

En el modo de control de LSP ordenado, un LSR solo puede crear un enlace local para una FEC si se reconoce como egress LSR para esa FEC o si el LSR ha recibido una etiqueta enlazada desde el próximo salto para esa FEC.

El control ordenado ofrece como ventajas una mejor ingeniería de trafico y mayor control en la red, aunque en comparación con el control independiente, presenta tiempos de convergencia mayor.

La desventaja de un control LSP independiente es que algunos LSR comienzan a intercambiar etiquetas de paquetes antes de que el LSP este completo de extremo a extremo. Por lo tanto, el paquete no es enviado en la manera que debía serlo. Si el LSP no es completamente establecido, los paquetes no podrán recibir el correcto tratamiento de envío en todos lados o podrían ser quitados de la red.

6- Distribución de etiquetas

La primera etiqueta es impuesta en el ingress LSR y la etiqueta pertenece a un LSP. El camino del paquete a través de la red MPLS esta limitada a ese LSP en particular. Lo único que cambia es la etiqueta tope en la pila de etiquetas, en cada salto (de un LSR a otro). El ingress LSR impone una o más etiquetas en el paquete apiladas. Los intermediate LSRs intercambian la etiqueta superior del paquete etiquetado recibido por otra etiqueta y transmiten el paquete por el enlace. Una vez recorrido todo el LSP, el egress LSR retira todas las etiquetas y reenvía el paquete.
Para que esto trabaje correctamente, los LSRs adyacentes deben ser capaz de entender como intercambiar la etiqueta de entrada con la de salida. Esto significa que se necesita un mecanismo para indicarle al router que etiquetas usar cuando reenvía un paquete. Las etiquetas son locales para cada par de routers adyacentes. Las etiquetas no tienen un significado global a través de la red. Para que routers adyacentes acuerden que etiqueta usar, estos necesitan alguna forma de comunicación. Lo que se necesita es un protocolo de distribución de etiquetas.
Se pueden distribuir etiquetas de dos maneras:

- Transportar las etiquetas sobre un protocolo IP existente.
- Tener un protocolo separado para distribuir etiquetas.

6.1- Utilizar un protocolo de ruteo IP existente

Este primer método tiene la ventaja de que no es necesario un nuevo protocolo corriendo en los LSRs, pero cada protocolo IP existente necesita ser extendido para acarrear las etiquetas, lo que no siempre es una cosa sencilla de hacer.
La gran ventaja de tener un protocolo de ruteo para acarrear las etiquetas es que el router y la distribución de etiquetas siempre están sincronizadas, lo que significa que no puede haber una etiqueta si el prefijo falta o viceversa. También elimina la necesidad de otro protocolo corriendo en el LSR para hacer la distribución de etiquetas.

6.2- Crear un protocolo para distribución de etiquetas

Este segundo método tiene la ventaja de ser independiente del protocolo de ruteo. Cualquiera sea el protocolo de ruteo IP, sea capaz de distribuir etiquetas o no, un protocolo separado distribuye las etiquetas y deja al protocolo de ruteo distribuir los prefijos. La desventaja de este método es que un nuevo protocolo es necesario en los LSRs.

6.3- Seleccionando el protocolo

La elección de todos los vendedores de routers fue la de tener a un nuevo protocolo de distribución de etiquetas para su distribución. Éste es el Label Distribution Protocol (LDP), aunque no es el único protocolo que se puede encontrar en el mercado.
Existen por el momento tres protocolos para distribuir etiquetas:

- Tag Distribution Protocol (TDP).
- Label Distribution Protocol (LDP).
- Resource Reservation Protocol (RSVP).

6.3.1- Tag Distribution Protocol (TDP)

Este fue el primer protocolo de distribución de etiquetas implementado por Cisco, formalizado más tarde por la IETF en LDP. En cuanto a funcionamiento, tanto TDP como LDP son similares, aunque este último presenta más funcionalidades, razón por la cual TDP resulta actualmente obsoleto.

6.3.2- Label Distribution Protocol (LDP)

Para que los paquetes viajen a través de un LSP en una red MPLS, todos los LSRs deben ejecutar un protocolo para la distribución de etiquetas e intercambiar las etiquetas enlazadas. Cuando todos los LSRs tienen las etiquetas para un FEC en particular, todos los paquetes pueden ser enviados por un LSP por medio de un intercambio de etiquetas en el paquete en cada LSR. Cada LSR conoce las operaciones de etiquetas (swap, push, pop) que debe realizar mediante una búsqueda en la LFIB. La LFIB es llenada con las etiquetas enlazadas encontradas en la LIB. La LIB es llenada con las etiquetas enlazadas recibidas por LDP. Los LSRs directamente conectados deben establecer una relación LDP para cada par o una sesión entre ellos. Los pares LDP intercambian mensajes de mapeo de etiquetas a través de una sesión LDP. Una etiqueta mapeada o enlazada es una etiqueta que es asociada a una FEC.

LDP tiene cuatro funciones principales:

- Detección de los LSR que tengan funcionando LDP.
- Establecer y mantener sesiones.
- Comunicación del mapeo de etiqueta.
- Gestión interna por medio de notificaciones.

Cuando dos LSRs hacen funcionar LDP y ellos comparten uno o más enlaces entre ellos, deberían detectar a otro LSR mediante un mensaje “Hello”. El segundo paso para ellos es establecer una sesión a través de una conexión TCP. A través de esta sesión, LDP comunica mensajes de mapeo de etiquetas entre los pares. Estos mensajes de mapeo de etiquetas sirven para avisar, cambiar o eliminar etiquetas enlazadas. LDP provee los medios para notificar a los LDP vecinos de alguna

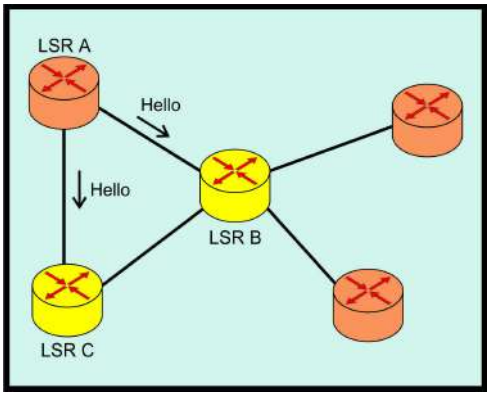
advertencia o mensajes de error, enviando mensajes de notificación.

6.3.2.1- Detección de los LSR que tengan funcionando LDP

Los LSRs que tienen funcionando LDP envían un mensaje "Hello" por todas las interfaces donde LDP esta habilitado. Este mensaje se envía por UDP (a través del puerto 646) a los routers vecinos. El protocolo de descubrimiento de LDP utiliza UDP como protocolo de transporte. Existen dos modalidades de descubrimiento:

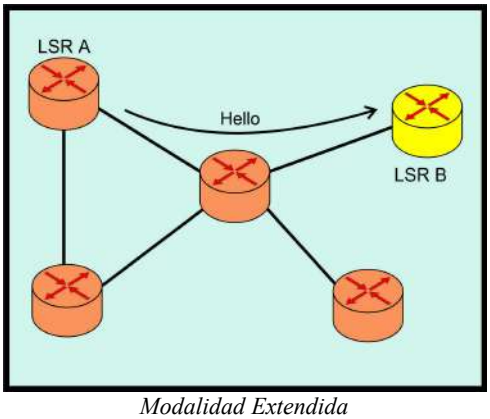
- Básica.
- Extendida.

En la modalidad **Básica** el LSR envía periódicamente mensajes "Hello" a un puerto bien conocido con la dirección multicast. Los routers están escuchando continuamente en este puerto a la espera de recibir mensajes "Hello". Por tanto, llegará un momento en el que el LSR conocerá todos los LSRs con los que tiene una conexión directa. Por tanto este mecanismo se utiliza si los LSRs están conectados directamente por medio de un enlace.



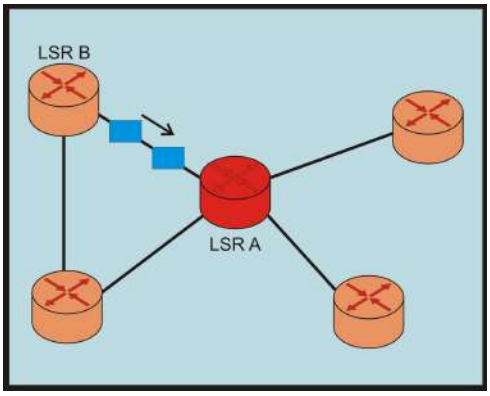
Modalidad Básica

Con la modalidad **Extendida** se permite que dos LSRs que no están conectados directamente establezcan una sesión LDP. Con esta modalidad, un LSR emite periódicamente mensajes "Hello" a un puerto (UDP) bien conocido y con una dirección específica, que habrá aprendido de algún modo (por ejemplo, por configuración). Los mensajes "Hello" transportarán el identificador LDP con el espacio de etiquetas que el LSR pretende usar, además de otro tipo de información. El LSR al que se le están enviando los mensajes "Hello" podrá responder o ignorar dicho mensaje. Si decide responder dicho mensaje deberá mandar periódicamente mensajes "Hello" al LSR que inició el proceso.



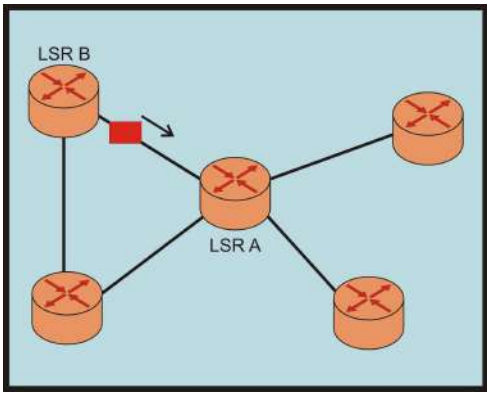
Modalidad Extendida

La modalidad extendida es útil cuando se ha configurado un LSP por ingeniería de tráfico, deseando mandar paquetes ya etiquetados a través de ese LSP. El LSR situado al principio del LSP necesitará saber como etiquetar los paquetes que le enviará el LSR situado al final del LSP.
El mensaje "Hello" tiene un **Tiempo de Respuesta** (Hold Time), si no se recibe un Hello desde un LSR antes de que este expire, el LSR remueve dicho LSR de la lista de vecinos LDP detectados. Si se envía y se recibe un mensaje Hello en una interfaz, entonces se genera una conexión a través de ese enlace entre los dos LSR que tienen funcionando LDP. Además, cada mensaje Hello tiene un intervalo de tiempo para ser reenviado denominado **Tiempo de Intervalo** (Interval Time). El **Tiempo de Respuesta** y el **Tiempo de Intervalo** están configurados por defecto en 15 y 5 segundos respectivamente. Si un par de LSR tiene distintos Hold Time configurados, se utiliza el más pequeño de ellos. Es necesario tener un especial cuidado si se desea cambiar estos tiempos de espera, ya que un **Tiempo de Respuesta** muy pequeño o muy grande puede ser problemático. Si este es muy chico, la sesión puede ser perdida inmediatamente aún cuando solo unos pocos paquetes son perdidos. Si el **Tiempo de Respuesta** es muy grande, la sesión podría ser demasiado prolongada en caso de un problema serio, y la reacción a tomar podría ser muy lenta. Y como resultado, muchos paquetes etiquetados pueden ser perdidos.
En el siguiente ejemplo, podemos ver como con un **Tiempo de Respuesta** grande, se puede tener una mayor perdida de paquetes, debido, a que el LSR B, sigue enviando paquetes, ya que desconoce que el LSR A, se encuentra caído.



Hold Time Grande

En este otro ejemplo, podemos ver como el **Tiempo de Respuesta**, es muy corto por lo que el mensaje no llega a ser recibido por el LSR A, provocando que el LSR B desconozca si ese salto es un router con LDP o no, cancelando el envío de los demás paquetes.



Hold Time Chico

Con estos dos ejemplos anteriores se puede ver claramente, lo dicho anteriormente con respecto a las consecuencias de configurar un **Tiempo de Respuesta** grande, ya que el LSR B, no tiene

conocimientos de que el LSR A esta caído, hasta que expira el tiempo de espera y deja de enviar mensajes, pero en este caso se pueden haber enviados varios mensajes, por lo tanto estos mensajes enviados se pierden. En el segundo caso no es tan critico, ya que en el peor de los casos se puede llegar a terminar el tiempo cuando el mensaje esta siendo reenviado por el LSR A.
Los LSR que ejecutan LDP poseen un identificador LDP (o LDP ID). Este LDP ID es un campo de 6 bytes que consiste de 4 bytes para identificar al LSR de forma única y 2 bytes para identificar el espacio de etiquetas que el LSR esta utilizando. Si los últimos 2 bytes son cero, se utiliza un espacio de etiquetas por plataforma. Si estos no son cero, se utiliza un espacio de etiquetas por interfaz. Los primeros 4 bytes del LDP ID representan una dirección IP tomados desde una interfaz operacional del router.
El LDP ID necesita estar presente en la tabla de ruteo de los routers LDP vecinos. Si esto no sucede, la sección LDP no es establecida. Por tanto, la dirección IP del router debe ser incluida en el protocolo de ruteo del LSR.

6.3.2.2- Establecer y mantener las sesiones

Si dos LSRs se han descubierto por medio de mensajes "Hello" LDP, intentaran establecer una sesión LDP entre ellos. Uno de los LSR tratará de abrir una comunicación TCP (por el puerto 646) hacia el otro LSR. Si la conexión es creada, ambos negociarán los parámetros de la sesión LDP mediante mensajes de iniciación. Esos parámetros incluyen entre otras cosas:

- Intervalos de tiempo.
- Método de distribución de etiquetas.
- Rangos de identificación de caminos virtuales (VPI) y rangos de identificador de canales virtuales (VCI), para etiquetas ATM.
- Rangos de identificador de conexión enlace de datos (DLCI) para Frame Relay.

Si el par LDP acuerdan los parámetros de sesión, mantienen la conexión TCP entre ellos. Si no logran ponerse de acuerdo, intentan crear una nueva sesión LDP entre ellos pero con parámetros regulados.
Después de establecer la sesión LDP, ésta es mantenida mediante la recepción de cualquier paquete LDP o por un mensaje periódico keepalive. Cada vez que se recibe un mensaje LDP o un mensaje de keepalive, se reinicia el valor del reloj para esa conexión. Si se agota el tiempo del reloj keepalive y no se recibieron mensajes la sesión es cerrada.
A continuación se muestra la Tabla de transición, y el diagrama de transición de estado de inicialización de la sesión.

ESTADO	EVENTO	NUEVO ESTADO
NO EXISTENTE	Conexión TCP de sesión establecida.	INICIALIZADO
INICIALIZADO	Transmitir mensaje de inicio (papel activo).	OPENSENT
INICIALIZADO	Recibir mensaje de inicio (papel pasivo). **Acción: transmitir mensajes de inicio y establecer cesión.**	OPENREC
INICIALIZADO	Recibir cualquier otro mensaje LDP. **Acción: transmitir mensaje de notificación de error (NAK) y cerrar la conexión de transporte.**	NO EXISTENTE
OPENREC	Recibir mensaje de establecer cesión.	OPERACIONAL
OPENREC	Recibir cualquier otro mensaje LDP. **Acción: transmitir mensaje de notificación de error (NAK) y cerrar la conexión de transporte.**	NO EXISTENTE

OPENSENT	Recibir un mensaje de iniciación aceptable. **Acción: transmitir un mensaje de mantenimiento.**	OPENREC
OPENSENT	Recibir cualquier otro mensaje LDP. **Acción: transmitir mensaje de notificación de error (NAK) y cerrar la conexión de transporte.**	NO EXISTENTE
OPERACIONAL	Recibir mensaje de finalización. **Acción: transmitir mensaje de finalización y cerrar la conexión de transporte.**	NO EXISTENTE
OPERACIONAL	Recibir cualquier otro mensaje LDP	OPERACIONAL
OPERACIONAL	Intervalo de tiempo sobrepasado. **Acción: transmitir mensaje de finalización y cerrar la conexión de transporte.**	NO EXISTENTE

La siguiente imagen muestra el diagrama de transición de estado al iniciar y mantener una sesión en el protocolo LDP:

Diagrama de estados de LDP

6.3.2.3- Número de sesiones

Cuando se trabaja en un espacio de etiquetas por plataforma entre un par de LSRs, una sola sesión LDP es suficiente. Esto se debe a que las etiquetas tienen el mismo significado para todas las interfaces. Cuando se trabaja en un espacio de etiquetas por interfaz, se necesita una sesión LDP por cada par que utilice este tipo de espacio de etiquetas. Esto se debe a que cada etiqueta tiene un

significado distinto para cada interfaz.
Por lo tanto para un espacio de etiquetas por interfaz, es necesario establecer una conexión para cada cesión.

6.3.2.4- Comunicación del mapeo de etiqueta

La comunicación del mapeo de etiqueta o etiquetas enlazadas es el propósito principal de LDP. Si bien hay distintas formas en la que un protocolo de distribución de etiquetas puede operar, no importa en que manera los pares LDP operan, el propósito es comunicar etiquetas enlazadas. En el modo **Sin Solicitud**, los pares LDP distribuyen etiquetas enlazadas no solicitadas para estos pares. Las etiquetas enlazadas son un conjunto de LDP ID y etiquetas por prefijo. Un router LDP recibe múltiples de estas etiquetas para cada prefijo y son almacenadas en la tabla LIB del router. No obstante, solo un par LDP es el próximo LSR para un prefijo en particular, claro esta que si existe un balanceo de carga es posible tener más de un próximo salto.
El próximo LSR es encontrado buscando el destino del salto para ese prefijo en la tabla de ruteo. Solo los enlaces remotos asociados con ese próximo salto, deberían ser usados para almacenarse en la LFIB. Esto significa que solo una etiqueta de todas las que se encuentran enlazadas por todos los vecinos LDP de ese LSR debería ser usada como salida en la LFIB para ese prefijo. El problema es que las etiquetas enlazadas son comunicadas como un LDP ID y la etiqueta sin la dirección IP de la interfaz. Esto significa que para encontrar la etiqueta de salida para un prefijo en particular, se debe construir un mapa entre el LDP ID y la dirección IP de la interfaz en el próximo LSR. Solamente se puede hacer esto si cada par LDP comunica todas sus direcciones IP. Estas direcciones son comunicadas por los pares LDP con **Mensajes de Direcciones** (Address Messsages) y eliminadas con mensajes de **Withdraw Address**.
Cada LSR asigna una etiqueta local a cada prefijo IGP en la tabla de ruteo. Ésta es la etiqueta enlazada local. Estos enlaces locales son almacenados en la LIB del router. Cada una de estas etiquetas y de estos prefijos son asignados para ser comunicados a través de LDP a todos los pares LDP. Estas etiquetas son los enlaces remotos en los pares LDP y son almacenados en la LIB. Para cada prefijo, el LSR siempre tiene un solo enlace local y un enlace remoto por par LDP.

6.3.2.5- Label Withdrawing

Cuando un par LDP comunica una etiqueta enlazada, el LDP receptor la mantiene hasta que la sesión LDP termine o hasta que la etiqueta sea retirada. Una etiqueta puede ser retirada si la etiqueta local cambia. Esto puede pasar por ejemplo cuando la interfaz de un cierto prefijo cae, pero otro LSR aún comunica el prefijo. Por lo tanto, la etiqueta local para ese prefijo cambia de implicit NULL a una etiqueta no reservada. Si esto sucede, la etiqueta implicit NULL es inmediatamente retirada por el envío de un mensaje "Label Withdraw" al par LDP. La nueva etiqueta es comunicada con un mensaje de "Label Mapping".

6.3.2.6- Gestión interna por medio de notificaciones

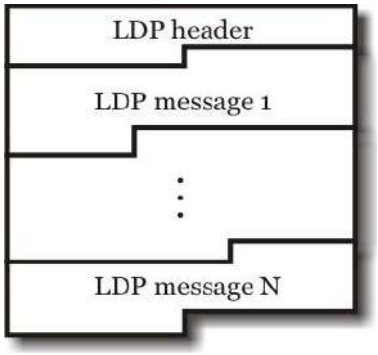
Los mensajes de notificación son necesarios para la gestión interna de las sesiones LDP. Un mensaje de notificación señala eventos importantes a los pares LDP. Estos eventos pueden ser errores fatales (notificaciones de error) o simplemente información de advertencia (notificaciones de advertencias). Si ocurre un error fatal, se debe terminar la sesión LDP inmediatamente. Las notificaciones de advertencias son usadas para enviar información acerca de la sesión LDP o de un mensaje recibido desde el par.

Algunos ejemplos de mensajes de notificación son los siguientes:

- Espiración de tiempos de keepalive.
- Caídas de sesiones.
- Mensajes de eventos de iniciación.
- Eventos resultantes de otros mensajes.
- Errores internos.
- Detecciones de loop.
- Eventos varios.

