

Configuración y pruebas de funcionamiento de la interconexión de redes heterogéneas con troncal MPLS

Jordi Lores Jacinto

Director: Xavier Hesselbach i Serra
Departament d'Enginyeria Telemàtica

Índice

1.	Introducción	11
1.1.	Conocimientos previos.....	13
1.1.1.	Modelo OSI/ISO.....	13
1.1.2.	IP (Internet Protocol)	14
1.1.3.	Enrutamiento	15
2.	Objetivos del proyecto	16
3.	MPLS (MultiProtocol Label Switching).....	17
3.1.	Conceptos básicos.....	18
3.2.	Componentes básicos	19
3.2.1.	Label (Etiqueta)	19
3.2.2.	FEC (Forwarding Equivalence Class).....	20
3.2.3.	LSR (Label Switched Router)	21
3.2.4.	LSP (Label Switched Path)	21
3.2.5.	Label Stack (Pila de Etiquetas)	22
3.2.6.	LDP (Label Distribution Protocol).....	23
3.3.	Funcionamiento del protocolo MPLS.....	24
3.3.1.	Creación y distribución de etiquetas	24
3.3.2.	Creación de la tabla LIB en cada LSR.....	24
3.3.3.	Creación de los LSPs.....	25
3.3.4.	Paso de un paquete por la red	26
3.3.5.	Fast Reroute	27
4.	QoS en redes MPLS	28
4.1.	Conceptos básicos.....	28
4.2.	Servicios Diferenciados (Diffserv)	29

4.2.1.	Diffserv y paquetes IP	29
4.2.2.	DiffServ y paquetes MPLS	31
5.	Configuración	33
5.1.	Escenario básico:.....	33
5.2.	Configuración del escenario:	34
5.2.1.	Reseteo de la configuración de los routers:	35
5.2.2.	Configuración básica del router:.....	35
5.2.3.	Configuración básica para IP y MPLS	36
5.2.4.	Configuración del “marcador”	45
5.3.	Pruebas de conexión extremo-extremo	47
5.4.	Pruebas de envío de vídeo extremo-extremo	49
5.4.1.	Configuración del VLC	49
5.4.2.	Resultados.....	50
5.5.	Configuración del camino Backup.....	51
5.6.	Configuración del Balanceo de Carga	59
5.7.	Configuración de un punto de acceso Wifi	62
5.8.	Configuración de Túneles IP-IP	64
5.8.1.	Configuración en la Fuente:.....	66
5.8.2.	Configuración para el Destino:.....	66
5.9.	Configuración de QoS	70
5.9.1.	Definir las clases de tráfico	71
5.9.2.	Definir políticas de QoS.....	72
5.9.3.	Asignar las políticas a los interfaces.....	73
6.	Pruebas a través de los routers IP-MPLS	75
6.1.	Pruebas a través de los routers IP-MPLS - Ethernet	75

6.1.1. Conclusiones	95
6.2. Pruebas a través de los routers IP-MPLS – Wifi.....	97
6.2.1. Conclusiones	114
6.3. Pruebas a través de los routers IP-MPLS – QoS.....	116
6.3.1. Conclusiones	118
6.4. Pruebas a través de los routers IP-MPLS – Túnel IP-IP	119
7. Conclusiones y línea futuras de trabajo.....	124
7.1. Conclusiones	124
7.2. Líneas futuras de trabajo	126
8. Bibliografía	127

Índice de Figuras

Figura 1: Modelo de la capa OSI	13
Figura 2: Posición de MPLS en el modelo OSI.....	17
Figura 3: Cabecera MPLS.....	19
Figura 4: Ejemplo del Label Stack a través de la red	22
Figura 5: Ejemplo del detalle de una "Pila de etiquetas"	23
Figura 6: Ejemplo de Label Information Base	25
Figura 7: Red MPLS, con un LSP definido.....	26
Figura 8: Detalle y relación de las cabeceras IP y MPLS.....	31
Figura 9: Mapeo del campo DSCP en el EXP	31
Figura 10: Escenario básico	33
Figura 11: Configuración del interfaz	46
Figura 12: Configuración de las reglas en el marcador.....	47
Figura 13: Ping desda la "Fuente" hacia el "Destino"	48
Figura 14: Ping desda el "Destino" hacia la "Fuente"	48
Figura 15: Configuración del VLC en la "Fuente"	49
Figura 16: Configuración del VLC en el "Destino"	50
Figura 17: Detalle de un paquete MPLS.....	51
Figura 18: Escenario con Backup	52
Figura 19: Escenario con Backup y Wifi	62

Figura 20: Configuración de red del interfaz móvil.....	63
Figura 21: Ping entre el dispositivo móvil y la "Fuente"	64
Figura 22: Escenario con túneles	65
Figura 23: Detalle de paquete IP-IP	70
Figura 24: Detalle de paquete IP.....	70
Figura 25: Diagrama de la aplicación del QoS.....	74
Figura 26: Escenario con Backup	75
Figura 27: Caudal en el Origen en bits por segundo	77
Figura 28: Caudal en el Analizador en bits por segundo.....	77
Figura 29: Caudal en el Destino en bits por segundo	78
Figura 30: Jitter del Caso 1.....	79
Figura 31: Caudal en el Origen en bits por segundo.....	80
Figura 32: Caudal en el Analizador en bits por segundo.....	81
Figura 33: Caudal en el Destino en bits por segundo	81
Figura 34: Detalle del destino de los paquetes en el Analizador en bits por segundo.....	82
Figura 35: Jitter del Caso 2.....	83
Figura 36: Caudal en el Origen en bits por segundo.....	84
Figura 37: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo.....	84
Figura 38: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo	85
Figura 39: Caudal en el Destino en bits por segundo	85
Figura 40: Caudal en el Origen en bits por segundo	86

Figura 41: Caudal en el Analizador en bits por segundo.....	87
Figura 42: Caudal en el Destino en bits por segundo	87
Figura 43: Jitter en el Caso 4.1.....	88
Figura 44: Caudal en el Origen en bits por segundo	89
Figura 45: Caudal en el Analizador en bits por segundo.....	89
Figura 46: Caudal en el Destino en bits por segundo	90
Figura 47: Detalle del destino de los paquetes en el Analizador en bits por segundo	91
Figura 48: Jitter en el Caso 4.2.....	91
Figura 49: Caudal en el Origen en bits por segundo.....	92
Figura 50: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo.....	93
Figura 51: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo	93
Figura 52: Caudal en el Destino en bits por segundo	94
Figura 53: Escenario con Backup y Wifi	97
Figura 54: Caudal en el Origen en bits por segundo	98
Figura 55: Caudal en el Analizador en bits por segundo.....	98
Figura 56: Caudal en el Destino en bits por segundo	99
Figura 57: Jitter en el Caso 1	99
Figura 58: Caudal en el Origen en bits por segundo	100
Figura 59: Caudal en el Analizador en bits por segundo.....	101
Figura 60: Caudal en el Destino en bits por segundo	101
Figura 61: Detalle del destino de los paquetes en el Analizador en bits por segundo	102

Figura 62: Jitter en el Caso 2	103
Figura 63: Caudal en el Origen en bits por segundo	104
Figura 64: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo	104
Figura 65: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo	105
Figura 66: Caudal en el Destino en bits por segundo	105
Figura 67: Caudal en el Origen en bits por segundo	106
Figura 68: Caudal en el Analizador en bits por segundo	107
Figura 69: Caudal en el Destino en bits por segundo	107
Figura 70: Jitter en el Caso 4.1	108
Figura 71: Caudal en el Origen en bits por segundo	109
Figura 72: Caudal en el Analizador en bits por segundo	109
Figura 73: Caudal en el Destino en bits por segundo	110
Figura 74: Detalle del destino de los paquetes en el Analizador en bits por segundo	110
Figura 75: Jitter en el caso 4.2	111
Figura 76: Caudal en el Origen en bits por segundo	112
Figura 77: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo	112
Figura 78: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo	113
Figura 79: Caudal en el Destino en bits por segundo	113
Figura 80: Escenario básico con Backup	116
Figura 81: Caudal en el Origen en bits por segundo	117
Figura 82: Caudal en el Analizador en bits por segundo	117

Figura 83: Caudal en el Destino en bits por segundo	118
Figura 84: Escenario con Túneles.....	120
Figura 85: Paquete entre el cliente y la red MPLS	121
Figura 86: Paquete a través de la red MPLS	121
Figura 87: Paquete ente la red MPLS y el servidor	121
Figura 88: ACK entre el servidor y la red MPLS.....	121
Figura 89: ACK a través de la red MPLS	122
Figura 90: ACK entre la red MPLS el cliente	122
Figura 91: Caudal en el Cliente en bits por segundo	122
Figura 92: Caudal en el Servidor en bits por segundo	123

Índice de Tablas

Tabla 1: LIB de MPLS4 en el escenario básico.....	43
Tabla 2: LIB de MPLSCore en el escenario básico	44
Tabla 3: LIB de MPLS100 en el escenario básico.....	45
Tabla 4: LIB en MPLS4 con dos caminos	55
Tabla 5: LIB en MPLS4 con el Backup configurado	57
Tabla 6: LIB en MPLS4 con Backup active	58
Tabla 7: LIB en MPLS4 con el enlace principal recuperado.....	59
Tabla 8: Ejemplo de funcionamiento del balanceo por paquete.....	61
Tabla 9: Tabla de enrutamiento de la "Fuente".....	68
Tabla 10: Tabla de enrutamiento del Tunnel1.....	69
Tabla 11: Tabla de enrutamiento del Tunnel2.....	69
Tabla 12: Tabla de enrutamiento del “Destino”	69
Tabla 13: Resultados de las pruebas MPLS – Ethernet.....	95
Tabla 14: Resultados de las pruebas MPLS – Wifi.....	114

1. Introducción

El protocolo IP (Internet Protocol) se ha convertido en el pilar donde se sustentan las actuales redes de telecomunicaciones, gracias al gran crecimiento de la red Internet. Actualmente cuenta con más de un 80% del tráfico cursado. La versión actual de IP, conocida como IPv4 y recogida en la RFC 791, lleva operativa desde 1980. Este protocolo está definido en la capa de red (Nivel 3 OSI), e implementa los mecanismos de la distribución o encaminamiento de paquetes. Lo hace de una manera no fiable y sin conexión, en redes heterogéneas, por lo que se suele utilizar junto con TCP (*Transmission Control Protocol*) (Nivel 4 de OSI) para garantizar la entrega de los paquetes.

Durante la década de los 90, se introdujo ATM (*Asynchronous Transfer Mode*) en la capa de enlace (Nivel 2 de OSI) de las redes a causa de la gran demanda por parte de los clientes de los ISP (*Internet Service Providers*) de aplicaciones multimedia con altas necesidades de ancho de banda y una calidad de servicio o QoS (*Quality of Service*) garantizada. Este nuevo modelo de IP sobre ATM satisfacía los requisitos de las nuevas aplicaciones, ya que utilizaba el encaminamiento inteligente de nivel 3 de los routers IP en la red de acceso, e incrementaba el ancho de banda y rendimiento con los switches ATM en la red central, basándose en la alta velocidad de los conmutadores de nivel 2 y los circuitos permanentes virtuales. Por otro lado, esta arquitectura, presentaba ciertas limitaciones, debido a:

- La dificultad de operar e integrar una red basándose en dos tecnologías muy distintas.
- La aparición de switches ATM e IP de alto rendimiento en las redes troncales.
- La mayor capacidad de transmisión ofrecida por SDH/SONET (*Synchronous Digital Hierarchy / Synchronous Optical NETwork*) y DWDM (*Dense Wavelength Division Multiplexing*) respecto a ATM.

En 1996, aparecieron soluciones de conmutación de nivel 2 propietarias diseñadas para la red troncal de Internet que integraban la conmutación ATM con el encaminamiento IP; como por ejemplo, **Tag Switching** de Cisco o **Aggregate Route-Based IP Switching** de IBM. La parte común de estas tecnologías, era tomar el software de control de un router IP, integrarlo con el rendimiento de reenvío con cambio de etiqueta de un switch ATM y crear un router extremadamente rápido y eficiente en cuanto a coste.

La integración en esta arquitectura era mayor, porque se utilizaban protocolos IP propietarios para distribuir y asignar los identificadores de conexión de ATM como etiquetas; pero los protocolos no eran compatibles entre sí y aún requerían infraestructura ATM.

Finalmente en 1997, el IETF (*Internet Engineering Task Force*) estableció el grupo de trabajo **MPLS** (**MultiProtocol Label Switching**) para producir un nuevo estándar que unificase las soluciones propietarias de conmutación de nivel 2. Como resultado se obtuvo la definición en 1998 del estándar conocido por MPLS, recogido en la RFC 3031. MPLS proporciona las ventajas de la ingeniería de tráfico del modelo de IP sobre ATM, pero además proporciona los siguientes beneficios:

- Una operación y diseño de red más sencillo y una mayor escalabilidad.
- MPLS está diseñado para operar sobre cualquier tecnología en el nivel de enlace, facilitando así la migración a las redes ópticas de próxima generación, basadas en infraestructuras SDH/SONET y DWDM, a diferencia de las soluciones de conmutación de nivel 2 propietarias.

1.1. Conocimientos previos

1.1.1. Modelo OSI/ISO

El modelo OSI (Open System Interconnection) es la referencia creada por la ISO (Organización Internacional de la Estandarización) para definir la arquitectura de la interconexión de sistemas.

The OSI Reference Model

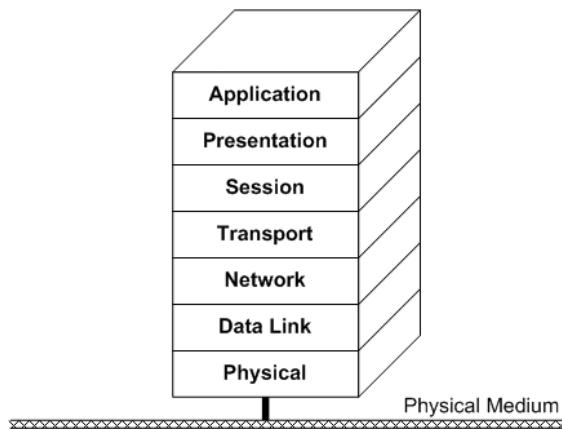


Figura 1: Modelo de la capa OSI

Está formada por 7 capas:

1. **Capa física:** Es la encargada de la conexión física del sistema con la red, tanto en la forma en cómo se transmite la información, cómo por qué medio.
2. **Capa de enlace:** En este capa se controla el direccionamiento físico, el acceso a la red y el control de flujo, así como de a notificación de errores y del envío ordenado de las tramas.
3. **Capa de red:** El objetivo de esta capa es que los paquetes lleguen a su destino, aunque no estén conectados directamente.
4. **Capa de transporte:** Esta capa es la encargada de transportar los paquetes de la capa de red hacia su destino.

5. **Capa de sesión:** El objetivo de esta capa es el de mantener la conexión establecida entre dos ordenadores mientras están transmitiendo datos.

6. **Capa de presentación:** La función de esta capa es la de representar de manera correcta la información que se transmite.

7. **Capa de aplicación:** Finalmente, la capa de aplicación es donde se accede a los servicios, como el HTTP o el SMTP del correo electrónico. Es la capa más “cercana” al usuario.

1.1.2. IP (Internet Protocol)

Internet Protocol o IP es un protocolo de clase 3 no orientado a conexión usado para la comunicación de datos a través de una red de paquetes conmutados.

Los datos en una red basada en IP son enviados en bloques conocidos como paquetes o datagramas. En particular, en IP no se necesita ninguna configuración antes de que un equipo intente enviar paquetes a otro con el que no se había comunicado antes.

IP provee un servicio de datagramas no fiable (también llamado del *mejor esfuerzo (best effort)*) que no provee ningún mecanismo para determinar si un paquete alcanza o no su destino. Simplemente proporciona seguridad (mediante *checksums* o sumas de comprobación) de sus cabeceras y no de los datos transmitidos. Si se necesita fiabilidad, ésta es proporcionada por los protocolos de la capa de transporte, como TCP.

Si la información a transmitir ("datagramas") supera el tamaño máximo "negociado" (MTU) en el tramo de red por el que va a circular podrá ser dividida en paquetes más pequeños, y reensamblada luego cuando sea necesario. Estos fragmentos podrán ir cada uno por un camino diferente dependiendo de cómo estén de congestionadas las rutas en cada momento.

Las cabeceras IP contienen las direcciones de las máquinas de origen y destino (direcciones IP), direcciones que serán usadas por los conmutadores de paquetes (switches) y los enrutadores (routers) para decidir el tramo de red por el que reenviarán los paquetes.

1.1.3. Enrutamiento

El encaminamiento (a veces conocido por el anglicismo ruteo o enrutamiento) es el mecanismo con el que se hacen llegar paquetes desde su origen a su destino final, siguiendo un camino o ruta a través de la red. En una red grande o en un conjunto de redes interconectadas el camino a seguir hasta llegar al destino final puede suponer transitar por muchos nodos intermedios.

Asociado al encaminamiento existe el concepto de métrica, que es una medida de lo "bueno" que es usar un camino determinado. La métrica puede estar asociada a distintas magnitudes: distancia, coste, retardo de transmisión, número de saltos, etc., o incluso a una combinación de varias magnitudes. Si la métrica es el retardo, es mejor un camino cuyo retardo total sea menor que el de otro. Lo ideal en una red es conseguir el encaminamiento óptimo: tener caminos de distancia (o coste, o retardo, o la magnitud que sea, según la métrica) mínimos. Típicamente el encaminamiento es una función implantada en la capa 3 (capa de red) del modelo de referencia OSI.

2. Objetivos del proyecto

El objetivo general de este proyecto es realizar un estudio sobre la interconexión de una red MPLS con distintas redes heterogéneas. Partiendo de una pequeña red privada MPLS se han implementado distintos mecanismos de balanceo de carga y de recuperación en caso de fallo, con la interconexión con redes Ethernet y Wifi.

Los objetivos específicos son los siguientes:

- Estudiar la arquitectura y el funcionamiento del protocolo MPLS, así como los comandos necesarios para la configuración de una pequeña red.
- Establecer una conexión extremo-extremo de la red anterior y verificar su correcto funcionamiento.
- Añadir redundancia a la red para aportar un camino de backup en caso de fallo, o para proporcionar balanceo de carga entre los dos caminos.
- Realizar diversas pruebas para comprobar cómo funcionan estos mecanismos en distintas redes heterogéneas y con tráfico de video real.
- Implementar distintos mecanismos de QoS a través de la red MPLS y probar su funcionamiento.
- Elaborar análisis y conclusiones sobre los resultados obtenidos con las pruebas llevadas a cabo.

3. MPLS (MultiProtocol Label Switching)

Como ya hemos comentado anteriormente, la necesidad de las operadoras de que sus redes tuvieran una cierta calidad de servicio, ha llevado a la búsqueda de una tecnología que ofreciese ese QoS por sí misma. Las tecnologías como ATM o SDH se han quedado obsoletas y aplicar calidad de servicio es una tarea muy complicada.

Por estas razones surgió MPLS. Definida en el RFC 3031, es una tecnología orientada a paquetes muy flexible. Si la situásemos en el modelo ISO/OSI (International Standard Organization / Open System Interconnection) se encontraría en la capa 2.5, entre la capa de enlace y de red, o sea, entre la capa 2 y 3. El hecho de que se encuentre entre dos capas, le proporciona el nombre de “Multi Protocol”. Este hecho le da la ventaja de poder usar las características de los protocolos de las capas adyacentes sin ninguna restricción.

Además de esto, MPLS ofrece adaptación total a IP. Esto es de gran importancia porque actualmente el mundo se mueve con este protocolo.

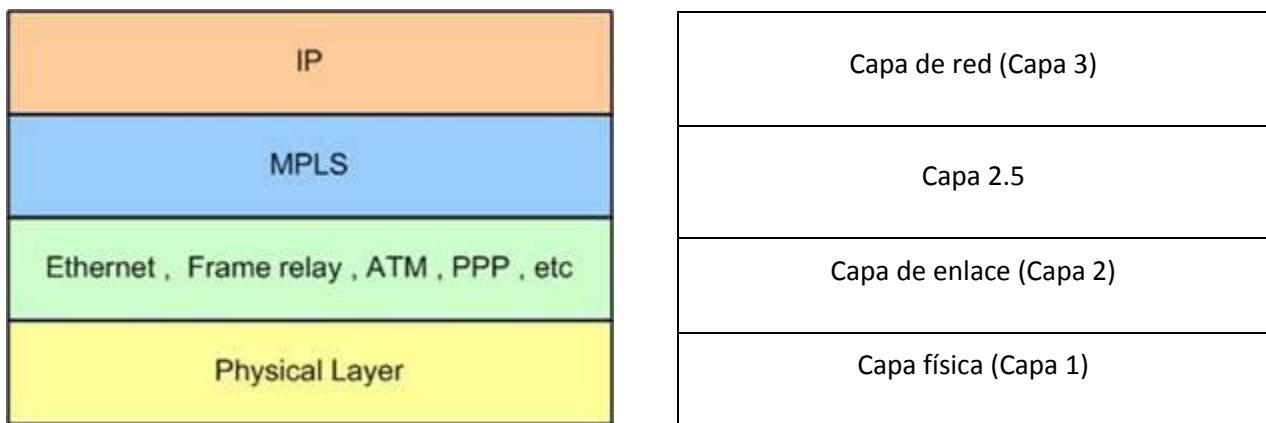


Figura 2: Posición de MPLS en el modelo OSI

Con MPLS ganamos en muchos aspectos en los que ATM presentaba carencias. Gracias al *label switching*, técnica usada en MPLS para enrutar paquetes, conseguimos hacer este enrutado a más velocidad, a la vez que disminuimos el retardo y el jitter. Estas características son de especial interés en

las redes troncales, donde uno de sus principales objetivos es enviar paquetes de una localización a otra en el mínimo tiempo posible. Pero MPLS va más allá que la velocidad y nos ofrece otras grandes ventajas como la posibilidad de tener el control de la ruta, asignar distintos anchos de banda a los enlaces o crear prioridades para la utilización de un enlace. Todas estas ventajas y otras constituyen lo que se llama Traffic-Engineering (TE) y que suele designarse como MPLS-TE.

3.1. Conceptos básicos

MPLS proporciona la posibilidad de administrar el tráfico de una red a través de etiquetas en las cabeceras de los paquetes y a routers específicos capaces de reconocerlas. Principalmente consiste en integrar los niveles de enlace y red eficazmente. Es decir, combina la inteligencia del routing con la velocidad del switching.

MPLS usa un esquema de etiquetado de tráfico, marcándolo en la entrada de la red, pero no en su salida. Es usado únicamente en los routers y es independiente del protocolo usado, lo que le permite ser utilizado sobre otros protocolos distintos a IP, como IPX, ATME, PPP, Ethernet, Frame Relay.... Los protocolos de enrutamiento de nivel 3 como OSPF o IS-IS se usan únicamente para funciones de control, ya que las decisiones de enrutamiento se toman en función de la etiqueta MPLS y no de la cabecera IP.

MPLS mejora la escalabilidad de la red (reduciendo las tablas de enrutamiento) y el retardo de proceso en los routers, combinando algunas prestaciones de las redes orientadas a conexión con la de las redes sin conexión. Así, un router asigna una etiqueta a cada una de las entradas de la tabla de enrutamiento y las distribuye a sus routers vecinos. Luego, cuando se pasan paquetes entre ellos, los routers solo tienen que leer la etiqueta MPLS para identificar el siguiente salto donde enviar el paquete. De esta forma los paquetes “fluyen” de un extremo a otro de la red sin que los routers tengan que mirar su dirección.

3.2. Componentes básicos

A continuación se detallan los principales componentes que forman parte de una red MPLS.

3.2.1. Label (Etiqueta)

La etiqueta MPLS es un identificador de 20 bits encapsulado dentro de la cabecera MPLS de 32 bits. Esta contiene la información necesaria para enrutar un paquete hasta su destino.

Las etiquetas se utilizan en los routers para diferenciar entre los distintos FECs (Forward Equivalence Class) y por lo tanto determinar el siguiente salto donde el paquete debe ser enviado. Generalmente el valor de la etiqueta se asigna a partir de la dirección IP destino y es meramente local, ya que solo tiene validez entre dos routers vecinos. Por otro lado, un mismo paquete puede ser etiquetado con un valor distinto dependiendo de cuál sea su router de entrada a la red MPLS (LSR). También cabe añadir que un paquete puede disponer de múltiples etiquetas (pila de etiquetas).

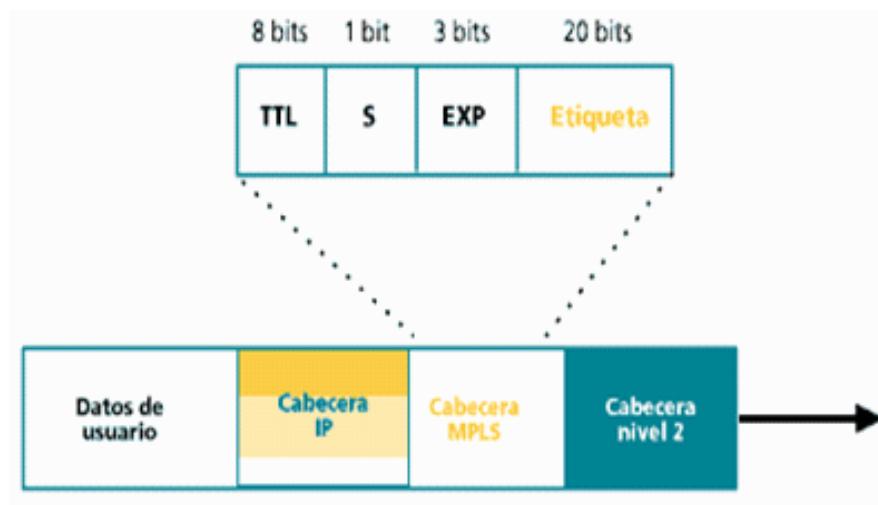


Figura 3: Cabecera MPLS

Como puede observarse en el gráfico superior, la cabecera MPLS se incluye entre las cabeceras de nivel 2 y 3 y contiene los siguientes campos:

- TTL (Time to Live): es un valor que se decrementa cada vez que el paquete es reenviado por un router de la red MPLS (LSR). Cuando el valor es 0, el paquete se descarta. Su función es evitar que un paquete viaje indefinidamente por la red, provocando tráfico innecesario.
- S (Bottom of stack): Si su valor es 1 indica que el paquete solo contiene una etiqueta. Si por el contrario vale 0, significa que el paquete posee una pila de etiquetas.
- EXP (Experimental): Anteriormente era denominado CoS (Class of Service) pero ahora se considera un campo experimental. Se suele usar para proporcionar QoS.
- Etiqueta (label): Valor local que usa el router para identificar un FEC en el proceso de forwarding, para determinar el próximo salto del paquete o su encapsulación.

3.2.2. FEC (Forwarding Equivalence Class)

El FEC es la agrupación de etiquetas que permite la asociación de un conjunto de paquetes sobre el mismo camino y con un destino común.

Todos los paquetes de un mismo FEC se tratan de la misma forma hacia su destino, y cuantos más FECs tengamos, mayor granularidad para diferenciar entre distintos tipos de flujos. Aunque el hecho de tener más FECs nos afecta en la escalabilidad de la red, y por lo tanto, tendremos que llegar a un compromiso entre el número de FECs y la eficiencia de la red.

Cada FEC tiene un camino específico a seguir a través de la red MPLS y es independiente en cada router. Puede darse el caso que para una misma dirección IP haya más de un FEC a través del mismo LSP (Label Switched Path), lo que significa que paquetes con un mismo destino pueden pertenecer a FECs distintos si se tienen que tratar de forma distinta.

Así, la etiqueta de un determinado paquete representa al FEC al cual pertenece. Los LSR de entrada, que son los que etiquetan a los paquetes, son los encargados de asociar cada paquete a un FEC y se basan principalmente en la dirección destino para hacerlo, aunque también puede depender de otros factores como de la dirección de origen, los puertos de origen y destino, el protocolo o los requerimientos de servicio.

A pesar de que, como ya hemos comentado, los FECs están pensados para agrupar a un conjunto de etiquetas, en la práctica lo normal es que cada FEC esté asociado a una única etiqueta.

3.2.3. LSR (Label Switched Router)

Los LSR son todos aquellos routers que son capaces de usar MPLS. A diferencia de un router convencional, estos routers reenvían los paquetes en función de las etiquetas de los paquetes recibidos, y no en función de la dirección IP de destino. En una red MPLS podemos encontrar dos tipos de LSR:

- **Label Edge Router (LER):** Los LER son los routers frontera que operan en los bordes de una red MPLS. Estos routers son los encargados de convertir los paquetes IP en paquetes MPLS, o viceversa. Dependiendo de esta función, podemos diferenciar entre los tipos Ingres Edge Router (router de ingreso) y los Egress Edge Router (router de salida). Los primeros se sitúan en la entrada de la red y se encargan de asignar un FEC a los paquetes que reciben y de etiquetarlos para que lleguen a su destino. Los routers de salida son los encargados de hacer la acción contraria, eliminar la etiqueta (label pop). Estos se sitúan al final de la red.
- **Core Router:** Estos son los routers que forman el núcleo de la red y permiten el tránsito de los paquetes hacia su destino. Estos routers están capacitados para hacer un label swapping (intercambio de etiquetas), label push y label pop.

3.2.4. LSP (Label Switched Path)

El LSP es el camino compuesto por uno o varios LSR a través del cual se transmiten todos los paquetes pertenecientes a un determinado FEC.

Estos caminos son unidireccionales (simplex) y solo transmiten hacia un sentido de tráfico. Si queremos que nuestra red sea dúplex, se deben establecer dos LSPs, uno para cada sentido.

Los LSPs se pueden diseñar para minimizar el número de saltos de los paquetes, para evitar congestiones en puntos críticos de la red, para tener un cierto ancho de banda o simplemente para forzar que el tráfico pase a través de un cierto nodo.

MPLS nos proporciona dos opciones para crear un LSP:

- Punto a punto: Desde el LSR de origen, se especifica punto a punto los saltos que tiene que dar el paquete. No suele ser el LSP más óptimo, pero es el más fácil de gestionar ya que el administrador controla por donde pasa el tráfico.
- Acoplados (merging): El camino escogido por los paquetes es determinado por el protocolo de enrutamiento interno (IGP), normalmente OSPF o IS-IS. Este camino puede ir variando si las condiciones de la red cambian.