6.3.2.7- Mensajes LDP

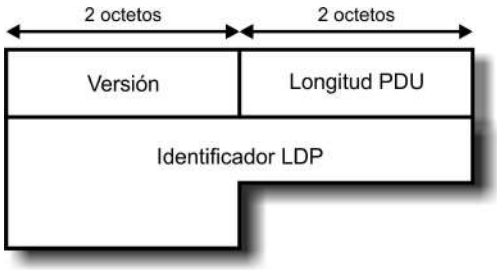
Los LSRs intercambian mensajes LDP a través del formato LDP PDU, el cual consiste de una cabecera LDP y uno o más mensajes LDP que pueden estar o no relacionados entre si.
El formato de los mensajes LDP PDU es el siguiente:



Mensaje LDP PDU

6.3.2.7.1- Cabecera del mensaje LDP

La Cabecera LDP se forma de la siguiente manera:



Cabecera LDP

Compuesta por los siguientes campos:

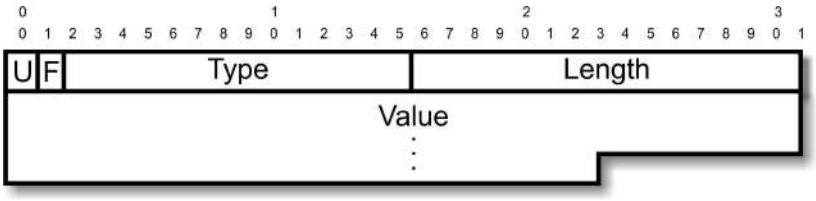
- **Versión:** Campo de 16 bits que indica la versión del protocolo.
- **Longitud PDU(PDU length):** Campo de 16 bits que indica la longitud total del mensaje LDP PDU en bytes, excluyendo la versión y el campo Longitud PDU.
- **Identificador LDP(LDP ID):** Campo de 48 bits, de los cuales 32 bits indican el id del router y los 16 bits restantes indican el número de espacio de etiqueta.

El formato de los mensajes LDP consisten de una cabecera seguido de parámetros mandatarios y

parámetros opcionales. Esta junto con los parámetros son codificados usando el esquema de valores de **Longitud y Tipo** (Type Length Value o TLV).

6.3.2.7.2- Codificación TLV de los mensajes LDP

La codificación TLV, esta formada de la siguiente manera:



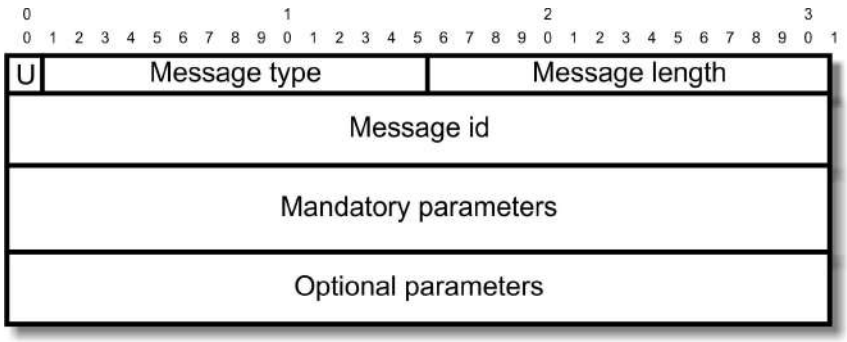
Codificación TLV

Compuesta por los siguientes campos:

- **U (bit TLV desconocido):** Bit usado cuando se recibe un TLV desconocido. Si U = 0, una notificación es devuelta al originador del mensaje y todo el mensaje es ignorado. Si U = 1, el TLV es ignorado y el resto del mensaje es procesado como si el TLV no hubiese existido.
- **F (bit de envío de TLV desconocido):** Este es utilizado únicamente cuando U = 1, y el mensaje LDP conteniendo un TLV desconocido tiene que ser enviado. Si F = 0, el TLV desconocido no es enviado con el resto del mensaje. Si F = 1, el TLV desconocido es enviado con el resto del mensaje.
- **Type:** Campo de 14 bits que indica como el campo Value debe ser interpretado.
- **Length:** Campo de 16 bits que indica la longitud del campo Value en bytes.
- **Value:** Contiene información que es interpretada como específica el campo Type. Este campo puede contener a su vez otras TLV.

6.3.2.7.3- Formato de los mensajes LDP

El formato de los mensajes LDP, es el siguiente:



Mensaje LDP

Compuesta por los siguientes campos:

- **U (bit de mensaje desconocido):** Cuando se recibe un mensaje desconocido, Si U = 0, una notificación es devuelta al originador del mensaje. Si U=1, el mensaje es simplemente ignorado.
- **Message type:** Campo de 15 bits que indica el tipo de mensaje.

- **Message length:** Campo de 16 bits que indica la longitud total (en bytes) del campo Message ID y de los parámetros mandatarios y opcionales.
- **Message ID:** Campo de 32 bits que identifica al mensaje. Mensajes posteriores y relacionados a este mensaje tendrán el mismo ID.

Los parámetros mandatarios y opcionales varían dependiendo los distintos mensajes:

- Notificación.
- HELLO.
- Iniciación.
- Mantenimiento.

Otros tipos de mensajes son:

- Dirección.
- Retiro de dirección.
- Asociación de etiqueta.
- Petición de etiqueta.
- Petición de abandono de etiqueta.
- Retiro de etiqueta.
- Liberación de etiquetas.

6.3.3- Resource Reservation Protocol (RSVP)

Un protocolo de señalización alternativo a LDP es el Resource Reservation Protocol-Traffic Engineering (RSVP-TE) el cual fue diseñado para soportar la arquitectura de Integrated Services (IntServ). Para poder entender RSVP-TE primero hay que entender como funciona RSVP.

La arquitectura IntServ fue desarrollada por IETF a mediados de 1990 con vistas a introducir QoS en una red IP. Se definieron las siguientes dos clases en IntServ:

- **Guaranteed Service:** Este servicio provee limites "rígidos" en el retardo de la cola extremo-a-extremo sin perdida de paquetes.
- **Controlled-load Services:** Este servicio le provee al usuario con una QoS muy similar a la del servicio de mejor esfuerzo que el usuario recibiría de una red sin carga. Específicamente, un usuario podría asumir lo siguiente:
    - Un porcentaje muy alto de paquetes transmitidos serán entregados exitosamente por la red al receptor.
    - El porcentaje de paquetes no entregados exitosamente debe ser muy similar a la tasa de error de paquetes básica del enlace de transmisión.
    - El retardo extremo-a-extremo experimentado por un gran porcentaje de los paquetes entregados no excederá en gran medida el mínimo retardo extremo-a-extremo.

En IntServ, el emisor especifica cuanto tráfico transmitirá a su receptor (o receptores), y un receptor

especifica cuanto tráfico éste puede recibir y la QoS requerida, expresada en términos de pérdida de paquetes y retardo extremo-a-extremo. Esta información permite a cada router IP a través del camino seguido por los paquetes enviados efectuar las siguientes funciones:

- **Policing:** Esta es usada para verificar que el tráfico transmitido por el emisor es conforme a un conjunto de descriptores de tráfico que caracterizan el tráfico transmitido por el emisor.
- **Admission control:** Éste es usado para decidir cuando un router IP tiene recursos adecuados para satisfacer la QoS solicitados.
- **Classification:** Éste es usado para decidir que paquetes IP deberían ser considerados como parte del tráfico del emisor y proveerles de la QoS solicitada.
- **Queueing and scheduling:** Para que un router IP pueda proveer diferentes QoS a diferentes receptores, éste tiene que ser capaz de encolar paquetes en diferentes colas y de transmitirlos fuera de estas colas de acuerdo a un plan.

La arquitectura IntServ requiere un protocolo de señalización para el establecimiento y mantenimiento fiable de la reserva de recursos. Como en MPLS, IntServ no requiere del uso de un protocolo de señalización específico y puede acomodar una gran variedad de protocolos de señalización, donde el RSVP es el más común. RSVP fue desarrollado para soportar la arquitectura IntServ, pero puede ser usado para acarrear otro tipo de información de control. Esto es porque RSVP no tiene en cuenta el contenido de los campos del protocolo RSVP que contienen tráfico e información de políticas de control usados por los router para reservar recursos. RSVP puede ser usado para hacer reservas de recursos en aplicaciones tanto unicast como multicast.

RSVP fue designado en vistas de soportar conferencias “multiparty” con receptores heterogéneos. En RSVP, la reserva de recursos es decidida e iniciada por el receptor, ya que éste es el único que conoce cuanto ancho de banda necesitará. Éste metodología también permite al receptor unirse o abandonar el multicast cuanto éste desee.

Un problema de ésta última metodología es que el receptor no conoce el camino del emisor hasta él. Por lo tanto, éste no puede pedir asignación de recursos en cada router a lo largo del camino ya que no conoce cuales son estos routers. Este problema es solucionado usando el mensaje path que tiene origen en el emisor y viaja a través de la ruta unicas o mulicast al receptor. El principal objetivo del mensaje path es almacenar la información de **Estado del Camino** (Path State Information) en cada nodo del mismo, y acarrear información acerca de las características de tráfico del emisor y las propiedades del camino extremo-a-extremo.

A continuación alguna de la información contenida en el mensaje path:

- **Phop:** Esta el la dirección del router anterior que reenvía el mensaje. Esta dirección es almacenada en la información de estado del camino en cada nodo, y es usada para enviar el mensaje de reserva upstream hacia el emisor.
- **Sender template:** Este campo acarrea la dirección IP del emisor y opcionalmente el puerto UDP/TCP.
- **Sender Tspec:** Este define las características del tráfico del flujo de datos que el emisor generará.
- **Adspec:** Este acarrea la información OPWA (one-pass with advertising). Esta es información juntada en cada nodo a través del camino seguido por el mensaje path. Esta información es entregada al receptor, quien puede usarla para construir un nuevo pedido de reserva o para modificar una reserva existente.

Cunado se recibe el mensaje path, el receptor envía un mensaje Resv hacia el emisor a través del camino inverso al del mensaje path. El mensaje Resv contiene la siguiente información:

- **Flowspec:** Especifica la QoS deseada. Este consiste del Tspec del receptor, el Rspec, y la clase de servicio. El Tspec del receptor es un conjunto de descriptores de tráfico que es usado por los nodos a lo largo del camino para reservar recursos. El Rspec define el ancho de banda deseado y las garantías de retardo. Cuando RSVP es usado en IntServ, la clase de servicio puede ser el **Servicio Garantizado** (Guaranteed Service) o el **Servicio de Carga Controlada** (Controlled-Load).
- **Filter spec:** Define un paquete que recibirá la QoS requerida que fue definida en el flowspec. Un Filter Spec simple podría ser solo la dirección IP del emisor y opcionalmente su puerto UDP/TCP.

Cuando un router recibe el mensaje Resv, este reserva recursos de acuerdo a las instrucciones del receptor y luego envía el mensaje Resv al router del salto anterior obtenido de la información de estado del camino.

Los mensajes RSVP son enviados en datagramas IP sin una encapsulación TCP o UDP (raw IP datagrams).

RSVP hace uso de las notaciones flujo de datos y sesión. Una sesión está definida por los parámetros:

- Dirección IP destino
- ID del protocolo
- Opcionalmente número de puerto destino

RSVP hace reservas para flujos de datos unidireccionales. De esta forma, para que dos usuarios X e Y se comuniquen en ambas direcciones, es necesario establecer dos sesiones, una sesión de Y a X y otra desde X a Y.

6.3.3.1- Estilos de reservas

Tres tipos de estilos de reservas pueden ser usados con RSVP (denominadas esquemas). Para poder entender estos esquemas se puede considerar el caso donde un número de emisores transmiten información a un mismo receptor. Cada emisor transmite su propio flujo de datos de paquetes en una sesión, la cual esta definida por la dirección IP del receptor y un ID de protocolo. Una opción consiste con la reserva de recursos de estas sesiones. En particular, se puede considerar que varios de estos flujos de datos pasan a través de un mismo router. Este router tiene la opción de establecer una reserva separada para cada flujo de datos o hacer una sola reserva para todos los flujos de datos.

Una segunda opción de reserva controla la selección de los emisores. Ésta puede ser **explicit** o **wildcard**. En la selección de emisor **explicit**, el receptor provee una lista de emisores de los cuales éste desea recibir datos. Un emisor no puede enviar paquetes a un receptor a menos que su dirección IP esté en la lista explícita. En la selección de emisor **wildcard**, cualquier emisor puede transmitir datos al receptor. Basados en estas dos opciones los siguientes tres estilos fueron definidos:

- **Estilo wildcard-filter (WF):** Cualquier emisor puede transmitir a la sesión. Existe solo una reserva de recursos compartida por todos los flujos de datos desde todos los emisores

upstream. La reserva de recursos es la más grande de todas las requeridas.

- **Estilo fixed-filter (FF):** Una reserva separada es creada para cada emisor en particular que es especificado en la lista explícita de emisores. Otros emisores identificados en la lista explícita que transmiten en la misma sesión no comparten esta reserva.
- **Estilo shared explicit (SE):** Una lista de emisores es explícitamente declarada y hay una sola reserva compartida por todos sus flujos.

6.3.3.2- Manejo del estado de las reservas

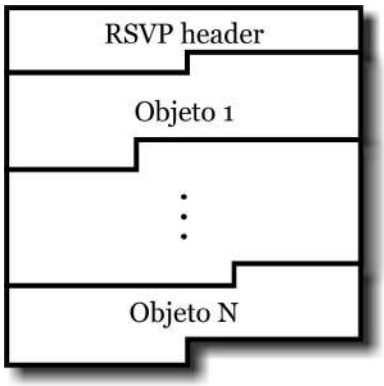
RSVP toma un "enfoque de estado blando" para manejar el estado de las reservas en los routers y hosts. Esto quiere decir que la información de estado en cada router y hosts tienen que ser periódicamente actualizada por mensajes Path y Resv. El estado de una reserva es eliminado si ningún mensaje de actualización arribo antes de que un **Timeout Expire**. Un estado también puede ser eliminado por un mensaje explícito.

Cuando una ruta cambia, el siguiente mensaje Path, inicializará el estado del camino en los routers a lo largo de la nueva ruta y el mensaje Resv establecerá una reserva en cada uno de estos routers. El estado de las rutas sin uso expirarán.

RSVP envía mensajes como datagramas IP sin garantías de que estos sean entregados. Un mensaje RSVP puede que nunca sea entregado debido a errores en la transmisión u overflows de buffers. Esta situación es controlada por mensajes de actualización periódicos. Enviando mensajes de actualización incrementa la carga de la red, pero elimina la necesidad de usar un protocolo confiable como TCP.

6.3.3.3- Formato del mensaje RSVP

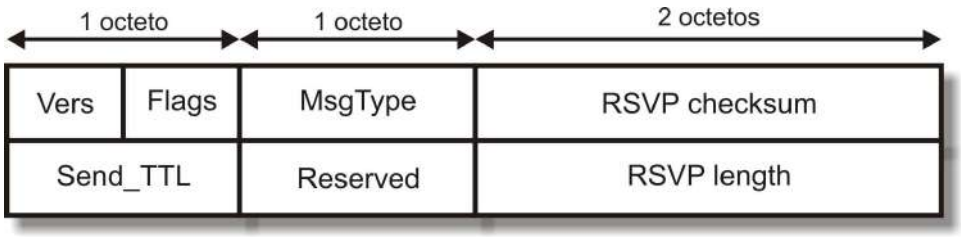
Un mensaje RSVP consiste de un encabezado común seguido de un número variable de objetos. Cada objeto contiene un grupo de parámetros relacionados y tiene una longitud variable.



Mensaje RSVP

6.3.3.3.1- Cabecera del mensaje RSVP

La cabecera del mensaje tiene la siguiente forma:



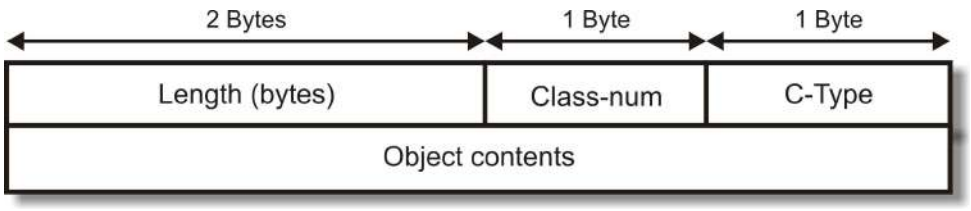
Cabecera RSVP

Compuesta por los siguientes campos:

- **Vers:** Campo de 4 bits usados para indicar el número de versión del protocolo.
- **Flags:** Campo de 4 bits usados para flags; ninguna flags ha sido especificada.
- **MsgType:** El tipo de mensaje es especificado por un número llevado en este campo de 8 bits. Los siguientes tipos de mensajes y respectivos números han sido especificados:
    - Path
    - Resv
    - PathErr
    - ResvErr
    - PathTear
    - ResvTear
    - ResvConf
- **RSVP checksum:** Un checksum de 16 bits es calculado sobre el mensaje entero.
- **Send_TTL:** Un campo de 8 bits que contiene el valor del tiempo de vida de IP.
- **RSVP length:** La longitud total en bytes es almacenada en este campo de 8 bits. La longitud incluye el encabezado común y todos los objetos siguientes.

6.3.3.3.2- Objetos del mensaje RSVP

Un objeto del mensaje tiene la siguiente forma:



Campos de los objetos

Compuesta por los siguientes campos:

- **Length:** Un campo de 16 bits usado para indicar el largo total en bytes del objeto. Éste tiene que ser un múltiplo de cuatro, y ser por lo menos igual a cuatro.
- **Class-num:** Un campo de 8 bits usado para identificar la clase del objeto.
- **C-Type:** Un campo de 8 bits usado para definir el tipo de objeto.

Se mencionan a continuación algunas de las clases de objetos que han sido definidas:

- NULL: El contenido de un objeto NULL es ignorado por el receptor.
- SESSION: Contiene la dirección IP destino, el ID del protocolo IP, y opcionalmente un puerto destino. Este objeto es requerido en cada mensaje RSVP.
- RSVP_HOP: Acarrea la dirección IP del router RSVP que envía este mensaje.
- TIME_VALUES: Contiene el valor para el período de actualización usado por el creador del mensaje. Éste es requerido en cada mensaje Path y Resv.
- STYLE: Define el estilo de reserva más información específica del estilo. Es requerido en cada mensaje Resv.
- FLOWSPEC: Acarrea la información necesaria en un mensaje Resv para hacer una reserva en un router.
- FILTER_SPEC: Define cuales paquetes de datos deberían recibir la QoS especificada en el objeto FLOWSPEC. Es requerido en un mensaje Resv.
- SENDER_TSPEC: Contiene las características del tráfico del flujo de datos del emisor. Es requerido en un mensaje Path.
- ADSPEC: Acarrea la información OPWA (one-pass with advertising).
- ERROR_SPEC: Especifica un error en un mensaje PathErr, ResvErr o una confirmación en un mensaje ResrConf.
- POLICY_DATA: Acarrea información que le permite al router decidir cuando una reserva es permitida administrativamente. Puede aparecer en un mensaje Path, Resv, PathErr o ResvErr.
- INTEGRITY: Acarrea datos criptográficos para autenticar el nodo originario para verificar el contenido de este mensaje RSVP.
- SCOPE: Acarrea una lista explícita de hosts emisores hacia los cuales la información en el mensaje será reenviada. Puede aparecer en mensajes Resv, ResvErr o ResvTear.
- RESV_CONFIRM: Acarrea la dirección IP de un receptor que pidió una confirmación. Puede aparecer en un mensaje Resv o ResvConf.

6.3.3.4- El mensaje Path

El mensaje Path consiste del encabezado común mencionado anteriormente seguido de los objetos:

- INTEGRITY (opcional)
- SESSION

- RSVP_HOP
- TIME_VALUES
- POLICY_DATA (opcional)
- Un descriptor de emisor formado por SENDER_TEMPLATE y el SENDER_TSPEC
- ADSPEC (opcional)

Cada host emisor envía un mensaje Path para cada flujo de datos al que desea transmitirle. El mensaje Path es reenviado desde un router a otro usando la "información de siguiente salto" de la tabla de routeo hasta que alcanza al receptor. Cada router a lo largo del camino captura y procesa el mensaje Path. El router crea un "estado del camino" para el par {emisor, receptor} definido en los objetos SENDER_TEMPLATE y SESSION del mensaje Path. También son gravados en el "estado del camino" cualquier objeto POLICY_DATA, SENDER_TSPEC y ADSPEC. Si se encuentra un error, un mensaje PathErr será enviado al originador del mensaje Path.

6.3.3.5- El mensaje Resv

Cuando un receptor recibe un mensaje Path, este genera un mensaje Resv es cual es enviado hacia el emisor a través del camino inverso al tomado por el mensaje Path. Recordar que los paquetes de datos siguen el mismo camino que el mensaje Path. El mensaje Resv es un pedido a cada nodo a lo largo del camino para que reserve recursos para el flujo de datos.

Este mensaje consiste del encabezado común mencionado anteriormente seguido de los objetos:

- INTEGRITY (opcional)
- SESSION
- RSVP_HOP
- TIME_VALUES
- RESV_CONFIRM (opcional)
- SCOPE (opcional)
- POLICY_DATA (opcional)
- STYLE

7- Soporte para calidad de servicio

QoS o Calidad de Servicio (Quality of Service, en inglés) está compuesto por un conjunto de tecnologías que garantizan la transmisión de cierta cantidad de información en un tiempo dado. Calidad de servicio es la capacidad de dar un buen servicio. Es especialmente importante para ciertas aplicaciones tales como la transmisión de vídeo o voz.

7.1- Calidad de servicio sobre IP

La Internet Engineering Task Force (IETF) ha diseñado dos maneras para implementar QoS en una red IP:

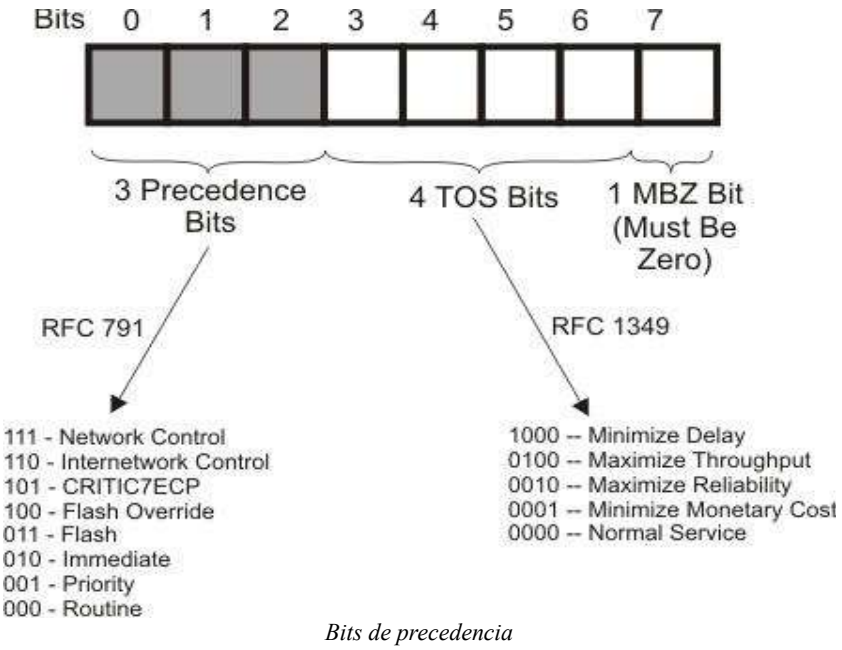
- Servicios integrados (Integrated Services o IntServ)
- Servicios diferenciados (Differentiated Services o DiffServ).

DiffServ usa los bits de DiffServ en la cabecera IP para calificar un paquete IP en una determinada QoS. Los routers examinan esos bits para marcar, encolar, formar y establecer la caída de precedencia de un paquete. La mayor ventaja de DiffServ sobre IntServ es que el modelo DiffServ no necesita protocolos de señalización. El modelo IntServ usa un protocolo de señalización que debe funcionar en los host y en los routers. Si la red tiene muchos flujos, los routers deben tomar información de estado para cada flujo que pasa a través de el. Esto es un problema serio de escalabilidad, que ha hecho que IntServ no se halla convertido en un modelo popular.

Se puede establecer la prioridad de un paquete IP tanto con el campo de precedencia IP (3 bits) o con el campo de DiffServ Codepoint (DSCP) de 6 bits. Originalmente, solo 3 bits del campo ToS del la cabecera IP eran reservados para calidad de servicio. Luego fue incrementado a 6 bits con la introducción de calidad de servicio DiffServ.

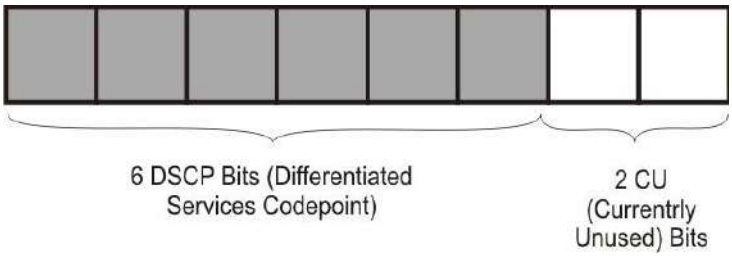
7.2- Señalización del campo ToS del la cabecera IP.

Los Bits de precedencia del campo TOS, en la cabecera IP esta formado de la siguiente manera:



Bits de precedencia

Los Bits de QoS en DiffServ esta formado de la siguiente manera:



Bits de QoS en DiffServ

Con los bits DSCP se puede elegir 64 valores (0 a 63) para la calidad de servicio, mientras que con los 3 bits de precedencia originales solo se podía elegir 8 niveles de calidad de servicio.

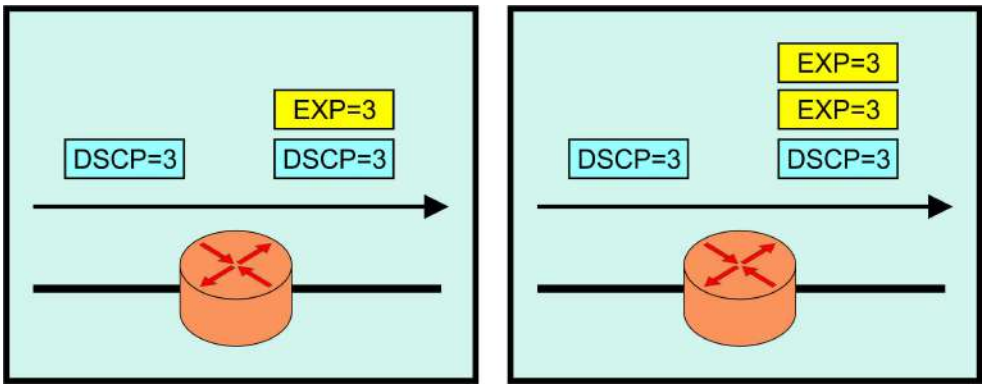
7.3- Paquetes MPLS con DiffServ

En MPLS los 3 bits de EXP pueden ser utilizados de la misma manera que los 3 bits de precedencia de la cabecera IP. Si se usan estos 3 bits para QoS, los LSP pueden ser llamados E-LSP, indicando que los LSRs usarán los bits EXP para catalogar a los paquetes y decidir sobre su importancia. Sin embargo, en MPLS existe otra opción para implementar QoS para los paquetes etiquetados. Se puede utilizar a la etiqueta tope del paquete para implicar parte de la calidad de servicio para dicho paquete. Esto implica la necesidad de una etiqueta por clase para cada flujo o tráfico entre dos puntos de un LSP. Por lo tanto, un protocolo de señalización tiene que ser capaz de señalar diferentes etiquetas para el mismo LSP o prefijo. En esta opción, a los LSP se los conoce como L-LSP, indicando que la etiqueta tiene implícitamente parte de la información de la calidad de servicio. Con un L-LSP, los bits de EXP continúan teniendo parte de la QoS, solo la caída de precedencia, mientras que la etiqueta indica la clase. Con E-LSP, los bits EXP mantienen información de la clase y de la caída de precedencia.

7.4- Reglas de QoS en MPLS

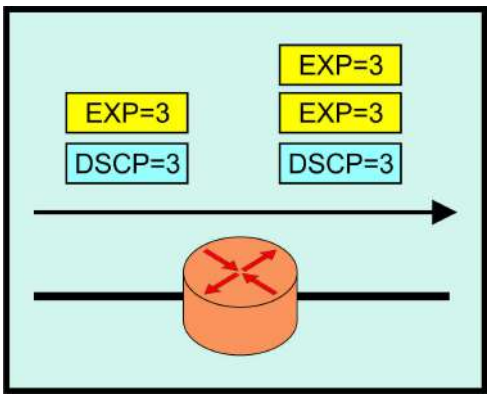
El comportamiento de los LSR en los sistemas Cisco IOS para dar soporte de calidad a través de un E-LSP se puede resumir en las siguientes reglas:

Regla 1: Los bits de precedencia o los primeros 3 bits del campo DSCP de la cabecera IP son copiados a los bits EXP de todas las etiquetas impuestas en un ingress LSR.



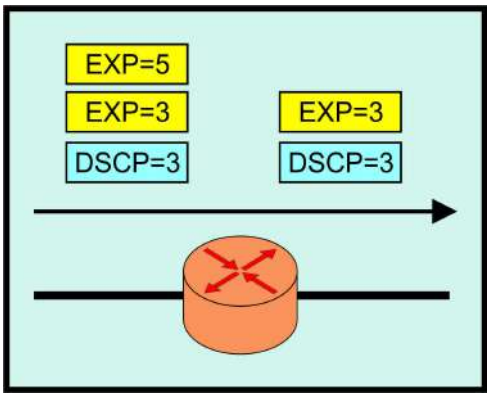
Ejemplos de la regla 1

Regla 2: Los bits EXP de la etiqueta tope de entrada son copiados a las etiquetas de salida intercambiadas (swapped) y a cualquier etiqueta puesta encima de esta.



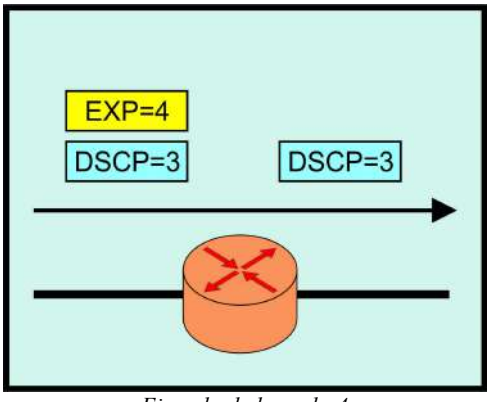
Ejemplo de la regla 2

Regla 3: Los bits EXP de la etiqueta tope de entrada no son copiados a las etiquetas nuevamente expuestas cuando la etiqueta de entrada es quitada.



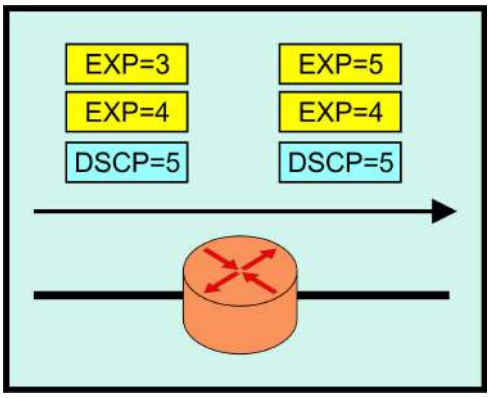
Ejemplo de la regla 3

Regla 4: Los bits EXP de la etiqueta tope de entrada no son copiados a los bits de precedencia o a los bits de DSCP cuando la pila de etiquetas es removida y el paquete queda sin etiquetas.



Ejemplo de la regla 4

Regla 5: Cuando se cambian los valores de los bits EXP mediante la configuración, el valor de los bits EXP en las etiquetas que no sean la etiqueta tope, las etiquetas intercambiadas (swapped), o las etiquetas impuestas por debajo de la etiqueta tope y los bits de precedencia o los bits DSCP se mantienen sin cambios.



Ejemplo de la regla 5

8- Ingeniería de Tráfico

La Ingeniería de Tráfico (TE) es una disciplina que procura la optimización de la performance de las redes operativas. Esta abarca la aplicación de la tecnología y los principios científicos a la medición, caracterización, modelado, y control del tráfico que circula por la red. Las mejoras del rendimiento de una red operacional, en cuanto a tráfico y modo de utilización de recursos, son los principales objetivos de la Ingeniería de Tráfico. El objetivo es en este caso el optimizar la utilización de los recursos de la red de manera que no se saturen partes de la misma mientras otras permanecen sub-utilizadas. Esto apunta a un objetivo global, que es minimizar la congestión del tiempo al intentar incrementar la eficiencia de la utilización de los recursos. Claramente, la congestión es un fenómeno nada deseable y es causada por la insuficiencia de recursos en la red. En casos de congestión de algunos enlaces, el problema se podría resolver, añadiendo más capacidad a los enlaces. La otra causa de congestión es la utilización ineficiente de los recursos debido al mapeo del tráfico. El objetivo básico de la Ingeniería de Tráfico es adaptar los flujos de tráfico a los recursos físicos de la red. La idea es equilibrar de forma óptima la utilización de esos recursos, de manera que no haya algunos que estén sobre-utilizados, creando cuellos de botella, mientras otros puedan estar sub-utilizados.

8.1- Ingeniería de Tráfico en redes MPLS

El RFC 2702, 'MPLS Traffic Engineering (TE)' establece que la ingeniería de tráfico concierne a la optimización de la performance de una red e involucra diversas áreas:

- Mediciones de tráfico.
- Modelado de tráfico y redes.
- Control del tráfico en Internet.
- Evaluación de performance.

Se establece que los principales Objetivos de TE son:

- Utilizar el exceso de ancho de banda sobre los enlaces sub-utilizados.
- Maximizar la utilización de los enlaces y nodos de la red.
- Aumentar la confiabilidad del servicio.
- Alcanzar requerimientos impuestos.

Los requerimientos pueden ser:

- **Orientados al tráfico:** basados en el estudio de la pérdidas de paquetes o retardos.
- **Orientados a los recursos:** basados en la utilización de la capacidad de la red.

Las acciones de control tomadas al realizar TE pueden involucrar:

- Modificación de los parámetros de Gestión de Tráfico.
- Modificación de los parámetros asociados al ruteo.
- Modificación de los parámetros y atributos asociados con los recursos.

En general se busca también minimizar la intervención manual para tomar acciones de control. En modelos con una sola clase, puede encapsular todo el tráfico entre dos LERs. Los caminos son objetos enrutables y deben diferenciarse del LSP que utiliza un camino en un momento dado. Esta distinción es importante porque el LSP puede cambiar pero el camino sigue siendo el mismo.

8.2- Ruta explicita

La principal característica de MPLS que permite realizar TE es el ruteo explícito. Una ruta explícita es una secuencia de nodos lógicos entre un nodo de ingreso y uno de egreso que se definen y establecen desde un nodo de la frontera. Una ruta explícita puede ser una lista de direcciones IP. También pueden especificarse los primeros N saltos solamente y luego la ruta definida por el protocolo de ruteo IP. Puede usarse también en una ruta explícita el concepto de Nodo Abstracto, que es una colección de nodos presentados como un solo paso en una ruta explícita. Si el nodo de entrada quiere establecer una ruta que no sigue el camino por defecto el protocolo de ruteo IP, debe utilizar un protocolo de distribución de etiquetas que soporte la definición de rutas explícitas, por ejemplo el LDP y RSVP.

El LSP puede ser restringido por la capacidad de recursos y la capacidad de los nodos de cumplir con los requerimientos de QoS. Esto lleva al concepto de **Constrained Route** (CR) o ruta con restricciones. Una ruta con restricciones se obtiene imponiendo un conjunto de restricciones que se deben cumplir, por ejemplo, información de QoS del enlace (ancho de banda disponible, retardo, etc.). EL LSR de ingreso calcula una ruta que satisfaga un conjunto de restricciones en el estado actual de la red.

Para encontrar una CR se debe correr un algoritmo de **Ruteo Basado en Restricciones** (Constrained Base Routing, CBR). En el RFC 2702 se establece que para realizar TE en una red MPLS debe ser posible definir, atributos asociados a los caminos de tráfico que en conjunto especifican su comportamiento. Dentro de estos atributos se encuentran:

- Parámetros del tráfico del camino, caracterización del tráfico que utilizaría ese camino.
- Atributos para el establecimiento y mantenimiento de caminos establecidos administrativamente.
- Reglas para establecer preferencias de ciertos caminos que pueden ser mandatarios o no, se

considera adecuado tener atributos que establezcan una jerarquía o preferencia para mapear un camino dentro de un conjunto de posibles caminos.

- Clase de afinidad con recursos, se recomienda contar con atributos que permitan establecer clases de afinidad entre los recursos y de establecer caminos usando aquellos recursos que le son afines.
- Adaptabilidad a cambios, se debe poder especificarse si ante cambios en el estado de la red se re-calculan o no los caminos establecidos.
- Prioridad de los diferentes caminos a la hora de establecer y de mantener un LSP.
- Atributos asociados al re-enrutamiento, se debe poder decidir ante cambios en la red, si un camino se re-enrutará solo si hay caminos con recursos suficientes o si se re-enrutará siempre.
- Políticas para definir qué acciones se toman si el camino no cumple con los parámetros de tráfico que se especificaron.
- Máximo ancho de banda que se permite reservar para los caminos que atraviesan dicho enlace.
- Clases de recursos, afinidad que restringe el mapeo de los caminos sobre los recursos.

Existen diversas propuestas académicas de algoritmos para realizar CBR. Uno de los principales problemas que enfrenta el ruteo basado en restricciones es que casi cualquier problema de interés es NP-completo.

A raíz de este problema se han propuesto numerosos algoritmos heurísticos, para encontrar caminos sujetos a un conjunto de restricciones. Algunos de estos algoritmos son aptos solo para trabajar fuera de línea ya que por su complejidad dificulta su implementación en línea. Otros que se proponen para aplicaciones en línea, tienen la dificultad que se basan en el conocimiento de determinados parámetros de la red que no son fáciles de implementar.

8.3- Ruteo basado en restricciones

Algunos de los principales tópicos de investigación y desarrollo en el área de Ingeniería de Tráfico en MPLS son:

- El ruteo basado en restricciones.
- El reparto de carga.

8.3.1- Ruteo basado en restricciones

El ruteo basado en restricciones (Constraint Based Routing, CBR) busca caminos entre puntos de la red que satisfagan un conjunto de restricciones explícitas. Estas pueden ser por ejemplo, que las pérdidas sean menores que un cierto valor, que el retardo extremo a extremo sea menor que un cierto valor de tiempo o que exista un ancho de banda mínimo. Sin embargo se ha observado, que el ruteo basado en restricciones para casi cualquier problema real es un problema NP-completo. Por esta razón se han propuesto múltiples algoritmos heurísticos sub-óptimos para realizar CBR.

El ruteo basado en restricciones es uno de los pilares de la ingeniería de tráfico en MPLS. Existen

muchas propuestas pero el algoritmo básico que se utiliza en la actualidad para hacer CBR es CSPF (Constrained Shortest Path First). La idea de este algoritmo es modificar el tradicional algoritmo SPF (Shortest Path First), sacando aquellos enlaces que no cumplen alguna restricción. En cuanto a restricciones asociadas a requerimientos de QoS en la práctica la única que se utiliza es la capacidad requerida por el camino a establecer respecto de la capacidad disponible del enlace.

8.3.2- Reparto de carga

El **Balanceo de Carga** (Load Balancing) plantea el problema de dividir el tráfico de un agregado de flujos entre diversos caminos basados en algún criterio de optimización de la red. En los últimos años se han propuesto diversos algoritmos para realizar reparto de carga fuera de línea entre diversos caminos (LSPs) en una red MPLS por ejemplo utilizando modelos basados en el ancho de banda efectivo, se han propuesto algoritmos para balancear carga en línea en una red MPLS.

El RFC 3031, habilita a realizar balanceo o reparto de carga entre diferentes LSPs. Balancear carga es una potente herramienta de ingeniería de tráfico en MPLS. Esta herramienta brinda la posibilidad de enrutar caminos cuyo tráfico es superior a las posibilidades de un único camino en la red, y permite también mejorar el uso de recursos de la red.

Para realizar reparto de carga se deben tener en cuenta dos aspectos:

- El primero es el algoritmo con el cual se decide los coeficientes de reparto de carga entre los LSPs.
- El segundo es el mecanismo (una vez fijados los coeficientes) que se utiliza para asignar los paquetes a uno u otro LSP.

En ambos casos se busca optimizar alguna medida de performance como por ejemplo las pérdidas promedio o el retardo medio en la red.

Los algoritmos fuera de línea obtienen formas de cálculo que son adecuadas para una optimización de largo plazo de la red pero que son difíciles de aplicar en escalas más cortas. Por otra parte en su aplicación real cabe preguntarse que quiere decir "óptimos" en el largo plazo, cuando la red sufre variaciones de diferente índole y estos algoritmos se basan en estimaciones estadísticas de diversos parámetros. Este tipo de algoritmos además tratan de optimizar algún parámetro de la red, por ejemplo que las pérdidas máximas en la red sean mínimas, pero como dijimos, esto puede no lograr una configuración donde cada agregado de flujo tenga las pérdidas que se requieren para cumplir sus requerimientos de QoS. Es decir, puede convenir que las pérdidas en algunos agregados sean mayores, pero gracias a eso disminuir las pérdidas para el tráfico más restrictivo en cuanto a la QoS.