3.2.5. Label Stack (Pila de Etiquetas)

Una de las características del protocolo MPLS es que nos permite apilar diversas etiquetas unas sobre las otras. Esto se denomina “Pila de Etiquetas” y consigue anidar un LSP dentro de otro. El objetivo de esta técnica es el de crear túneles dentro de los otros LSPs, como se puede observar en la siguiente figura:

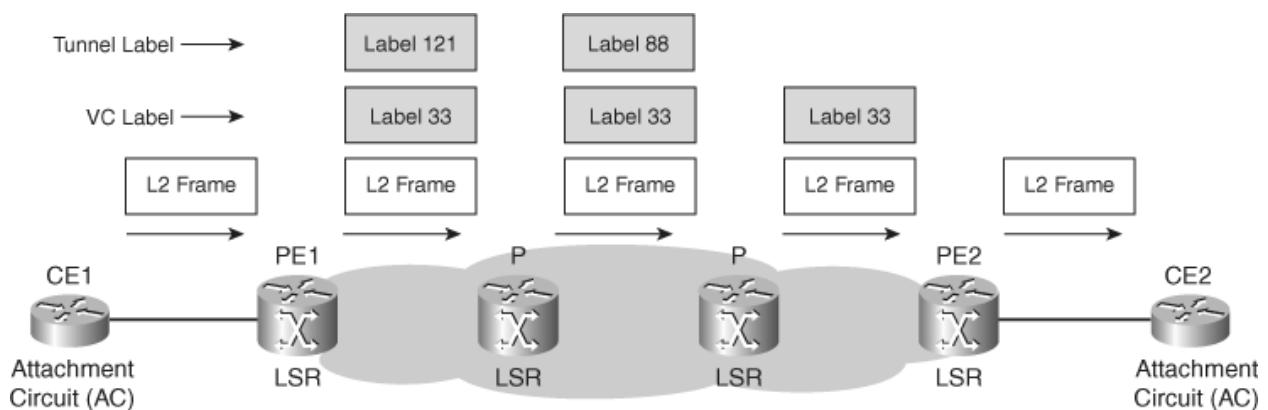


Figura 4: Ejemplo del Label Stack a través de la red

Para conseguirlo, un LSR en vez de intercambiar las etiquetas lo que hace es añadir una etiqueta nueva encima de la pila. Las etiquetas se añaden siguiendo un sistema LIFO (Last-in, First-out) y no altera el funcionamiento de enrutado, simplemente el router lee la etiqueta más externa y actúa únicamente en función de ese valor.



Figura 5: Ejemplo del detalle de una "Pila de etiquetas"

3.2.6. LDP (Label Distribution Protocol)

El LDP es el protocolo más extendido para la distribución de etiquetas y comunicación de ellas a los LSRs, aunque existen otros como RSVP, PIM o TDP. Está definido en el RFC 3036, funciona sobre TCP y usa las tablas de enrutamiento IP existentes creadas por el protocolo de enrutamiento, como OSPF, para propagarse.

El LDP, por un lado, asocia un FEC con cada camino LSP que se crea, y por el otro, intercambia y distribuye esta información de asociación de las etiquetas entre dos LSR vecinos. Esta asociación es bidireccional y permite que un LSR aprenda del otro.

La distribución de las etiquetas usa uno de los dos siguientes métodos:

- Unsolicited Downstream: En este método, el LSR distribuye su información sobre las etiquetas cuando las tiene disponibles, aunque no se la hayan solicitado.
- Downstream on Demand: Solo se envía información sobre las etiquetas cuando el vecino LSR pide información sobre ella.

3.3. Funcionamiento del protocolo MPLS

Con el enrutamiento IP los paquetes avanzan de “salto en salto” a través de la red, es decir que en cada router se encamina el paquete hacia el siguiente salto en función de su dirección IP destino y de la tabla de enrutamiento. En MPLS, los LSR también encaminan los paquetes “salto a salto” pero simplemente basándose en la etiqueta de longitud fija, lo que significa que no usan la información de la cabecera IP.

3.3.1. Creación y distribución de etiquetas

Antes de que se inicie el tráfico de datos, cada router de ingreso (LER) une ciertas etiquetas con determinados FECs y construye la tabla de etiquetas FER. Una vez completado este proceso, se distribuyen estas uniones usando el protocolo LDP entre los distintos LSRs.

Como ya se ha comentado anteriormente, el protocolo LDP usa TCP para comunicar las etiquetas, ya que aporta fiabilidad a la red. Un error en la distribución de las etiquetas resultaría fatal para el funcionamiento de la red.

3.3.2. Creación de la tabla LIB en cada LSR

Cada LSR construye una tabla de etiquetas LIB (Label Information Base) a medida que va recibiendo las etiquetas con el protocolo LDP. Las tablas LIB es donde se especifica el mapeo de cada etiqueta con un interfaz, tanto de entrada como de salida. Esta tabla se actualiza cada vez que se efectúa una renegociación de las uniones de etiquetas.



Figura 6: Ejemplo de Label Information Base

Esta tabla guía al LSR cuando tiene que realizar un swap de etiquetas, indicándole a qué interfaz tiene que dirigir el paquete. En el ejemplo de esta figura, un paquete de entrada por la interfaz 2 con la etiqueta 51, se redirigiría a la interfaz 5 con la etiqueta 37.

3.3.3. Creación de los LSPs

El siguiente paso es la creación de los LSP, los cuales se crean en orden inverso a la trayectoria del paquete. Lo que significa que el LSP se crea en el Nodo Destino hacia el Nodo Origen.

El Nodo Origen, al recibir un paquete del cual no tiene etiqueta en la tabla LIB, solicita mediante un paquete “request” la ruta que necesita. Este paquete “request” se irá propagando hasta llegar al nodo LER de salida. Una vez recibido este paquete, el LER enviará un paquete de “mapping” en dirección upstream. Este paquete, al pasar por los nodos hacia el Nodo Origen, irá completando la tabla LIB relacionada con el LSP que se está creando.

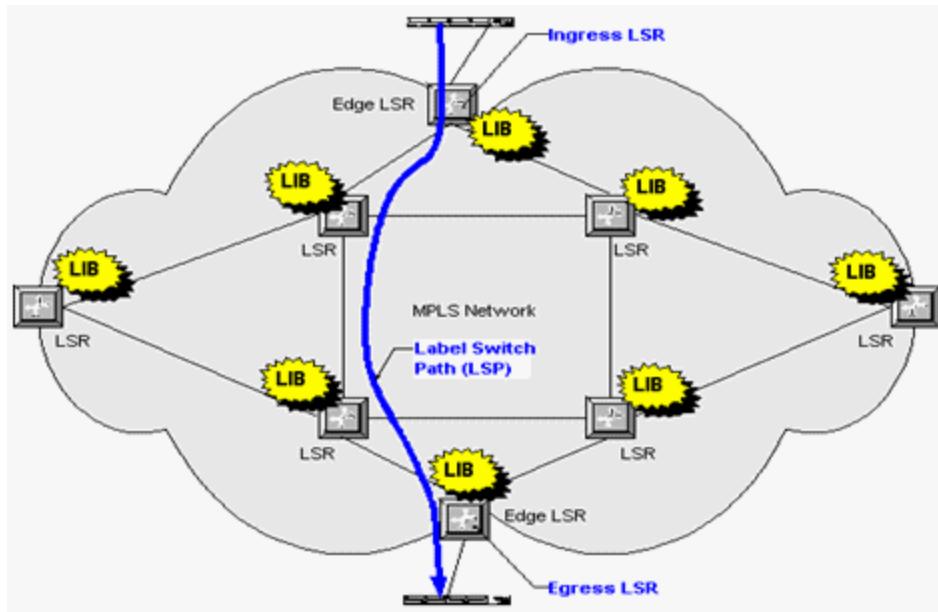


Figura 7: Red MPLS, con un LSP definido

3.3.4. Paso de un paquete por la red

Una vez ya tenemos definidos los FECs, LDP y etiquetas, solo nos queda analizar el proceso que sigue un paquete al entrar en una red MPLS.

Primero, llega un paquete sin etiquetar a un router LER de ingreso. El router entonces decide a qué FEC pertenece y le asigna las etiquetas correspondientes (push). El proceso de asignar un paquete a un FEC solo se hace una vez, a diferencia de lo que ocurriría con un paquete IP tradicional, que se evalúa en cada nodo.

Una vez el paquete ya está etiquetado, se envía al siguiente salto LSR usando la tabla LIB.

Este paquete va saltando de LSR en LSR basándose en la tabla LIB de cada router. Normalmente lo que hacen estos routers es hacer un swap de la etiqueta.

Finalmente, el paquete llega el router LER de salida, el cual es el encargado de quitar la última etiqueta (pop) y enviar el paquete hacia su destino por routing convencional. En este punto, el paquete ya no es del tipo MPLS porque ya no tiene etiquetas. Este último paso suele realizarlo en penúltimo router de la

red (**Penultimate Hop Popping**). La razón de esto es para liberar al último router del trabajo, ya que este tiene que enrutar un paquete IP y si además tuviera que eliminar la etiqueta, tendría dos trabajos. De esta forma, el penúltimo router de la red MPLS hace un pop en el momento de enviar el paquete al interfaz que le indica la tabla LIB y el último router ya recibe un paquete IP convencional.

3.3.5. Fast Reroute

Existen muchos factores que pueden provocar un fallo en la red, pero a grandes rasgos y a nivel del router se pueden resumir en fallos en los enlaces o en los nodos. Estos fallos pueden repercutir en la pérdida de paquetes por parte de la red.

MPLS introduce el mecanismo de Fast Reroute para redirigir el tráfico por nuevas rutas no definidas por el protocolo IGP y minimizar el número de paquetes perdidos.

Normalmente, cuando se produce un fallo en un enlace o un nodo, se señaliza en las cabeceras de los LSPs que usan ese enlace o nodo. En ese momento el protocolo IGP recalcula la ruta para redirigir los paquetes. En este tiempo se pueden producir las pérdidas de paquetes, las cuales pueden ser significativas en aplicaciones en tiempo real como la voz o el video.

El mecanismo, aunque no asegura no tener pérdidas de paquetes, si que las minimiza. Para poder usar este mecanismo eficientemente, se tiene que configurar un camino principal por donde se enrutará el tráfico, y simultáneamente implementar un camino de backup para dotar al enlace, o nodo de redundancia. Cuando se detecte un fallo en el enlace principal, Fast Reroute desviarán el tráfico hacia el camino de backup, que como ya estaba configurado, no perderá tiempo en recalcular la ruta.

Lamentablemente, solo uno de los routers de los que disponemos en el laboratorio puede soportar la versión adecuada de IOS donde se implementó Fast Reroute, y por lo tanto se ha tenido que implementar otros métodos para dotar de backup a la red, que aunque no son tan eficientes como el Fast Reroute, se han intentado ajustar para que sean lo más parecido posibles.

4. QoS en redes MPLS

4.1. Conceptos básicos

El término de QoS apareció en 1987 para englobar las características del rendimiento de la red. Con la extensión de IP, el término QoS pasó a ser usado para describir las técnicas encargadas de controlar la **pérdida de paquetes**, el retraso (**delay**) y el **jitter**. Con el aumento del ancho de banda que consumen las aplicaciones y su demanda de menor retardo y pérdida de paquetes, el QoS se ha convertido en un criterio muy importante a la hora de definir una red.

La **pérdida de paquetes**, como su nombre indica, se encarga de controlar el número de paquetes que alcanzan su destino correctamente.

El **retraso**, o **delay**, controla el tiempo que tardan los paquetes en llegar a su destino.

Y finalmente el **jitter**, controla las fluctuaciones del tráfico, es decir, que los paquetes lleguen “uniformemente” a su destino.

Así pues, dependiendo del tipo de aplicación que queramos, serán necesarios unos criterios u otros de QoS. Una videoconferencia, por ejemplo, necesita un retraso y un jitter muy bajo, pero no es crítico si se pierden algunos paquetes. Por otro lado, el envío de un email requiere que no se pierdan paquetes, ya que algún error en estos podría provocar que se recibiera un mensaje totalmente distinto al original. El delay en este caso es muy poco importante.

A día de hoy, hay dos grandes arquitecturas para QoS:

- Integrated Services (**IntServ**): Se usa para redes pequeñas y medianas, pero no es escalable ya que usa mucha señalización entre los hosts de la red.
- Differentiated Services (**DiffServ**): Si que es escalable, ya que se basa en una clasificación previa de los paquetes, de forma que se reduce mucho la señalización.

Como MPLS está pensado para ser usado como una red troncal o “backbone”, nos decantaremos por usar la arquitectura de DiffServ.

4.2. Servicios Diferenciados (Diffserv)

El modelo de arquitectura DiffServ está especificado en el RFC 2475, y permite distinguir diferentes clases de servicio marcando los paquetes.

El tráfico entra en la red, se clasifica y se asigna a un conjunto de comportamiento. Cada uno de estos conjuntos se identifica con un *codepoint* DS que se añade a la cabecera del paquete. Luego, estos paquetes son enviados por la red en función de lo que decida cada nodo en referencia a dicho *codepoint*.

Según los requisitos de cada usuario, Diffserv permite diferenciar distintos servicios como tráfico web, correo electrónico o transferencia de ficheros, donde el retardo no es muy importante, o servicios como videollamada y VoIP donde sí lo es.

4.2.1. Diffserv y paquetes IP

Diffserv tiene un campo en los encabezados de los paquetes IP conocido como DiffServ CodePoint (DSCP). Los hosts o routers que envían el tráfico a una red diffserv, marcan los paquetes IP con un valor DSCP, y los routers de la red, clasifican estos paquetes en función de dicho valor. Los tráficos con requisitos de QoS parecidos son marcados de igual forma.

Este proceso de envío de paquetes con el campo DSCP modificado desde los routers, se conoce como “Per Hop Behavior” PHB, o comportamiento por salto. Estos PHB nos indican las políticas aplicadas en los routers, la gestión del tráfico o los encolamientos.

Para proporcionar diferentes niveles de servicio, el campo DSCP consta de 8 bits, estando los dos últimos reservados. Estos 6 bits útiles nos dan un total de 64 combinaciones distintas para clasificar los paquetes.

Pero este marcado de los paquetes IP para ofrecer QoS ha ido variando a lo largo del tiempo. La cabecera IP siempre ha tenido 8 bits destinados a ofrecer este servicio, pero ha ido evolucionando tal como se muestra a continuación con los 4 primeros bytes de la cabecera:

Version (4)	IHL(4)	PRE (3)	ToS (3)	0	0	Total Length (16)
-------------	--------	---------	---------	---	---	-------------------

RFC 791 (1981)

Version (4)	IHL(4)	PRE (3)	ToS (4)	0	Total Length (16)
-------------	--------	---------	---------	---	-------------------

RFC 1349 (1992)

Version (4)	IHL(4)	DSCP (6)	0	0	Total Length (16)
-------------	--------	----------	---	---	-------------------

RFC 2474 (1998)

Version (4)	IHL(4)	DSCP (6)	ECN (2)	Total Length (16)
-------------	--------	----------	---------	-------------------

RFC 3168 (2001)

Originalmente en la cabecera existían 3 bits de precedencia (PRE) y 3 bits de “Type of Service”, con 2 bits no utilizados. Los bits de PRE se usaban para tomar decisiones para el tratamiento del paquete y los de ToS se querían usar para marcar los paquetes a los que queríamos darle un trato especial, pero las arquitecturas nunca se prepararon para usarlo.

A partir del RFC 2474, se unieron el PRE y ToS para formar el DSCP que conocemos actualmente (6 bits), y más tarde, los dos bits en desuso, para definir el “Explicit Congestion Notification” ECN.

4.2.2. DiffServ y paquetes MPLS

En la cabecera de los paquetes MPLS, tenemos el campo EXP para controlar el QoS. Como hemos podido observar, la cabecera IP tiene 6 bits destinados al DSCP para clasificar los distintos paquetes, pero la cabecera MPLS solo dispone de 3 bits de EXP. Por lo tanto se tendrán que mapear las distintas 64 clases en las 8 que permite MPLS. Esto no es un gran problema, ya que 8 clases de servicio suelen ser más que suficiente.

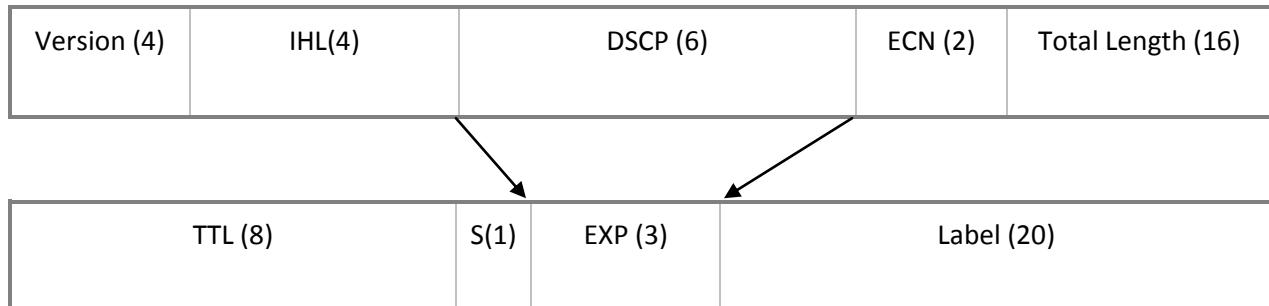


Figura 8: Detalle y relación de las cabeceras IP y MPLS

Por defecto, cuando un paquete llega a la red, el router MPLS de ingreso encapsula el paquete IP con su etiqueta correspondiente y, el campo EXP con los 3 primeros bits del campo DSCP (los 3 bits más significativos). Luego, cuando el paquete MPLS viaja por la red, se va copiando el valor del campo EXP en la etiqueta más externa de la Pila de Etiquetas. Así pues, el mapeo que se realizará será el siguiente:

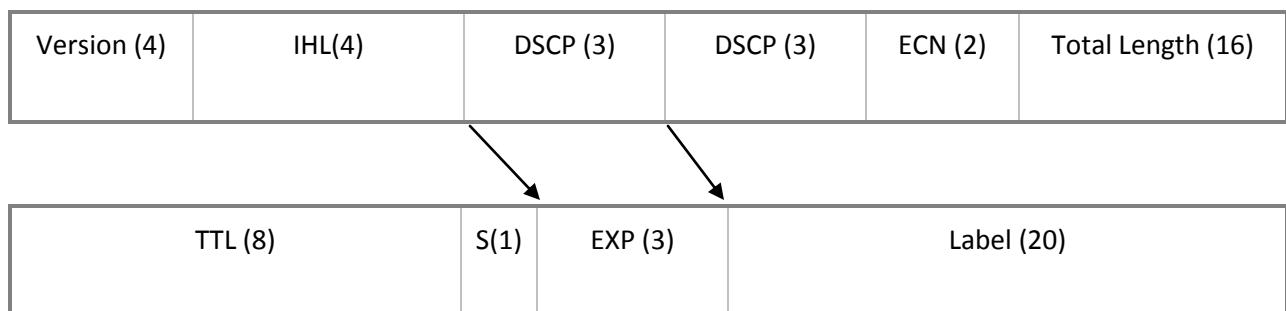


Figura 9: Mapeo del campo DSCP en el EXP

Cabe destacar, que paquetes con distintos DSCP, pero con los 3 primeros bits de este iguales, tendrán el mismo valor de EXP, y por lo tanto serán tratados de igual forma por la red MPLS.

Para que esto no ocurra, podemos definir un PHB para que modifique el valor del EXP en función de todo el valor del campo DSCP (6 bits). Entonces, cuando un paquete llegue a una red MPLS, el PHB asignará un valor preestablecido al campo EXP del nuevo paquete MPLS, y otro PHB podrá actuar para ese valor de EXP.

Es importante darse cuenta que los PHB definidos para los DSCP no tendrán efecto dentro de una red MPLS, ya que la ventaja de esta red es que no revisa los valores del paquete IPs, y solo mira los valores del MPLS. Por lo tanto se tendrán que definir PHB para los valores de EXP.

5. Configuración

En este capítulo se va a explicar cómo configurar los distintos routers y aplicaciones para poder realizar las distintas pruebas del capítulo 6.

5.1. Escenario básico:

El escenario está compuesto por tres routers que configuran dos tramos MPLS, un Hub entre cada uno de los routers y un host de origen y destino en los extremos de la red. También se ha incluido un "marcador" entre el host de origen y el primer router MPLS de la red para poder diferenciar distintos tipos de tráfico. Finalmente se ha conectado un Analizador a los Hubs para poder observar los paquetes cuando viajan dentro de la red. La siguiente figura ilustra el escenario implementado.

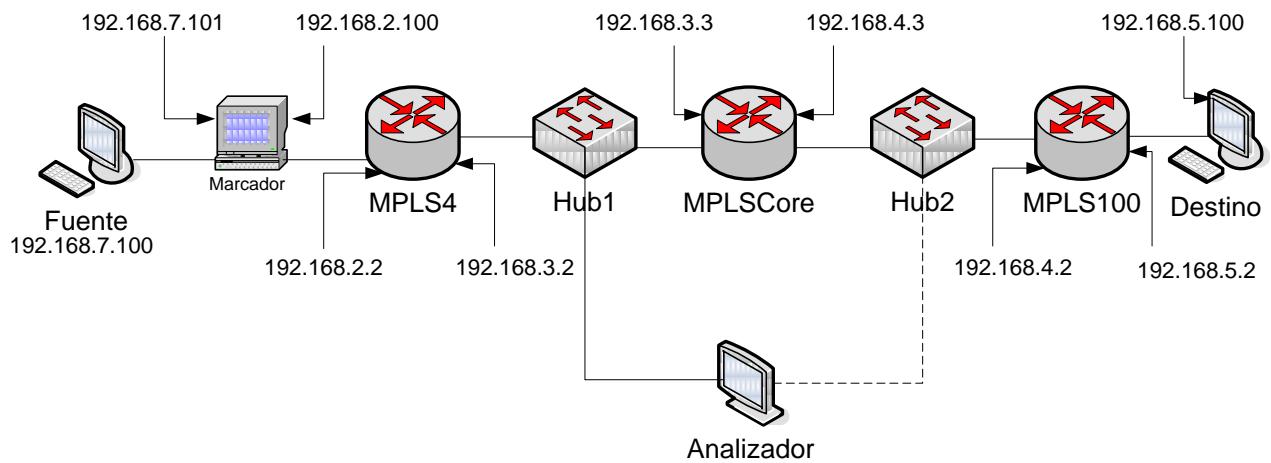


Figura 10: Escenario básico

La razón por la que se ha elegido este escenario es porque es el mínimo para poder observar correctamente la conectividad extremo-extremo de la red MPLS.

Como se ha explicado en el capítulo 3.3.4, es el penúltimo router de la red el que quita la etiqueta MPLS (**Penultimate Hop Popping**). Si solo tuviéramos 2 routers MPLS en la red, el router de entrada sería a su vez el penúltimo de salida. Al ser este el que añadiría (push) y quitaría (pop) la etiqueta, no podríamos observar su comportamiento. Por lo tanto el número mínimo de routers necesarios es el de 3.

También se han escogido Hubs, en lugar de Switches, para que actúen de repetidores y envíen una copia de cada paquete al Analizador. Para ganar esta funcionalidad, tenemos que sacrificar un poco la eficiencia de la red, ya que con un Hub se producen más colisiones de paquetes (más pérdidas) que con un Switch.

Finalmente, se ha incluido el “marcador”, como ya hemos dicho, para diferenciar distintos tipos de tráfico. El ordenador “marcador” dispone del software Xmarker, desarrollado por el Dr. Xavier Hesselbach en 2008 e implementado por Marc Suñé, que nos permite modificar el campo DSCP de los paquetes IP siguiendo un conjunto de reglas que definiremos.

5.2. Configuración del escenario:

Nuestro objetivo es establecer un enlace MPLS que esté capacitado para ofrecer Traffic Engineering, por eso vamos a orientar la configuración para que soporte la implementación de DiffServ

Para trabajar con una red MPLS con Traffic Engineering debemos comprobar que la red tiene habilitada una serie de protocolos:

- La red tiene que tener habilitado el protocolo CEF (Cisco Express Forwarding)
- Un protocolo de routing, en nuestro caso OSPF
- Una interfaz de Loopback para poder ser usada como router ID (RID)

5.2.1. Reseteo de la configuración de los routers:

El primer paso que realizamos fue el de resetear los routers para eliminar cualquier configuración de usos anteriores. Además, los tres routers usados disponían de contraseñas tanto para conectarse a ellos como para editar sus configuraciones, y estas contraseñas eran desconocidas para nosotros. Para ellos realizamos los siguientes pasos:

Primero reiniciamos el router manualmente.

Antes de que pasen 60 segundos pulsamos **Ctrl+Break** para parar la carga de la configuración.

RouterMPLS # confreg 0x2142

RouterMPLS # reset

Al volver a reiniciar el router nos pregunta si queremos realizar una configuración básica siguiendo un plantilla. Le decimos que no.

5.2.2. Configuración básica del router:

Para empezar la configuración le asignaremos un nombre al router, así como contraseñas para la consola y para editar la configuración.

Router > enable

Router # configure terminal

Router (config) # hostname MPLS4 //Introducimos el nombre de MPLS4 para el router

MPLS4 (config) # **line con 0**

MPLS4 (config - con) # **password**

MPLS4 (config - con) # <**contraseña de la consola**> //Definimos el password de la consola

MPLS4 (config - con) # **login**

MPLS4 (config - con) # **exit**

MPLS4 (config) # **enable secret <contraseña enable>** //Habilitamos el password del enable

Finalmente guardamos todos los cambios en la memoria.

MPLS4 (config) # **config-register 0x2102**

Pulsamos **Ctrl+z** para salir del modo configuración

MPLS4 # **write memory** // Guardamos los cambios

5.2.3. Configuración básica para IP y MPLS

Ahora que ya tenemos configurado el router de forma básica, procederemos a configurar sus interfaces y a habilitar MPLS.

Para empezar ejecutaremos unos comandos para habilitar el enrutamiento MPLS en el router de manera global.

MPLS4 # **config terminal**

MPLS4 (config) # **ip cef** //Habilitamos el protocolo CEF en el router

MPLS4 (config) # mpls label protocol ldp //Definimos el protocolo LDP como protocolo para la distribución de las etiquetas

MPLS4 (config) # mpls ip //Habilitamos MPLS a nivel global

Una vez configurado el protocolo que usará el router para realizar la conmutación de etiquetas (CEF) y su distribución (LDP), procedemos a realizar la configuración de OSPF para poder hacer el routing interno.

MPLS4 (config) # router ospf 1 // Configuramos el enrutamiento interno con OSPF con el identificador 1

MPLS4 (config-router) # mpls traffic-eng router-id loopback0 //Usaremos la interfaz de Loopback como identificador del router para Traffic Engineering

MPLS4 (config-router) # mpls traffic-eng area 0 // Configuramos el area 0 como la area en la que habilitamos el traffic engineering

MPLS4 (config-router) # network 192.168.1.1 0.0.0.0 area 0 // Habilitamos el interfaz de Loopback para usar OSPF y lo asignamos al area 0

MPLS4 (config-router) # network 192.168.3.0 0.0.0.255 area 0 // Habilitamos la subred 192.168.3.0/24 para usar OSPF y lo asignamos al area 0

MPLS4 (config-router) # exit

La configuración de los otros dos routers es prácticamente la misma. Simplemente tenemos que realizar los siguientes cambios:

MPLSCore

Sustituimos *network 192.168.1.1 0.0.0.0 area 0* por *192.168.1.2 0.0.0.0 area 0* ya que su interfaz de Loopback es la 192.168.1.2 y no la 192.168.1.1

Añadimos el comando *network 192.168.4.0 0.0.0.255 area 0* ya que el router MPLSCore tiene dos interfaces vecinas que realizan MPLS.

MPLS100

Sustituimos *network 192.168.1.1 0.0.0.0 area 0* por *192.168.1.3 0.0.0.0 area 0* ya que su interfaz de Loopback es la 192.168.1.3 y no la 192.168.1.1

También tenemos que sustituir el comando *network 192.168.3.0 0.0.0.255 area 0* por el *network 192.168.4.0 0.0.0.255 area 0* ya que la red 192.168.4.0 es la adyacente a este router y no la 192.168.3.0

Finalmente solo nos queda configurar las interfaces de los routers.

MPLS4 (config) # interface loopback0 // Accedemos al interfaz de Loopback

MPLS4 (config-if) # ip address 192.168.1.1 255.255.255.255 //Le asignamos una IP y mascara de subred

MPLS4 (config-if) # exit

MPLS4 (config) # interface f0/0 // Accedemos al interfaz f0/0

MPLS4 (config-if) # ip address 192.168.2.2 255.255.255.0 //Le asignamos una IP y mascara de subred

MPLS4 (config-if) # no shutdown // Habilitamos el interfaz

MPLS4 (config-if) # exit

```
MPLS4 (config) # interface f0/1 // Accedemos al interfaz f0/1
```

```
MPLS4 (config-if) # mpls ip //Habilitamos MPLS en el interfaz
```

```
MPLS4 (config-if) # ip address 192.168.3.2 255.255.255.0 //Le asignamos una IP y mascara de subred
```

```
MPLS4 (config-if) # no shutdown // Habilitamos el interfaz
```

```
MPLS4 (config-if) # exit
```

```
MPLS4 (config) # ip route 192.168.5.0 255.255.255.0 192.168.3.0
```

```
MPLS4 (config-if) # ip route 192.168.7.0 255.255.255.0 192.168.2.0
```

En este último comando hemos añadido la ruta estática para que el router sepa por donde enviar los paquetes con destino a la subred 192.168.5.0/24 y a la 192.168.7.0/24. También podemos ver como se ha habilitado el protocolo MPLS en la subred 192.168.3.0/24 ya que es la única que es adyacente a la red MPLS.