Una vez definidos los coeficientes de reparto de carga, existen dos formas de repartir carga entre LSPs:

- Por paquete.
- Por flujo.

El primer modo es más simple de implementar y más preciso. El segundo es más complejo de implementar y si no se tienen muchos flujos o estos son muy diferentes, es poco preciso.

Sin embargo, este último método presenta una fuerte ventaja al mantener el ordenamiento de los paquetes de un flujo.

Actualmente en los routers comerciales se puede realizar reparto de carga pero no se puede ajustar en línea los coeficientes de reparto. Es decir, se configuran los coeficientes de reparto y luego estos quedan fijos no existiendo un mecanismo de ajuste si hay variaciones de tráfico. Lo que sucede por defecto, es que si se establece más de un LSP para un camino, la carga se reparte por partes iguales entre los LSPs establecidos. En algunos casos se implementa una variante algo más sofisticada que es repartir carga inversamente proporcional al ancho de banda reservado por cada LSP.

En otros routers se permite configurar coeficientes de reparto de carga y repartir de acuerdo a estos coeficientes. En general también se permite configurar reparto de carga por paquete o por flujo.

8.4- Soporte a las clases de servicio

MPLS soporta diferentes clases de servicio para cada LSP. Como caso particular, puede soportar servicios diferenciados en el mismo LSP.

Históricamente, Internet ha ofrecido un solo nivel de servicio el de **Mejor Esfuerzo** (Best Effort). Con la aparición de aplicaciones multimedia y aplicaciones en tiempo real, surgió la necesidad de la diferenciación de servicios en Internet. De esta forma se podrán diferenciar servicios como el correo electrónico de otros que dependen mucho más del retardo y de la variación del mismo como el video y la voz interactiva.

El modelo de los servicios diferenciados definen los mecanismos para poder clasificar el tráfico en clases de servicio con diferentes prioridades. Para clasificar el tráfico se emplea el campo ToS (Type of Service). Una vez clasificados los paquetes en la frontera de la red, los paquetes se reenvían basándose en el campo ToS. El reenvío se realiza por salto, es decir, el nodo decide por sí solo como se deberá realizar el reenvío. A este concepto se le denomina comportamiento por salto (PHB: Per-Hop Behavior).

MPLS se adapta bien a este modelo, ya que las etiquetas MPLS tienen el campo EXP para poder propagar la clase de servicio QoS en el correspondiente LSP. Por tanto, una red MPLS puede transportar distintas clases de tráfico. Entre cada par de LSRs exteriores se pueden tener distintos LSPs con distintas prestaciones y distintos anchos de banda.

9- Redes Privadas Virtuales

Una **Red Privada Virtual** (VPN) se construye basado en conexiones realizadas sobre una infraestructura compartida, con funcionalidades de red y de seguridad equivalentes a las que se obtienen con una red privada. El objetivo de las VPNs es el soporte de aplicaciones intra/extranet, integrando aplicaciones multimedia de voz, datos y video sobre infraestructuras de comunicaciones eficaces y rentables. La seguridad supone aislamiento, y privacidad que el usuario "cree" que posee los enlaces. Las IP VPNs son soluciones de comunicación VPN basada en el protocolo de red IP de la Internet.

Las VPNs tradicionales se han venido construyendo sobre infraestructuras de transmisión compartidas con características implícitas de seguridad y respuesta predeterminada. Tal es el caso de las redes de datos Frame Relay, que permiten establecer PCVs (Permanent Virtual Circuit), conexiones del tipo permanente, entre los diversos nodos que conforman la VPN. La seguridad y las garantías proporcionan la separación de tráficos por PVC y el contrato de ancho de banda determinado en un tiempo determinado CIR (Commited Information Rate). Algo similar se puede

hacer con ATM, con diversas clases de garantías. Los inconvenientes de este tipo de solución es que la configuración de las rutas se basa en procedimientos más bien artesanales, al tener que establecer cada PVC entre nodos, con la complejidad que esto supone al proveedor en la gestión. Si se quiere tener conectados a todos con todos, en una topología lógica totalmente mallada, añadir un nuevo emplazamiento supone retocar todos los CPEs (Customer Premises Equipment), los cuales proporcionan una forma rentable de brindar un seguro privado de red basado en IP, del cliente y restablecer todos los PVCs.

La popularidad de las aplicaciones TCP/IP, así como la expansión de las redes de los NSPs (Network Security Program), ha llevado a tratar de utilizar estas infraestructuras IP para el soporte de VPNs, tratando de conseguir una mayor flexibilidad en el diseño e implantación y unos menores costes de gestión y provisión de servicio. La forma de utilizar las infraestructuras IP para servicio VPN (IP VPN) ha sido la de construir túneles IP de diversos modos.

El objetivo de un túnel sobre IP es crear una asociación permanente entre dos extremos, de modo que funcionalmente aparezcan conectados. Lo que se hace es utilizar una estructura no conectiva como IP para simular esas conexiones: una especie de tuberías privadas por las que no puede entrar nadie que no sea miembro de esa IP VPN.

Los túneles IP en conexiones dedicadas se pueden establecer de dos maneras:

- En el nivel 3, mediante el protocolo IPSec del IETF.
- En el nivel 2, mediante el encapsulamiento de paquetes privados (IP u otros) sobre una red IP pública de un NSP.

En las VPNs basadas en túneles IPSec, la seguridad requerida se garantiza mediante el cifrado de la información de los datos y de la cabecera de los paquetes IP, que se encapsulan con una nueva cabecera IP para su transporte por la red del proveedor. Es relativamente sencillo de implementar, bien sea en dispositivos especializados, tales como cortafuegos, como en los propios routers de acceso del NSP. Además, como es un estándar, IPSec permite crear VPNs a través de redes de distintos NSPs que sigan el estándar IPSec. Pero como el cifrado IPSec oculta las cabeceras de los paquetes originales, las opciones QoS son bastante limitadas, ya que la red no puede distinguir flujos por aplicaciones para asignarles diferentes niveles de servicio. Además, sólo vale para paquetes IP nativos, IPSec no admite otros protocolos.

En los túneles de nivel 2 se encapsulan paquetes multiprotocolo (no necesariamente IP), sobre los datagramas IP de la red del NSP. De este modo, la red del proveedor no pierde la visibilidad IP, por lo que hay mayores posibilidades de QoS para priorizar el tráfico por tipo de aplicación IP. Los clientes de la VPN pueden mantener su esquema privado de direcciones, estableciendo grupos cerrados de usuarios, si así lo desean (Además de encapsular los paquetes, se puede cifrar la información por mayor seguridad, pero en este caso limitando las opciones QoS). A diferencia de la opción anterior, la operación de túneles de nivel 2 está condicionada a un único proveedor.

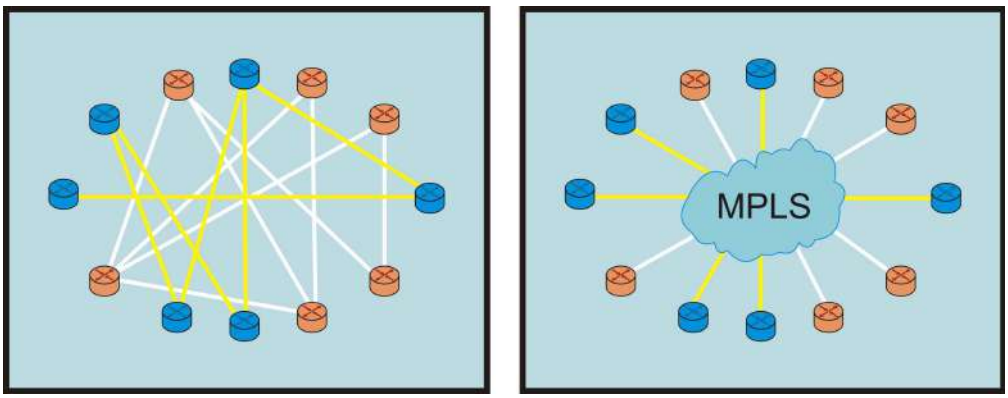
A pesar de las ventajas de los túneles IP sobre los PVCs, ambos enfoques tienen unas características comunes que las hacen menos eficientes frente a la solución MPLS:

- Están basadas en conexiones punto a punto (PVCs o túneles).
- La configuración es manual.
- La provisión y gestión son complicadas, una nueva conexión supone alterar todas las configuraciones.
- Plantean problemas de crecimiento al añadir nuevos túneles o circuitos virtuales.

La gestión de QoS es posible en cierta medida, pero no se puede mantener extremo a extremo a lo largo de la red, Ya que no existen mecanismos que sustenten los parámetros de calidad durante el transporte.

Realmente, el problema que plantean estas IP VPNs es que están basadas en un modelo topológico superpuesto sobre la topología física existente, basados en túneles extremos a extremo (o circuitos virtuales) entre cada par de routers de cliente en cada VPN. De ahí las desventajas en cuanto a la poca flexibilidad en la provisión y gestión del servicio, así como en el crecimiento cuando se quieren añadir nuevos emplazamientos. Con una arquitectura MPLS se obvian estos inconvenientes ya que el modelo topológico no se superpone sino que se acopla a la red del proveedor. En el modelo acoplado MPLS, en lugar de conexiones extremo a extremo entre los distintos emplazamientos de una VPN, lo que hay son conexiones IP a una "nube común" en las que solamente pueden entrar los miembros de la misma VPN. Las "nubes" que representan las distintas VPNs se implementan mediante los caminos LSPs creados por el mecanismo de intercambio de etiquetas MPLS. Los LSPs son similares a los túneles en cuanto a que la red transporta los paquetes del usuario (incluyendo las cabeceras) sin examinar el contenido, a base de encapsularlos sobre otro protocolo.

Esta es la diferencia existente entre ambos métodos, en los túneles se utiliza el encaminamiento convencional IP para transportar la información del usuario, mientras que en MPLS esta información se transporta sobre el mecanismo de intercambio de etiquetas, que no ve para nada el proceso de ruteo IP. Sin embargo, sí se mantiene en todo momento la visibilidad IP hacia el usuario, que no sabe nada de rutas MPLS sino que ve una **Internet Privada** (intranet) entre los miembros de su VPN. De este modo, se pueden aplicar técnicas de QoS basadas en el examen de la cabecera IP, que la red MPLS podrá propagar hasta el destino, pudiendo así reservar ancho de banda, priorizar aplicaciones, establecer QoS y optimizar los recursos de la red con técnicas de ingeniería de tráfico.



Modelo "superpuesto" (túneles/PVCs) vs. modelo "acoplado" (MPLS)

En la figura anterior se representa una comparación entre ambos modelos. La diferencia entre los túneles IP convencionales (o los circuitos virtuales) y los "túneles MPLS" (LSPs) está en que éstos últimos se crean dentro de la red, basados en LSPs, y no de extremo a extremo a través de la red.

Como resumen, las ventajas que MPLS ofrece para IP VPNs son:

- Proporcionan un modelo "acoplado" o "inteligente", ya que la red MPLS "sabe" de la existencia de VPNs (lo que no ocurre con túneles ni PVCs).
- Evita la complejidad de los túneles y PVCs.
- La provisión de servicio es sencilla: una nueva conexión afecta a un solo router.
- Tiene mayores opciones de crecimiento modular.

- Permiten mantener garantías QoS extremo a extremo, pudiendo separar flujos de tráfico por aplicaciones en diferentes clases, gracias al vínculo que mantienen el campo EXP de las etiquetas MPLS con las clases definidas a la entrada.
- Permite aprovechar las posibilidades de ingeniería de tráfico para poder garantizar los parámetros críticos y la respuesta global de la red (ancho banda, retardo, fluctuación...), lo que es necesario para un servicio completo VPN.

10- Diez razones para migrar a MPLS VPN

En los últimos tiempos, no sólo se viene hablando de la famosa convergencia de Voz, Video y Datos sobre una misma plataforma, sino también de la necesidad de la migración de servicios "Legacy" (heredados) como ATM o Frame Relay a una nueva generación de "IPbased VPNs" (Redes Privadas Virtuales basadas en protocolo IP) como los son las "MPLS VPNs".

Sin embargo, resistencia sigue siendo la primera palabra que se asocia cuando se habla de "cambios", mucho más aún, cuando se trata de migraciones de servicios de comunicaciones, críticos para una empresa.

A continuación, se enumeran 10 razones claves para hacer frente a la mencionada "resistencia" a los cambios cuando una empresa, corporación u organismo este pensando en migrar su infraestructura Legacy actual a una IP-Based MPLS VPN

10.1- Flexibilidad

Cada empresa, corporación u organismo tiene desarrollada su propia estructura interna, tanto en infraestructura como en recursos humanos, generadas en base a sus necesidades y recursos disponibles. En base a ésta estructura, muchas veces única, se montan los servicios de comunicaciones para acomodar de la mejor manera posible y al menor costo, el transporte de la información interna, así como también externa, con sus clientes y proveedores.

La topología de una MPLS VPN puede acomodarse acorde a cada necesidad, dada su naturaleza que brinda conexiones "Any-to-Any" (cualquiera con cualquiera) entre los distintos puntos que comprenden la VPN, contando así con el mejor camino o ruta entre cada punto. A su vez se puede obtener mayor flexibilidad realizando configuraciones híbridas con Hub-and-Spoke (estrella), por ejemplo en las conexiones con clientes.

10.2- Escalabilidad

Con un nuevo concepto de aprovisionamiento, llamado "Point-to-Cloud" (punto a la nube), se implementan los nuevos puntos de la VPN. Este concepto proviene del hecho de que cada vez que sea necesario "subir" un nuevo punto a la VPN, sólo habrá que configurar el equipamiento del Service Provider que conecte este nuevo punto. De esta forma, evitamos tareas complejas y riesgosas, como las que se producen cuando se activa un nuevo punto en una red basada en circuitos virtuales de Frame Relay o ATM, en donde es necesario re-configurar TODOS los puntos involucrados.

10.3- Accesibilidad

La arquitectura de MPLS VPN permite utilizar prácticamente todos las tecnologías de acceso para interconectar las oficinas del cliente con su "Service Provider" (Proveedor de Servicios).

Por dicho motivo, la versatilidad que nos permite utilizar xDSL o un enlace Wireless Ethernet en las oficinas más pequeñas y hasta incluso en usuarios móviles, mientras que en el headquarter utilizamos leased lines (TDM) en altas capacidades como E3/T3, nos permite dimensionar cada punto de la VPN acorde a sus necesidades sin limitar o restringir la de otros puntos.

10.4- Eficiencia

En una infraestructura 100% IP, es decir, aquellas empresas en donde todo el equipamiento involucrado y las aplicaciones utilizadas son IP-based, el uso de servicios de transporte ATM o Frame Relay someten al cliente a incurrir en un costo adicional por el overhead que los protocolos de transporte introducen. Mediante IFX MPLS VPN - un servicio IP-Based VPN - este costo extra desaparece.

10.5- Calidad de servicio (QoS) y Clases de servicio (CoS)

Las necesidades de comunicación entre dos lugares remotos, hoy en día van mucho más allá de la simple transferencia de datos vía email, web u otras aplicaciones. Siendo incluso insuficiente muchas veces, la interesante combinación de voz y datos bajo una misma plataforma. Es por ésto, que la ya mencionada convergencia de datos con aplicaciones real-time y/o interactivas, voz y también video de alta calidad, necesitan de una eficiente plataforma de transporte.

Mediante la utilización de técnicas y herramientas de Calidad de Servicio (QoS), se ofrecen distintas Clases de Servicio (CoS) dentro de una MPLS VPN para cumplimentar los requerimientos de cada servicio o aplicación.

10.6- Administración

Las MPLS VPN son denominadas Network-Based, ésta característica proviene del hecho en que el servicio es implementado sobre la infraestructura del Service Provider; implicando, entre otras cosas, que la administración de enrutamiento es llevada a cabo por el Service Provider; quien por su naturaleza, es especialista en dicha tarea desligando así al cliente de llevarla a cabo.

10.7- Monitoreo

Las MPLS VPN son monitoreadas, controladas y con un constante seguimiento en forma permanente, las 24 horas los 7 días de la semana, por parte del Service Provider. Además, se

extienden "Service Level Agreements" (acuerdos de nivel de servicio) para garantizar y asegurar la estabilidad y performance que el cliente necesite.

10.8- Fácil Migración

La simplicidad de la tecnología determina que las tareas de aprovisionamiento, administración y mantenimiento sean actividades sencillas para el Service Provider; lo cual se traslada directamente al cliente, obteniendo una migración del servicio actual sin complicaciones.

10.9- Seguridad

Análisis y estudios realizados por los distintos fabricantes y entidades especializadas en el área, determinaron que los niveles de seguridad entregados por una MPLS VPN son comparables con los entregados por los circuitos virtuales de Frame Relay y ATM.

Sin embargo, en escenarios donde estos niveles no son suficientes, como por ejemplo en las necesidades de entidades financieras, una MPLS VPN puede también ser combinada con la encriptación y autenticación que IPSec brinda, elevando aún más la seguridad de la VPN.

10.10- Bajo Costo

Son varios los motivos que permiten afirmar que un servicio MPLS VPN ofrece "más por menos", entre ellos podemos destacar:

- **Independencia de equipos de cliente (CPE):** al ser un servicio Network-based, la implementación de la VPN no requiere un hardware específico ni costoso para ser instalado en las oficinas del cliente.
- **Convergencia:** por ser una VPN CoS-Aware (Soporte de Clases de Servicio) se puede integrar distintos servicios y aplicaciones sobre una misma plataforma. De este modo, empresas que al día de hoy mantienen distintos y costosos servicios para soportar sus necesidades de voz, datos y video, pueden unificar estos requerimientos concluyendo en un ahorro significativo y manteniendo relación con un único proveedor de servicios.

11- Conclusión

MPLS es un protocolo que pretende mejorar no solo la calidad de servicio brindado en el envío de paquetes, sino, que también proporciona, la posibilidad de realizar ingeniería de trafico, utilización de VPN. Siempre manteniendo la mayor compatibilidad posible, con tecnología WAN (ATM, Frame Relay) existentes y actualmente en uso, siendo un soporte de estas para mejorar su utilización y eliminar el cuello de botella que se presenta entre una red LAN y una red WAN, mejorando el rendimiento de la red y brindándole a los usuarios un mejor servicio, ya sea de video, conferencias, telefonía IP, entre otras, con un costo mínimo y totalmente transparente.

Como se planteó en un primer momento se dice que este es un protocolo de capa 2,5, por lo tanto esta independencia total de la capa 2 y la capa 3, permite que este protocolo funcione indistintamente, ya sea dentro de una LAN o una WAN, ya que no depende de ninguno de los protocolos de ninguna de las dos capas del modelo OSI.

Una de las grandes ventajas de este protocolo es el contar con LDP, que básicamente se encarga de conectar todos los LSR, y generar los diferentes LSP, para enviar los paquetes de un extremo a extremo, generando esta nube MPLS, que permite al usuario una total transparencia en la utilización de la red.

12- Bibliografía

La bibliografía utilizada en la parte teórica, fue la siguiente:

- MPLS Fundamentals, Luc De Ghein, CCIE. 2007, Nº 1897. Indianapolis: 800 East 96th Street. ISBN: 1-58705-197-4.
- Connection-oriented Networks, Harry G. Perros. 2005, Inglaterra: West Sussex PO19 8SQ. ISBN 0-470-02163-2.

13- Indice

TESIS FINAL
INGENIERÍA EN SISTEMAS

REDES MPLS

(Parte de Laboratorio)

DATOS PERSONALES:

NOMBRE COMPLETO: Matias Eric Grassi

DNI: 31.192.578

DIRECTOR: Abel Crespo

1- Introducción

En el siguiente informe se va a configurar, mediante la herramienta **UML** (User Mode Linux), una red MPLS, esto es posible ya que se dispone de un parche (mpls-linux-1.962), para el kernel linux-2.6.25.14, el cual permitirá configurar el kernel agregando opciones y poder utilizar un laboratorio virtual, en el que se podrá ver el funcionamiento del mismo.
Primeramente, se indicará los requerimientos necesarios para realizar este ejercicio, luego se veran la configuraciones necesarias que se deben de realizar antes de compilar el kernel para que el parche pueda funcionar, y finalmente se realizará una instalación de los paquetes necesarios para que los experimento funcionen.
Luego se mostrará la topología utilizada, para llevar a cabo el ejercicio, junto con la configuración de cada PC y los comandos básicos utilizados para poder llevar a cabo los experimentos.
Teniendo todo configurado, primeramente se realizarán experimentos utilizando Ingeniería de Tráfico en MPLS, utilizando cuatro situaciones posibles:

- **La protección de link:** En la cual se garantiza el envío de mensajes por un LSP en el caso de que caiga una determinada interfaz por medio de un LSP secundario.
- **La protección de nodo:** Este caso es similar a *La protección de link* solo que ahora se intenta garantizar el envío de mensajes por un LSP, en el caso de que caiga un LSR determinado.
- **Balanceo de carga:** Se realizara un filtrado por el valor en el campo DSCP de la cabecera IP, enviando a los paquetes que tienen una determinada calidad de servicio por un LSP y los que tienen otra calidad de servicio por otro LSP.
- **QoS basado en la Ingeniería de Tráfico:** El cual permite realizar un cambio de LSP, en el momento en que se comiencen a enviar paquetes que superen una determinada tasa de transferencia, que provoquen el deterioro de la calidad de la red.

Y finalmente, se realizarán pruebas utilizando servicios diferenciados, configurando las siguientes situaciones:

- E-LSP.
- L-LSP.

2- Requerimientos MPLS-Linux laboratorio.

Todas las pruebas se llevaron acabo para un File System con slackware 12.1, donde se instalaron los siguientes paquetes:

- kernel linux-2.6.25.14 (www.kernel.org)
- mpls-linux-1.962 (http://mpls-linux.sourceforge.net/fedora/8/i386/mpls/)
- ebtables-2.0.8.tgz (http://mpls-linux.sourceforge.net/fedora/8/i386/mpls/)
- iproute-2.6.26.tgz (http://mpls-linux.sourceforge.net/fedora/8/i386/mpls/)
- iptables-1.4.1.1.tgz (http://mpls-linux.sourceforge.net/fedora/8/i386/mpls/)
- glibc-2.7-i486-17.tgz (ftp://mirror.pacific.net.au/linux/slackware/slackware-12.1/slackware/d/gcc-4.2.3-i486-1.tgz).

Estos paquetes son requeridos en común para todos los experimento, luego serán mencionado otros paquetes que serán utilizados específicamente para cada experimento.

3- Instalación

En primer lugar se va a realizar la compilación del kernel con el parche mpls-linux-1.962, descomprimimos tanto el kernel y, como el parche mpls-linux-1.962:

- tar xvf linux-2.6.25.14.tar
- tar xvf mpls-linux-1.962.tar

Luego que se termina de descomprimir los dos archivos anteriores, se debe aplicar el parche del kernel para poder configurar antes de compilar el servicio de MPLS. Para ingresar el parche en el kernel se deben realizar los siguientes pasos:

- Ingresar a la carpeta linux-2.6.25.14 (es decir ejecutar el comando cd linux-2.6.25.14).
- Suponiendo que la carpeta de mpls-linux-1.962 se encuentra un nivel por debajo de la actual se debe de ejecutar el siguiente comando:
  - patch -p1 < ../mpls-linux-1.962/patches/linuxkernel.diff.

Una ves ejecutado este comando aparecerán las siguientes lineas:

patching file include/linux/genetlink.h
patching file include/linux/if_arp.h
patching file include/linux/mpls.h
patching file include/linux/netdevice.h
patching file include/linux/ppp_defs.h
patching file include/linux/rtnetlink.h
patching file include/linux/shim.h
patching file include/linux/socket.h
patching file include/net/ip6_fib.h
patching file include/net/ip_fib.h
patching file include/net/mpls.h
patching file include/net/shim.h
patching file net/bridge/Kconfig
patching file net/bridge/Makefile
patching file net/bridge/mplsbr.c
patching file net/bridge/netfilter/ebt_mpls.c
patching file net/bridge/netfilter/Kconfig
patching file net/bridge/netfilter/Makefile
patching file net/core/dev.c
patching file net/core/Makefile
patching file net/core/shim.c
patching file net/core/shim_procfs.c
patching file net/ipv4/fib_semantics.c
patching file net/ipv4/ip_input.c
patching file net/ipv4/ip_output.c
patching file net/ipv4/Kconfig
patching file net/ipv4/Makefile
patching file net/ipv4/mpls4.c
patching file net/ipv4/netfilter/ipt_mpls.c
patching file net/ipv4/netfilter/Kconfig
patching file net/ipv4/netfilter/Makefile
patching file net/ipv4/route.c
patching file net/ipv6/ip6_output.c
patching file net/ipv6/ipv6_syms.c
patching file net/ipv6/Kconfig
patching file net/ipv6/Makefile
patching file net/ipv6/mpls6.c
patching file net/ipv6/netfilter/ip6t_mpls.c
patching file net/ipv6/netfilter/Kconfig
patching file net/ipv6/netfilter/Makefile
patching file net/ipv6/route.c
patching file net/Kconfig
patching file net/Makefile
patching file net/mpls/af_mpls.c
patching file net/mpls/Makefile
patching file net/mpls/mpls_dst.c
patching file net/mpls/mpls_if.c
patching file net/mpls/mpls_ilm.c
patching file net/mpls/mpls_init.c
patching file net/mpls/mpls_input.c
patching file net/mpls/mpls_instr.c

patching file net/mpls/mpls_netlink.c
patching file net/mpls/mpls_nhlfe.c
patching file net/mpls/mpls_opcode.c
patching file net/mpls/mpls_output.c
patching file net/mpls/mpls_procfs.c
patching file net/mpls/mpls_proto.c
patching file net/mpls/mpls_shim.c
patching file net/mpls/mpls_sysfs.c
patching file net/mpls/mpls_tunnel.c
patching file net/mpls/mpls_utils.c
patching file net/mpls/TODO
patching file net/xfrm/xfrm_shim.c

Ahora con el nuevo parche que se le instalo al kernel, en el menú de configuración del mismo aparecerán nuevas opciones relacionadas con el parche aplicado, por lo tanto procedemos a realizar las configuraciones necesarias (con el comando make menuconfig ARCH=um, o make xconfig ARCH=um en caso de tenerlo instalado), para utilizar los servicios de una red MPLS, para esto es necesario habilitar las siguientes opciones:

- *Networking*
- *Networking options*
  - *(*) Multiprotocol Label Switching*
  - *(*) MPLS: Virtual tunnel interface*
  - *(*) IP: MPLS support*
  - *(*) IP: tunneling*
  - *(*) IP: GRE tunnels over IP*
  - *(*) Network packet filtering framework*
    - *Core Netfilter Configurationtion Netfilter*
      - *NFQUEUE over NFNETLINK interface*
      - *Netfilter LOG over NFNETLINK interface*
      - *(*) Nerfilter connection tracking support*
      - *(*) Connection tracking netlink interface*
      - *(*) Netfilter Xtables support (required for ip_tables)*
      - *(*) "DSCP" and "TOS" target support*
      - *(*) "mpls" target support*
      - *(*) "dscp" and "tos" match support*
    - *IP: Netfilter Configuration*
      - *(*) Ipv4 connection tracking support (required for NAT)*
      - *(*) proc/sysctl compatibility with old connection tracking*
      - *(*) IP Userspace queueing via NETLINK*
      - *(*) IP tables support (required for filtering/masq/NAT)*
      - *(*) "ttl" match support*
      - *(*) Packet filtering*
      - *(*) Full NAT*
      - *(*) MASQUERADE target support*
      - *(*) REDIRECT target support*
      - *(*) NETMAP target support*
      - *(*) Basic SNMP-ALG support*
      - *(*) Packet mangling*
      - *(*) TTL target support*
      - *(*) raw table support*
    - *Bridge: Netfilter Configuration (TODAS LAS OPCIONES)*
  - *(*) 802.1d Ethernet Briging*
  - *(*) Bridge: MPLS support*
  - *(*) 802.q VLAN Support*
  - *(*) QoS and/or fair queueing (TODAS LAS OPCIONES)*

Ahora se procederá a compilar el kernel, para eso se debe de ejecutar los siguientes comandos:

1. make clean ARCH=um (para borrar compilaciones anteriores)
2. make ARCH=um

Luego de esperar unos minutos, vamos a tener compilado finalmente el kernel. En este punto es necesario que se realice una copia de la imagen de un file system (con preferencia de slackware 12.1, que es con el que se desarrollaron las pruebas, en caso contrario los paquetes se pueden convertir a otras distribuciones con el Alien, ver **anexo Alien**) y colocarlo en la carpeta linux-2.6.25.14.
Una vez copiado el File System se iniciará una maquina virtual, para instalar los paquetes necesarios para implementar una red MPLS, y también es necesario actualizar el glibc-2.7-i486-17 (descargado desde: ftp://mirror.pacific.net.au/linux/slackware/slackware-12.1/ slackware/d/gcc-4.2.3-i486-1.tgz).
Para generar una máquina virtual con el File System base que se utilizará debemos ejecutar el siguiente comando:

- ./linux ubd0=uml_fs

Se procederá a instalar los paquetes necesarios para poder realizar todos los experimentos, pero primero debemos desinstalar dos paquetes, ya que estos fueron modificados para el correcto funcionamiento del laboratorio. Para esto primero debemos dirigirnos a la carpeta donde se encuentran instalados todos los paquetes del kernel linux-2.6.25.14, ejecutando el siguiente comando:

- cd /var/log/pakages

Una vez ubicados en esta carpeta, simplemente eliminamos los paquetes que vamos a actualizar, y lo hacemos de la siguiente manera:

- removepkg iptables
- removepkg iproute

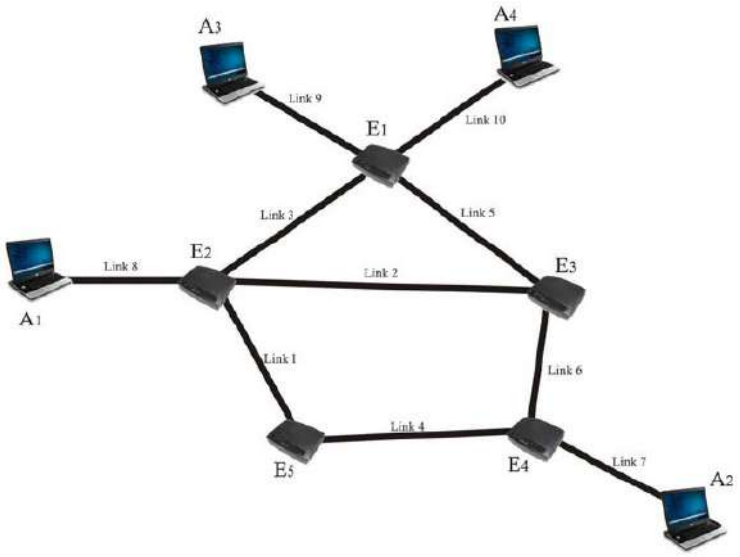
Ya eliminados los paquetes, debemos instalar el resto de los paquetes para poder implementar una red MPLS, es necesario que la instalación se lleve a cabo en el orden que se muestra a continuación, por una cuestión de dependencias. Lo hacemos ejecutando los siguientes comandos:

- installpkg iptables-1.4.1.1.tgz
- installpkg iproute-2.6.26.tgz
- installpkg ebtables-2.0.8.tgz
- installpkg glibc-2.7-i486-17.tgz

Finalizada estas operaciones ya tenemos todo listo para empezar a implementar una red MPLS.
Se creo un script **(anexo rc.local)**, donde en dicho archivo verificamos, en el momento de iniciar cada máquina en particular, cuál es el nombre del archivo .cow, y ejecuta el script conf.sh determinado para esa máquina, el cual configura los IP, la máscara, la interfaz y, dependiendo de cada caso, la salida por defecto.

4- Topología

Se definirá una topología común para todos los experimentos la cual estará formada por 4 PC's y por 5 LSR's, la siguiente imagen muestra como estarán conectados cada componente con sus respectivas interfaces:



Topología

Se creo un script que genera la topologia mostrada en la figura, el cuál genera las máquinas, routers y switch que sean necesarios (Para ejecutar dicho script, debe ubicarse en la carpeta donde se encuentra el archivo y ejecutar el comando ./red-mpls.sh, para ver el script visualizar el **anexo redmpls.sh**).

4.1- Configuración de las PC`s

Para realizar la configuración de las PC vamos a ejecutar un script, al cual llamamos conf.sh, y el que va a contener los comandos de configuración necesarios. Se desarrolló de esta manera la configuración, ya que permite de una manera simple el trabajar con los .cow, ya que al ejecutar el archivo, ubicado en /etc/rc.d/rc.local, este se encargara automáticamente de cargar en cada PC el conf.sh que sea necesario. Para ver los script de configuración de cada máquina ver el **anexo PC´s**.

4.1.1- PC A1

La PC A1 se encuentra conectado al link 8 con las siguientes características:

- *Dirección de red: 172.16.10.0*
- *Dirección IP: 172.16.10.10*
- *Mascara: 255.255.255.0*
- *Salida por defecto: 172.16.10.2*
- *Nombre de la interfaz: eth0*

4.1.2- PC A2

La PC A2 se encuentra conectado al link 7 con las siguientes características:

- *Dirección de red: 172.16.20.0*
- *Dirección IP: 172.16.20.20*
- *Mascara: 255.255.255.0*
- *Salida por defecto: 172.16.20.4*
- *Nombre de la interfaz: eth0*

4.1.3- PC A3

La PC A3 se encuentra conectado al link 9 con las siguientes características:

- *Dirección de red: 172.16.30.0*
- *Dirección IP: 172.16.30.30*
- *Mascara: 255.255.255.0*
- *Salida por defecto: 172.16.30.1*
- *Nombre de la interfaz: eth0*

4.1.4- PC A4

La PC A4 se encuentra conectado al link 10 con las siguientes características:

- *Dirección de red: 172.16.40.0*
- *Dirección IP: 172.16.40.40*
- *Mascara: 255.255.255.0*
- *Salida por defecto: 172.16.40.2*
- *Nombre de la interfaz: eth0*

4.2- Configuración de los Routers

Nuevamente para realizar la configuración de los router se va a ejecutar un script, al cual llamamos conf.sh, y el cual contiene los comandos de configuración necesarios.
Nuevamente tenemos el archivo ubicado en /etc/rc.d/rc.local que ejecuta el conf.sh correspondiente para cada router.
Si prestamos atención no se van a configurar, ni las salidas por defecto, ni redes que estén fuera del alcance del router, ya que esto se encargara mas tarde MPLS. Para ver los script de configuración de cada routers ver el **anexo Routers**.

4.2.1- Router E1

El router E1 se encuentra conectado al link 3, 5, 9 y 10 ahora vamos a mostrar como son las configuraciones en cada uno de ellos.
La configuración en el link 3:

- *Dirección de red: 10.0.3.0*
- *Dirección IP: 10.0.3.1*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth2*

La configuración en el link 5:

- *Dirección de red: 10.0.5.0*
- *Dirección IP: 10.0.5.1*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth3*

La configuración en el link 9:

- *Dirección de red: 172.16.30.0*
- *Dirección IP: 172.16.30.1*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth1*

La configuración en el link 10:

- *Dirección de red: 172.16.40.0*
- *Dirección IP: 172.16.40.1*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth0*

4.2.2- Router E2

El router E2 se encuentra conectado al link 1, 2, 3 y 8 ahora vamos a mostrar como son las configuraciones en cada uno de ellos.

La configuración en el link 1:

- *Dirección de red: 10.0.1.0*
- *Dirección IP: 10.0.1.2*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth3*

La configuración en el link 2:

- *Dirección de red: 10.0.2.0*
- *Dirección IP: 10.0.2.2*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth0*

La configuración en el link 3:

- *Dirección de red: 10.0.3.0*
- *Dirección IP: 10.0.3.2*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth2*

La configuración en el link 8:

- *Dirección de red: 172.16.10.0*
- *Dirección IP: 172.16.10.2*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth1*

4.2.3- Router E3

El router E3 se encuentra conectado al link 2, 3 y 6 ahora vamos a mostrar como son las configuraciones en cada uno de ellos.

La configuración en el link 2:

- *Dirección de red: 10.0.2.0*
- *Dirección IP: 10.0.2.3*
- *Mascara: 255.255.255.0*

- *Nombre de la interfaz: eth0*

La configuración en el link 5:

- *Dirección de red: 10.0.5.0*
- *Dirección IP: 10.0.5.3*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth2*

La configuración en el link 6:

- *Dirección de red: 10.0.6.0*
- *Dirección IP: 10.0.6.3*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth1*

4.2.4- Router E4

El router E4 se encuentra conectado al link 4, 6 y 7, ahora vamos a mostrar como son las configuraciones en cada uno de ellos.

La configuración en el link 4:

- *Dirección de red: 10.0.4.0*
- *Dirección IP: 10.0.4.4*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth0*

La configuración en el link 6:

- *Dirección de red: 10.0.6.0*
- *Dirección IP: 10.0.6.4*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth1*

La configuración en el link 7:

- *Dirección de red: 172.16.20.0*
- *Dirección IP: 172.16.20.4*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth2*

4.2.5- Router E5

El router E5 se encuentra conectado al link 1 y 4 ahora vamos a mostrar como son las configuraciones en cada uno de ellos.

La configuración en el link 1:

- *Dirección de red: 10.0.1.0*
- *Dirección IP: 10.0.1.5*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth1*

La configuración en el link 4:

- *Dirección de red: 10.0.4.0*
- *Dirección IP: 10.0.4.5*
- *Mascara: 255.255.255.0*
- *Nombre de la interfaz: eth0*

5- Comandos Básicos del parche MPLS-Labs

En esta sección se verán todos los comandos básicos del parche MPLS necesarios para realizar los distintos experimentos, que se realizaron. En total se utilizan dos comandos:

- **mpls**
- **ip route**

Cada uno con sus respectivos parámetros y configuraciones.

5.1- Generar el próximo salto

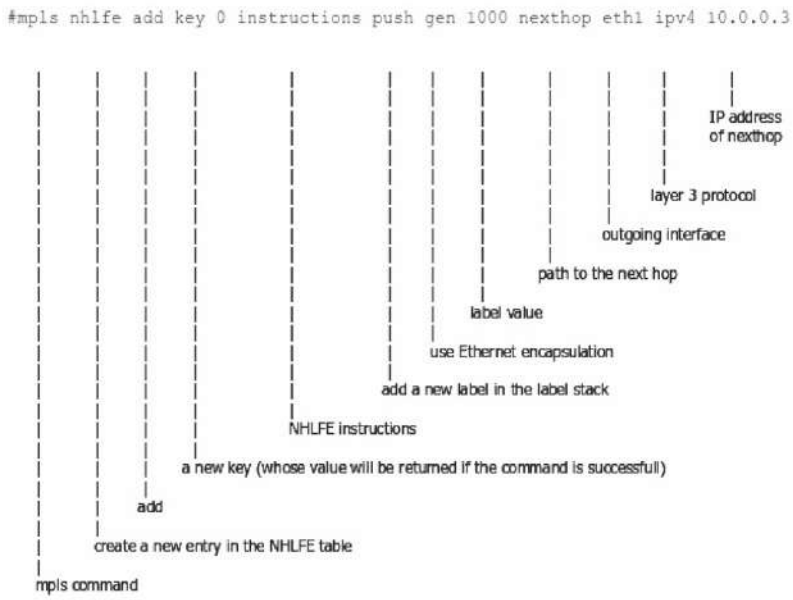
Para poder configurar una red MPLS se utiliza el comando **mpls**. Para indicar el próximo salto (LFIB) que un paquete predeterminado debe realizar, se le indica cual es el IP del router y porque interfaz se debe de realizar el próximo salto, para esto hay que ejecutar el siguiente comando:

- mpls nhlfe add key 0 instructions push gen **(#etiqueta)** nexthop **(interfaz)** ipv4 **(IP)**.

Esta marcado con negrita la información que se debe de indicar:

- **#etiqueta**: es el número de etiqueta entrante.
- **interfaz**: es la interfaz de salida.
- **IP**: dirección IP del próximo salto.

En el siguiente ejemplo se puede ver el comando **mpls**, con todos los parámetros necesarios para que los experimentos funcionen correctamente:



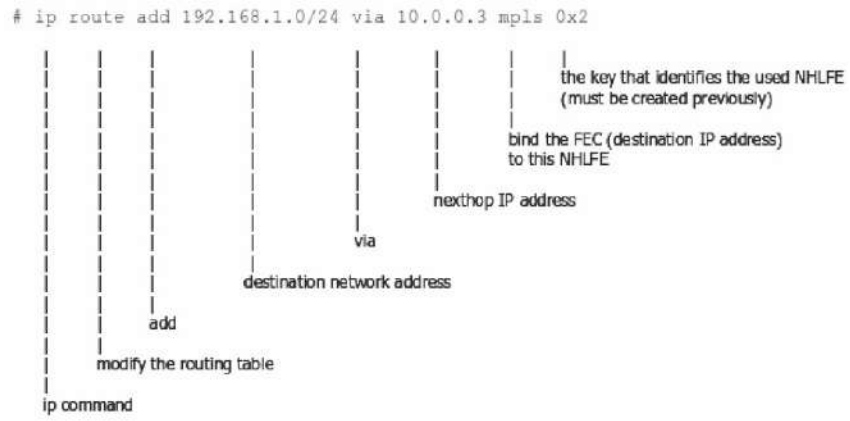
Utilizando el comando anterior, con sus respectivas opciones bien configuradas, ya le indicamos al protocolo cual es el próximo salto, ahora vamos a configurar a una determinada red destino y por cual interfaz debe ser enviada:

- ip route add (**red IP destino**)/(**máscara**) via (**IP próximo salto**) mpls (**key**)

Nuevamente se encuentra marcado con negrita los parámetros que se deben ingresar:

- **red IP destino**: dirección IP de donde proviene el mensaje.
- **máscara**: mascara de la red por la que se enviaran los paquetes.
- **IP del próximo salto**: dirección IP del próximo salto.
- **key**: es un valor que se asigna a esa entrada al utilizar la configuración anterior.