La configuración de los otros dos routers son parecidas.

```
MPLSCore (config) # interface loopback0
```

```
MPLSCore (config-if) # ip address 192.168.1.2 255.255.255.255
```

```
MPLSCore (config-if) # exit
```

```
MPLSCore (config) # interface GigabitEthernet0/0
```

```
MPLSCore (config-if) # mpls ip
```

```
MPLSCore (config-if) # ip address 192.168.3.3 255.255.255.0
```

```
MPLSCore (config-if) # no shutdown
```

```
MPLSCore (config-if) # exit
```

```
MPLSCore (config) # interface GigabitEthernet0/1
```

```
MPLSCore (config-if) # mpls ip
```

```
MPLSCore (config-if) # ip address 192.168.4.3 255.255.255.0
```

```
MPLSCore (config-if) # no shutdown
```

```
MPLSCore (config-if) # exit
```

```
MPLSCore (config) # ip route 192.168.5.0 255.255.255.0 192.168.4.0
```

```
MPLSCore (config) # ip route 192.168.2.0 255.255.255.0 192.168.3.0
```

```
MPLSCore (config) # ip route 192.168.7.0 255.255.255.0 192.168.3.0
```

```
MPLS100 (config) # interface loopback0
```

```
MPLS100 (config-if) # ip address 192.168.1.3 255.255.255.255
```

```
MPLS100 (config-if) # exit
```

```
MPLS100 (config) # interface f0/0
```

```
MPLS100 (config-if) # mpls ip
```

```
MPLS100 (config-if) # ip address 192.168.4.2 255.255.255.0
```

```
MPLS100 (config-if) # no shutdown
```

```
MPLS100 (config-if) # exit
```

```
MPLS100 (config) # interface f0/1
```

```
MPLS100 (config-if) # ip address 192.168.5.2 255.255.255.0
```

```
MPLS100 (config-if) # no shutdown
```

```
MPLS100 (config-if) # exit
```

```
MPLS100 (config) # ip route 192.168.2.0 255.255.255.0 192.168.3.0
```

```
MPLS100 (config) # ip route 192.168.7.0 255.255.255.0 192.168.3.0
```

Una vez tenemos los tres routers configurados, vamos a ver cómo han quedado sus configuraciones usando los siguientes comandos.

```
MPLS4# show ip route // Nos muestra la tabla de rutas ip
```

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2

i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2

ia - IS-IS inter area, * - candidate default, U - per-user static route

o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

O 192.168.4.0/24 [110/11] via 192.168.3.3, 00:02:19, FastEthernet0/1

S 192.168.5.0/24 [1/0] via 192.168.3.3

S 192.168.7.0/24 [1/0] via 192.168.2.101

192.168.0.0/32 is subnetted, 3 subnets

C 192.168.0.1 is directly connected, Loopback0

O 192.168.0.2 [110/11] via 192.168.3.3, 00:02:19, FastEthernet0/1

O 192.168.0.3 [110/12] via 192.168.3.3, 00:02:19, FastEthernet0/1

C 192.168.2.0/24 is directly connected, FastEthernet0/0

C 192.168.3.0/24 is directly connected, FastEthernet0/1

Las rutas marcadas con una C son las que están conectadas directamente al router, las marcadas con una S son las rutas estáticas que hemos definido, y finalmente, las rutas marcadas con una O son las rutas que se han obtenido del protocolo OSPF.

Para ver el comportamiento del protocolo MPLS que hemos configurado, ejecutamos:

MPLS4# show mpls forwarding-table

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing Interface	Next Hop
16	17	192.168.5.0/24	0	Fa0/1	192.168.3.3
17	Untagged	192.168.7.0/24	296	Fa0/0	192.168.2.101
18	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.3

19	Pop tag	192.168.0.2/32	0	Fa0/1	192.168.3.3
20	20	192.168.0.3/32	0	Fa0/1	192.168.3.3

Tabla 1: LIB de MPLS4 en el escenario básico

Observamos como los paquetes con destino a la subred 192.168.5.0/24 cambiarán su etiqueta del 16 al 17 al pasar por este router. Por otro lado, los paquetes con destino a la subred 192.168.7.0/24 no serán etiquetados ya que esa red está fuera de la red MPLS.

Análogamente, si ejecutamos estos comandos en los otros dos routers, obtendremos su configuración.

MPLSCore# show ip route

Gateway of last resort is not set

C 192.168.4.0/24 is directly connected, GigabitEthernet0/1

S 192.168.5.0/24 [1/0] via 192.168.4.2

S 192.168.7.0/24 [1/0] via 192.168.3.2

192.168.0.0/32 is subnetted, 3 subnets

O 192.168.0.1 [110/11] via 192.168.3.2, 00:07:08, GigabitEthernet0/0

C 192.168.0.2 is directly connected, Loopback0

O 192.168.0.3 [110/2] via 192.168.4.2, 00:07:08, GigabitEthernet0/1

S 192.168.2.0/24 [1/0] via 192.168.3.2

C 192.168.3.0/24 is directly connected, GigabitEthernet0/0

MPLS100# show mpls forwarding-table

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing	
				interface	Next Hop
16	17	192.168.7.0/24	1227842	Gi0/0	192.168.3.2
17	Pop tag	192.168.5.0/24	56196236	Gi0/1	192.168.4.2
18	Pop tag	192.168.2.0/24	0	Gi0/0	192.168.3.2
19	Pop tag	192.168.0.1/32	0	Gi0/0	192.168.3.2
20	Pop tag	192.168.0.3/32	0	Gi0/1	192.168.4.2

Tabla 2: LIB de MPLS100 en el escenario básico

Cabe destacar que como es el penúltimo router el que elimina la etiqueta, será el router central el que haga esta acción en nuestro escenario, tal como muestra la tabla anterior.

MPLS100# show ip route

- C 192.168.4.0/24 is directly connected, FastEthernet0/0
- C 192.168.5.0/24 is directly connected, FastEthernet0/1
- S 192.168.7.0/24 [1/0] via 192.168.4.3
 - 192.168.0.0/32 is subnetted, 3 subnets
 - O 192.168.0.1 [110/12] via 192.168.4.3, 00:08:51, FastEthernet0/0
 - O 192.168.0.2 [110/2] via 192.168.4.3, 00:08:51, FastEthernet0/0
- C 192.168.0.3 is directly connected, Loopback0
- S 192.168.2.0/24 [1/0] via 192.168.4.3

O 192.168.3.0/24 [110/11] via 192.168.4.3, 00:08:51, FastEthernet0/0

MPLS100# show mpls forwarding-table

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing	
				interface	Next Hop
16	16	192.168.7.0/24	0	Fa0/0	192.168.4.3
17	18	192.168.2.0/24	0	Fa0/0	192.168.4.3
18	Pop tag	192.168.3.0/24	0	Fa0/0	192.168.4.3
19	19	192.168.0.1/32	0	Fa0/0	192.168.4.3
20	Pop tag	192.168.0.2/32	0	Fa0/0	192.168.4.3

Tabla 3: LIB de MPLS100 en el escenario básico

5.2.4. Configuración del “marcador”

Al realizar el envío del vídeo también marcaremos los paquetes en su campo DSCP. Esto nos permitirá diferenciar distintos flujos de tráfico en el futuro.

Primero de todo, y como el ordenador usa una distribución del Sistema Operativo Linux, vamos a habilitar la opción de enrutar los paquetes que lleguen a sus interfaces. Para ello ejecutamos el siguiente comando:

```
root@marcador: home/Jordi# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Ahora que el ordenador ya se comporta como un router, configuraremos sus interfaces utilizando el asistente que nos ofrece el Sistema Operativo.

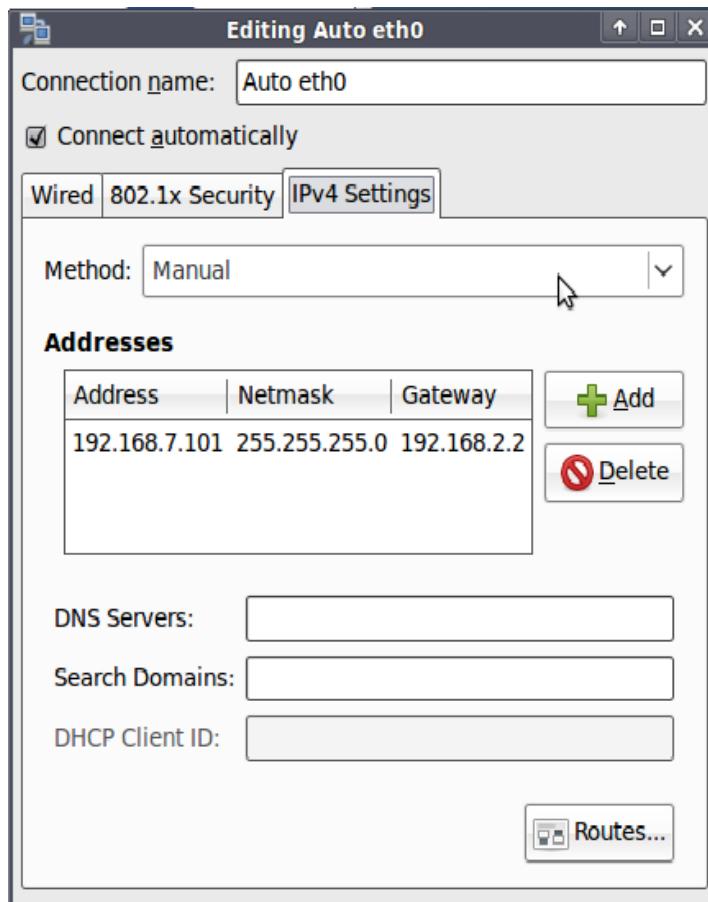


Figura 11: Configuración del interfaz

De igual forma configuraremos el interface eth1, con la dirección 192.168.2.100.

Como último paso, en el ordenador “marcador” disponemos del software Xmarker, el cual nos permite asignar “reglas” a los distintos interfaces de la máquina. En nuestro caso crearemos una regla que nos marque el campo DSCP de los paquetes que entran por el interfaz eth0 (red 192.168.7.0/24) con destino al Host “Destino” (192.168.5.100), con el valor CS4 (0x20).

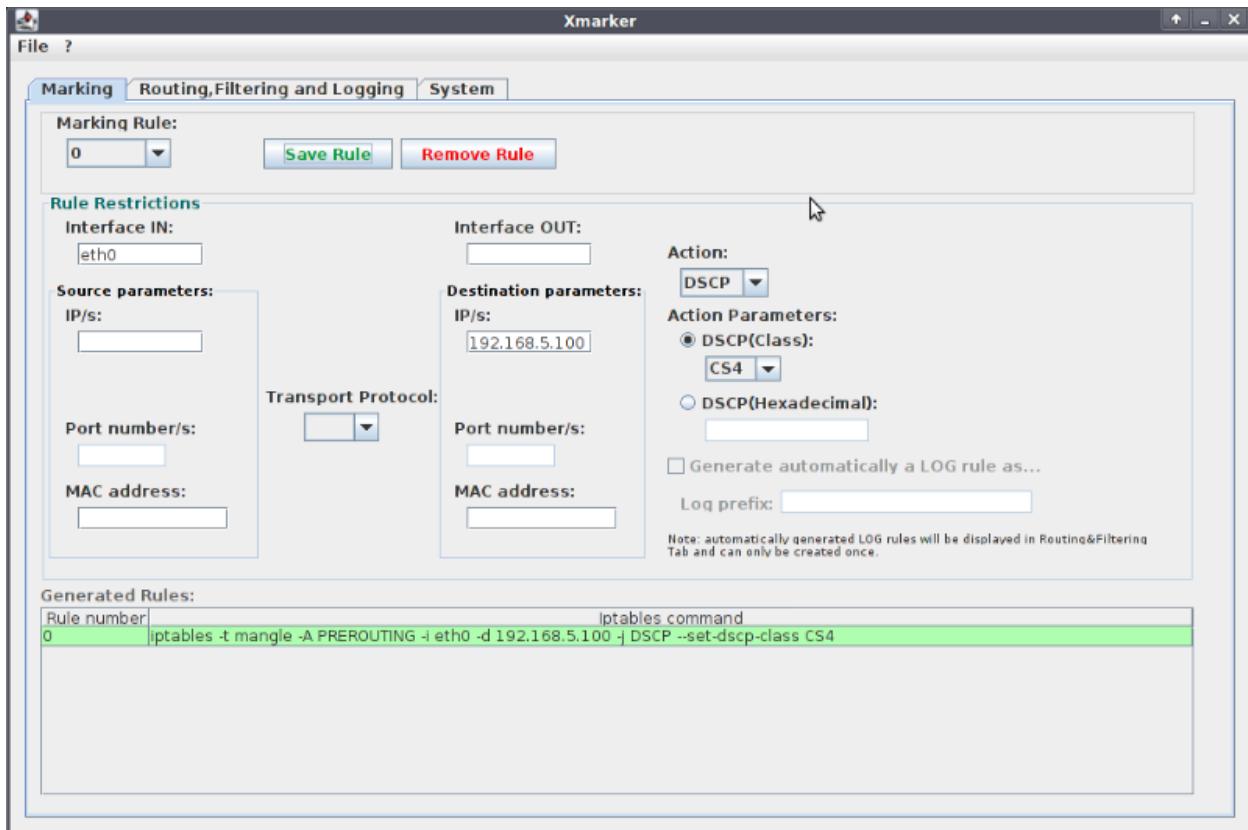


Figura 12: Configuración de las reglas en el marcador

5.3. Pruebas de conexión extremo-extremo

Una vez tenemos configurado todo el escenario, debemos comprobar que funciona la conexión extremo-extremo en ambos sentidos.

Se entiende como conexión extremo-extremo aquella que va desde el Host “Fuente” al Host “Destino” pasando a través de los 3 routers MPLS y del “marcador”.

Para ello enviaremos una serie de paquetes ICMP (Internet Control Message Protocol), o sea, pings a través de la red.

Nos conectamos al Host “Fuente” (192.168.7.100) y enviamos 4 paquetes al Host “Destino” (192.168.5.100)

```
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\jlores>ping 192.168.5.100

Haciendo ping a 192.168.5.100 con 32 bytes de datos:

Respuesta desde 192.168.5.100: bytes=32 tiempo=1ms TTL=124

Estadísticas de ping para 192.168.5.100:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 1ms, Máximo = 1ms, Media = 1ms

C:\Documents and Settings\jlores>
```

Figura 13: Ping desda la "Fuente" hacia el "Destino"

Podemos observar cómo se han recibido los 4 paquetes y ninguno se ha perdido, por lo tanto podemos confirmar que hay conectividad extremo-extremo.

Finalmente vamos a realizar la misma prueba pero en sentido contrario, del Host “Destino” al Host “Fuente”

```
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\jlores>ping 192.168.7.100

Haciendo ping a 192.168.7.100 con 32 bytes de datos:

Respuesta desde 192.168.7.100: bytes=32 tiempo=1ms TTL=124

Estadísticas de ping para 192.168.7.100:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 1ms, Máximo = 1ms, Media = 1ms

C:\Documents and Settings\jlores>
```

Figura 14: Ping desda el "Destino" hacia la "Fuente"

Igual que en el caso anterior la prueba de conexión ha sido un éxito.

5.4. Pruebas de envío de vídeo extremo-extremo

5.4.1. Configuración del VLC

Ahora que ya tenemos la red configurada y con conexión, vamos a enviar un archivo de vídeo por streaming con el programa VLC.

En el Host “Fuente” seleccionamos el vídeo a transmitir, protocolo (HTTP), por qué interfaz y puerto enviarlo (192.168.7.100:8080) y que no haga Transcoding.

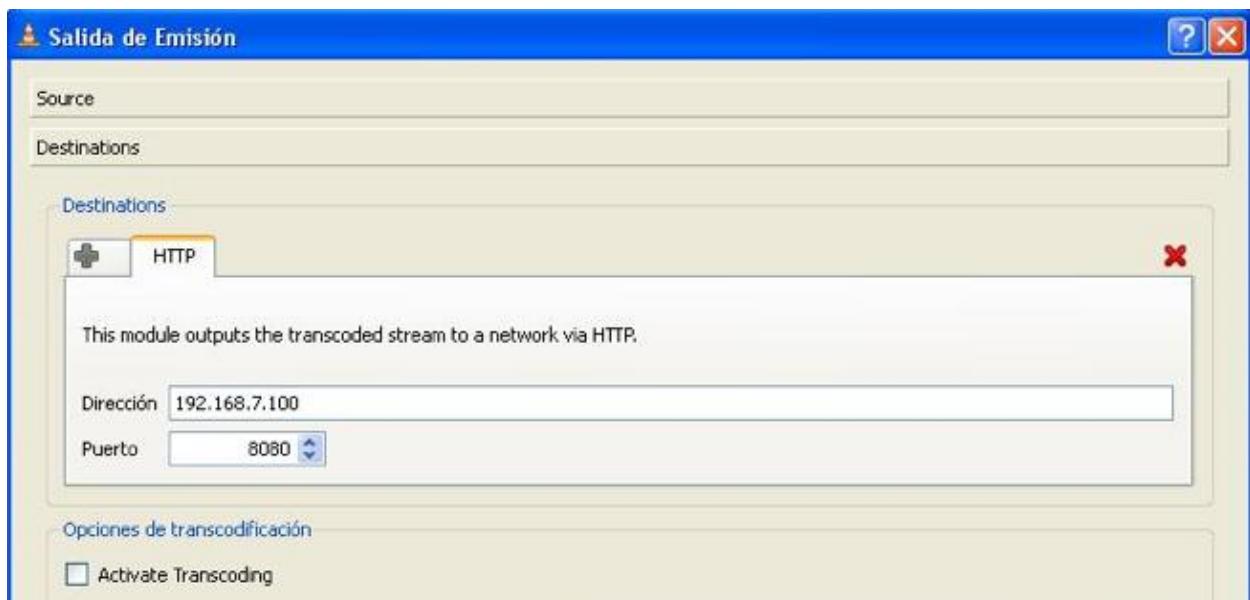


Figura 15: Configuración del VLC en la "Fuente"

En el Host “Destino” simplemente seleccionamos el protocolo y origen de los datos, los cuales deben ser igual a los de la fuente.

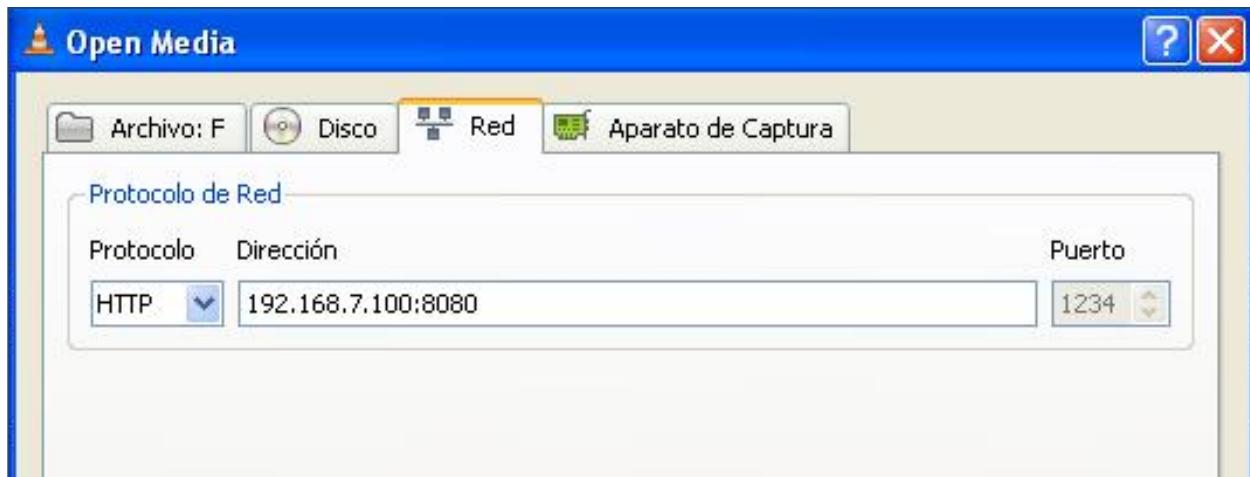


Figura 16: Configuración del VLC en el "Destino"

5.4.2. Resultados

Con esta configuración el vídeo se recibe correctamente en el Host “Destino” aunque con un pequeño retraso con respecto al vídeo original, inferior al segundo. Este hecho es lógico ya que los datos tienen que ser reenrutados por 2 routers y por el marcador y representarse de nuevo en el Host “Destino”.

Para poder observar los paquetes usaremos el Analizador conectado al Hub 1, ya que si los mirásemos en el Host “Destino” no podríamos ver los paquetes MPLS ya que estaríamos fuera de la red MPLS, pues el penúltimo router (MPLSCore) hace un Pop de las etiquetas.

Con el Analizador podemos ver el siguiente paquete:

```
No.      Time      Source          Destination        Protocol Info  
537 4.462910  192.168.7.100    192.168.5.100    HTTP      Continuation or non-HTTP t  
  
Frame 537 (1514 bytes on wire, 1514 bytes captured)  
Ethernet II Src: Cisco f7:d3:e1 (00:d0:bb:f7:d3:e1) Dst: Cisco a4:9a:78 (00:13:80:a4:9a:78)  
MultiProtocol Label Switching Header, Label: 17, Exp: 4, S: 1, TTL: 126  
  MPLS Label: 17  
  MPLS Experimental Bits: 4  
  MPLS Bottom Of Label Stack: 1  
  MPLS TTL: 126  
Internet Protocol, Src: 192.168.7.100 (192.168.7.100), Dst: 192.168.5.100 (192.168.5.100)  
  Version: 4  
  Header length: 20 bytes  
  Differentiated Services Field: 0x80 [DSCP 0x20: Class Selector 4; ECN: 0x00]  
  Total Length: 1496  
  Identification: 0x1ed5 (7893)  
  Flags: 0x04 (Don't Fragment)  
  Fragment offset: 0  
  Time to live: 126  
  Protocol: TCP (0x06)  
  Header checksum: 0x49b2 [correct]  
  Source: 192.168.7.100 (192.168.7.100)  
  Destination: 192.168.5.100 (192.168.5.100)  
Transmission Control Protocol, Src Port: http-alt (8080), Dst Port: bnetgame (1119), Seq: 516637, A  
Hypertext Transfer Protocol
```

Figura 17: Detalle de un paquete MPLS

Donde se puede observar que es un paquete MPLS con la etiqueta 17 y que es la única etiqueta del paquete, que su origen es el HOST “Fuente” y su destino el Host “Destino”. Que el paquete tenga la etiqueta 17 concuerda con la configuración que habíamos obtenido al ejecutar el comando **show mpls forwarding-table** en el router MPLS4. También se puede ver muy claramente en la cabecera IP como el campo DSCP está marcado con el valor 0x20 que se corresponde con el Class Selector 4, tal y como lo habíamos configurado.

5.5. Configuración del camino Backup

Una vez tenemos nuestra red funcionando, vamos a añadirle un nodo de Backup para dotar de redundancia al router MPLSCore. El nuevo escenario es el siguiente:

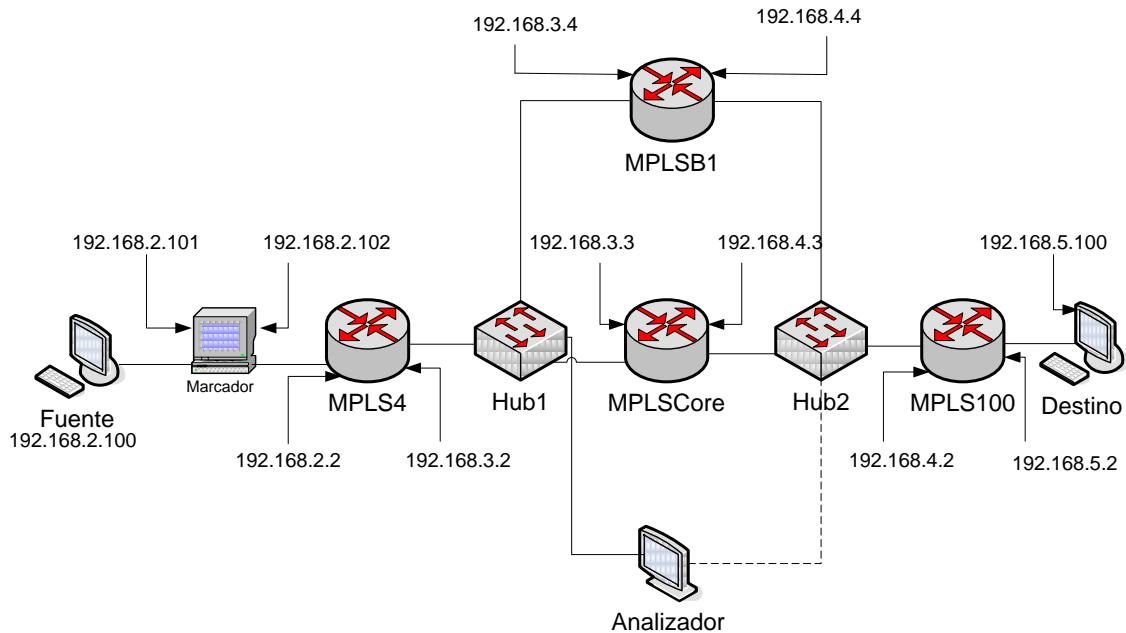


Figura 18: Escenario con Backup

Como se puede observar se ha añadido el router MPLSB1 entre los dos extremos, de modo que está en paralelo con el MPLSCore. También se han modificado las IP's de la Fuente y del Marcador para que este último actúe de una forma más “transparente” en la red.

El objetivo del router de Backup es que solo funcione cuando el router central (MPLSCore) no pueda hacerlo, ya sea por un problema en el propio router o en alguno de sus enlaces.

Como hemos visto anteriormente, la forma más sencilla de hacerlo sería aplicando el mecanismo de Fast Rerouteing (FRR), pero los modelos de los 3 primeros routers no los soportan. Para solucionarlo usaremos el protocolo OSPF, modificando el coste de los enlaces para que los routers solo vean el camino con menos coste.

El primer paso a realizar es configurar el router de Backup de la misma forma como hemos hecho con el MPLSCore, pero modificando las IP's de las interfaces. Así pues la única diferencia será:

Sustituir el comando `ip address 192.168.1.2 255.255.255.255` de la interfaz de loopback por el `ip address 192.168.1.4 255.255.255.255` y los comandos `ip address 192.168.3.3 255.255.255.0` y `ip address`

192.168.4.3 255.255.255.0 de sus otros interfaces por los *ip address 192.168.3.4 255.255.255.0* y *ip address 192.168.3.4 255.255.255.0* respectivamente.

Como ahora vamos a utilizar el protocolo OSPF para la realización del Backup, este protocolo será el encargado de construir toda la tabla de rutas, y por lo tanto no necesitaremos las rutas estáticas. En su lugar añadiremos las siguientes redes en las distintas configuraciones del OSPF de cada router:

MPLS4

MPLS4 (config-router) # network 192.168.2.0 0.0.0.255 area 0 // Habilitamos la subred 192.168.2.0/24 para usar OSPF y lo asignamos al area 0

MPLS4 (config-if) # ip route 192.168.2.100 255.255.255.0 192.168.2.101 // Solo tenemos que añadir esta ruta estática porque el marcador no usa el protocolo OSPF y no puede notificarlo al router MPLS4

MPLS100

MPLS100 (config-router) # network 192.168.5.0 0.0.0.255 area 0 // Habilitamos la subred 192.168.5.0/24 para usar OSPF y lo asignamos al area 0

Con esta configuración la red ya es capaz de transmitir un paquete desde el Host “Origen” hasta el Host “Destino” por el camino con menos coste, que ahora mismo es idéntico, ya sea a través del router MPLSCore como por el MPLSB1. Podemos ver cómo queda la tabla de rutas y de etiquetas MPLS en el router MPLS4:

MPLS4# show ip route

O 192.168.4.0/24 [110/2] via 192.168.3.4, 00:00:03, FastEthernet0/1

[110/2] via 192.168.3.3, 00:00:03, FastEthernet0/1

O 192.168.5.0/24 [110/3] via 192.168.3.4, 00:00:03, FastEthernet0/1

[110/3] via 192.168.3.3, 00:00:03, FastEthernet0/1

192.168.0.0/32 is subnetted, 4 subnets

- C 192.168.0.1 is directly connected, Loopback0
- O 192.168.0.2 [110/2] via 192.168.3.3, 00:00:03, FastEthernet0/1
- O 192.168.0.3 [110/3] via 192.168.3.4, 00:00:03, FastEthernet0/1
[110/3] via 192.168.3.3, 00:00:04, FastEthernet0/1
- O 192.168.0.4 [110/2] via 192.168.3.4, 00:00:04, FastEthernet0/1

192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks

- S 192.168.2.100/32 [1/0] via 192.168.2.101
- C 192.168.2.0/24 is directly connected, FastEthernet0/0
- C 192.168.3.0/24 is directly connected, FastEthernet0/1

Local tag	Outgoing tag or VC	Prefix	Bytes tag switched	Outgoing interface	
		or Tunnel Id		Next Hop	
16	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.4
	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.3
18	Pop tag	192.168.0.2/32	0	Fa0/1	192.168.3.3
19	16	192.168.0.3/32	0	Fa0/1	192.168.3.4
	18	192.168.0.3/32	0	Fa0/1	192.168.3.3
20	19	192.168.5.0/24	0	Fa0/1	192.168.3.4

	19	192.168.5.0/24	0	Fa0/1	192.168.3.3
21	Pop tag	192.168.0.4/32	0	Fa0/1	192.168.3.4
22	Untagged	192.168.2.100/32	0	Fa0/0	192.168.2.101

Tabla 4: LIB en MPLS4 con dos caminos

En este caso todo el tránsito con destino a la red 192.168.5.0/24 iría con la etiqueta 20 a través del router MPLSB1 (192.168.3.4), porque es la primera que se ha añadido a la red, y no podríamos decir que router priorizamos. Si hubiera un problema con el router MPLSB1 el tránsito se redirigiría automáticamente hacia el otro router (MPLSCore 192.168.3.3), pero no volvería al router MPLSB1 cuando volviera a estar online porque su etiqueta se añadiría debajo de la del salto 192.168.3.3 . Para solucionar este comportamiento, que no es el deseado, tenemos que modificar el coste de los interfaces.

Lo que haremos es aumentar el coste de los interfaces entre el router MPLS4 y MPLSB1, y entre el MPLSB1 y el MPLS100. De este modo, el protocolo OSPF encontrará dos caminos del Host “Origen” al Host “Destino”, pero uno de ellos con un coste superior al otro, y por tanto lo desestimará. Cuando el router MPLSCore falle, solo habrá un camino disponible (aunque tenga un coste alto) y todo el tráfico irá a través de él. En el momento en que se recupere el router MPLSCore, volveremos a estar en la primera situación y por lo tanto desestimaremos el camino con más coste.

Para modificar el coste, simplemente tenemos que entrar en la configuración del interfaz e introducir el siguiente comando:

```
MPLSB1 (config-if) # ip ospf cost 10 //Asignamos el coste del interfaz a 10
```

```
MPLSB1 (config-if) # ip ospf hello-interval 2 //Modificamos el intervalo entre los mensajes “hello”
```

```
MPLSB1 (config-if) # ip ospf dead-interval 5 //Modificamos el intervalo entre los mensajes “dead”
```

El intervalo de los mensajes “dead” es el tiempo que tarda el router para decidir que un router ya no está disponible, y por lo tanto eliminarlo de las tablas de enrutamiento. Por otro lado, el intervalo de

mensajes “hello” es el tiempo entre que el router envía mensajes a los otros routers para notificar que está activo. Estos dos tiempos tienen que ser los mismos en todos los interfaces de una misma subred, y por lo tanto se tendrán que modificar en los routers MPLS4 y MPLS100.

Con esta modificación, las rutas y etiquetas quedan del siguiente modo:

O 192.168.4.0/24 [110/2] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.5.0/24 [110/3] via 192.168.3.3, 00:10:44, FastEthernet0/1

192.168.0.0/32 is subnetted, 4 subnets

C 192.168.0.1 is directly connected, Loopback0

O 192.168.0.2 [110/2] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.0.3 [110/3] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.0.4 [110/2] via 192.168.3.4, 00:10:44, FastEthernet0/1

192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks

S 192.168.2.100/32 [1/0] via 192.168.2.101

C 192.168.2.0/24 is directly connected, FastEthernet0/0

C 192.168.3.0/24 is directly connected, FastEthernet0/1

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing interface	Next Hop
16	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.3
18	Pop tag	192.168.0.2/32	0	Fa0/1	192.168.3.3

19	18	192.168.0.3/32	0	Fa0/1	192.168.3.3
20	19	192.168.5.0/24	0	Fa0/1	192.168.3.3
21	Pop tag	192.168.0.4/32	0	Fa0/1	192.168.3.4
22	Untagged	192.168.2.100/32	0	Fa0/0	192.168.2.101

Tabla 5: LIB en MPLS4 con el Backup configurado

Se puede observar como el protocolo OSPF ha eliminado las rutas perteneciente al router MPLSB1 (Solo queda la ruta de su interfaz de Looback, ya que es el único router que tiene acceso a esa interfaz).

Con esta configuración todo el tránsito irá a través del router MPLSCore. Ahora, si desconectamos el interfaz 192.168.3.3 del router MPLSCore, el protocolo OSPF encontrará el único camino disponible:

- O 192.168.4.0/24 [110/11] via 192.168.3.4, 00:01:41, FastEthernet0/1
- O 192.168.5.0/24 [110/12] via 192.168.3.4, 00:01:41, FastEthernet0/1

192.168.0.0/32 is subnetted, 4 subnets

- C 192.168.0.1 is directly connected, Loopback0
- O 192.168.0.2 [110/12] via 192.168.3.4, 00:01:41, FastEthernet0/1
- O 192.168.0.3 [110/12] via 192.168.3.4, 00:01:41, FastEthernet0/1
- O 192.168.0.4 [110/2] via 192.168.3.4, 00:01:41, FastEthernet0/1

192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks

- S 192.168.2.100/32 [1/0] via 192.168.2.101
- C 192.168.2.0/24 is directly connected, FastEthernet0/0
- C 192.168.3.0/24 is directly connected, FastEthernet0/1

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing	
				interface	Next Hop
16	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.4
18	17	192.168.0.2/32	0	Fa0/1	192.168.3.4
19	16	192.168.0.3/32	0	Fa0/1	192.168.3.4
20	19	192.168.5.0/24	0	Fa0/1	192.168.3.4
21	Pop tag	192.168.0.4/32	0	Fa0/1	192.168.3.4
22	Untagged	192.168.2.100/32	0	Fa0/0	192.168.2.101

Tabla 6: LIB en MPLS4 con Backup active

Si ahora el interfaz original se vuelve a recuperar, todo el tránsito vuelve a él:

O 192.168.4.0/24 [110/2] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.5.0/24 [110/3] via 192.168.3.3, 00:10:44, FastEthernet0/1

192.168.0.0/32 is subnetted, 4 subnets

C 192.168.0.1 is directly connected, Loopback0

O 192.168.0.2 [110/2] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.0.3 [110/3] via 192.168.3.3, 00:10:44, FastEthernet0/1

O 192.168.0.4 [110/2] via 192.168.3.4, 00:10:44, FastEthernet0/1

192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks

S 192.168.2.100/32 [1/0] via 192.168.2.101

C 192.168.2.0/24 is directly connected, FastEthernet0/0

C 192.168.3.0/24 is directly connected, FastEthernet0/1

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag	Outgoing	
			switched	interface	Next Hop
16	Pop tag	192.168.4.0/24	0	Fa0/1	192.168.3.3
18	Pop tag	192.168.0.2/32	0	Fa0/1	192.168.3.3
19	18	192.168.0.3/32	0	Fa0/1	192.168.3.3
20	19	192.168.5.0/24	0	Fa0/1	192.168.3.3
21	Pop tag	192.168.0.4/32	0	Fa0/1	192.168.3.4
22	Untagged	192.168.2.100/32	0	Fa0/0	192.168.2.101

Tabla 7: LIB en MPLS4 con el enlace principal recuperado

Como se ha podido observar, el tránsito siempre, y únicamente, viaja a través del router MPLSCore que hemos definido como el router “principal”. Sólo en el caso de que el tráfico a través de este camino sea imposible, se usará el router de backup MPLSB1.

5.6. Configuración del Balanceo de Carga

Para configurar la opción del Balanceo de Carga tenemos dos opciones.

En la primera opción usaremos el mismo método que con el Backup: modificar el coste de los interfaces para que el tráfico viaje a través del router que indiquemos en cada momento.

De este modo, los comandos que tenemos que usar son los siguientes en cada interfaz del que queremos modificar el coste:

```
MPLSB1 (config-if) # ip ospf cost 10 //Asignamos el coste del interfaz a 10
```

```
MPLSB1 (config-if) # ip ospf hello-interval 2 //Modificamos el intervalo entre los mensajes "hello"
```

```
MPLSB1 (config-if) # ip ospf dead-interval 5 //Modificamos el intervalo entre los mensajes "dead"
```

Tenemos que tener en cuenta, como ya hemos dicho antes, que los intervalos de tiempo del hello y dead tienen que ser iguales en todos los interfaces de una misma red.

El tráfico viajará siempre por la ruta con menos coste y solo cambiará de ruta cuando cambiemos el coste manualmente.

Por otro lado, tenemos la opción de que el router envíe un paquete por cada una de las rutas disponibles, dividiendo así la carga en cada una de las rutas. Para configurar esta segunda opción, simplemente tenemos que asignar el mismo coste a cada una de las rutas y habilitar el siguiente comando en el interfaz que queremos que balancee la carga:

```
MPLS4 (config-if) # ip load-sharing per-packet //Indicamos al interfaz que balancee la carga paquete a paquete
```

Si por ejemplo tuviéramos 3 rutas disponibles, y de igual coste, los paquetes se distribuirían de la siguiente manera:

PAQUETES	RUTAS
Paquete 1	Ruta 1
Paquete 2	Ruta 2
Paquete 3	Ruta 3
Paquete 4	Ruta 1
Paquete 5	Ruta 2
....

Tabla 8: Ejemplo de funcionamiento del balanceo por paquete

Es posible que este método implique que se reciban los paquetes en destino de manera desordenada, ya que una ruta puede ser más rápida que otra. Será responsabilidad del protocolo de transporte, o de la aplicación, reordenar los paquetes.

5.7. Configuración de un punto de acceso Wifi

Con la red ya configurada y con un camino de Backup para darle redundancia, el siguiente paso a seguir es añadirle un punto de acceso Wifi al final de los routers. De esta manera, el escenario es el siguiente:

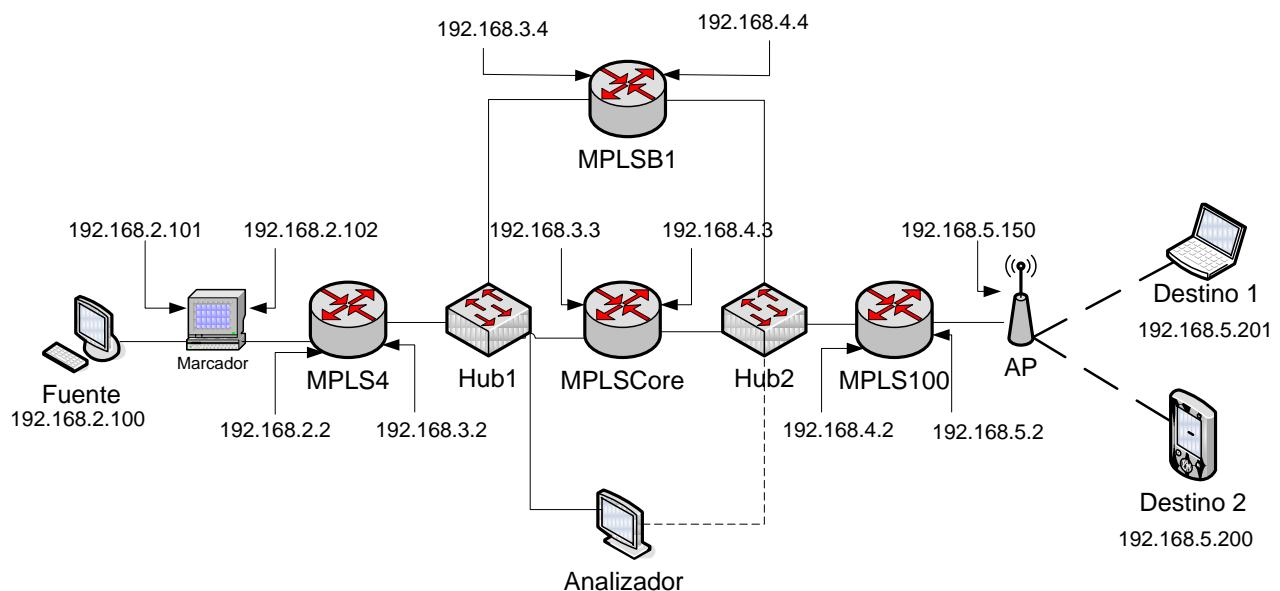


Figura 19: Escenario con Backup y WiFi

Como lo que queremos es ver cómo se comporta el tráfico MPLS a través de un punto de acceso inalámbrico, vamos a configurar este de la manera más sencilla posible: Sin autenticación ni protocolo DHCP. Por lo tanto el cliente que se conecte al punto WiFi, tendrá que introducir manualmente su ip, máscara de subred y Gateway.

De la misma forma como hemos configurado los routers, vamos a realizar la configuración básica del punto de acceso.

AccessPoint > enable

AccessPoint # configure terminal

AccessPoint (config) # hostname ap //Introducimos el nombre de ap al Access Point

ap (config)# ip default-gateway 192.168.5.2 //Como el Access Pont no es un router, necesitamos indicarle en que ip se encuentra el router.

ap (config)# interface FastEthernet0 //Entramos en el modo conf-if del interfaz conectado al router

ap (config-if)# ip address 192.168.5.150 255.255.255.0

ap (config-if)# exit

ap (config)# interface Dot11Radio0 // Entramos en el interfaz radio

ap (config-if)# ssid tsunami //Le asignamos un nombre a la red inalámbrica

ap (config-if)# authentication open //Quitamos la seguridad de la red

ap (config-if)# exit

Una vez tenemos configurado el punto de acceso, pasmos a configurar los terminales clientes.

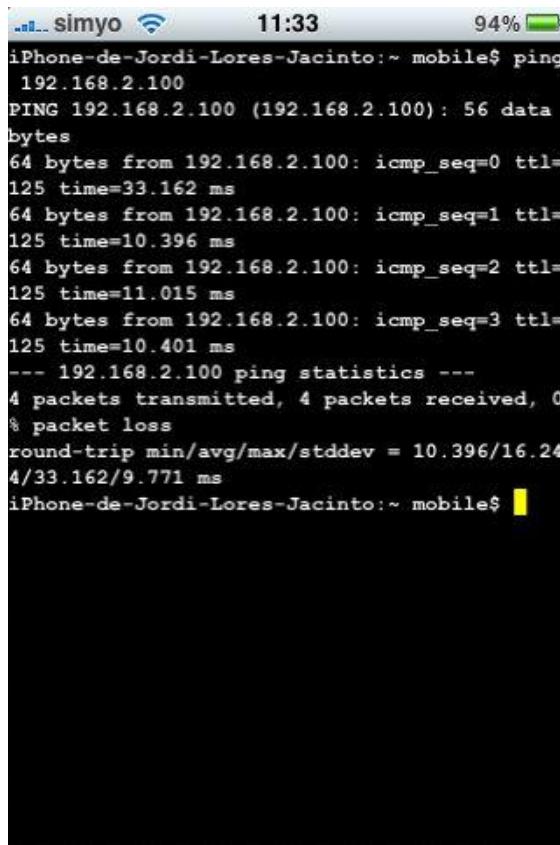
Para probar el correcto funcionamiento de la interfaz, hemos utilizado un iPhone:



Figura 20: Configuración de red del interfaz móvil

La dirección del Gateway (o router) debe ser la del router y no la ip del punto de acceso, ya que este solo interviene para realizar la conversión de medios de los paquetes, y no para enrutarlos.

Con esta configuración hemos realizado un ping a la Fuente (192.168.2.100) para comprobar que la transmisión es correcta.



```
simyo 11:33 94%
iPhone-de-Jordi-Lores-Jacinto:~ mobile$ ping
192.168.2.100
PING 192.168.2.100 (192.168.2.100): 56 data
bytes
64 bytes from 192.168.2.100: icmp_seq=0 ttl=
125 time=33.162 ms
64 bytes from 192.168.2.100: icmp_seq=1 ttl=
125 time=10.396 ms
64 bytes from 192.168.2.100: icmp_seq=2 ttl=
125 time=11.015 ms
64 bytes from 192.168.2.100: icmp_seq=3 ttl=
125 time=10.401 ms
--- 192.168.2.100 ping statistics ---
4 packets transmitted, 4 packets received, 0
% packet loss
round-trip min/avg/max/stddev = 10.396/16.24
4/33.162/9.771 ms
iPhone-de-Jordi-Lores-Jacinto:~ mobile$
```

Figura 21: Ping entre el dispositivo móvil y la "Fuente"

5.8. Configuración de Túneles IP-IP

Para que tráfico proveniente de Internet pueda atravesar nuestra red MPLS, vamos a configurar dos túneles bidireccionales. Uno a cada extremo de la red, tal como se puede observar en el siguiente escenario. Los túneles que vamos a usar son del tipo IPIP, ya que son los más sencillos de configurar, y

consisten en encapsular un paquete IP dentro de otra cabecera IP. De este modo, los paquetes tienen dos direcciones IP origen y destino, una dentro de la otra.

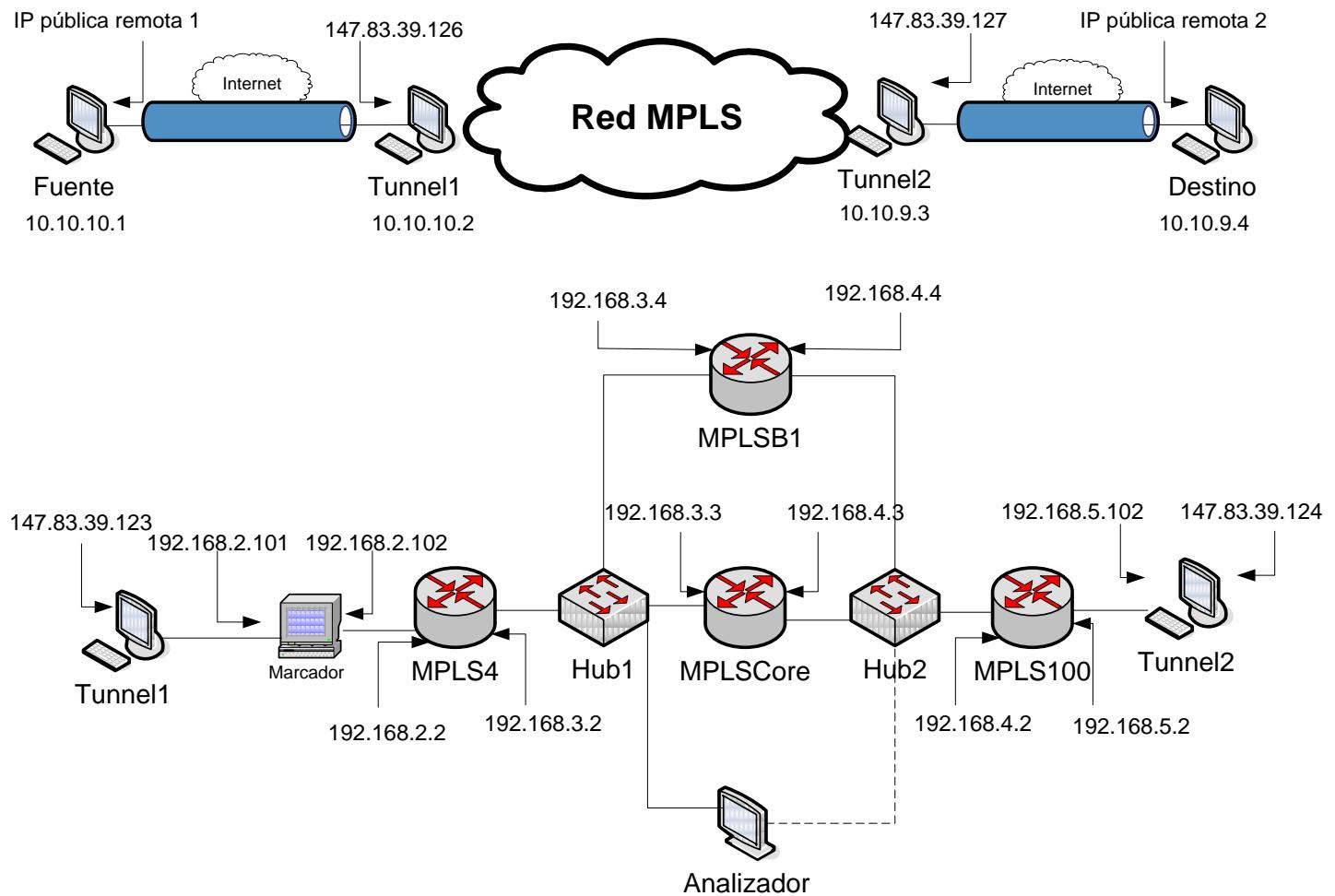


Figura 22: Escenario con túneles

En esta nueva configuración se puede observar como la Fuente y el Destino tienen direcciones públicas de Internet. Para que se puedan comunicar a través de la red MPLS, tenemos que redirigir su tráfico a través de los túneles.

El primer paso consiste en configurar dos túneles, uno de “entrada” del tránsito y otro de “salida”.

5.8.1. Configuración en la Fuente:

Tenemos que introducir los siguientes comandos en la consola de Linux:

modprobe ipip : Habilitamos el modo ipip para los túneles, ya que es posible que no lo esté por defecto.

ip tu add MiTunel mode ipip remote 147.83.39.123 local X.X.X.X: Creamos un túnel ipip hacia la dirección 147.83.39.126. La dirección local X.X.X.X es su ip pública desde la que se creara el túnel.

ifconfig MiTunel 10.10.10.1/24 up : Habilitamos la interfaz MiTunel y le asignamos una ip y máscara.

Finalmente tenemos que añadir la ruta hacia el otro túnel y red destino.

route add -net 10.10.9.0 netmask 255.255.255.0 MiTunel : Añadimos la red 10.10.9.0/24 (la red del otro túnel) a través de nuestro túnel, ya que es además la red destino del tráfico.

5.8.2. Configuración para el Destino:

Los comandos son los mismos, pero intercambiado las IP's.

modprobe ipip : Habilitamos el modo ipip para los túneles, ya que es posible que no lo esté por defecto.

ip tu add MiTunel mode ipip remote 147.83.39.124 local X.X.X.X: Creamos un túnel ipip hacia la dirección 147.83.39.127. La dirección local X.X.X.X es su ip pública desde la que se creara el túnel.

ifconfig MiTunel 10.10.9.4/24 up : Habilitamos la interfaz MiTunel y le asignamos una ip y máscara.

Finalmente tenemos que añadir la ruta hacia el otro túnel y red destino.

route add -net 10.10.10.0 netmask 255.255.255.0 MiTunel : Añadimos la red 10.10.10.0/24 (la red del otro túnel) a través de nuestro túnel.

Para dotar de bidireccionalidad a los túneles, se tienen que configurar de forma idéntica los ordenadores Tunnel1 y Tunnel2, pero con las IP's públicas y privadas que detalla el escenario.

Como último paso simplemente queda añadir las nuevas redes a las tablas de rutas del "Marcador" y de la red de routers MPLS. Estos últimos no han aprendido las nuevas reglas automáticamente ya que el protocolo OSPF solo está activo dentro de los routers, y las dos redes 10.10.10.0/24 y 10.10.9.0/24 son externas a estas.

Estas redes se podrían añadir manualmente en cada router, pero entonces no serían compatibles con la configuración del Backup anterior, ya que serían ajenas al protocolo OSPF. Para solucionarlo, se tienen que introducir los siguientes comandos:

MPLS4 (config) # ip route 10.10.10.0 255.255.255.0 192.168.2.102 //En el router más cercano a la nueva red, introducimos la ruta estática.

MPLS4 (config) # router ospf 1 // Configuramos el enrutamiento interno con OSPF con el identificador 1

MPLS4 (config-router) # redistribute static subnets // Compartimos las subredes estáticas con el resto de routers de nuestra área

En el otro extremo de la red, repetimos el procedimiento, pero con la otra subred:

MPLS100 (config) # ip route 10.10.9.0 255.255.255.0 192.168.5.100 //En el router más cercano a la nueva red, introducimos la ruta estática.

MPLS100 (config) # router ospf 1 // Configuramos el enrutamiento interno con OSPF con el identificador 1

MPLS100 (config-router) # redistribute static subnets // Compartimos las subredes estáticas con el resto de routers de nuestra área.

Una vez tenemos configurada por completo la red, observamos cómo han quedado las tablas de rutas de cada dispositivo:

Para la prueba hemos usado con “IP pública remota 1” 147.83.39.127, y como “IP pública remota 2” 147.83.39.126.

Fuente:

Destino	Pasarela	Genmask	Indic	Interfaz
localnet	*	255.255.255.0	U	eth0
10.10.10.0	*	255.255.255.0	U	MiTunel
10.10.9.0	10.10.10.2	255.255.255.0	UG	MiTunel
link-local	*	255.255.0.0	U	eth0
default	147.83.39.1	0.0.0.0	UG	eth0

Tabla 9: Tabla de enrutamiento de la "Fuente"

Tunnel1:

Destino	Pasarela	Genmask	Indic	Interfaz
192.168.2.0	192.168.2.101	255.255.255.0	U	eth2
147.83.39.0	*	255.255.255.0	UG	eth1
10.10.10.0	*	255.255.255.0	U	MiTunel
10.10.9.0	192.168.2.101	255.255.255.0	UG	eth2

link-local	*	255.255.0.0	U	eth1
default	147.83.39.1	0.0.0.0	UG	eth1

Tabla 10: Tabla de enrutamiento del Tunnel1

Tunnel2:

Destino	Pasarela	Genmask	Indic	Interfaz
192.168.5.0	192.168.5.2	255.255.255.0	U	eth1
147.83.39.0	*	255.255.255.0	UG	eth0
10.10.10.0	192.168.5.2	255.255.255.0	U	eth1
10.10.9.0	*	255.255.255.0	UG	MiTunel
link-local	*	255.255.0.0	U	eth1
default	147.83.39.1	0.0.0.0	UG	eth1

Tabla 11: Tabla de enrutamiento del Tunnel2

Destino:

Destino	Pasarela	Genmask	Indic	Interfaz
localnet	*	255.255.255.0	U	1th1
10.10.10.0	10.10.9.3	255.255.255.0	U	MiTunel
10.10.9.0	*	255.255.255.0	UG	MiTunel
link-local	*	255.255.0.0	U	eth0
default	147.83.39.1	0.0.0.0	UG	eth0

Tabla 12: Tabla de enrutamiento del "Destino"

Si ahora capturamos un paquete que entra por la interfaz pública de Tunnel 1, se puede observar cómo está encapsulado un paquete IP dentro de otro.

```
[+] Frame 2 (1514 bytes on wire, 1514 bytes captured)
[+] Ethernet II, Src: 3com_6a:43:cd (00:60:08:6a:43:cd), Dst: Asiarock_df:7d:e2 (00:13:8f:df:7d:)
[+] Internet Protocol, Src: 147.83.39.127 (147.83.39.127), Dst: 147.83.39.123 (147.83.39.123)
[+] Internet Protocol, Src: 10.10.10.1 (10.10.10.1), Dst: 10.10.9.4 (10.10.9.4)
[+] Transmission Control Protocol, Src Port: http-alt (8080), Dst Port: 39314 (39314), Seq: 1429
[+] Hypertext Transfer Protocol
```

Figura 23: Detalle de paquete IP-IP

Si observamos el mismo paquete, pero en la interfaz de entrada del “Marcador”, el paquete ya no está “tunelado”:

```
[+] Frame 2 (1494 bytes on wire, 1494 bytes captured)
[+] Ethernet II, Src: CisTechn_39:fc:83 (00:20:18:39:fc:83), Dst: 3Com_92:5f:ce (00:04:76:92:5f:ce)
[+] Internet Protocol, Src: 10.10.10.1 (10.10.10.1), Dst: 10.10.9.4 (10.10.9.4)
[+] Transmission Control Protocol, Src Port: http-alt (8080), Dst Port: 39314 (39314), Seq: 1429, A
[+] Hypertext Transfer Protocol
```

Figura 24: Detalle de paquete IP

5.9. Configuración de QoS

Como se ha explicado anteriormente en la configuración básica del escenario, los paquetes se marcan en el campo DSCP antes de entrar en la red MPLS con el ordenador "Marcador". Esto nos permitirá definir distintos comportamientos y calidades para el tráfico que pasa a través de la red.

Para asignar QoS en MPLS se tienen que seguir 3 pasos:

- Definir las clases de tráfico
- Definir las políticas de QoS
- Asignar a qué interfaces se aplican las políticas

5.9.1. Definir las clases de tráfico

Para definir las clases de tráfico se utiliza en comando **class-map** seguido de **match-any** o **match-all**, dependiendo de si queremos que cumpla todas las condiciones o solo una de ellas, y del nombre de la clase.

En nuestro caso podríamos diferenciar dos tipos de clase: los paquetes con destino una red Ethernet, y los paquetes con destino una red Wifi.

```
MPLS4 (config) # class-map match-all Ethernet // creamos la clase "Ethernet" que debe cumplir todas las condiciones
```

```
MPLS4 (config-cmap) # match dscp CS1 // creamos la condición que el paquete tenga el campo DSCP igual a CS1
```

Ahora aplicamos los mismos comandos, pero para crear la clase adecuada para la Wifi

```
MPLS4 (config) # class-map match-all Wifi // creamos la clase "Wifi" que debe cumplir todas las condiciones
```

```
MPLS4 (config-cmap) # match dscp CS2 // creamos la condición que el paquete tenga el campo DSCP igual a CS2
```

```
MPLS4 (config-cmap) # end // salimos de la configuración de la clase
```

```
MPLS4 (config) # class-map match-all videoEthernet // creamos la clase "videoEthernet" que debe cumplir todas las condiciones
```

```
MPLS4 (config-cmap) # match mpls experimental 1 // creamos la condición que el paquete tenga el campo EXP igual a 1
```

```
MPLS4 (config-cmap) # end // salimos de la configuración de la clase
```

```
MPLS4 (config) # class-map match-all videoWifi // creamos la clase "videoWifi" que debe cumplir todas las condiciones
```

```
MPLS4 (config-cmap) # match mpls experimental 2 // creamos la condición que el paquete tenga el campo EXP igual a 2
```

```
MPLS4 (config-cmap) # end // salimos de la configuración de la clase
```

Ahora mismo puede que no quede muy clara la función de estas dos últimas clases, pero se entenderá mejor después de asignar las políticas a los interfaces.

5.9.2. Definir políticas de QoS

Para definir las políticas de QoS tenemos que usar el comando **policy-map** seguido del nombre de la política.

```
MPLS4 (config) # policy-map marcarEXP1 // creamos la política "marcarEXP1"
```

```
MPLS4 (config-pmap) # class Ethernet // asignamos la clase "Ethernet" a esta política
```

```
MPLS4 (config-pmap) # set mpls experimental 1 // la política marcará los paquetes de la clase con el campo EXP igual a 1
```

```
MPLS4 (config-pmap) # end // salimos de la configuración de la política
```

```
MPLS4 (config) # policy-map marcarEXP2 // creamos la política "marcarEXP2"
```

```
MPLS4 (config-pmap) # class Wifi // asignamos la clase "Wifi" a esta política
```

```
MPLS4 (config-pmap) # set mpls experimental 2 // la política marcará los paquetes de la clase con el campo EXP igual a 2
```

```
MPLS4 (config-pmap) # end // salimos de la configuración de la política
```

Ahora vamos a crear las políticas que realmente modificarán el comportamiento del tráfico:

Por ejemplo podemos definir que todo el tráfico que supere los 5Mbps con destino una red Ethernet sea descartado. Así mismo, todo el que supere 3Mbps con destino una red Wifi también será descartado.