En el siguiente ejemplo se puede ver el comando mencionado con todos los parámetros que utiliza:



5.1.1- Generar el valor key

Este valor es muy importante, ya que se utiliza para poder identificar cual sera la etiqueta que llevara un paquete, sabiendo cuales son las direcciones fuente y destino, esta asignación se realiza en el momento que se utiliza el comando **ip route**, y el valor key se genera cuando se utiliza el comando **mpls**. Por lo tanto utilizando la recomendación del autor del parche esto se realiza de la siguiente manera:

1. var=`**mpls** nhlfe add key 0 instructions push gen 1000 nexthop eth0 ipv4 10.0.2.3 | grep key | cut -c 17-26`
2. **ip route** add 172.16.20.0/24 via 10.0.2.3 mpls $var

Básicamente lo que sucede es que el comando **mpls** retorna el valor de **key**, el cual se lo asigna a la **var**, la cual será utilizada posteriormente en el comando **ip route**.

5.2- Asignación de espacios de etiquetas

Para realizar la asignación del espacio de etiquetas, ya sea a una interfaz o a una etiqueta en particular, se utiliza el comando **mpls**, de la siguiente manera:

- **mpls** labelspace set dev (**interfaz**) labalspace (**X**).
- **mpls** ilm add label gen (**#etiqueta**) labelspace (**X**).

En este caso siempre se deben de ejecutar los dos comandos **mpls**, con sus respectivos parámetros en forma conjunta. Los parámetros que se deben de ingresar en el primer comando son:

- **interfaz**: es la interfaz entrante asociada a un espacio de etiquetas.
- **X**: es el espacio de etiquetas asociada.

Para el segundo comando los parámetros que se deben ingresar son:

- **#etiqueta**: es la etiqueta entrante asociada a un espacio de etiquetas.
- **X**: es el espacio de etiquetas asociada.

En las siguientes imágenes podemos ver a los dos comandos con sus respectivos parámetros:

```
#mpls labelspace set dev eth0 labelspace 0

                                                  labelspace value (signifies per platform labelspace)
                                            keyword
                                      incoming interface for MPLS packets
                                keyword
                          set interfaces to a specific labelspace
                    keyword
mpls command
```

```
#mpls ilm add label gen 1000 labelspace 0

                                                  labelspace value
                                            keyword
                                      incoming label value
                                Ethernet encapsulation
                          incoming label
                    add
              associate an ILM with a labelspace
mpls command
```

5.3- Intercambio de etiquetas

Para realizar el intercambio de etiquetas en los LSR, se utiliza el comando **mpls**, de la siguiente forma:

- mpls xc add ilm_label gen (**#etiqueta entrada**) ilm_labelspace (**X**) nhlfe_key (**key**)

Los parámetros que se deben ingresar para realizar esta configuración son:

- **#etiqueta entrada**: indica el valor de la etiqueta entrante.
- **X**: es el espacio de etiquetas asociada.
- **key**: que indica la referencia de la etiqueta de salida, la interfaz y el IP del próximo salto. Este valor se extrae como se mostró en la *sección 5.1.1***.**

En la siguiente imagen se puede ver mas detallado este comando:

```
#mpls xc add ilm_label gen 1000 ilm_labelspace 0 nhlfe_key 0x2

                                                  the NHLFE key used
                                            link to a NHLFE
                                      labelspace value
                                keyword
                          incoming label value
                    Ethernet encapsulation
              specify the incoming label
        add
  label switching (eXChange)
mpls command
```

6- Distribución de etiquetas

Para manejar la configuración del espacio de etiquetas, es necesario asignar un valor al parámetro **X** que se utiliza en el comando **mpls**, esta configuración es independiente para cada router y va a depender del espacio asociado a cada etiqueta en cada router en particular.
Al manejar los espacios de etiquetas nos va a permitir realizar los dos tipos de distribuciones de etiquetas que existen:

- por plataforma.
- por interfaz

6.1- Distribución de etiquetas por plataforma

Para poder realizar una distribución de etiquetas por plataforma, es necesario, poner un mismo valor al espacio de etiquetas a cada par de comandos **mpls**.
Por ejemplo, si ingresamos a la LFIB el valor de dos o más etiquetas que ingresan por alguna de las interfaces, y este valor **X** es igual en todos los ingresos a la tabla con distintos valores de etiquetas, entonces, significa que se esta configurando el LSR que trabaje con una distribución de etiquetas por plataforma.
Por ejemplo si en un mismo router realizamos las siguientes configuraciones:

- Configuración de la salida por eth0:
  - mpls labelspace set dev eth0 labelspace 0
  - mpls ilm add label gen 1000 labelspace 0
- Configuración de la salida por eth1:
  - mpls labelspace set dev eth1 labelspace 0
  - mpls ilm add label gen 2000 labelspace 0

En los comandos utilizados, se puede ver como las dos entradas a la tabla manejan el mismo espacio de etiquetas (en este caso el valor cero), pero a su vez, mapean distintas etiquetas de entradas dependiendo de la interfaz por donde ingresen al router. Es decir una vez que un paquete ingresa a este LSR si el paquete sale por eth0 se le asigna la etiqueta 1000, y si sale por eth1 se le asigna la etiqueta 2000. En este caso el valor de las etiquetes deben de ser diferentes, para toda aquella configuración donde el espacio de etiquetas sea igual a cero.

6.2- Distribución de etiquetas por interfaz

Para poder realizar una configuración del LSR por interfaz, es decir que para saber el próximo salto el router va a mapear tanto por la etiqueta, como por la interfaz de entrada.
Por ejemplo, si en un mismo router realizamos la siguiente configuraciones:

- Configuración de la salida por eth0:
  - mpls labelspace set dev eth0 labelspace 0
  - mpls ilm add label gen 1000 labelspace 0
- Configuración de la salida por eth1:
  - mpls labelspace set dev eth1 labelspace 1
  - mpls ilm add label gen 1000 labelspace 1

En los comandos anteriores se puede ver que existen dos etiquetas iguales, pero una es enviada por un interfaz eth0 y la otra por eth1. Un paquete ingresa a este LSR no importa porque interfaz se envíe siempre se asignará la etiqueta 1000. En este caso se puede repetir el valor de las etiquetas, ya

que el valor asignado al espacio de etiquetas es diferente.

7- Apilamiento de etiquetas

El parche de laboratorio de MPLS, nos permite realizar apilamiento de etiquetas, para realizarlo es necesario ejecutar las siguientes operaciones en el siguiente orden:

1. mpls nhlfe add key 0 instructions push gen (**#etiqueta1**) nexthop (**interfaz**) ipv4 (**IP**).
2. mpls nhlfe add key 0 instructions push gen (**#etiqueta2**) forward (**ID1**).
3. mpls xc add ilm_label gen (**#etiqueta de entrada**) ilm_labelspace (**X**) nhlfe_key (**ID2**).

Los parámetros que se deben ingresar para realizar esta configuración son:

- **#etiqueta1**: es el valor de la etiqueta que quedará en el fondo de la pila.
- **#etiqueta2**: es el valor de la etiqueta tope.
- **#etiqueta de entrada**: es el valor de la etiqueta de entrada.
- **interfaz**: se le asigna la interfaz de salida.
- **X**: es el espacio de etiquetas asociada.

Para obtener los valores de **ID1** e **ID2** se realizará de la misma manera que en la *sección 5.1.1*. El **ID1** se obtiene del primer comando y el **ID2** del segundo comando.
Ahora vamos a ver como se realiza la configuración para poder sacar la etiqueta tope, y cambiarla por la del próximo salto, la configuración es la siguiente:

1. mpls ilm add label gen (**#etiqueta_tope**) labelspace (**X**).
2. mpls ilm add label gen (**#etiqueta_a_intercambiar**) labelspace (**X**).
3. mpls nhlfe add key 0 instructions push gen (**#etiqueta entrada**) nexthop (**interfaz**) ipv4 (**IP**).
4. mpls xc add ilm_label gen (**#etiqueta nueva**) ilm_labelspace 0 nhlfe_key (**key**).

En el primer comando se quita la etiqueta tope, esta operación se puede realizar todas las veces que sean necesarias para que el router pueda desapilar todas las etiquetas que sean necesarias. En los pasos posteriores se encarga de realizar el intercambio de la etiqueta, y enviarla por la interfaz correspondiente.
Se realizó una captura con el comando tcpdump para visualizar el anidamiento de las etiquetas y luego se imprimió el resultado utilizando el *wireshark*, se obtuvieron los siguientes resultados:

Frame 4 (106 bytes on wire, 96 bytes captured)
Ethernet II, Src: fe:25:d3:0a:88:1f (fe:25:d3:0a:88:1f), Dst: ce:2e:2d:36:ce:ac (ce:2e:2d:36:ce:ac)
MultiProtocol Label Switching Header, Label: 2000, Exp: 0, S: 0, TTL: 62
MPLS Label: 2000
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 0
MPLS TTL: 62
MultiProtocol Label Switching Header, Label: 200, Exp: 0, S: 1, TTL: 62
MPLS Label: 200
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 62
Internet Protocol, Src: 172.16.10.10 (172.16.10.10), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol

En esta captura se puede ver con color verde la etiqueta tope, y con color rojo la etiqueta inferior, la cual tiene un valor igual a 1 en el campo Botton Of Label Stack, también se puede ver que el valor de la etiqueta tope es 2000, y el valor de la etiqueta inferior es 200.

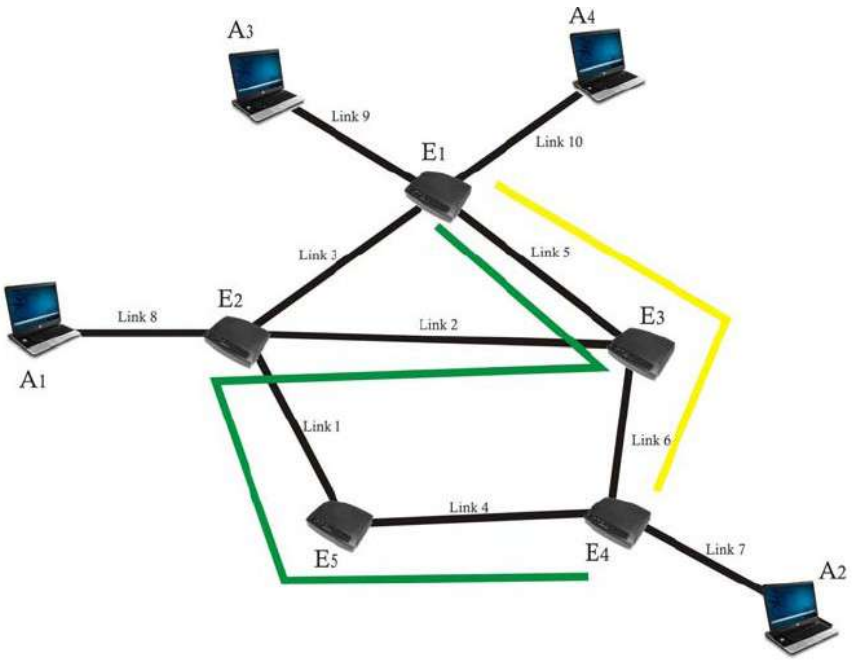
8- Ingeniería de tráfico

Como mencionamos en un principio, se realizaron algunos ejercicios de Ingeniería de Tráfico:

- **Protección de los Links**, en este experimento se tratará de resolver el problema que surge cuando se cae un enlace, de esta manera se tiene que utilizar un LSP secundario, en la cual solo se utilizará cuando el LSP principal caiga.
- **Protección de Nodos** que es muy similar a la anterior, pero en este caso no solo existe la posibilidad de que caiga el enlace, sino que también de que caiga el nodo.
- **Balanceo de carga** donde se definirá como clasificar el tráfico en diferentes LSP para el mismo destino, por medio de diferentes rutas físicas.
- **QoS basado en la Ingeniería de Tráfico**, donde se realizará un filtrado por el tamaño del paquete que se enviará. Dicho control se realiza debido a que, si supera este límite, la performance de la red decae. Por lo tanto se dispone de dos LSP (uno principal LSPp y otro secundario LSPs), para mantener la calidad de servicio por el LSPp.

8.1- Protección de link

En este experimento lo que se va a probar que es lo que sucede cuando se cae un enlace, y como hacemos para solucionar este inconveniente. Una solución es enviar el tráfico por otro LSP opcional. Para realizar esta tarea, se construirá un LSP secundario, en la cual solo será utilizada en el caso de que el LSP principal caiga. Esta solución es utilizada para proteger los enlaces muy importantes, donde la caída de la conexión pueda afectar a un gran número de usuarios o en los casos en que la información que circula por el LSP principal es critica.
En otras palabras el LSP secundario se utiliza inmediatamente luego de que el LSP principal cae, provocando solo la pérdida de unos pocos paquetes. Y cuando el LSP principal se recupera la información es enviada nuevamente, por su camino original.
La topología que se utilizara es la siguiente:



Se puede ver con color amarillo el **LSPp** (LSP principal) y con color verde el **LSPs** (LSP secundario).

Para realizar la configuración de este experimento debemos ejecutar el script ubicado en */home/router/te_link*, dentro de esta carpeta se encontrará un archivo el cual ejecutaremos individualmente en cada router, esto automáticamente realizará la configuración para correr el ejercicio.
Se hará un especial enfoque en la configuración de los LSR E4 y LSR E3, ya que este en el punto crítico del experimento, el resto de los LSR se configurarán como se explico anteriormente, y se crearon los script necesarios para su configuración.
La configuración para el LSRp es:

1. var_default=`mpls nhlfe add key 0 instructions push gen (#etiqueta) nexthop (**interfaz_p**) ipv4 (IP próximo salto_p) |grep key | cut -c 17-26`
2. ip route add (red IP destino)/(mascara) via (**IP próximo salto_p**) mpls $var_default

Ahora veamos la configuración del LSPs:

1. var_default=`mpls nhlfe add key 0 instructions push gen (#etiqueta) nexthop (**interfaz_s**) ipv4 (IP próximo salto_s) |grep key | cut -c 17-26`
2. ip route add (red IP destino)/(mascara) via (**IP próximo salto_s**) mpls $var_default

Estas configuraciones se diferencian en los parámetros que indican la interfaz, y el IP del próximo salto, esta configuración va a variar dependiendo de los próximos saltos elegidos, para armar el camino principal y secundario. En nuestro experimento ambos comandos son ejecutados al comienzo de los script de los LSR E3 y LSR E4.
Ya tenemos definido cuales serán los próximos saltos en el caso de que el enlace caiga, ahora se utilizara, primeramente, para controlar el enlace el código aportado por los creadores del laboratorio MPLS, al cual luego se le realizaran unas modificaciones para realizar un control mas exhaustivo del enlace, y garantizar el envío de paquetes. A continuación, se muestra el código extraído de los autores del laboratorio MPLS:

```
default_route=1
while [ "1" = "1" ] ; do
    sleep 0.5
    status=`ethtool eth1 | grep "Link detected" | grep yes`
    if [ -z "$status" ] ; then
        if [ $default_route = 1 ] ; then
            default_route=0 ;
            ip route del 172.16.30.0/24 via 10.0.6.3 mpls $var_default
            ip route add 172.16.30.0/24 via 10.0.4.5 mpls $var_indirect
            date +%r
            echo "deleting default route"
            echo "adding backup route"
        fi
    fi
    if [ ! -z "$status" ] ; then
        if [ $default_route = 0 ] ; then
            default_route=1 ;
            ip route del 172.16.30.0/24 via 10.0.4.5 mpls $var_indirect
            ip route add 172.16.30.0/24 via 10.0.6.3 mpls $var_default
            date +%r
            echo "deleting indirect route"
            echo "adding default route"
        fi
    fi
done
```

El funcionamiento del código es el siguiente:

1. Realiza un while infinito (al comparar 1=1).
2. Utiliza una bandera la cual es llamada *default_route*, que es seteada en un principio con el valor 1.

3. Se utiliza el comando sleep, que es configurado en 0,5 segundos (este valor puede ser cambiado para realizar más o menos verificaciones por segundos, esto va a depender, de la confiabilidad del enlace).
4. En el caso de que ingrese a uno de los dos if, se remueve el enlace de la tabla de ruteo al LSPp (LSR E3), e ingresa un enlace al LSPs (LSR E5), y viceversa.

Un código muy similar se puede encontrar en el LSR E3, solo que cambian los valores de las direcciones IP, que en este caso apuntan hacia el LSR E4 (LSRp) o el LSR E2 (LSRs).
Este código fue modificado, ya que presentaba algunos problemas para detectar la caída del link. Esta falla se encontraba cuando se utilizaba el comando ***ethool***, dicho comando verifica si su propia interfaz se encuentra activa, y no si la conexión entre ambos LSR. Esto se debe, a que en nuestro laboratorio UML, contamos con un switch virtual, para conectar los dos LSR virtuales, y no un enlace punto a punto, en el caso de que se utilizara este tipo de enlaces seria posible detectar la caída del link con este comando, pero en nuestro caso si se ejecuta el comando ***ifconfig eth1 down***, en le LSR E4, el LSR E3 no detecta la caída de la conexión ya que este esta conectado directamente a un switch virtual, y no directamente al LSR E4. Para poder solucionar este problema lo que se realiza en la nueva versión del código, es cambiar el comando ***ethool***, por el comando ***ping***, con la opción ***-c1*** la cual nos garantiza de que se enviará un solo ping hacia el LSR E3, si nos encontramos en el LSR E4, por medio de un pipe, utilizamos el comando ***greep*** con el cual buscamos que el LSR E3 respondió correctamente, y luego un último pipe, se utiliza el comando ***wc*** con la opción ***-l*** para contar la cantidad de linea que se encontraron, (debido a que anteriormente se limito la cantidad de ping a uno el valor siempre va a variar entre 1 y 0). En el caso de que el valor sea 0, nos va a indicar que se cayo el enlace, y en caso contrario nos indica que el enlace se encuentra estable. A continuación se mostrara como quedan planteados las modificaciones realizadas al código en los dos LSR:

- status='ping -c1 10.0.6.3 | grep 'from 10.0.6.3' | wc -l" (en el LSRe4)
- status='ping -c1 10.0.6.4 | grep 'from 10.0.6.4' | wc -l" (en el LSRe3)

Como se cambió el valor de la variable *status*, el control de los ***if***, tambien deben de ser modificados, por lo tanto quedaron de la siguiente manera:

- if [$status != 1] (el primer if).
- if [$status = 1] (el segundo if).