```
MPLS4 (config) # policy-map shapeEthernet // creamos la política "shapeEthernet"
```

```
MPLS4 (config-pmap) # class videoEthernet // asignamos la clase "videoEthernet" a esta política
```

```
MPLS4 (config-pmap) # shape peak 500000 // la política descartará los paquetes que superen los 5Mbps
```

```
MPLS4 (config-pmap) # end // salimos de la configuración de la política
```

```
MPLS4 (config) # policy-map shapeWifi // creamos la política "shapeWifi"
```

```
MPLS4 (config-pmap) # class videoWifi // asignamos la clase "videoWifi" a esta política
```

```
MPLS4 (config-pmap) # shape peak 300000 // la política descartará los paquetes que superen los 3Mbps
```

```
MPLS4 (config-pmap) # end // salimos de la configuración de la política
```

5.9.3. Asignar las políticas a los interfaces

Ahora que ya tenemos definidas las clases de servicios y las políticas, solo nos queda asignar dichas políticas a los interfaces. Para hacerlo debemos usar el comando **service-policy**.

```
MPLS4 (config)# interface f0/0 // Entramos en la configuración del interfaz f0/0
```

```
MPLS4 (config-if)# service-policy input marcarEXP1 // Asignamos la política "marcarEXP1" a la entrada de la interfaz
```

```
MPLS4 (config-if)# exit // Salimos de la configuración del interfaz
```

```
MPLS4 (config)# interface f0/1 // Entramos en la configuración del interfaz f0/1
```

```
MPLS4 (config-if)# service-policy output shapeEthernet // Asignamos la política "shapeEthernet" al interfaz de salida f0/1
```

El total de pasos que hemos seguido se pueden resumir en el siguiente diagrama:

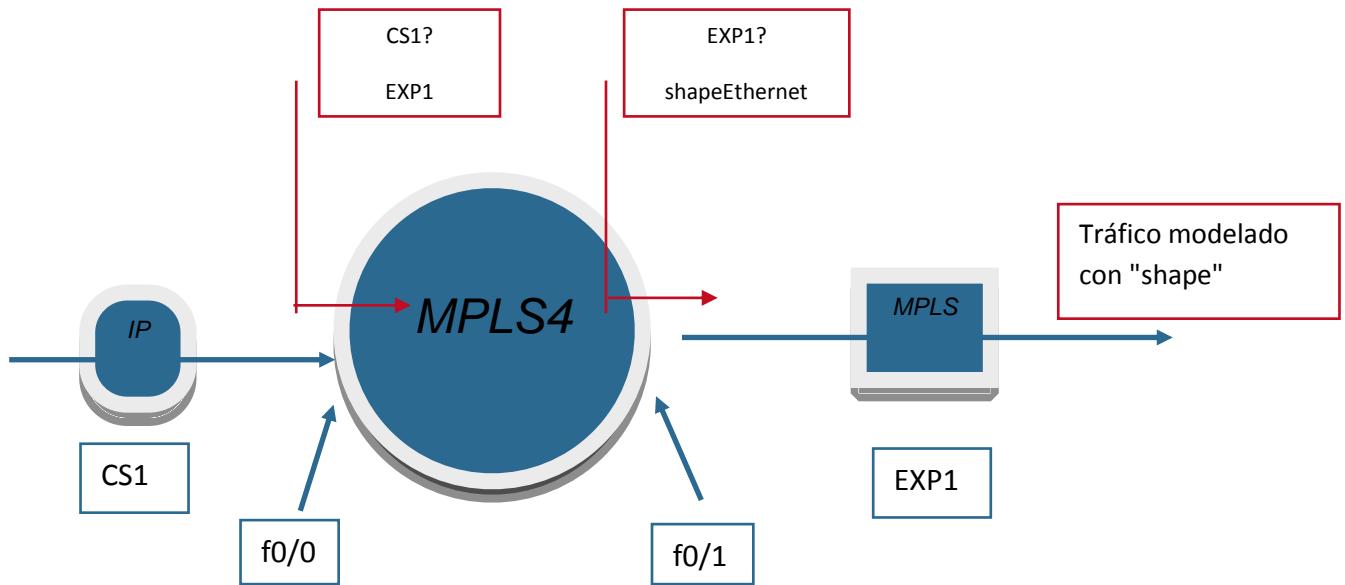


Figura 25: Diagrama de la aplicación del QoS

En este diagrama se recoge de forma esquemática el proceso que sigue un paquete que entra en la red MPLS cuando está configurada una política de QoS para modelar el tráfico de Ethenet. El caso de Wifi sería equivalente:

1. Llega un paquete IP
2. La interfaz de entrada (f0/0) comprueba si el campo DSCP del paquete es CS1 (Si su destino es una red Ethernet, tendrá marcado su campo DSCP con el valor “CS1”). En caso afirmativo, marca el paquete con el campo EXP igual a 1
3. La interfaz de salida (f0/1) comprueba si el campo EXP del paquete es igual a 1, si lo es, aplica la política “shapeEthernet”

Esta solución parece un poco rebuscada, ya que parece más sencillo que el interfaz de salida comprobara si el campo DSCP del paquete tiene el valor “CS1” y entonces aplique la política. El problema es que el paquete que sale de la interfaz de salida ya no es un paquete IP, sino que es un paquete MPLS, y por lo tanto el interfaz no encuentra el campo DSCP.

6. Pruebas a través de los routers IP-MPLS

6.1. Pruebas a través de los routers IP-MPLS - Ethernet

Medidas: Caudal, retardo extremo-extremo, jitter extremo-extremo, pérdidas

Estas medidas se tomarán mediante ping e Jperf (duración 2 minutos), capturando mediante Wireshark las trazas de todo el tráfico emitido y recibido, para los siguientes casos de uso:

Caso 1: Medidas sin afectación de ningún tipo (medidas libres)

Caso 2: Medidas con rotura de enlace y activación de Backup. La rotura se emulará mediante la simple desconexión de uno de los enlaces.

Caso 3: Medidas con balanceo de carga.

Caso 4: Transmisión de tráfico real enviando video con el programa VLC, repitiendo los casos 1, 2 y 3.

Escenario: Como usaremos el camino de Backup, el escenario será el siguiente:

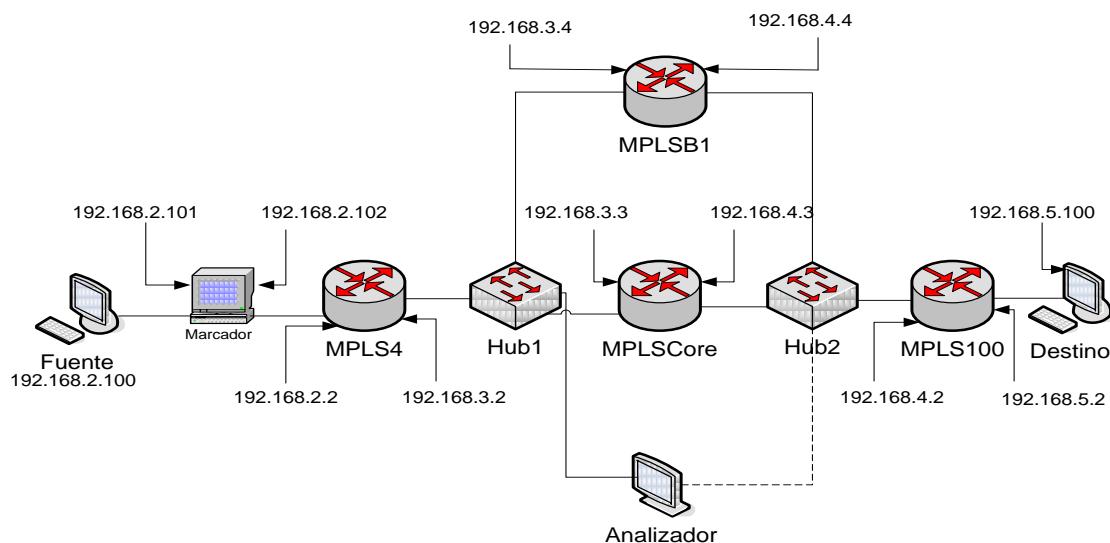


Figura 26: Escenario con Backup

Caso 1:

Con el comando ping obtenemos los siguientes paquetes:

No.	Time	Source	Destination	Protocol	Info
37	15.996779	192.168.2.100	192.168.5.100	ICMP	Echo (ping) request
Internet Protocol, Src: 192.168.2.100 (192.168.2.100), Dst: 192.168.5.100 (192.168.5.100)					

No.	Time	Source	Destination	Protocol	Info
38	15.996824	192.168.5.100	192.168.2.100	ICMP	Echo (ping) reply

Esto nos indica que el retraso es $(15.996824 - 15.996779)/2 = 22.5 \mu s$

En media, el retardo del comando ping es: **22.33 μs**

Ahora medimos el caudal, con la herramienta Jperf, configurada con el protocolo UDP para que no introduzca retraso.

El caudal capturado en cada uno de los puntos de observación es el siguiente:

Origen:

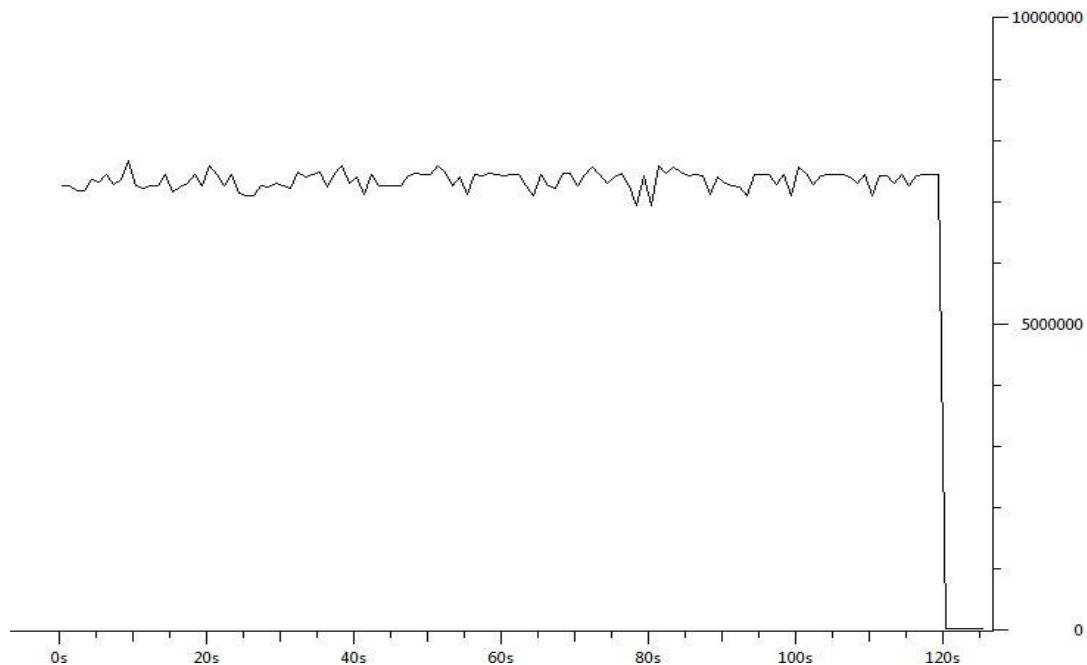


Figura 27: Caudal en el Origen en bits por segundo

Analizador:

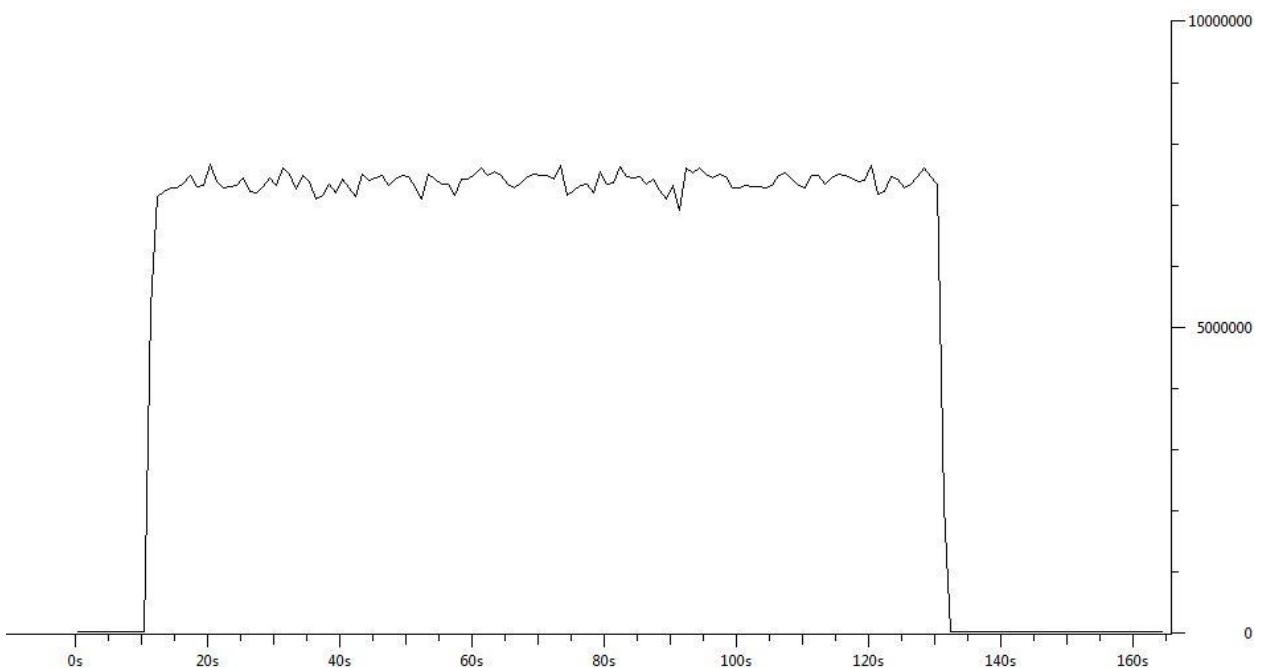


Figura 28: Caudal en el Analizador en bits por segundo

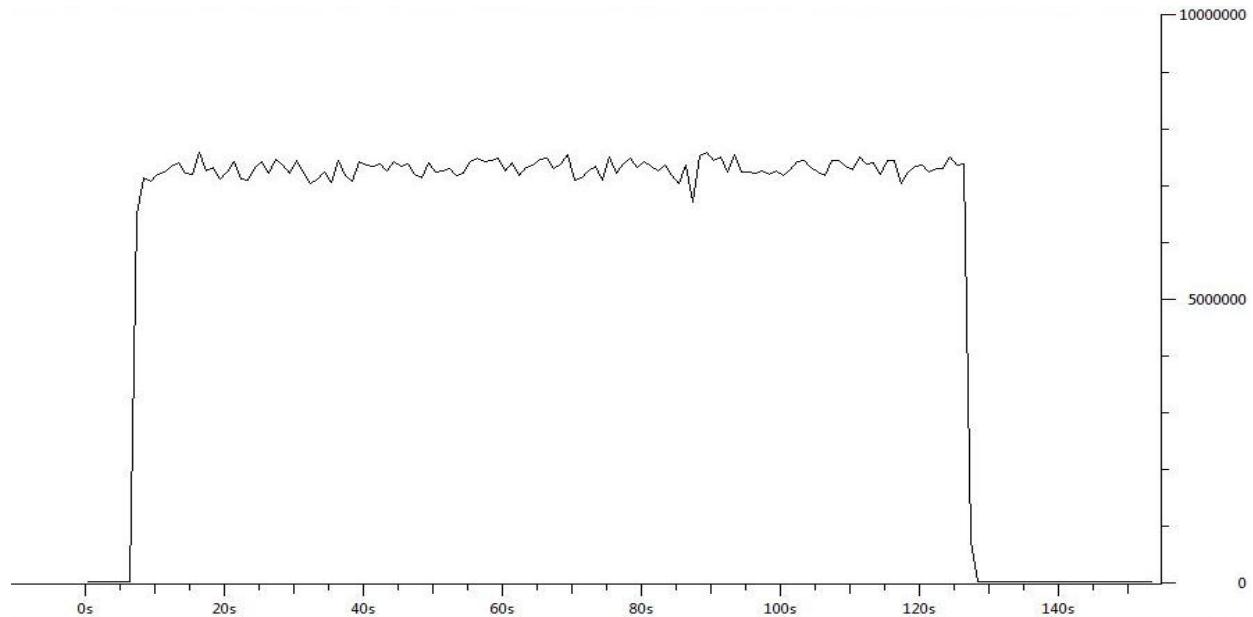
Destino:

Figura 29: Caudal en el Destino en bits por segundo

Donde podemos observar como el tráfico está limitado a unos 7,5 Mbps y es prácticamente idéntico en los 3 puntos. Aunque esta red MPLS puede llegar a una velocidad de 100Mbps, la tarjeta de red entre el “Origen” y el “Marcador” limita la red a 10Mbps como máximo.

También obtenemos el siguiente jitter extremo a extremo:

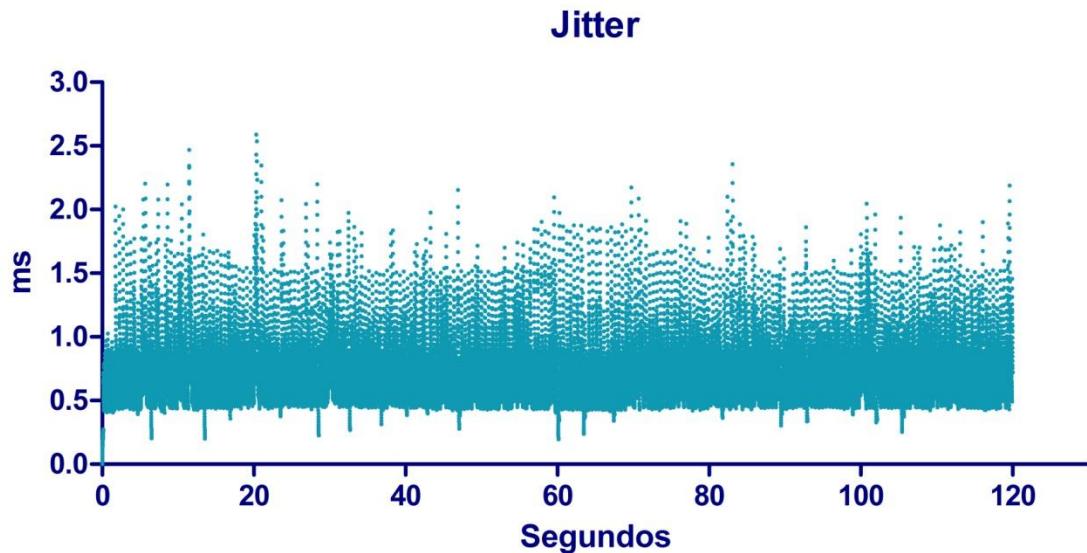


Figura 30: Jitter del Caso 1

Con valor medio: **0.7818 ms**

Cabe destacar que cada uno de los puntos de la gráfica corresponde a un paquete enviado a través de la red. Se puede observar que el jitter de una muestra depende de la anterior, ya que se ha calculado siguiendo la siguiente fórmula:

$$D(i,j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$$

Donde R_n es el tiempo en que se recibe el paquete “n” y S_n el momento en que se envía ese paquete.

$$J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16$$

$J(x)$ es la función del jitter, y por lo tanto $J(i)$ es el jitter del paquete “i”.

Finalmente, si observamos en número de paquetes emitidos y los recibidos, podemos calcular que el tanto por ciento de pérdidas es: $(278779/280528)=99.37\%$, **pérdidas = 100-99,37= 0.63%**

Caso 2:

En este segundo caso desconectaremos el cable entre el Hub1 y el rotuer MPLSCore sobre el segundo 30 de la transmisión. Luego, a los 60 segundos lo volveremos a conectar.

El caudal capturado en cada uno de los puntos de observación es el siguiente:

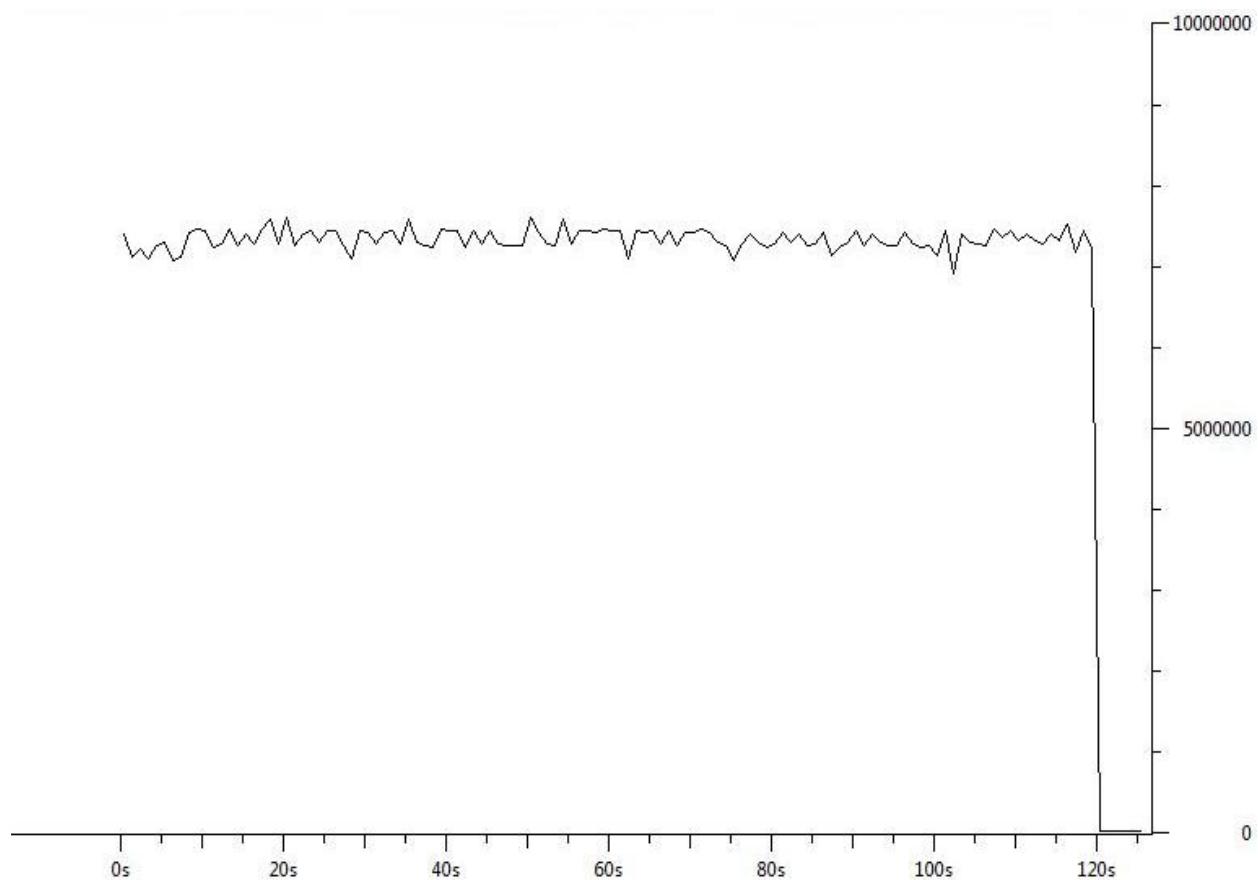
Origen:

Figura 31: Caudal en el Origen en bits por segundo

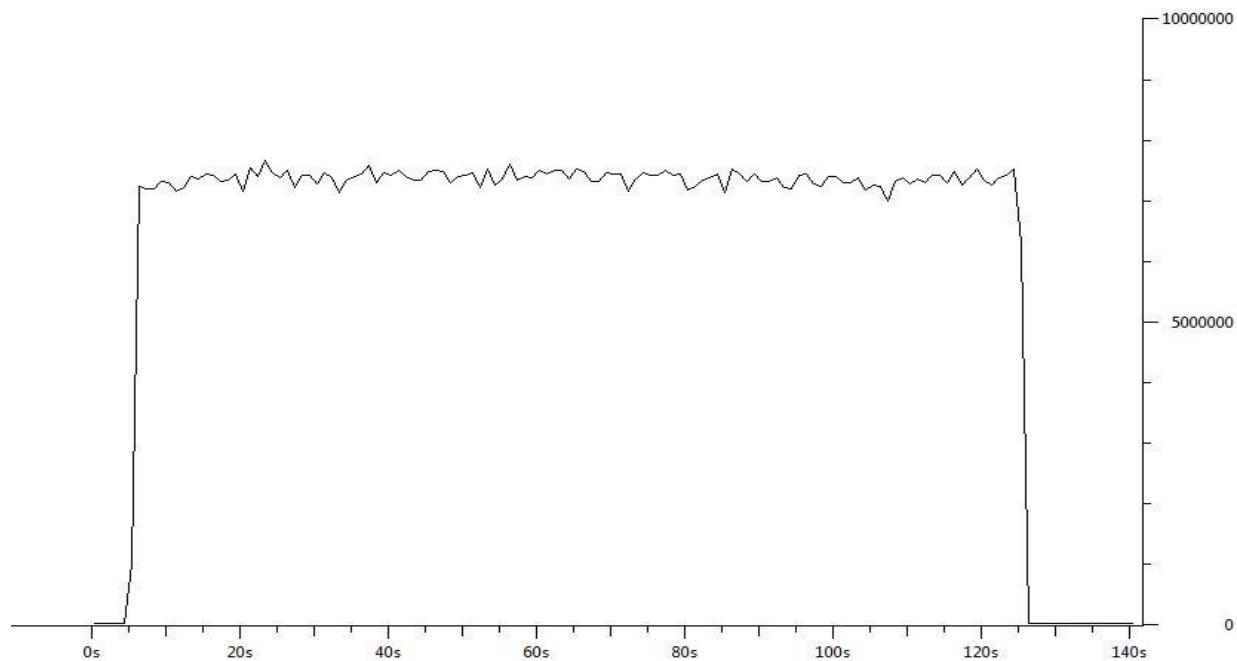
Analizador:

Figura 32: Caudal en el Analizador en bits por segundo

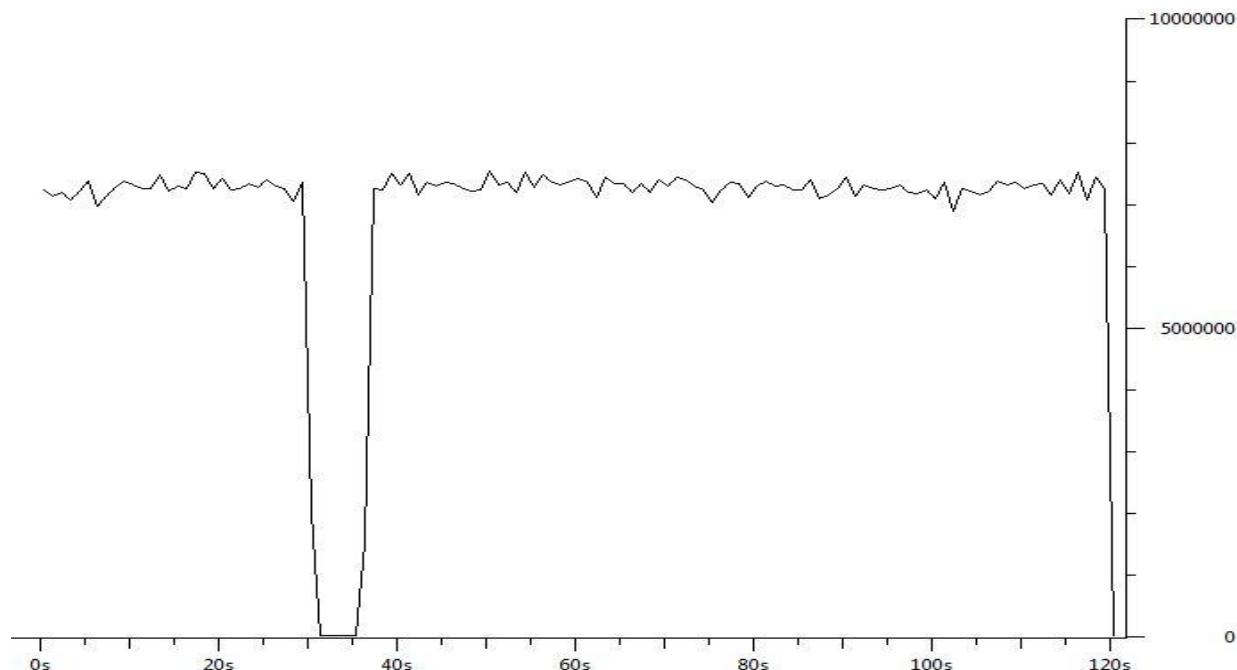
Destino:

Figura 33: Caudal en el Destino en bits por segundo

Si separamos el ancho de banda capturado por el “Analizador” en función del destino del paquete, podemos ver claramente cómo actúa el mecanismo de Backup que hemos implementado:

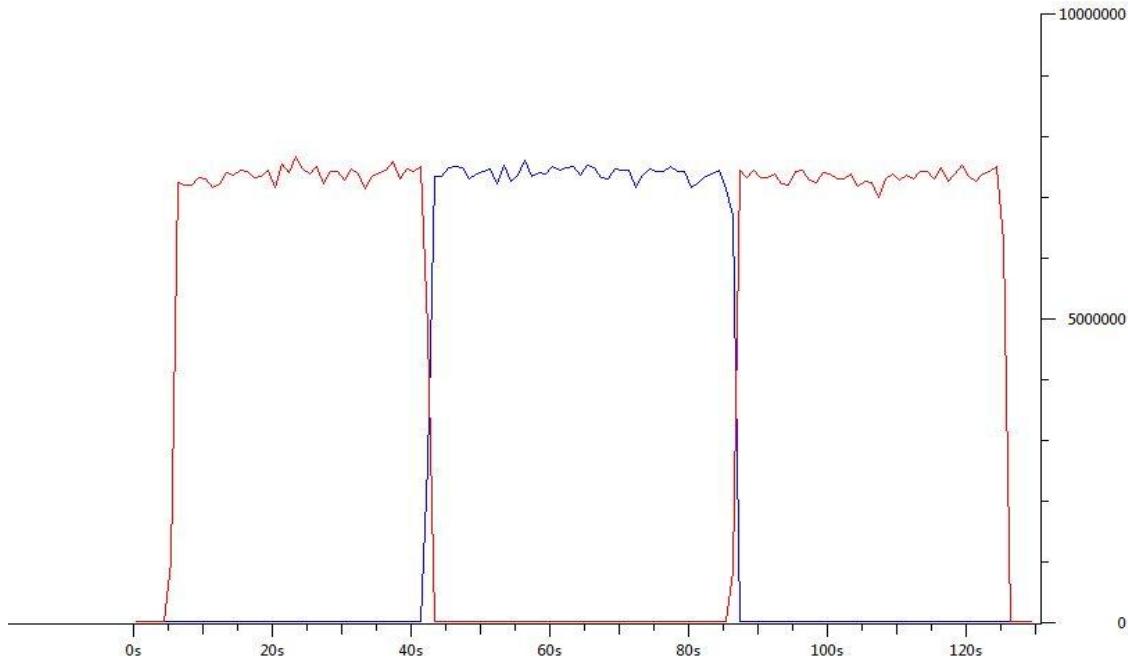


Figura 34: Detalle del destino de los paquetes en el Analizador en bits por segundo

El jitter extremo-extremo en este caso es el siguiente:

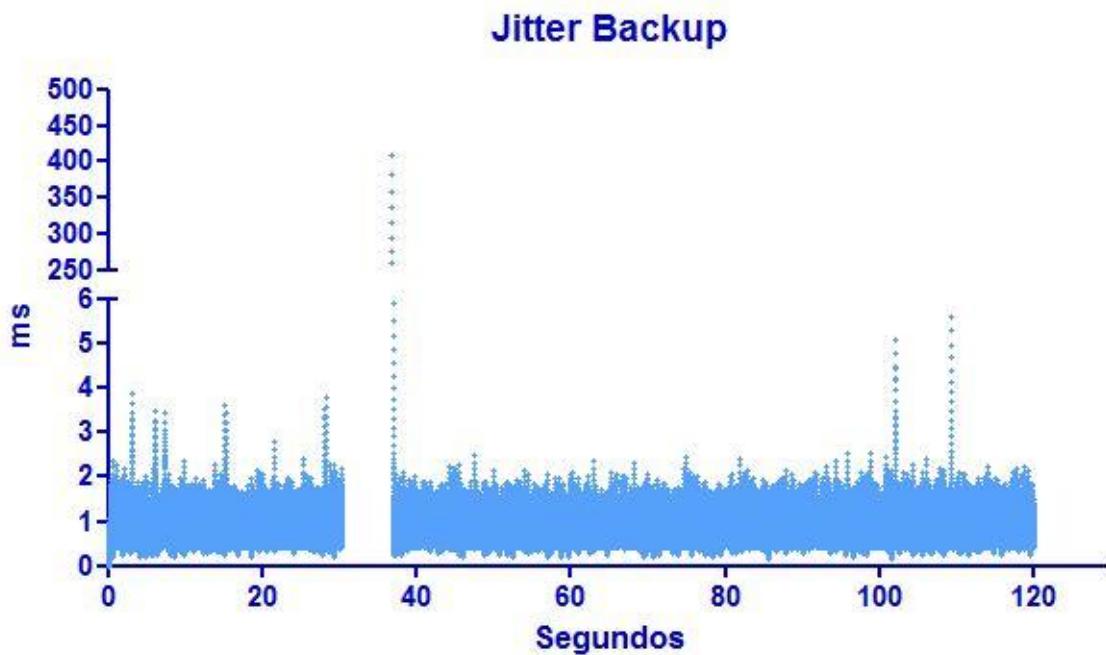


Figura 35: Jitter del Caso 2

Y con valor medio: **0.8046 ms**

En este caso el porcentaje de pérdidas será muy superior debido al corte: $(263119/280045)=93.95\%$ recibidos, por lo tanto las **pérdidas** = $100-93.95= 6.05\%$ Cabe destacar que el enlace ha estado caído 6.6seg, que representa un **5.5%** del tiempo total.

Caso 3:

Transmitimos 100 segundos de tráfico con Jperf, pero balanceando la carga entre los dos caminos disponibles.