La siguiente modificación surge de las pruebas realizadas, ya que al ejecutar el comando ***ifconfig eth1 down***, en el LSR E3, se obtiene como resultado que esa interfaz cae, y se eliminan las etiquetas asociadas con esta interfaz, por lo tanto, en el segundo *if* que controla el estado del enlace, se agregó la siguiente linea:

- var_default=`mpls nhlfe add key 0 instructions push gen 600 nexthop eth1 ipv4 10.0.6.4 | grep key | cut -c 17-26`

Luego de realizar estas observaciones, el código que se utiliza para controlar el enlace en el LSR E4, es el siguiente:

```
default_route=1
while [ "1" = "1" ] ; do
   sleep 2
   status=`ping -c1 10.0.6.3 | grep 'from 10.0.6.3' | wc -l`
   if [ $status != 1 ] ; then
      if [ $default_route = 1 ] ; then
         default_route=0 ;
         ip route del 172.16.30.0/24 via 10.0.6.3 mpls $var_default
         ip route add 172.16.30.0/24 via 10.0.4.5 mpls $var_indirect
         date +%r
         echo "deleting default route"
```

```
            echo "adding backup route"
        fi
    fi
    if [ $status = 1 ] ; then
        if [ $default_route = 0 ] ; then
            #need to switch to default route
            default_route=1 ;
            ip route del 172.16.30.0/24 via 10.0.4.5 mpls $var_indirect
            ip route add 172.16.30.0/24 via 10.0.6.3 mpls $var_default
            date +%r
            echo "deleting indirect route"
            echo "adding default route"
        fi
    fi
done
```

El código fina que se utilizará en el LSR E3 es le siguiente:

```
default_route=1;
while [ "1" = "1" ]; do
    sleep 2
    status=`ping -c1 10.0.6.4 | grep 'from 10.0.6.4' | wc -l`
    if [ $status != 1 ]; then
      if [ $default_route = 1 ]; then
        default_route=0;
        mpls xc del ilm_label gen 500 ilm_labelspace 0 nhlfe_key $var_default
        mpls xc add ilm_label gen 500 ilm_labelspace 0 nhlfe_key $var_indirect
        date +%r
        echo "deleting default route"
        echo "adding backup route"
      fi
    fi
    if [ $status = 1 ]; then
      if [ $default_route = 0 ]; then
        default_route=1;
        var_default=`mpls nhlfe add key 0 instructions push gen 600 nexthop eth1 ipv4 10.0.6.4 | grep key | cut -c 17-26`
        mpls xc del ilm_label gen 500 ilm_labelspace 0 nhlfe_key $var_indirect
        mpls xc add ilm_label gen 500 ilm_labelspace 0 nhlfe_key $var_default
        date +%r
        echo "deleting backup route"
        echo "adding default route"
      fi
    fi
done
```

8.1.1- Capturas realizadas

En esta sección se mostraran las capturas realizadas del experimento con el comando **tcpdump**, todas las capturas realizadas se realizarán en el LSR E4 con la opción **-i any** para ver todas las interfaces involucradas en el envio de paquetes, y se realizará una breve explicación de lo que está sucediendo.
Para ver mejor los paquetes enviados se pintarán el fondo de amarillo para los paquetes del LSPp, de color del fondo verde para los paquetes del LSPs y con color celeste es la comunicación entre el LSR E4 y la PC A2.
Para poder visualizar mejor los diferentes paquetes, voy a mencionar primeramente las direcciones IP y MAC de cada interfaz:

- MAC del LSR E4:

    - eth0: **IP: 10.0.4.4, MAC: 0a:9a:58:86:82:30**
    - eth1: **IP: 10.0.6.4, MAC: 9a:eb:01:37:4c:b3**
- MAC del LSR E3:
    - eth1: **IP: 10.0.6.3, MAC: fa:49:37:66:05:18**
- MAC del LSR E5:
    - eth0: **IP: 10.0.4.5, MAC: 1e:4e:2c:2f:51:49**

Primero veremos la captura de los ping que se realizan entre el LSR E4 y el LSR E3, para controlar que la conexión del LSPp se encuentra activa:

Frame 1 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
*Source: **fa:49:37:66:05:18** (fa:49:37:66:05:18)*
Protocol: IP (0x0800)
Internet Protocol, Src: 10.0.6.3 (10.0.6.3), Dst: 10.0.6.4 (10.0.6.4)
Internet Control Message Protocol
Ping enviado desde el LSRe3:
Frame 2 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
*Source: **9a:eb:01:37:4c:b3** (9a:eb:01:37:4c:b3)*
Protocol: IP (0x0800)
Internet Protocol, Src: 10.0.6.4 (10.0.6.4), Dst: 10.0.6.3 (10.0.6.3)
Internet Control Message Protocol

En la siguiente captura, se realizara un **ping** desde la PC2 y la PC3, y realizaré una captura desde el LSR E4 con el comando **tcpdump**, utilizando la opción **-i any**, para poder visualizar todas las interfaces:

Frame 19 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
Source: 6e:81:12:01:c1:ea (6e:81:12:01:c1:ea)
Protocol: IP (0x0800)
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol
Frame 20 (104 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
*Source: **9a:eb:01:37:4c:b3** (9a:eb:01:37:4c:b3)*
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 100, Exp: 0, S: 1, TTL: 63
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol
Frame 21 (104 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
*Source: **fa:49:37:66:05:18** (fa:49:37:66:05:18)*
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 600, Exp: 0, S: 1, TTL: 62

Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol
Frame 22 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
Source: 0e:6e:69:9c:7a:98 (0e:6e:69:9c:7a:98)
Protocol: IP (0x0800)
Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol

En la captura anterior podemos ver en negrita las direcciones MAC, tanto del LSR E3 como la del LSR E4, también se puede visualizar, las etiquetas MPLS utilizadas para cada paquete, en el Frame 20, se ve la etiqueta 100, que es utilizada para los mensajes enviados desde el LSR E4 al LSR E3, y en el Frame 21, se puede ver la etiqueta 600, que es utilizada para los mensajes enviados desde el LSR E3 hacia el LSR E4. Y finalmente podemos ver con fondo azul, la comunicación entre el LSR E4 y la PC2. Esta serie de 4 Frame se repetirá en el tiempo con los mismos parámetros siempre y cuando el LSPp este activo.
En esta última captura, se simulará un corte en el enlace entre el LSR E3 Y el LSR E4, por lo tanto el LSPp cae y comienza a transferir los paquetes por el LSPs, comunicándose el LSR E4 con el LSR E5:

Frame 139 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
Source: 6e:81:12:01:c1:ea (6e:81:12:01:c1:ea)
Protocol: IP (0x0800)
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol
Frame 140 (104 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
*Source: **0a:9a:58:86:82:30** (0a:9a:58:86:82:30)*
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 100, Exp: 0, S: 1, TTL: 63
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol
Frame 141 (104 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
*Source: **1e:4e:2c:2f:51:49** (1e:4e:2c:2f:51:49)*
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 600, Exp: 0, S: 1, TTL: 60
Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol
Frame 142 (100 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
Source: 0e:6e:69:9c:7a:98 (0e:6e:69:9c:7a:98)
Protocol: IP (0x0800)
Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol

En esta serie de mensajes se puede ver como las etiquetas MPLS siguen siendo las mismas, pero las direcciones MAC, cambian comunicando el LSR E4 con el LSR E5.
Una observación final que se puede hacer utilizando las dos últimas capturas realizadas, es la independencia de capas en el protocolo, se subrayo en todos los paquetes el IP destinos y el IP fuente de los mensajes, y se puede ver que siempre es el mismo para ir de la PC2 hacia la PC3, y viceversa, solo se agrega en el momento que el paquete ingresa a la nube MPLS, dos lineas que indica que se utiliza este protocolo, y una linea donde se indica el valor de la etiqueta, el TTL, el valor EXP y S, que corresponden a la cabecera de MPLS, que se vio en el primer informe.

8.2- Protección de nodo

En el siguiente experimento se realizará una protección de nodo, la cual es básicamente es similar al anterior, pero en este caso no solo existe la posibilidad de que caiga el enlace, sino que también existe la posibilidad de que caiga el LSR. Por lo tanto las configuraciones a realizar son muy similares que en el ejemplo anterior, ya que el LSR E3 y el LSR E4 se configuran de la misma manera para que realicen una protección sobre el enlace que los comunica. Por lo tanto vamos a centrarnos en la configuración de LSR E2 y el LSR E1, que deben de verificar si el LSR E3 se mantiene en funcionamiento.
Para realizar esta configuración debemos ejecutar el script ubicado en */home/router/te_nodo*, dentro de esta carpeta se encontrará un archivo el cual ejecutaremos individualmente en cada LSR, esto automáticamente realizará la configuración para correr el experimento.
A continuación se mostrarán las configuraciones en el LSR E2 y LSR E1, los cuales tienen una configuración similar que el ejercicio anterior.
Este es el correspondiente al LSR E1:

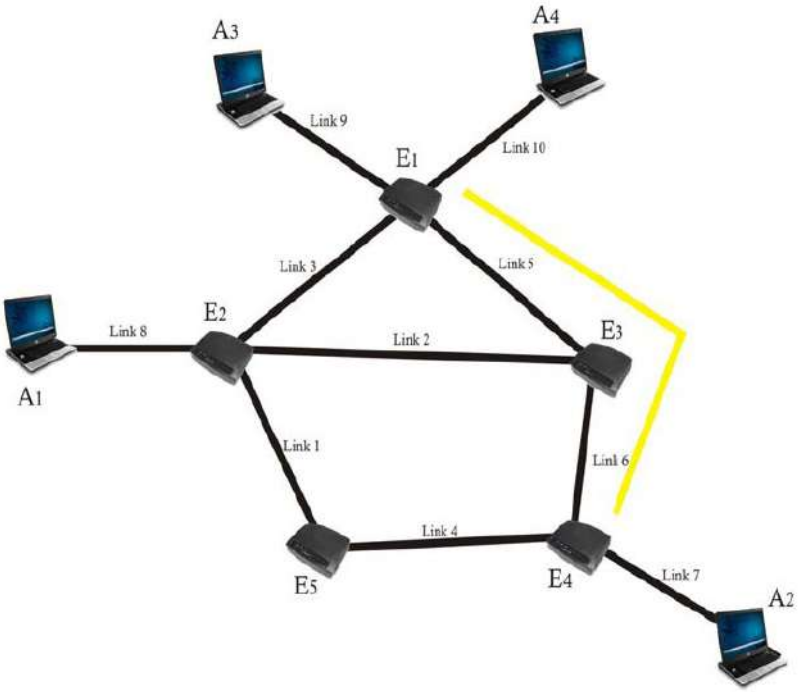
```
default_route=1
while [ "1" = "1" ] ; do
sleep 2
status=`ping -c1 10.0.5.3 | grep 'from 10.0.5.3' | wc -l`
if [ $status != 1 ] ; then
   if [ $default_route = 1 ] ; then
      default_route=0
      ip route del 172.16.20.0/24 via 10.0.5.3 mpls $var_best
      ip route add 172.16.20.0/24 via 10.0.3.2 mpls $var_low
      date +%r
      echo "LSRe3 is down"
   fi
fi
if [ $status = 1 ] ; then
   if [ $default_route = 0 ] ; then
      default_route=1
      ip route del 172.16.20.0/24 via 10.0.3.2 mpls $var_low
      ip route add 172.16.20.0/24 via 10.0.5.3 mpls $var_best
      date +%r
      echo "LSRe3 is up"
   fi
fi
done
```

Y el código en el LSR E2 es el siguiente:

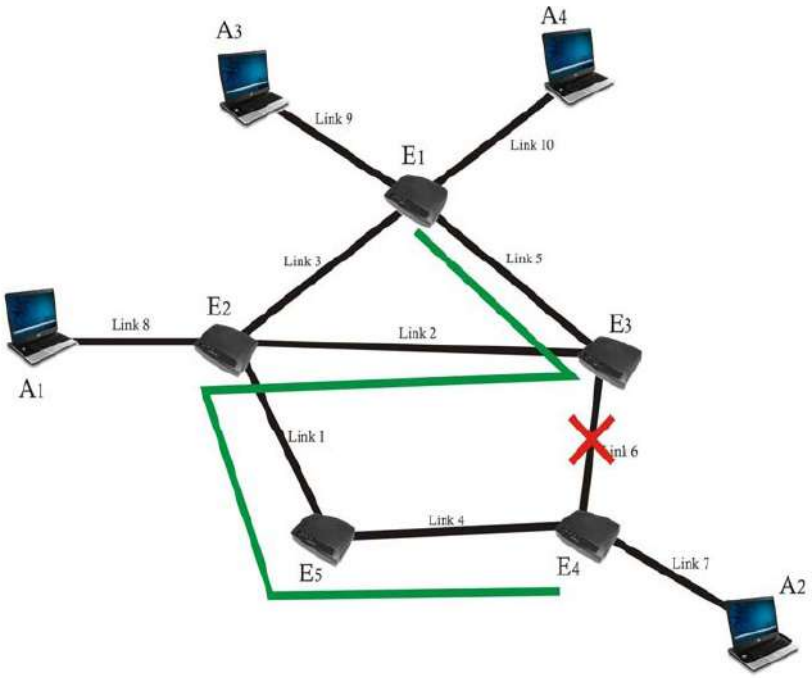
```
default_route=1
while [ "1" = "1" ] ; do
sleep 2
status=`ping -c1 10.0.2.3 | grep 'from 10.0.2.3' | wc -l`
if [ $status != 1 ] ; then
   if [ $default_route = 1 ] ; then
```

```
        default_route=0 ;
        mpls xc del ilm_label gen 100 ilm_labelspace 0 nhlfe_key $var_medium
        mpls xc add ilm_label gen 100 ilm_labelspace 0 nhlfe_key $var_low
        date +%r
        echo "LSRe3 is down"
    fi
fi
if [ $status = 1 ] ; then
    if [ $default_route = 0 ] ; then
        default_route=1 ;
        mpls xc del ilm_label gen 100 ilm_labelspace 0 nhlfe_key $var_low
        mpls xc add ilm_label gen 100 ilm_labelspace 0 nhlfe_key $var_medium
        date +%r
        echo "LSRe3 is up"
    fi
fi
done
```

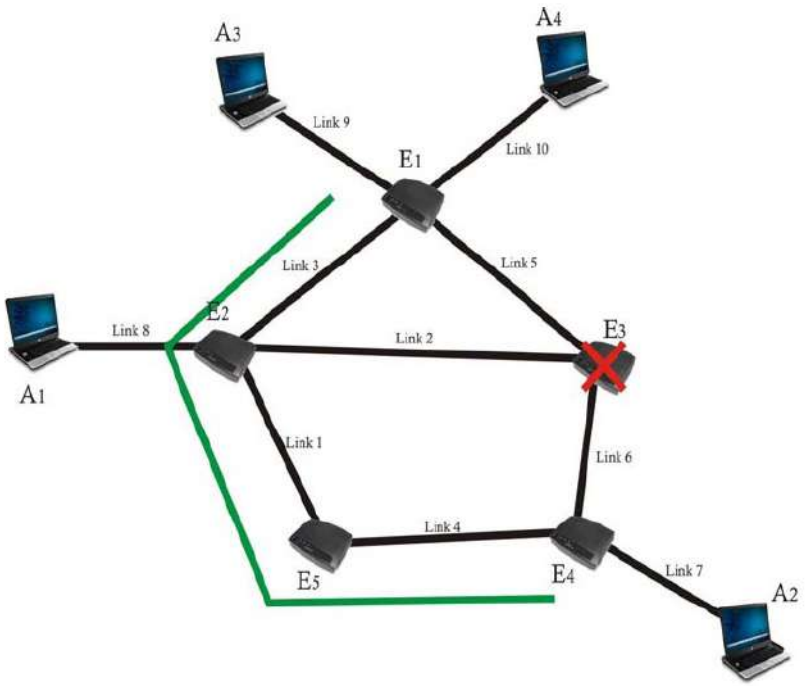
Se puede notar que estos últimos dos códigos, son muy similares a los utilizados en el apartado anterior, esto es cierto, solo que ahora se controla que el LSR E3 se encuentre operativo, para poder asegurar el envío de paquetes, realizando un control de todas sus interfaces.
En este caso no se realizaron capturas, ya que son muy similares a las explicadas en el apartado anterior, solo que se genera un nivel complicación ya que ahora son tres los routers los que cambian su interfaz para enviar los paquetes. Por lo tanto se realizará una explicación mediante imágenes, donde se va a poder visualizar de una manera más clara el experimento del apartado 8.1 y 8.2.
En la siguiente imagen se puede ver como funcionaría el servicio de no presentarse ningún inconveniente.



En la siguiente imagen se puede ver lo que sucede se el *Link 6* cae, este seria el caso planteado en el apartado 8.1.



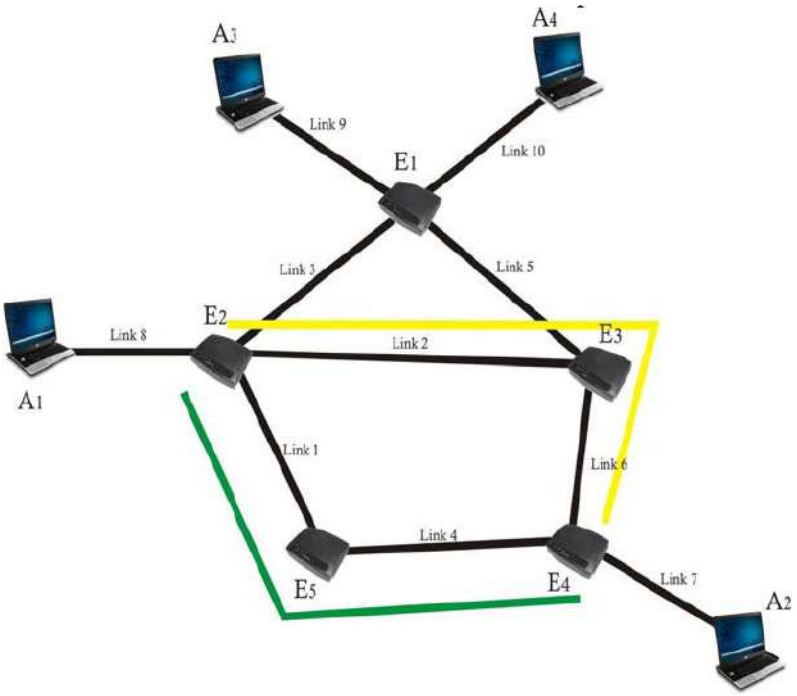
Y finalmente en esta ultima imagen se plantea el caso en el que el LSR E3 cae, por lo tanto se genera un tercer LSP, para evitar la perdida de paquetes.



8.3- Balanceo de carga

En este experimento vamos a ver como clasificar el tráfico y enviarlo por medio de diferentes rutas físicas para un mismo destino.
En este caso los paquetes de un mismo flujo siguen el mismo camino, pero estos pueden utilizar diferentes caminos dependiendo de un filtro, en nuestro caso, nuestro filtro se va a centrar en el

campo DSCP del paquete IP que se desea enviar entre la PC A1 y la PC A2.
A continuación se muestra el escenario sobre el cual se basa el experimento:



En la topología anterior podemos ver como vamos a tomar los dos LSP, uno es el verde y el otro el amarillo. El LSR va a enviar el paquete por el LSP amarillo siempre que la PC A1 envíe un paquete con un valor de 10 en el campo DSCP, y va a recorrer el LSP verde cuando envíe un paquete con el valor de 30 en el campo DSCP.
Para realizar esta configuración debemos ejecutar el script ubicado en */home/router/balanceo* en cada uno de los routers y en la PC A1 ejecutar el script que se encuentra en */home/a1/balanceo*, esto automáticamente realizará la configuración para correr el ejercicio.
Para poder generar el tráfico necesario se utilizará un generador de tráfico, el mgen 4.0x5, para crear tráfico desde la PC A1 hacia la PC A2.
Ahora veremos como configurar la PC A1 para indicar en su tabla IP los dos tipos de paquetes que van a ser enviados con distinta calidad en el campo DSCP. Para esto debemos ejecutar las siguientes líneas de código:

1. *iptables -t mangle -F*
2. *iptables -t mangle -A OUTPUT -p udp --sport 13900 -j DSCP --set-dscp 10*
3. *iptables -t mangle -A OUTPUT -p udp --sport 13902 -j DSCP --set-dscp 30*

Como se puede ver en estas líneas se configuran como salida (OUTPUT), en dos puertos diferentes (sport), y con dos calidades distintas (10 y 30).
Para finalizar con la configuración de la PC A1 se debe crear un archivo donde el nombre finalice con .mgn, el cual va a ser utilizado por el generador de tráfico para poder enviar los mensajes. Este archivo contendrá las siguientes líneas de código:

- *0.0 ON 1 UDP SRC 13900 DST 172.16.20.20/13901 PERIODIC [100 1024]*
- *0.0 ON 2 UDP SRC 13902 DST 172.16.20.20/13903 PERIODIC [100 1024]*

En estas dos líneas se puede observar que se indica por medio de que puerto enviar el paquete (13901 y 13903), y a que IP va dirigido el paquete.
En este experimento el único LSR que hay que configurar para esta experiencia es el LSR E2, ya que este es el encargado de multiplexar la información y enviar el paquete, dependiendo del valor en el campo DSCP, por el LSR E3 o por el LSR E5.

Primero vamos a mostrar en forma genérica como se debe configurar cada LSP:

1. *key1=`mpls nhlfe add key 0 instructions push gen* ***(#etiqueta)*** *nexthop* ***(interfaz)*** *ipv4* ***(IP próximo salto)*** *|grep key |cut -c 17-26`*
2. *iptables -A FORWARD -m dscp --dscp* ***(QoS)*** *-j mpls --nhlfe $key1*
3. *ip route add* ***(red IP destino)/(mascara)*** *via* ***(IP próximo salto)*** *mpls $key1*

Estas líneas corresponde a la configuración de uno de los LSP, por lo tanto debemos de realizar nuevamente estas líneas dependiendo de la cantidad de LSP que queramos configurar.
La diferencia entre esta configuración y la que se venía utilizando, es que se le agrega un comando intermedio, en el cual se genera una entrada en la tabla IP, donde mapea una calidad de servicio, con un LSP ($key1).
En este experimento el LSR E2, clasifica los paquetes y los envía por una interfaz o por la otra. Los LSR E3 y LSR E5 lo único que hacen es recibir el paquete, intercambiar la etiqueta y reenviar el paquete hacia el próximo salto. El LSR E4 recibe los paquetes etiquetados y quita la etiqueta del mismo y rutea el paquete hacia la PC A2.