Origen:

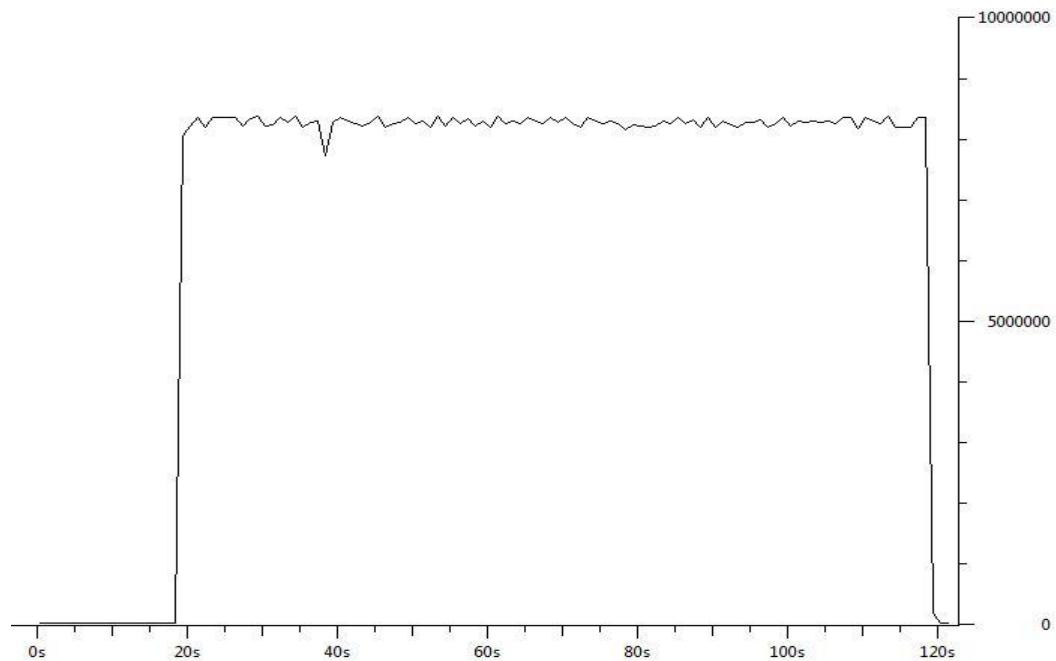


Figura 36: Caudal en el Origen en bits por segundo

Analizador:

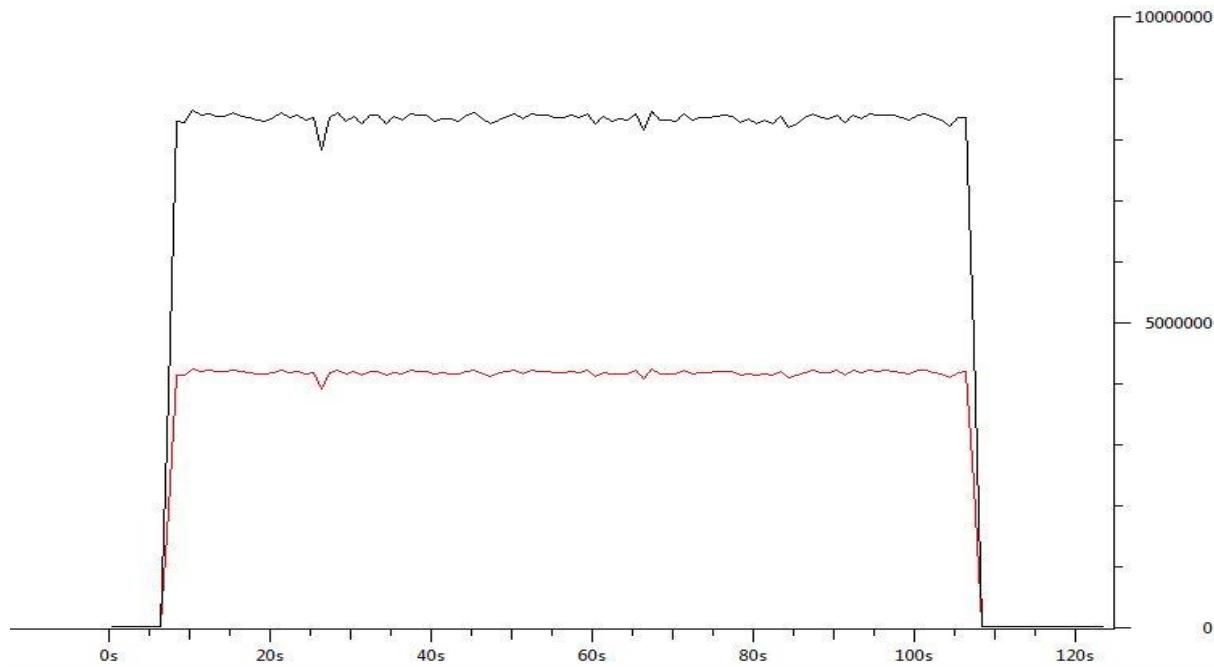


Figura 37: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo

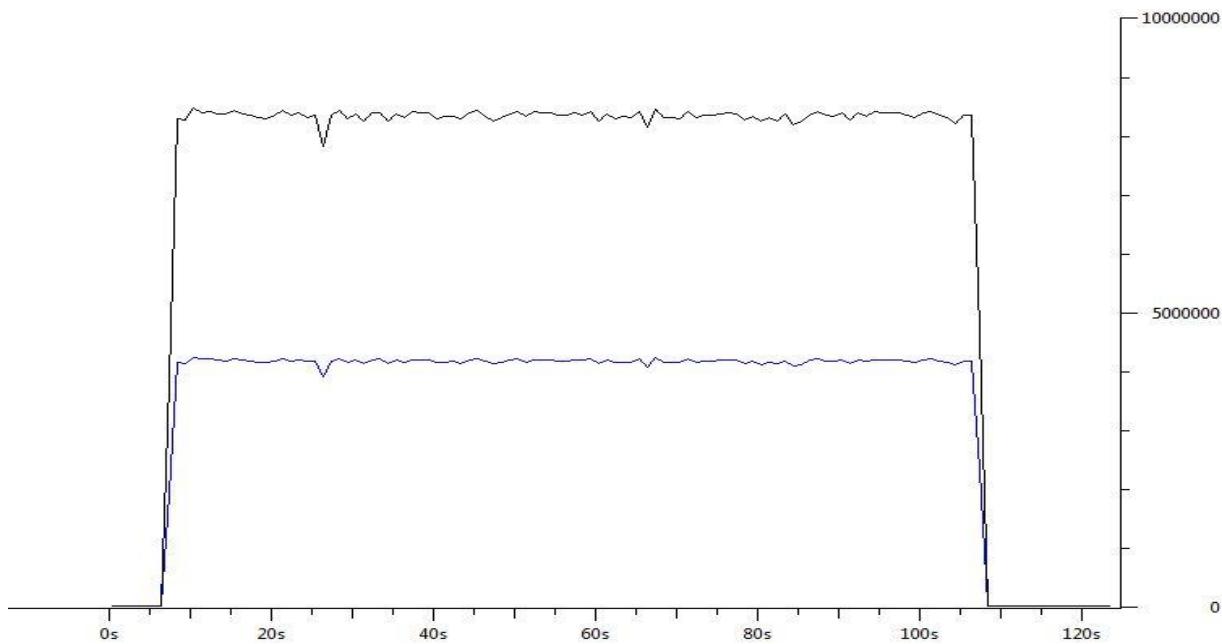


Figura 38: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo

Donde el tráfico marcado en negro es el tráfico total que pasa por el analizador, el rojo el que se dirige al router MPLSCore y el azul el que se dirige a router de Backup.

Destino:

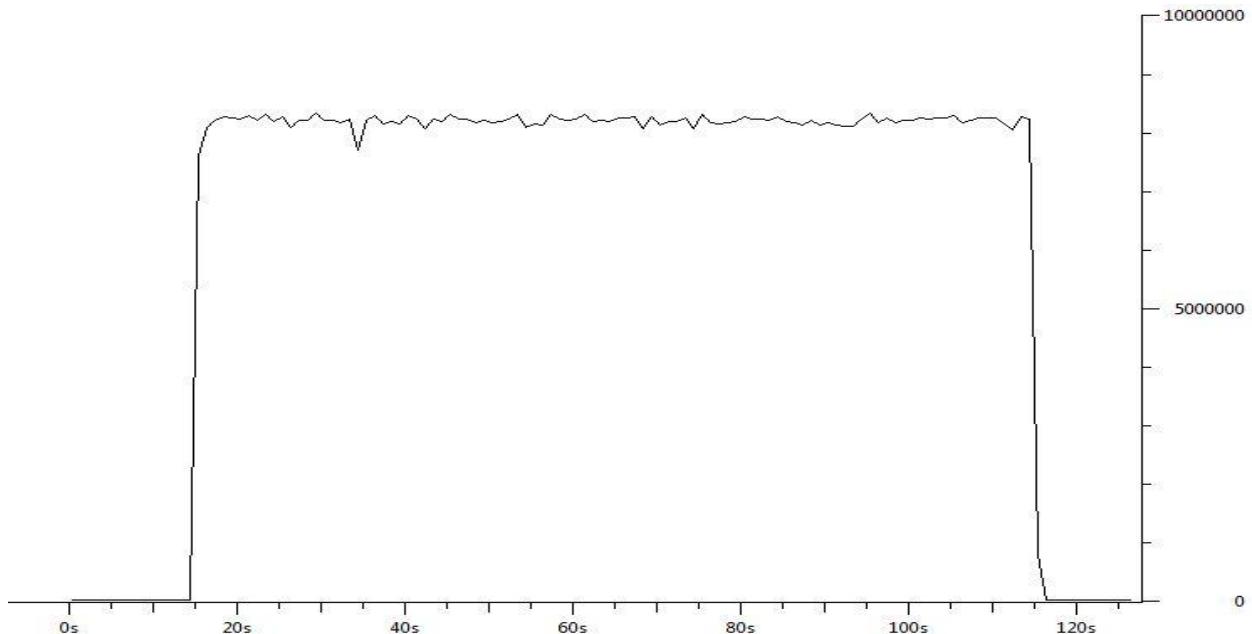


Figura 39: Caudal en el Destino en bits por segundo

En este caso es muy complicado calcular el jitter, ya que cada camino puede tener una velocidad distinta, y es tremadamente difícil emparejar cada paquete en el destino, con el paquete en el origen.

Las **pérdidas** en este caso son: $1 - (423169/426561) = 0.8\%$

Caso 4:

Caso 4.1:

Transmitimos 2 minutos de vídeo sin afectación. Para este caso, usaremos para el envío el mismo protocolo de transporte UDP con el Jperf, ya que es el más adecuado para la transmisión de vídeo, porque no introduce retardo.

Origen:

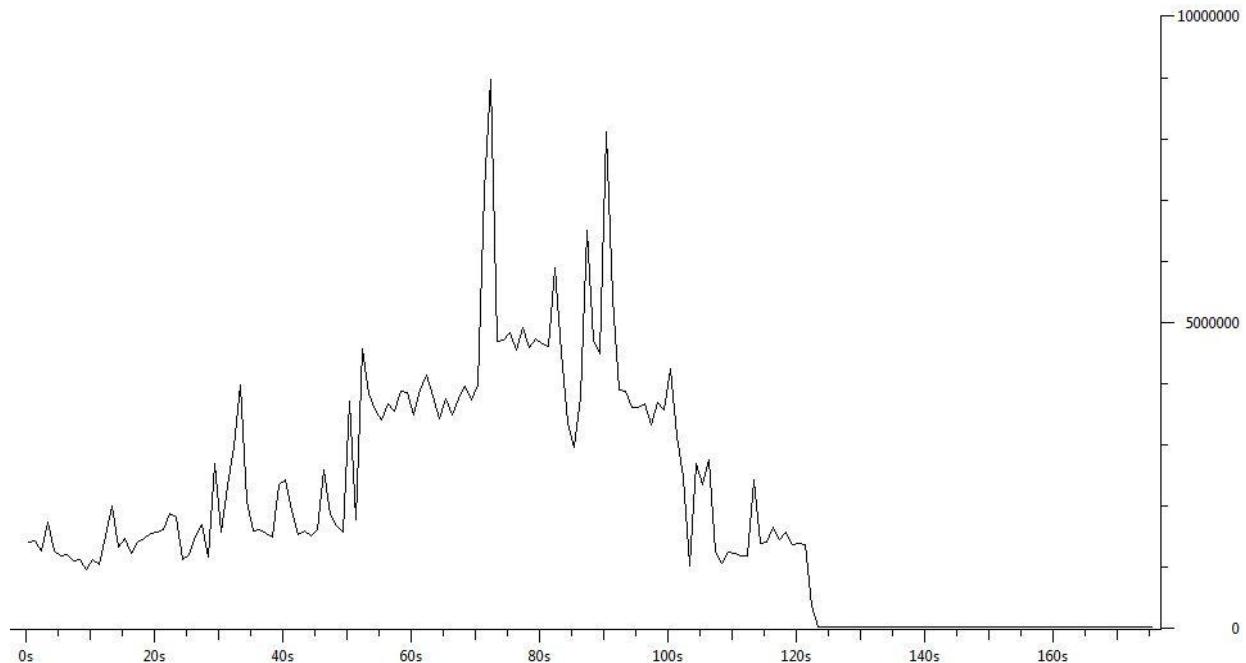


Figura 40: Caudal en el Origen en bits por segundo

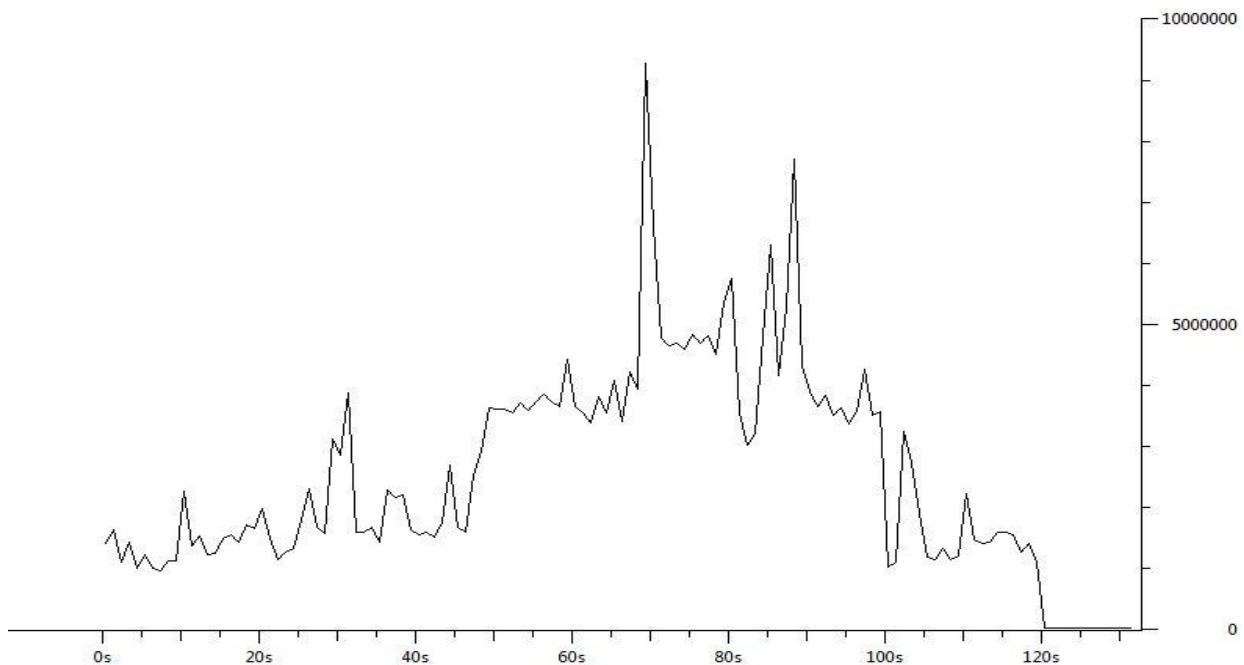
Analizador:

Figura 41: Caudal en el Analizador en bits por segundo

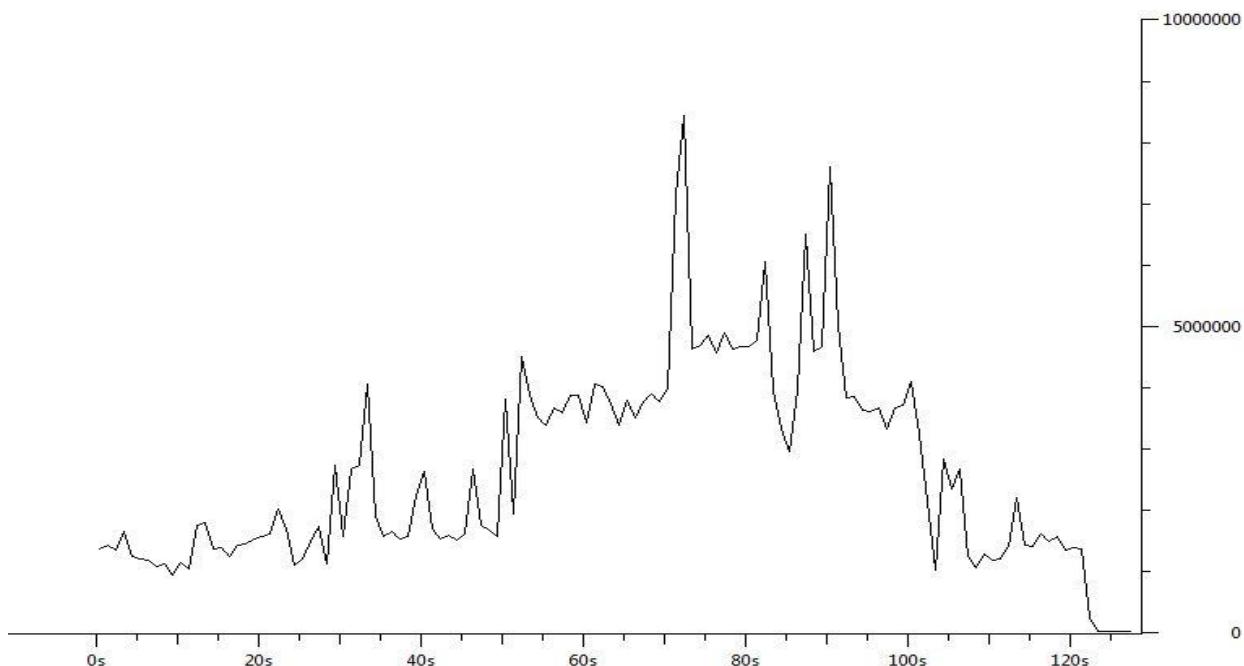
Destino:

Figura 42: Caudal en el Destino en bits por segundo

Y el jitter en este caso es:

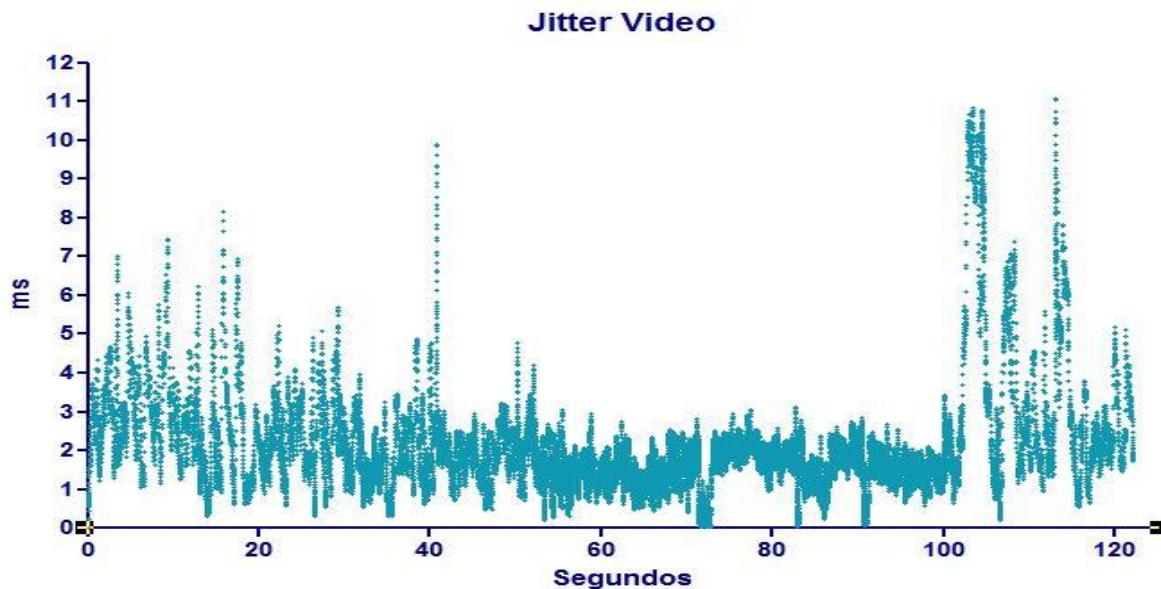


Figura 43: Jitter en el Caso 4.1

Con valor medio: **2.81 ms**

Las pérdidas sin corte son: $1 - (30745/30917) = 0.56\%$

Caso 4.2:

Ahora repetimos la misma transmisión pero desconectando el mismo enlace que en el **Caso 2**, en los mismos instantes.

Origen:

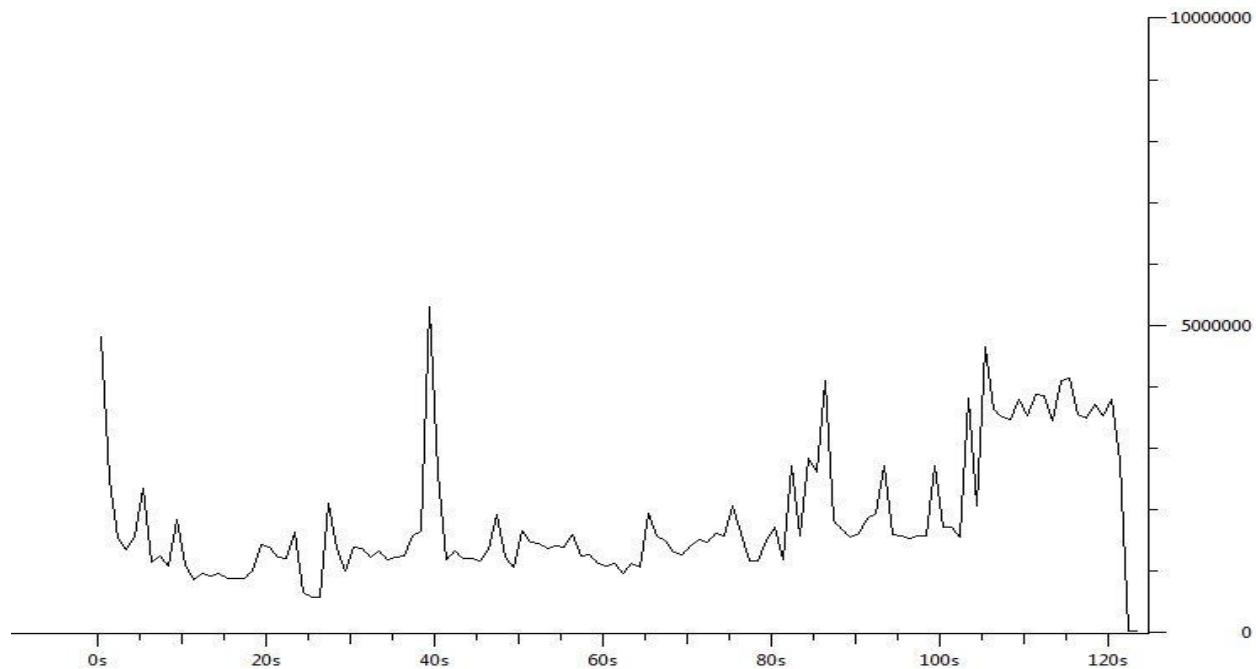


Figura 44: Caudal en el Origen en bits por segundo

Analizador:

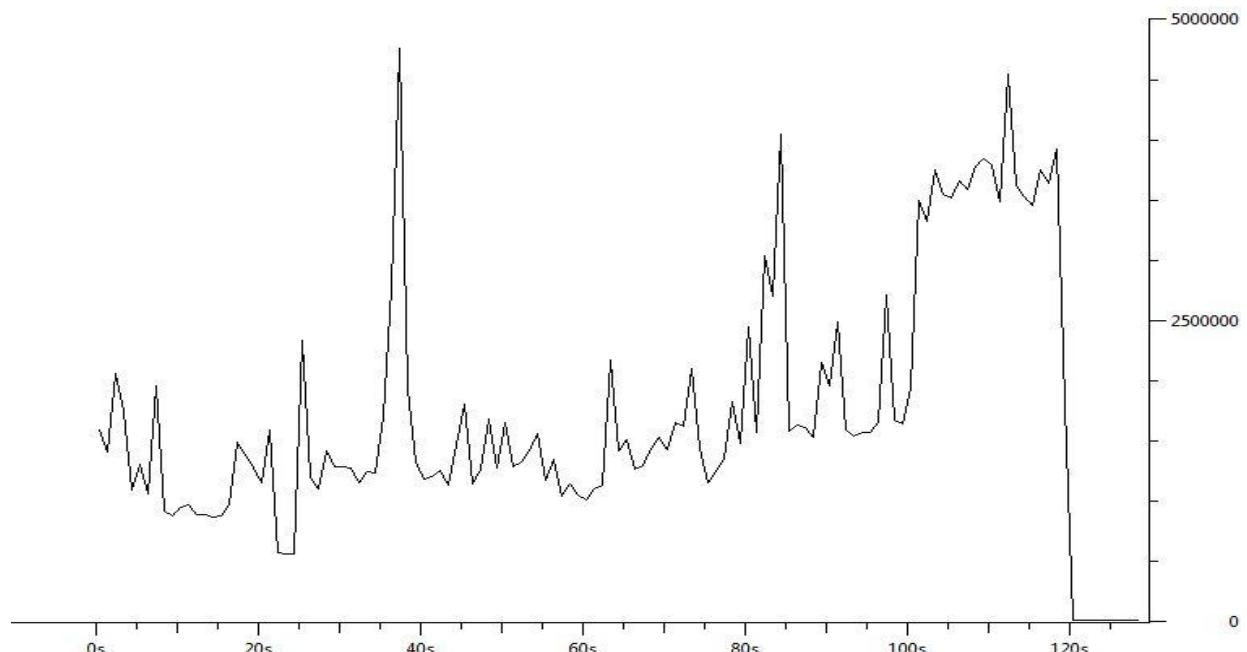


Figura 45: Caudal en el Analizador en bits por segundo

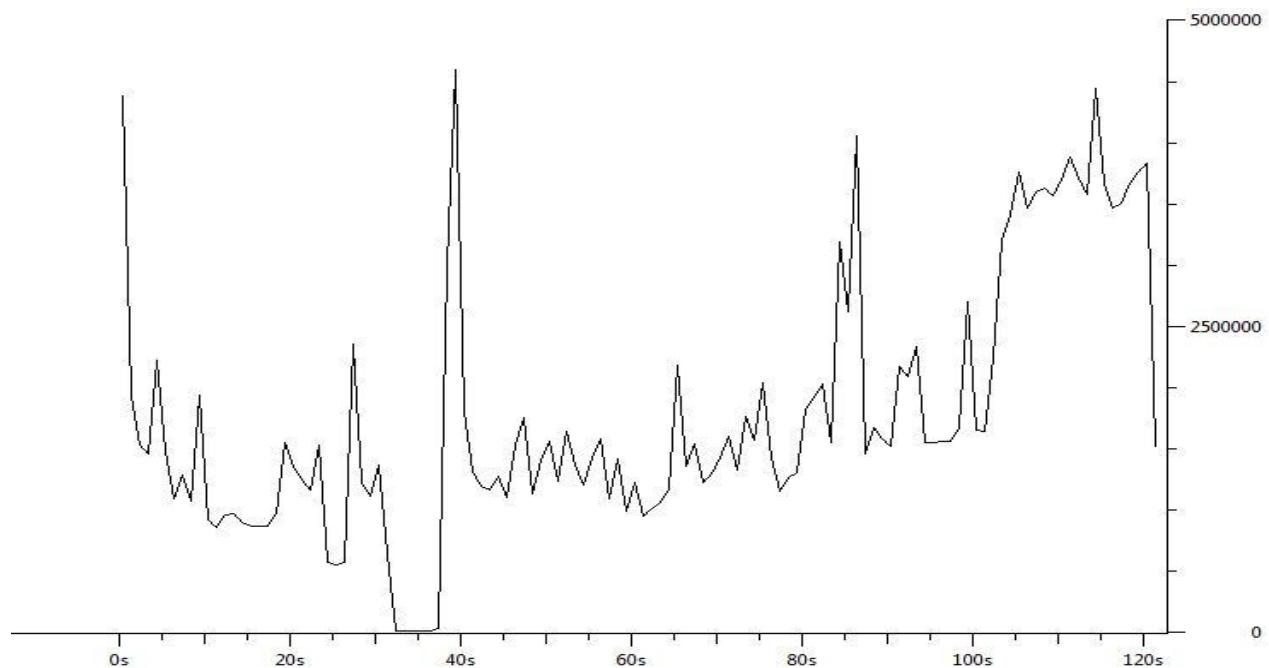
Destino:

Figura 46: Caudal en el Destino en bits por segundo

Como en el caso anterior, si separamos el ancho de banda capturado por el “Analizador” en función del destino del paquete, podemos ver claramente cómo actúa el mecanismo de Backup que hemos implementado:

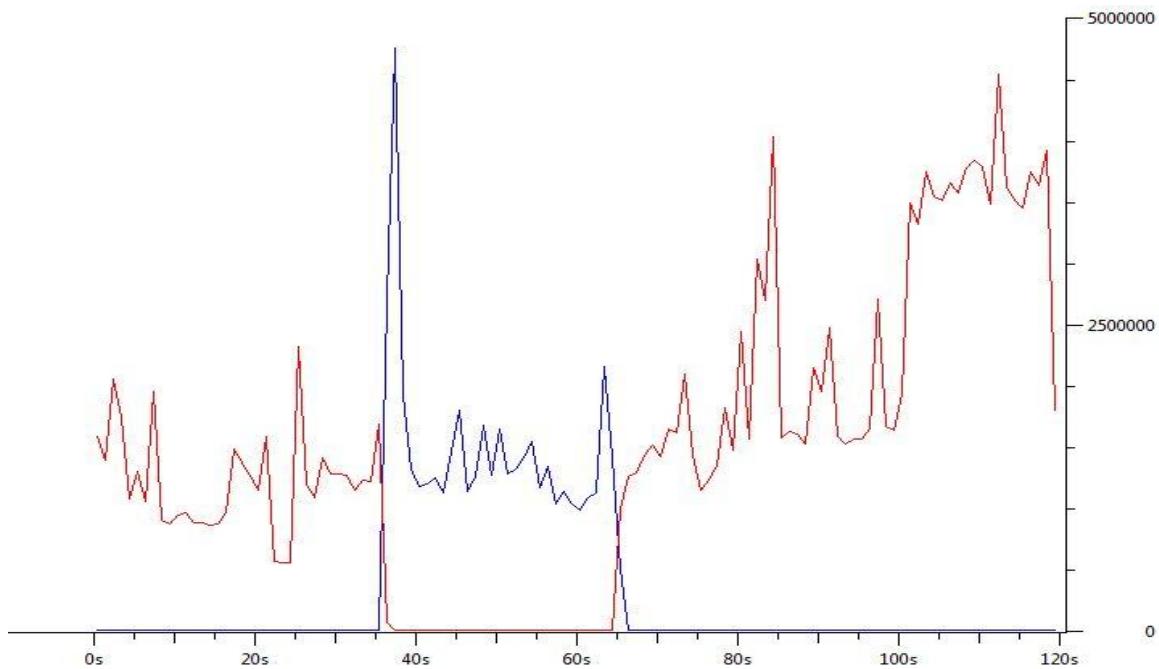


Figura 47: Detalle del destino de los paquetes en el Analizador en bits por segundo

Y con Jitter:

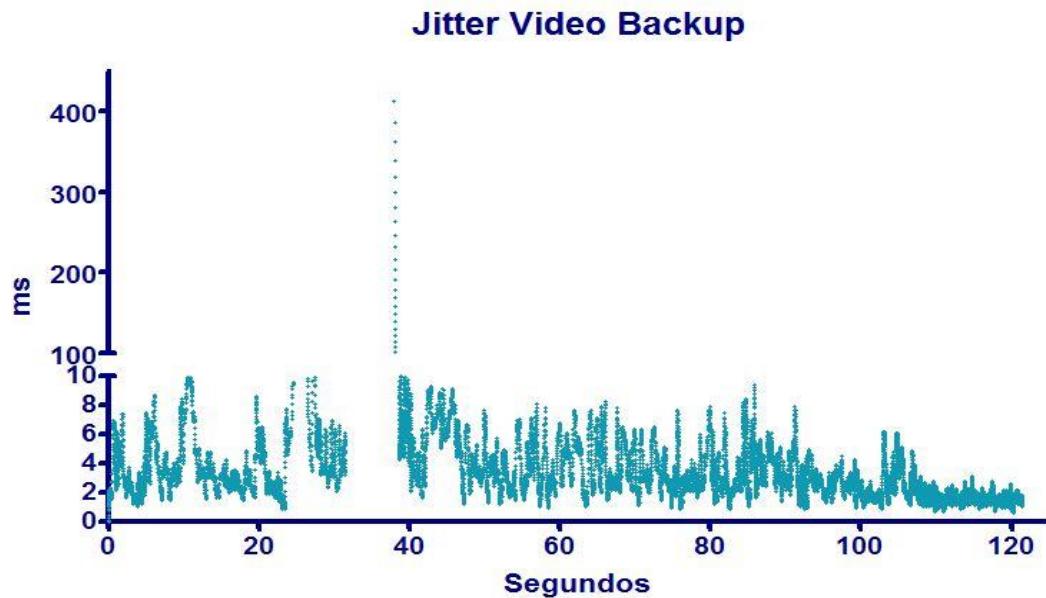


Figura 48: Jitter en el Caso 4.2

Con valor medio: 3.733 ms

Las **pérdidas** con el corte son: $1 - (19986/20900) = 4.38\%$. Aunque el enlace está interrumpido durante **6.5 seg** que es equivalente al **5.3%** del tiempo, la interrupción se ha producido durante un periodo con tasa de envío baja, lo que explica las bajas pérdidas.

Caso 4.3:

Ahora analizamos 100 segundos de tráfico real balanceando la carga. Usaremos el método de balanceo “por-paquete”, lo que significa que cada paquete se irá alternando entre los dos caminos disponibles.

Origen:

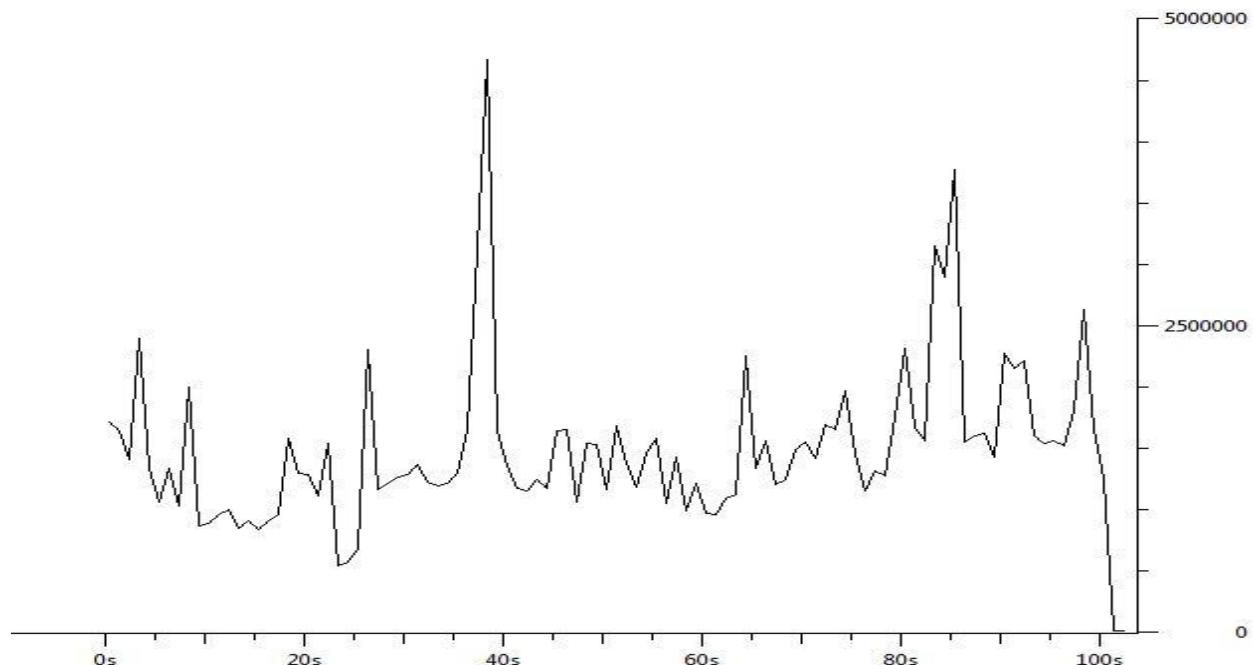


Figura 49: Caudal en el Origen en bits por segundo

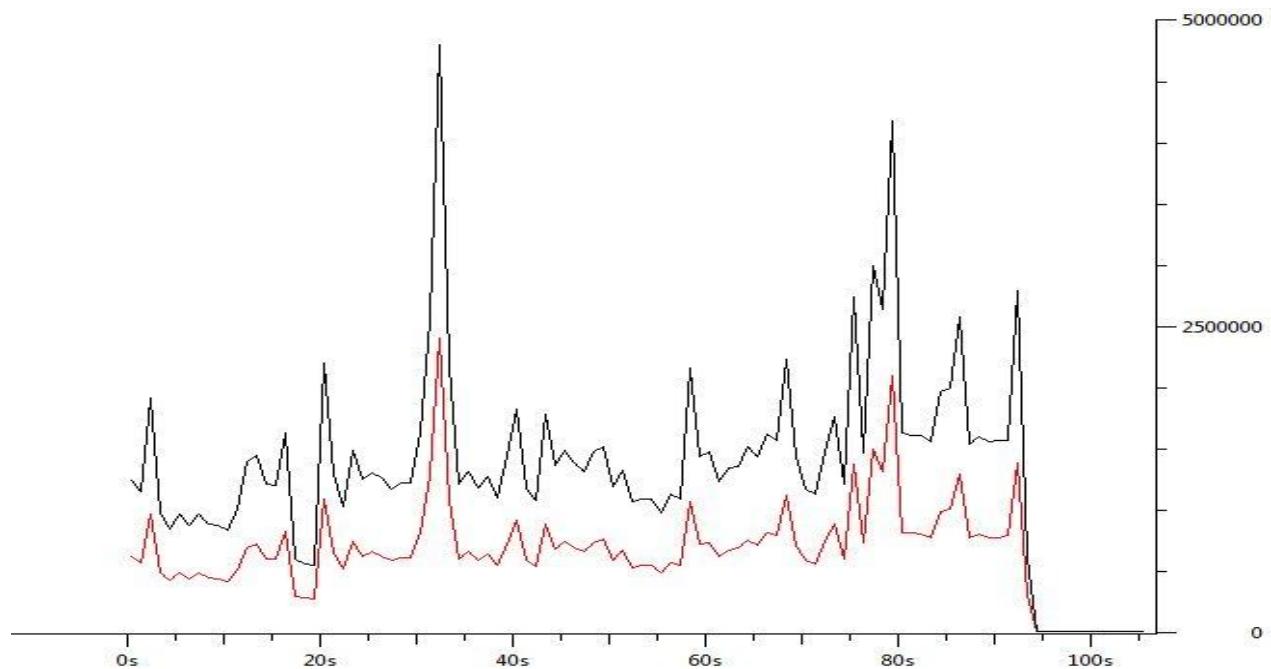
Analizador:

Figura 50: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo

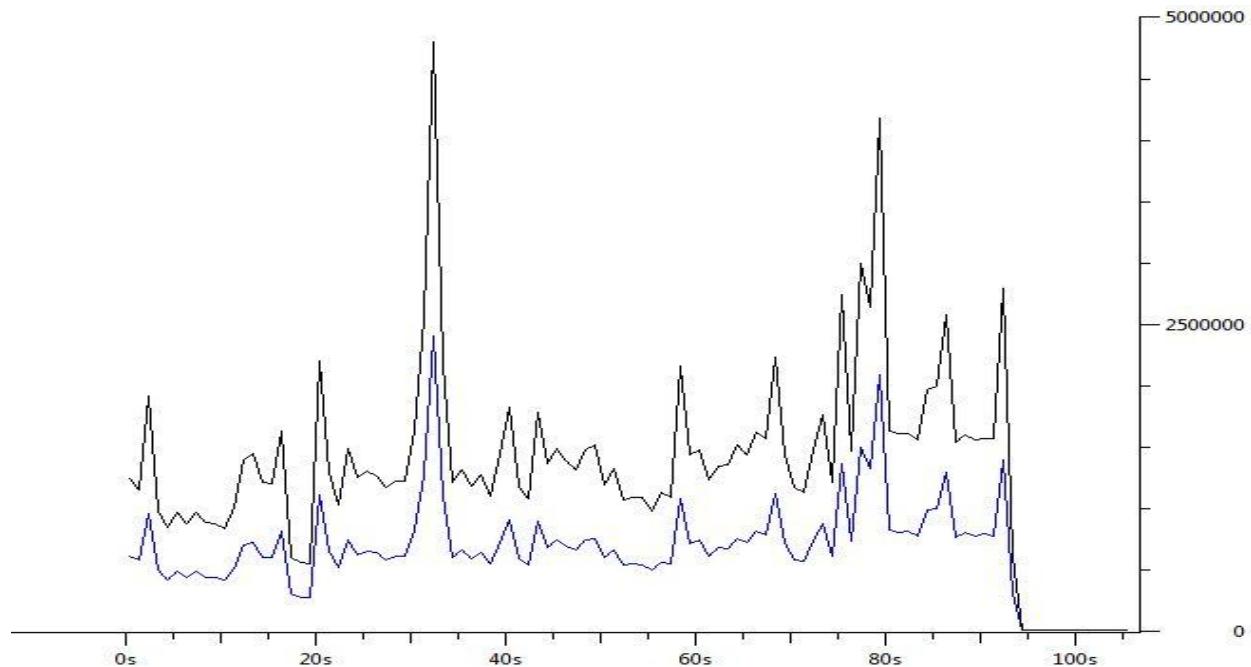


Figura 51: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo

Donde el tráfico marcado en negro es el tráfico total que pasa por el analizador, el rojo el que se dirige al router MPLSCore y el azul el que se dirige a router de Backup.

Destino:

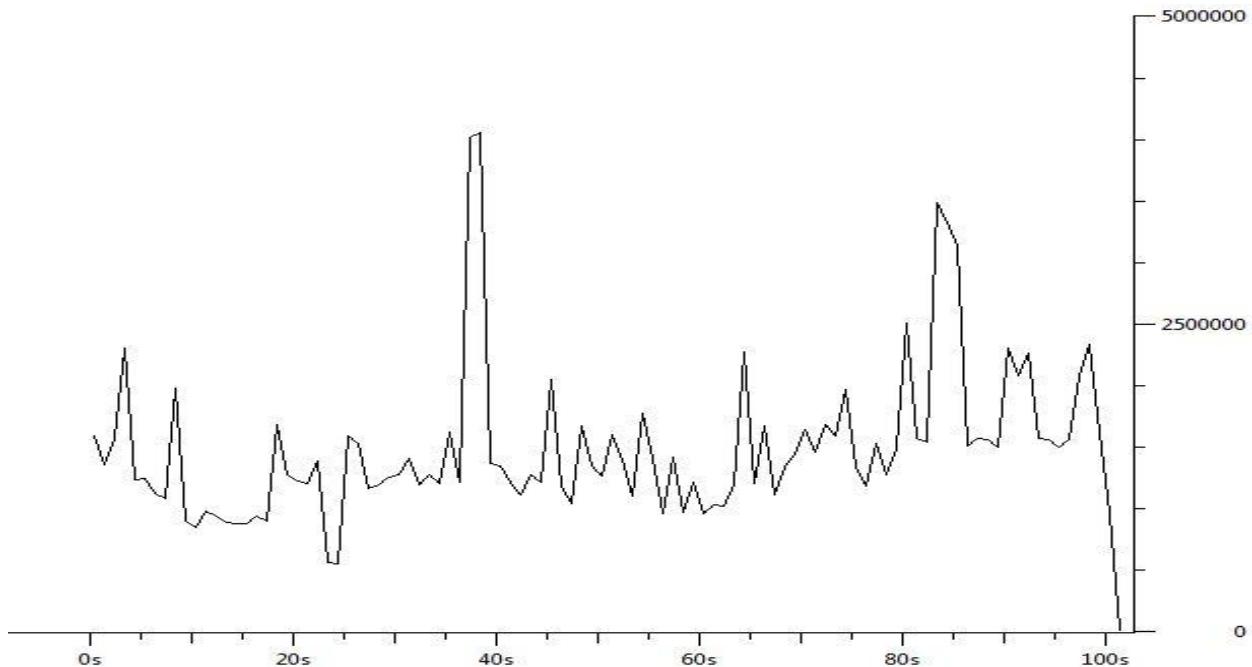


Figura 52: Caudal en el Destino en bits por segundo

Las **pérdidas** para este caso son: $1 - (423169/426561) = \mathbf{0.79\%}$

6.1.1. Conclusiones

Si recogemos los resultados obtenidos en los anteriores casos, obtenemos la siguiente tabla:

Caso	Porcentaje de tiempo		Valor medio del Jitter
	sin conexión	Porcentaje de pérdidas	
Caso 1	0%	0.63%	0.7818 ms
Caso 2	5.5%	6.05%	0.8046 ms
Caso 3	0%	0.8%	-
Caso 4.1	0%	0.56%	2.1 ms
Caso 4.2	5.3%	4.38%	3.733 ms
Caso 4.3	0%	0.79%	-

Tabla 13: Resultados de las pruebas MPLS – Ethernet

En el caso 1 hemos sometido a la red a su máxima capacidad. Los resultados de este caso nos sirven como control para las medidas del resto de casos. Vemos que la red se comporta de forma bastante transparente (la forma de las gráficas del caudal del origen y destino son prácticamente iguales), y solo se pierden el 0.63% de los paquetes, con un jitter que oscila entre 0.5 y 2 ms, pero con valor medio bastante inferior a 1ms.

Cuando en el caso 2 se pierde la conectividad en un nodo durante 30 segundos, debido a que se ha configurado el “dead time” del OSPF a 5 segundos, la red tarda esos 5 segundos en darse cuenta que el nodo ha sido desconectado. Luego realiza el cambio hacia el router de Backup, tardando en total 6.6 segundos en recuperarse. En total se pierden un 6.05% de los paquetes, aunque el 5.5% se pierden debido a la rotura del enlace. Por lo que respecta al jitter, aumenta ligeramente si lo comparamos con el caso 1, debido obviamente a la desconexión que ha sufrido la red, pero sigue siendo inferior a 1 ms.

En el caso 3 simplemente se ha optado por balancear la carga entre los dos caminos posibles, con la intención de repartir el ancho de banda utilizado en el enlace principal. De este modo se pretende que al ir los enlaces menos congestionados, se produzcan menos errores. Las gráficas obtenidas en el analizador demuestran que se reparte perfectamente el tráfico de forma idéntica por los dos nodos. Sorprendentemente, el porcentaje de paquetes perdidos es superior al caso 1, lo que indica que el balanceo no nos es beneficioso.