8.3.1- Capturas realizadas

En esta sección se mostraran una serie de capturas realizadas en el LSR E2, para poder visualizar mejor las capturas se pintará de color amarillo los paquetes con DSCP 10 y con verde los paquetes con DSCP 30, y se podrá observar como en el primer caso el paquete se etiqueta con el valor 1000, y el segundo con la etiqueta 2000.

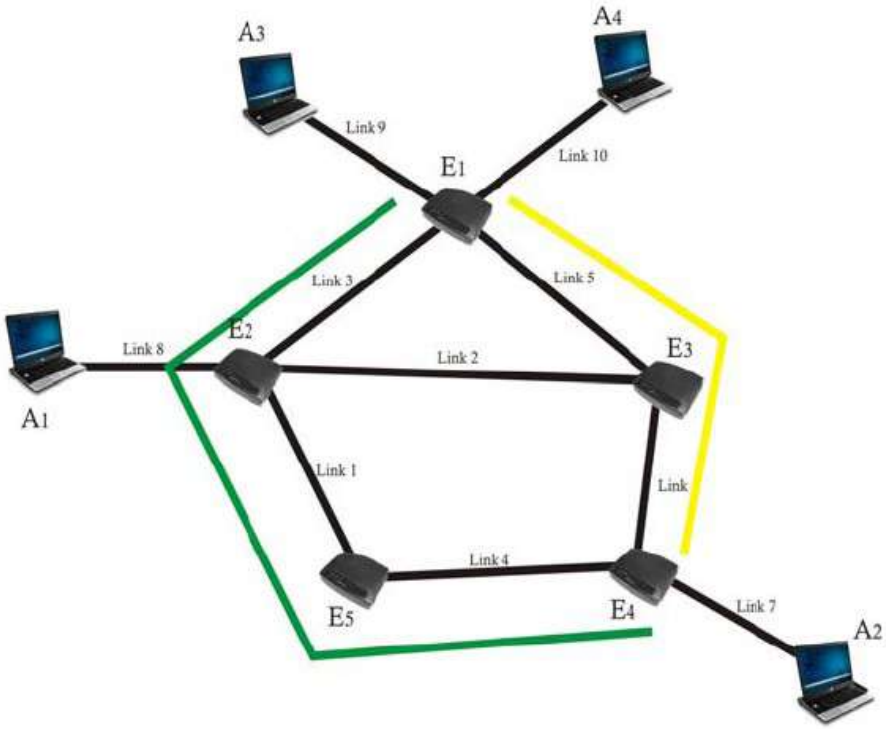
Frame 1 (1068 bytes on wire, 96 bytes captured)
Linux cooked capture
Internet Protocol, Src: 172.16.10.10 (172.16.10.10), Dst: 172.16.20.20 (172.16.20.20)
Versión: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
Total Length: 1052
Identification: 0x0000 (0)
Flags: 0x04 (Don't Fragment)
Fragment offset: 0
Time to live: 64
Protocol: UDP (0x11)
Header checksum: 0xc092 [correct]
Source: 172.16.10.10 (172.16.10.10)
Destination: 172.16.20.20 (172.16.20.20)
User Datagram Protocol, Src Port: 13900 (13900), Dst Port: 13901 (13901)
Data (52 bytes)
Frame 2 (1072 bytes on wire, 96 bytes captured)
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
Source: ae:52:a4:b3:46:fb (ae:52:a4:b3:46:fb)
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 1000, Exp: 0, S: 1, TTL: 63
MPLS Label: 1000
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 63
Frame 3 (1068 bytes on wire, 96 bytes captured)
Linux cooked capture
Internet Protocol, Src: 172.16.10.10 (172.16.10.10), Dst: 172.16.20.20 (172.16.20.20)
Version: 4
Header length: 20 bytes

Differentiated Services Field: 0x78 (DSCP 0x1e: Assured Forwarding 33; ECN: 0x00)
Total Length: 1052
Identification: 0x0000 (0)
Flags: 0x04 (Don't Fragment)
Fragment offset: 0
Time to live: 64
Protocol: UDP (0x11)
Header checksum: 0xc01a [correct]
Source: 172.16.10.10 (172.16.10.10)
Destination: 172.16.20.20 (172.16.20.20)
User Datagram Protocol, Src Port: 13902 (13902), Dst Port: 13903 (13903)
Data (52 bytes)
Frame 4 (1072 bytes on wire, 96 bytes captured)
Linux cooked capture
MultiProtocol Label Switching Header, Label: 2000, Exp: 0, S: 1, TTL: 63
MPLS Label: 2000
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 63
Internet Protocol, Src: 172.16.10.10 (172.16.10.10), Dst: 172.16.20.20 (172.16.20.20)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x78 (DSCP 0x1e: Assured Forwarding 33; ECN: 0x00)
Total Length: 105
Identification: 0x0000 (0)
Flags: 0x04 (Don't Fragment)
Fragment offset: 0
Time to live: 63
Protocol: UDP (0x11)
Header checksum: 0xc11a [correct]
Source: 172.16.10.10 (172.16.10.10)
Destination: 172.16.20.20 (172.16.20.20)
User Datagram Protocol, Src Port: 13902 (13902), Dst Port: 13903 (13903)
Data (48 bytes)

Como vemos en esta captura el paquete mantiene el mismo valor del campo DSCP. También podemos observar que se indica la utilización del protocolo MPLS, la etiqueta utilizada (en este caso es 2000), y los demás campos del header MPLS.

8.4- QoS basado en ingeniería de tráfico

En este experimento, se va a suponer que un proveedor de servicios, dispone de dos LSP (uno principal LSPp y otro secundario LSPs) para el envío de información, donde el LSPp es más critico que el LSPs. El proveedor desea garantizar un cierto ancho de banda para el LSPp. Por lo tanto, si un cliente no tiene mucho tráfico, se va a utilizar el camino 'principal' (LSPp), y en el caso de que exceda este ancho de banda, el tráfico es desviado por el camino 'secundario', (LSPs).
En la siguiente hoja se muestra el escenario sobre el cual se basa el experimento.



Para realizar esta configuración debemos ejecutar el script ubicado en */home/router/congestión*, dentro de esta carpeta se encontrará un archivo el cual ejecutaremos individualmente en cada router, esto automáticamente realizara la configuración para correr este experimento.
El LSR E1 es el encargado de verificar el tamaño de los paquetes enviados, y habilitar el LSPp o el LSPs, para realizar esto vamos a utilizar un pequeño programa llamado, bmon (la versión utilizada es 2.1.0), el cual sirve para realizar un filtrado de las interfaces y ver cual es el tamaño de los paquetes que circulan por una interfaz determinada. Para realizar esto vamos a ejecutar los siguientes comandos:

1. *bmon -p eth1 -o ascii:quitafter=2 >temp_bmon*
2. *var=`tail -1 temp_bmon `*
3. *var=`echo $var |cut -d " " -f 2`*

Luego vamos a revisar el valor de 'var', para saber cual es el tamaño de los paquetes, y dependiendo del valor de 'var' de dicho paquete se transmitirá por un LSP u otro. Esto se realiza por medio del siguiente código:

```
if [ $((100<$var)) = 0 ] ; then
   if [ $default_route = 1 ] ; then
      default_route=0 ;
      i p route del 172.16.20.0/24 via 10.0.3.2 mpls $var_indirect
      ip route add 172.16.20.0/24 via 10.0.5.3 mpls $var_default
      date +%r
      echo "ruta primaria"
   fi
fi
if [ $((100<$var)) = 1 ] ; then
   if [ $default_route = 0 ] ; then
      default_route=1 ;
      ip route del 172.16.20.0/24 via 10.0.5.3 mpls $var_default
      ip route add 172.16.20.0/24 via 10.0.3.2 mpls $var_indirect
      date +%r
      echo "ruta secundaria"
   fi
fi
```

8.4.1- Capturas realizadas

Se realizaron una series de capturas en el LSR E1 utilizando el comando **ping** enviando paquetes de diferentes tamaños. Cuando el paquete es menor a 100 KBps, el cual el paquete viaja a través del LSPp, en la siguiente captura se puede ver con color amarillo el paquete que va de la PC A3 hacia la PC A2 y con verde se puede ver como la PC A2 le responde a la PC A3:

*Frame 2 (**104 bytes on wire**, 96 bytes captured)*
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
Source: 36:3d:15:22:4e:a2 (36:3d:15:22:4e:a2)
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 300, Exp: 0, S: 1, TTL: 63
MPLS Label: 300
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 63
Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol
*Frame 3 (**104 bytes on wire**, 96 bytes captured)*
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
Source: ae:16:c7:ae:e8:ab (ae:16:c7:ae:e8:ab)
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 200, Exp: 0, S: 1, TTL: 62
MPLS Label: 200
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 62
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol

Ahora se utilizará el comando **ping** pero en esta ocasión superando el tamaño mínimo que permite el LSPp. Nuevamente se utilizará el color amarillo para los paquetes que viajan desde la PC A3 hacia la PC A2, y con verde la respuesta de la PC A2 a la PC A3:

*Frame 26 (**248 bytes on wire**, 96 bytes captured)*
Linux cooked capture
Packet type: Sent by us (4)
Link-layer address type: 1
Link-layer address length: 6
Source: a2:00:66:7d:b2:e9 (a2:00:66:7d:b2:e9)
Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 500, Exp: 0, S: 1, TTL: 63
MPLS Label: 500
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 63
Internet Protocol, Src: 172.16.30.30 (172.16.30.30), Dst: 172.16.20.20 (172.16.20.20)
Internet Control Message Protocol
*Frame 27 (**248 bytes on wire**, 96 bytes captured)*
Linux cooked capture
Packet type: Unicast to us (0)
Link-layer address type: 1
Link-layer address length: 6
Source: 42:59:cc:bb:7c:0b (42:59:cc:bb:7c:0b)

Protocol: MPLS label switched packet (0x8847)
MultiProtocol Label Switching Header, Label: 1000, Exp: 0, S: 1, TTL: 61
MPLS Label: 1000
MPLS Experimental Bits: 0
MPLS Bottom Of Label Stack: 1
MPLS TTL: 61
Internet Protocol, Src: 172.16.20.20 (172.16.20.20), Dst: 172.16.30.30 (172.16.30.30)
Internet Control Message Protocol

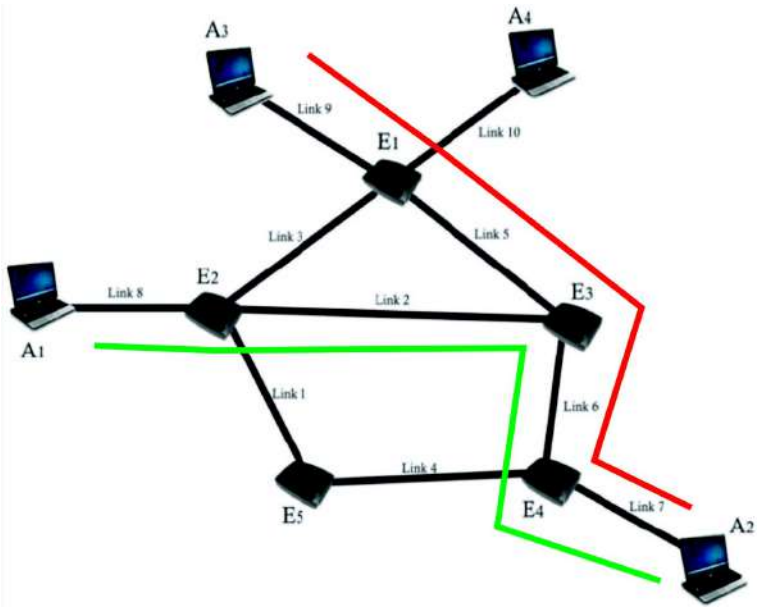
En las capturas realizadas anteriormente, se puede ver en negrita como son las etiquetas MPLS que se le asignan a cada paquete, solo dependiendo del tamaño del paquete, si este es menor a 100, se le asigna la etiqueta 300, para ir de la PC A3 hacia la PC A2, y la 200 para ir de la PC A2 hacia la PC A3. Y en el caso de que se supere los 100, se utilizará la etiqueta 500 para ir de la PC A3 hacia la PC A2, y la etiqueta 1000 para ir de la PC A2 hacia la PC A3. En el primer caso se utiliza el LSPp y en el segundo el LSPs.

9- Servicios diferenciado

En los siguientes experimentos se realizaron pruebas utilizando E-LSP y L-LSP, con servicio diferenciado. Las pruebas se realizaron, con el envío de paquetes de vídeo utilizando el servidor de vídeo VLC, para que esto sea posible se instalaron tanto en el cliente como en el servidores los siguientes paquete:

- vlc-0.8.6h-i486-2alien.tgz.
- dbus-1.1.20-i486-1.tgz.
- libogg-1.1.3-i486-2.tgz.
- libvorbis-1.2.0-i486-1.tgz.

En este experimento se realiza un envío de paquetes de vídeo, desde dos servidores diferentes, hacia un mismo cliente, procurando que el camino que recorren estos paquetes tengan un punto en común en uno o más de los LSR, en la siguiente figura podemos ver la topología utilizada.



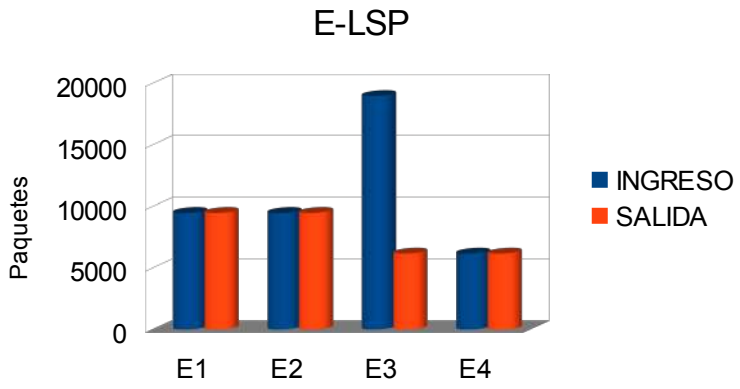
En la topología se puede ver que los puntos en común son los LSR E3 y LSR E4. Tanto para las pruebas realizadas con E-LSP y L-LSP, se utilizó el comando ***tc*** para limitar el tráfico en la salida de LSR E3, para simular un ancho de banda menor y forzar el descarte de paquetes, dejando pasar solo los de mayor privilegios.

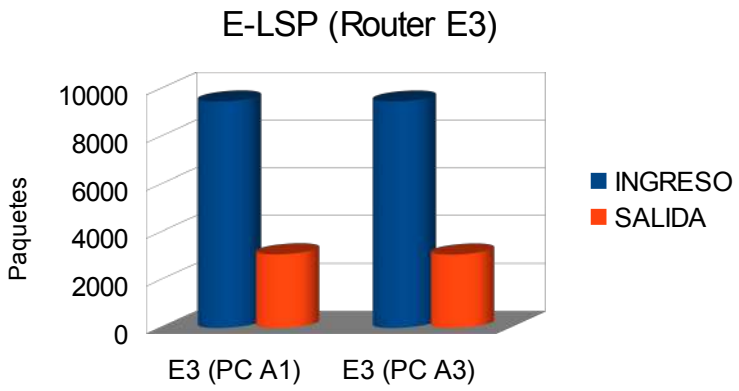
9.1- E-LSP

A continuación se mostraran los resultados obtenidos, luego de realizar la ejecución de este experimento utilizando los script de configuración.

E-LSP				
	INGRESO	SALIDA	PERDIDA	%
E1	**9478**	**9478**	**0**	**0,00**
E2	**9480**	**9480**	**0**	**0,00**
E3	**18958**	**6164**	**12794**	**67,49**
E3 (PC A1)	*9480*	*3087*	*6393*	*67,44*
E3 (PC A3)	*9478*	*3077*	*6401*	*67,54*
E4	**6164**	**6164**	**0**	**0,00**

A partir de estos datos se generaron dos gráficos, en el primero se puede ver con barras azules el ingreso de paquetes a cada LSR, y con rojo la salida de paquetes de cada LSR. Se puede observar que en los LSR E1, E2 y E4, la entrada es igual a la salida, esto se deba a que no poseen ningún tipo de filtro para reducir el ancho de banda. En el LSR E3 se puede ver una diferencia marcada entre la entrada y la salida debido a que se utilizó el comando ***tc***. Finalmente, en el último gráfico, podemos ver específicamente en el LSR E3 el ingreso y la salida separando los paquetes provenientes de la PC A1 y A2.





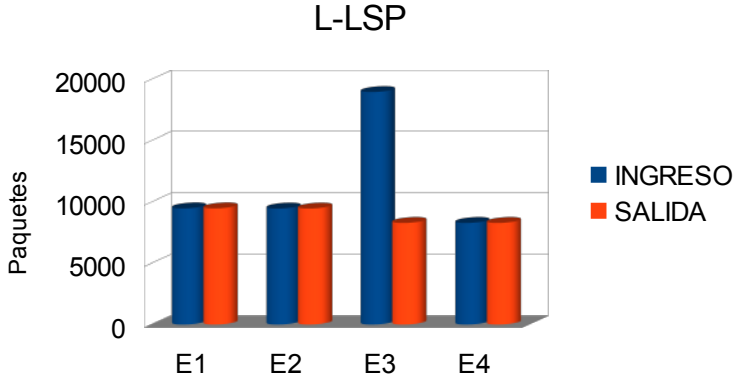
En este ultimo gráfico podemos ver claramente, que no diferencia entre los paquetes provenientes de una PC o de otra, simplemente descarta paquetes, y en este caso es prácticamente la misma cantidad de paquetes perdidos de la PC A1 y la PC A2.

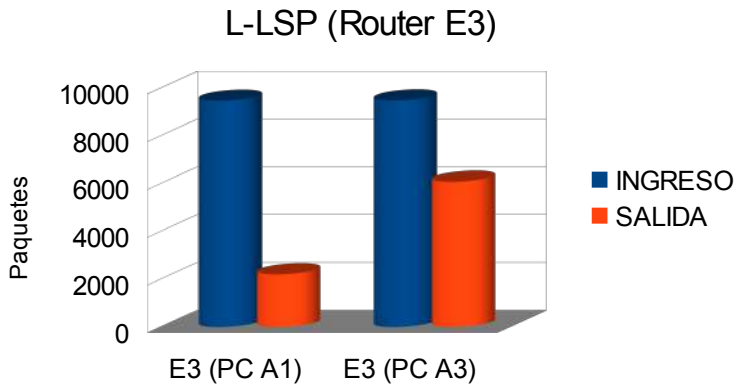
9.2- L-LSP

A continuación se mostraran los resultados obtenidos, luego de realizar la ejecución de este experimento utilizando los script de configuración.

L-LSP				
	INGRESO	SALIDA	PERDIDA	%
E1	**9486**	**9486**	**0**	**0,00**
E2	**9473**	**9473**	**0**	**0,00**
E3	**18959**	**8292**	**10667**	**56,26**
E3 (PC A1)	*9473*	*2205*	*7268*	*76,72*
E3 (PC A3)	*9486*	*6087*	*3399*	*35,83*
E4	**8292**	**8292**	**0**	**0,00**

Nuevamente para este experimento se generaron dos gráficos, donde en el primero se puede ver como nuevamente en los LSR E1, E2 y E4, no se registran pérdidas de paquetes, debido a que estos no contienen ningún tipo de filtro. Y un segundo gráfico donde podemos ver en detalle lo que sucede en el LSR E3, donde si se generan pérdidas de paquetes debido al que se está utilizando el comando ***tc***.





En este último gráfico podemos ver claramente, como se priorizan los paquetes provenientes de la PC A3 ya que estos presentan menos pérdidas.

10- Conclusión

En este trabajo se puede observar a través de diferentes experimentos como se puede garantizar calidad y servicio diferenciado con MPLS, cabe destacar que todos los experimentos se realizaron con etiquetas estáticas, ya que este paquete carece de cualquier protocolo de distribución de etiquetas, tales como LDP, RSVP o TDP. Es debido a la falta de estos protocolos que se debe de utilizar los script que realizan tareas de control de ancho de banda, o controlan la permanencia de un enlace determinado.

Finalmente se concluye que MPLS-Labs es una herramienta práctica a la hora de poder ver el funcionamiento de una red MPLS, junto con los intercambios de etiquetas en los diferentes LSR, a través de un LSP determinado, realizando túneles MPLS, las distintas formas de distribución de etiquetas (por interfaz o por plataforma), y utilizando código se puede simular calidad de servicio y servicio diferenciado.

Se puede decir como desventaja de este parche MPLS-Labs, que no es fácil realizar las configuraciones de cada LSR, ya que esto requiere de comandos muy largos, los cuales no son fáciles de recordar, y en algunas ocasiones, es necesario utilizar más de dos comandos ***mpls***. Otra desventaja a destacar es la falta de un protocolo de distribución de etiquetas, que facilite la creación de la red MPLS, y lleve todos los controles necesarios para poder realizar un verdadero control de calidad y servicio diferenciado.

Índice