Finalmente, cuando repetimos las medidas pero con tráfico real (caso 4.1), vemos que el porcentaje de paquetes perdidos disminuye. Este hecho es lógico, ya que el vídeo transmitido no usa el máximo caudal disponible por el enlace, y por lo tanto se producen menos errores. Por otro lado, al no tener el vídeo un caudal constante, el jitter aumenta considerablemente (2.1 ms de media) y fluctúa entre 0 y 11 ms. A pesar de este aumento, como el programa VLC usa un buffer donde guardar momentáneamente los paquetes recibidos, el vídeo se reproduce correctamente y sin saltos en el destino.

Cuando se desconecta el enlace en el caso 4.2, la red se recupera como en el caso 2, pero un observador en el destino, que esté viendo el vídeo, no se percata de la desconexión. Esto se debe a que en el instante de la desconexión se estaban transmitiendo pocos datos (vemos que el porcentaje de paquetes perdidos es inferior al porcentaje de tiempo desconectado) y por lo tanto si el buffer tarda más en vaciarse que el tiempo que el enlace está desconectado, el vídeo no se interrumpe.

El jitter para este caso aumenta más que en la relación del caso 1 y 2, pero los instantes de vídeo transmitidos en los casos 4.1 y 4.2 no son exactamente los mismos, y por lo tanto no es comparable.

Para terminar, cuando realizamos el balanceo del tráfico real, vemos que se comporta de igual forma que en el caso 3, aumentando ligeramente las pérdidas con referencia al caso 4.1, pero realizándose correctamente.

Con esta prueba se puede concluir que la red troncal MPLS funciona tal como se esperaba al conectar a su destino una red Ethernet. Simplemente el porcentaje de paquetes perdidos ha aumentado ligeramente al balancear la carga, pero sin afectación al visualizar el vídeo.

6.2. Pruebas a través de los routers IP-MPLS - Wifi

Medidas: Caudal, retardo extremo-extremo, jitter extremo-extremo, pérdidas

Estas medidas se tomarán mediante ping e Jperf (duración 2 minutos), capturando mediante Wireshark las trazas de todo el tráfico emitido y recibido, para los siguientes casos de uso:

Caso 1: Medidas sin afectación de ningún tipo (medidas libres)

Caso 2: Medidas con rotura de enlace y activación de Backup. La rotura se emulará mediante la simple desconexión de uno de los enlaces.

Caso 3: Medidas con balanceo de carga.

Caso 4: Transmisión de tráfico real enviando video con el programa VLC, repitiendo los casos 1, 2 y 3.

Escenario: Como usaremos el Wifi, el escenario será el siguiente:

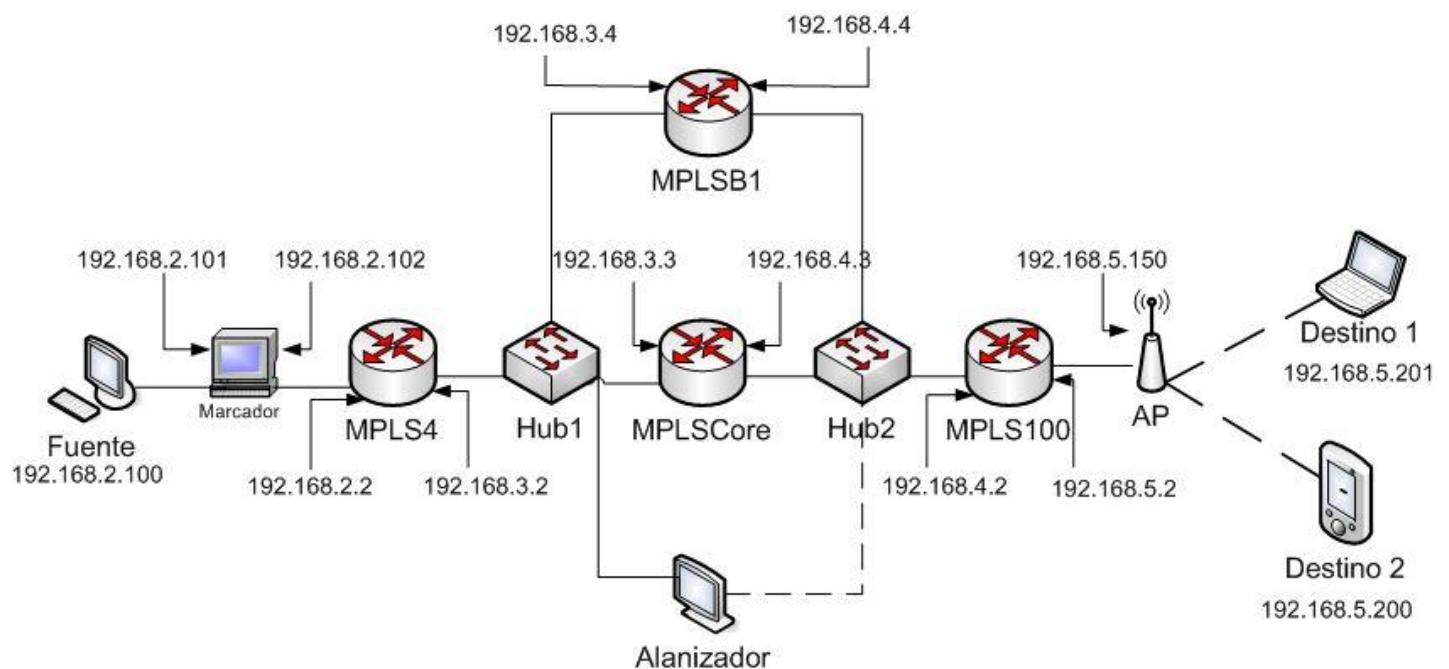


Figura 53: Escenario con Backup y WiFi

Caso 1:

El caudal capturado en cada uno de los puntos de observación es el siguiente:

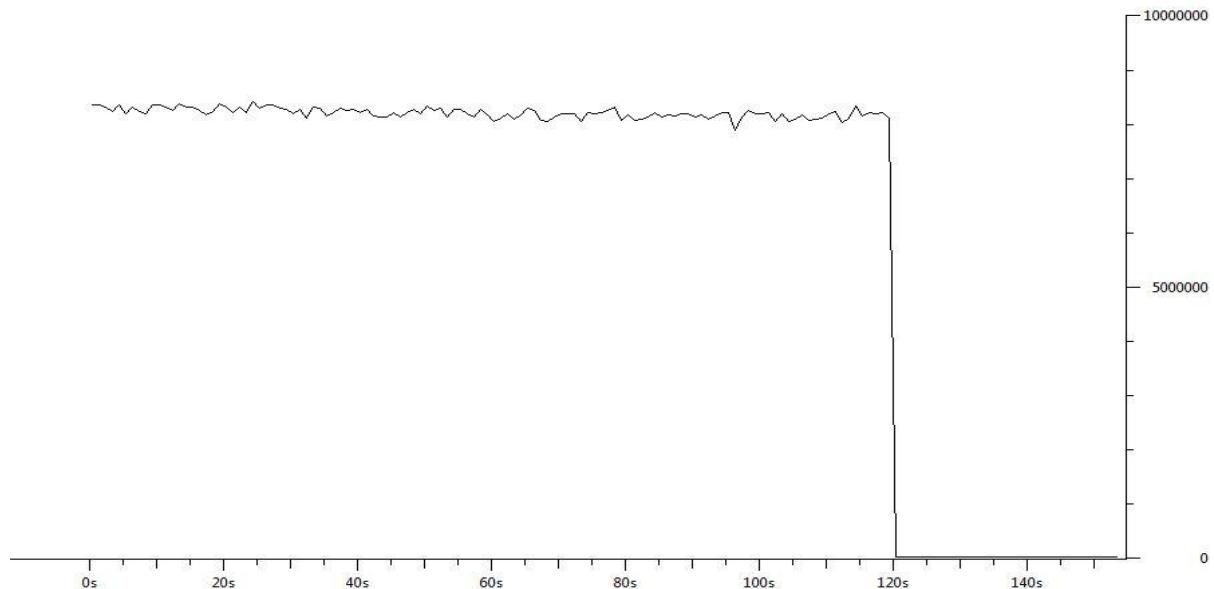
Origen:

Figura 54: Caudal en el Origen en bits por segundo

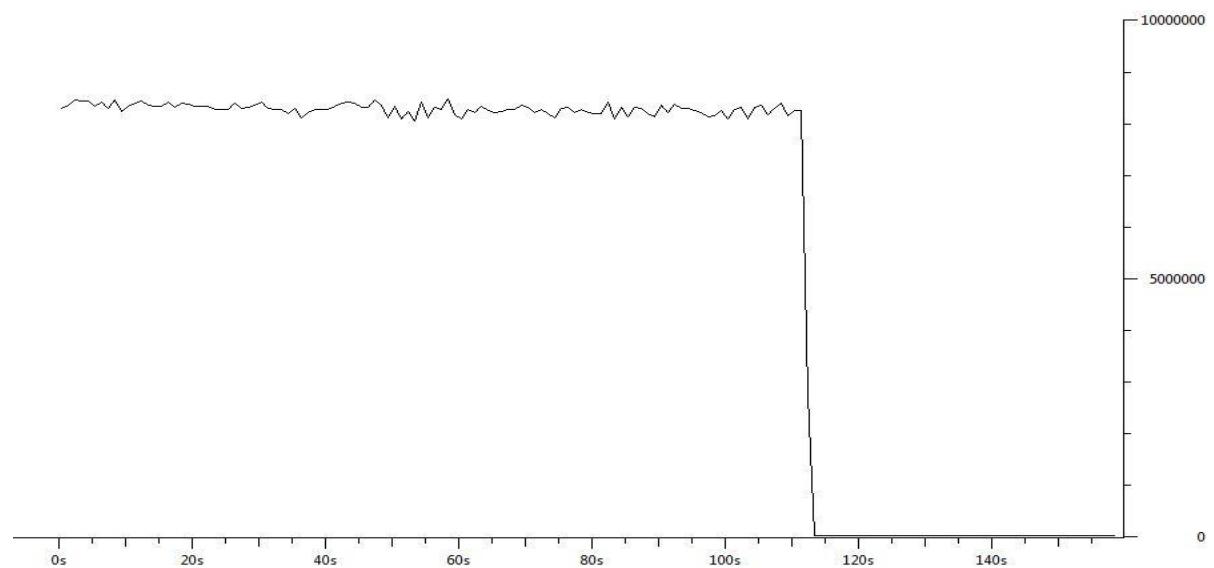
Analizador:

Figura 55: Caudal en el Analizador en bits por segundo

Destino:

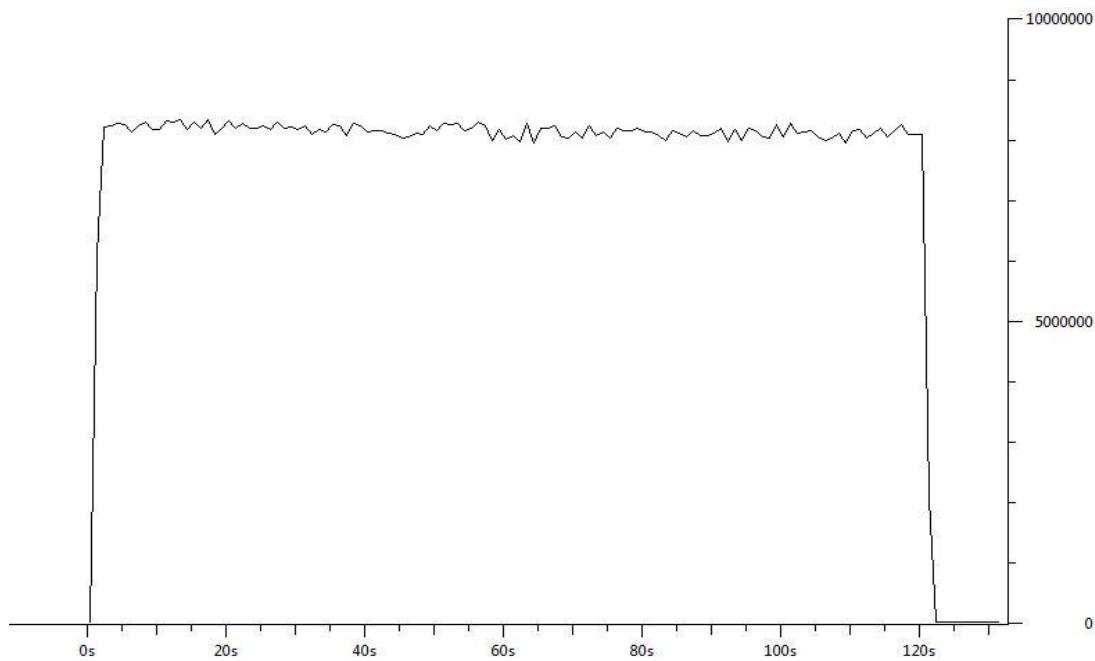


Figura 56: Caudal en el Destino en bits por segundo

También obtenemos el siguiente jitter extremo a extremo:

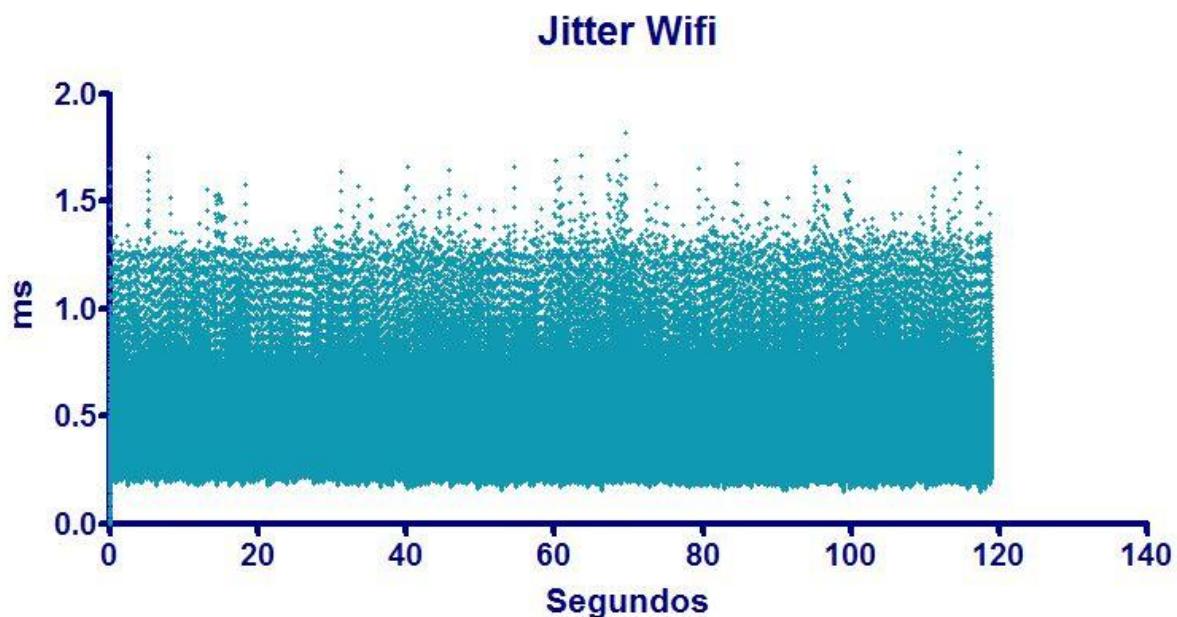


Figura 57: Jitter en el Caso 1

Con valor medio: **0.4129 ms**. En el gráfico no se aprecia este valor medio, porque hay muchos paquetes, y por lo tanto muchos puntos, muy juntos.

El tanto por ciento de **paquetes perdidos** es: $1 - (504333/507753) = 0.67\%$

Caso 2:

En este segundo caso desconectaremos el cable entre el Hub1 y el rotuer MPLSCore sobre el segundo 30 de la transmisión. Luego, a los 60 segundos lo volveremos a conectar.

El caudal capturado en cada uno de los puntos de observación es el siguiente:

Origen:

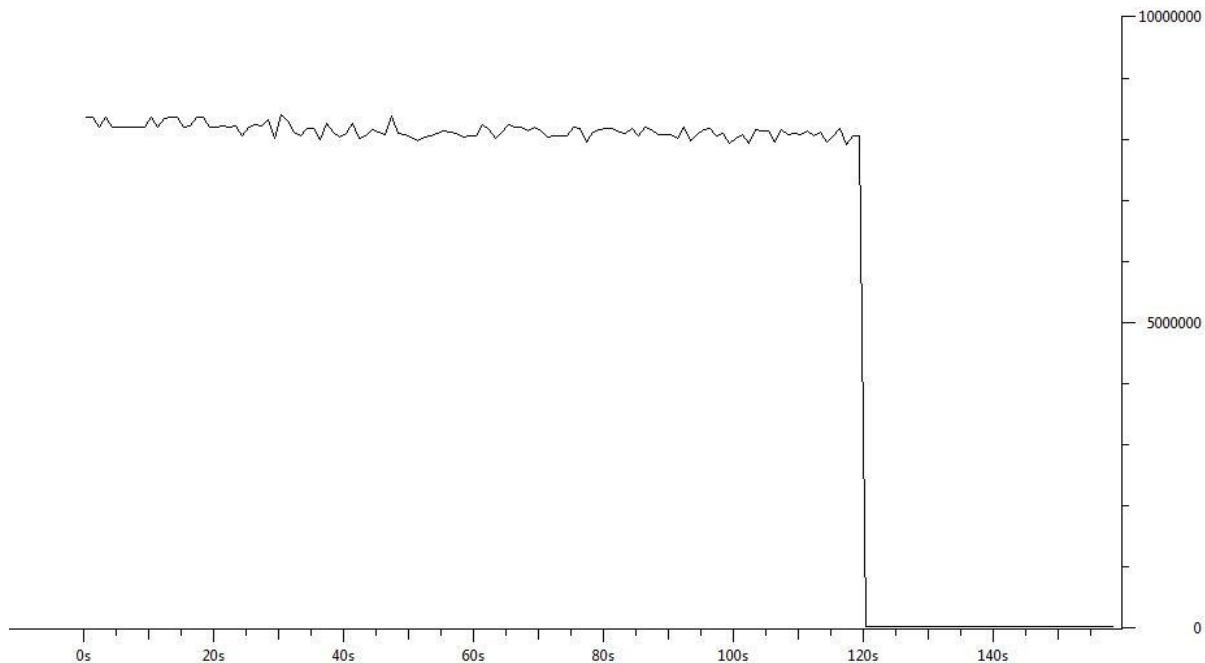


Figura 58: Caudal en el Origen en bits por segundo

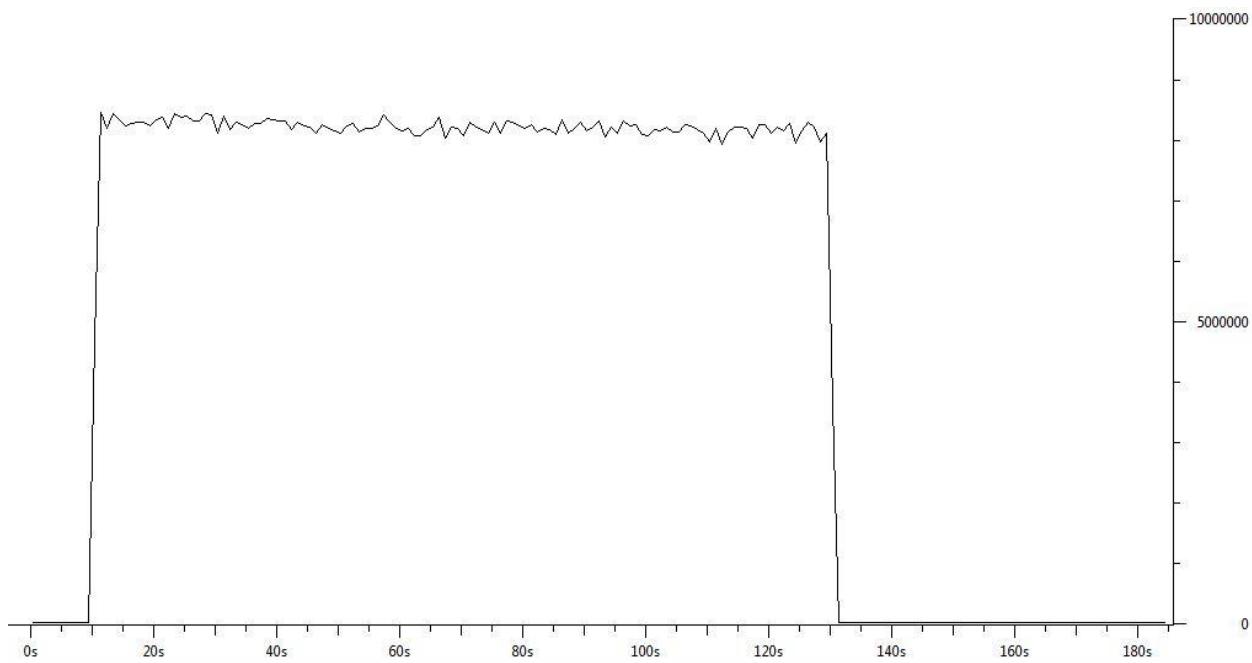
Analizador:

Figura 59: Caudal en el Analizador en bits por segundo

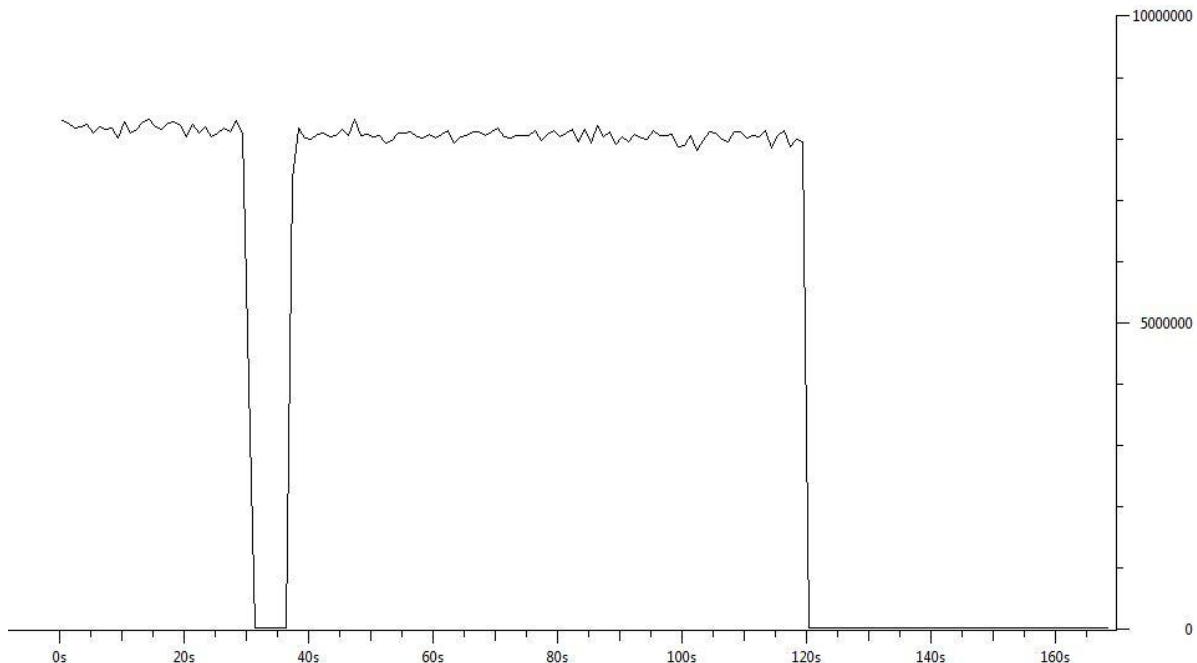
Destino:

Figura 60: Caudal en el Destino en bits por segundo

Si sepáramos el ancho de banda capturado por el “Analizador” en función del destino del paquete, podemos ver claramente cómo actúa el mecanismo de Backup que hemos implementado. Siendo el gráfico en rojo los paquetes que se dirigen al router MPLSCore, y el gráfico en azul los paquetes que se dirigen al router MPLSB1.

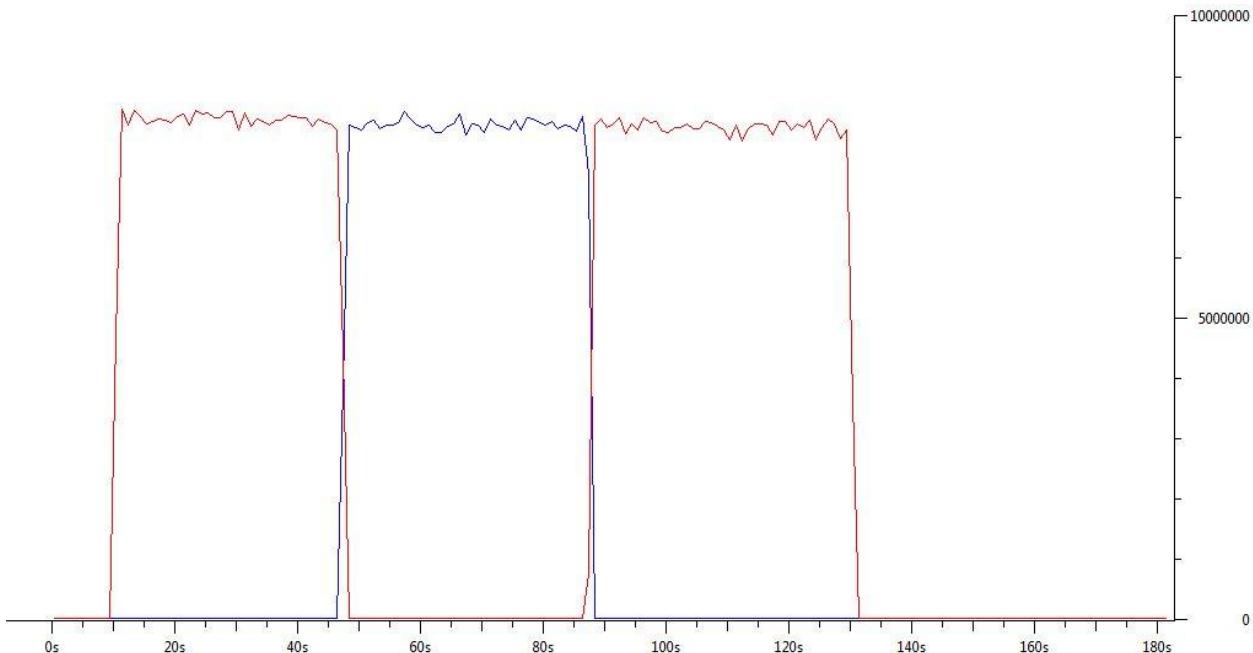


Figura 61: Detalle del destino de los paquetes en el Analizador en bits por segundo

Y el jitter extremo-extremo es el siguiente:

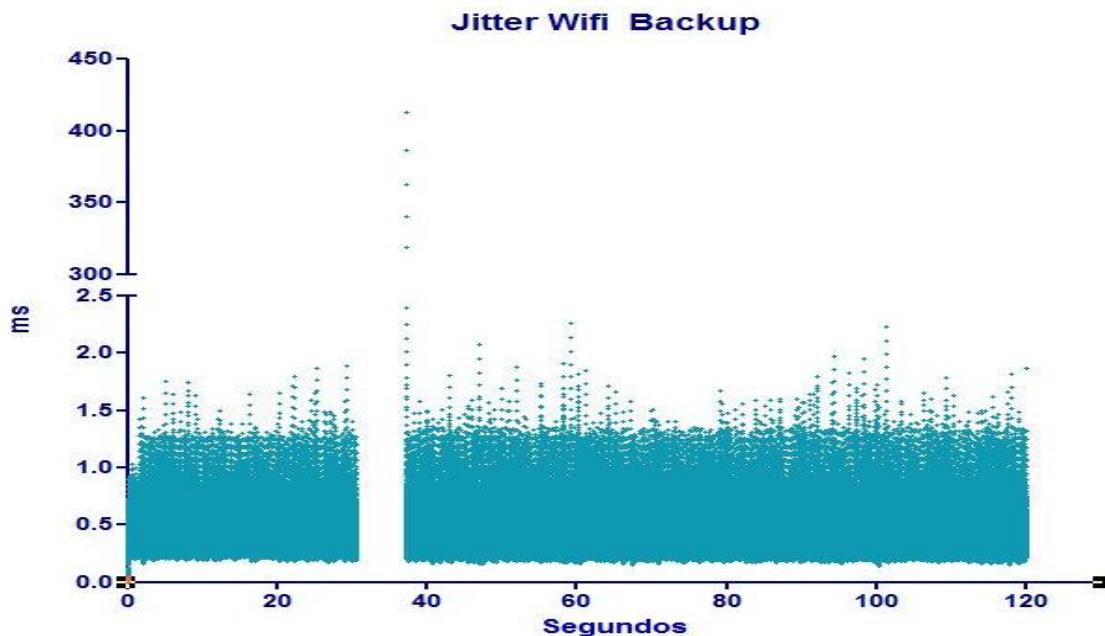


Figura 62: Jitter en el Caso 2

Con valor medio: **0.4283 ms**

El porcentaje de pérdidas es: $1 - (472333/503254) = 6,14\%$. El tiempo durante el que el enlace ha estado desconectado ha sido de: **6,6 seg, un 5,49% del tiempo**.

Caso 3:

Transmitimos 100 segundos de tráfico con Jperf, pero balanceando la carga entre los dos caminos disponibles.

Origen:

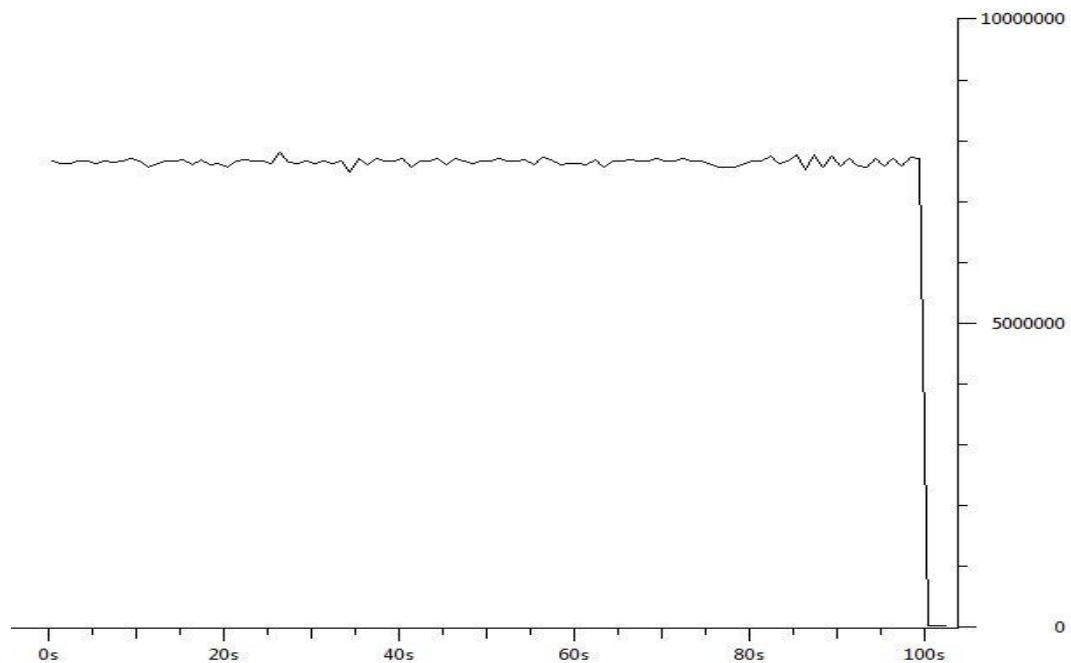


Figura 63: Caudal en el Origen en bits por segundo

Analizador:

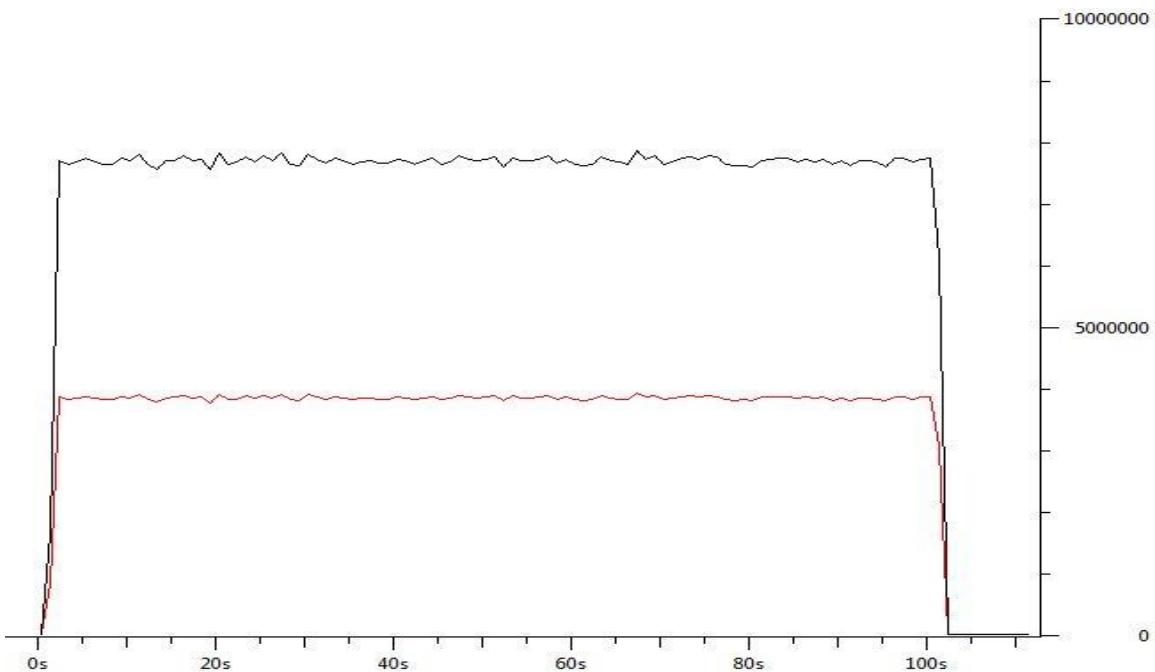


Figura 64: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo

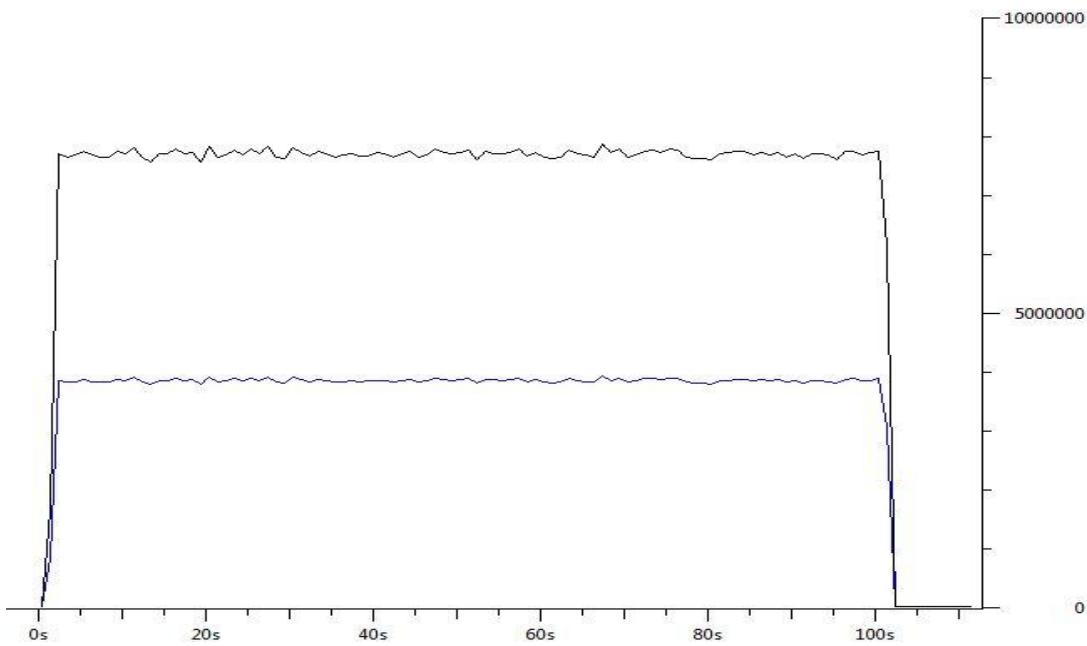


Figura 65: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo

Donde el tráfico marcado en negro es el tráfico total que pasa por el analizador, el rojo el que se dirige al router MPLSCore y el azul el que se dirige a router de Backup.

Destino:

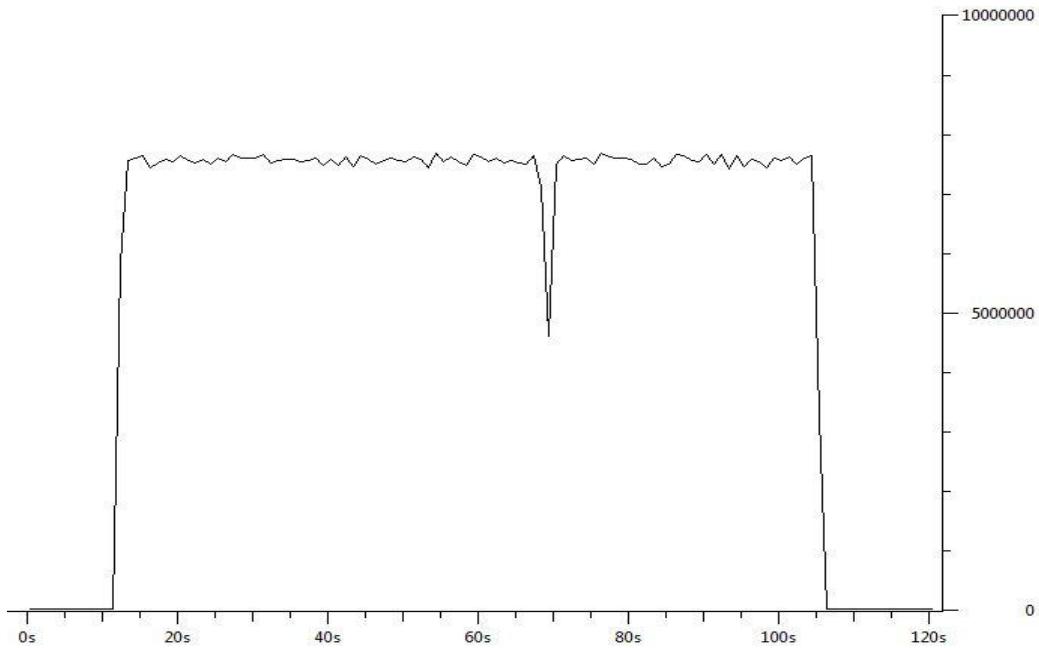


Figura 66: Caudan en el Destino en bits por segundo

Y el porcentaje de **pérdidas** es: $1 - (256013 / 279024) = \mathbf{8.24\%}$

Caso 4:

Caso 4.1:

Transmitimos 2 minutos de vídeo sin afectación. Para este caso, usaremos para el envío el mismo protocolo de transporte UDP que con el Jperff, ya que es el más adecuado para la transmisión de vídeo, porque no introduce retardo.

Origen:

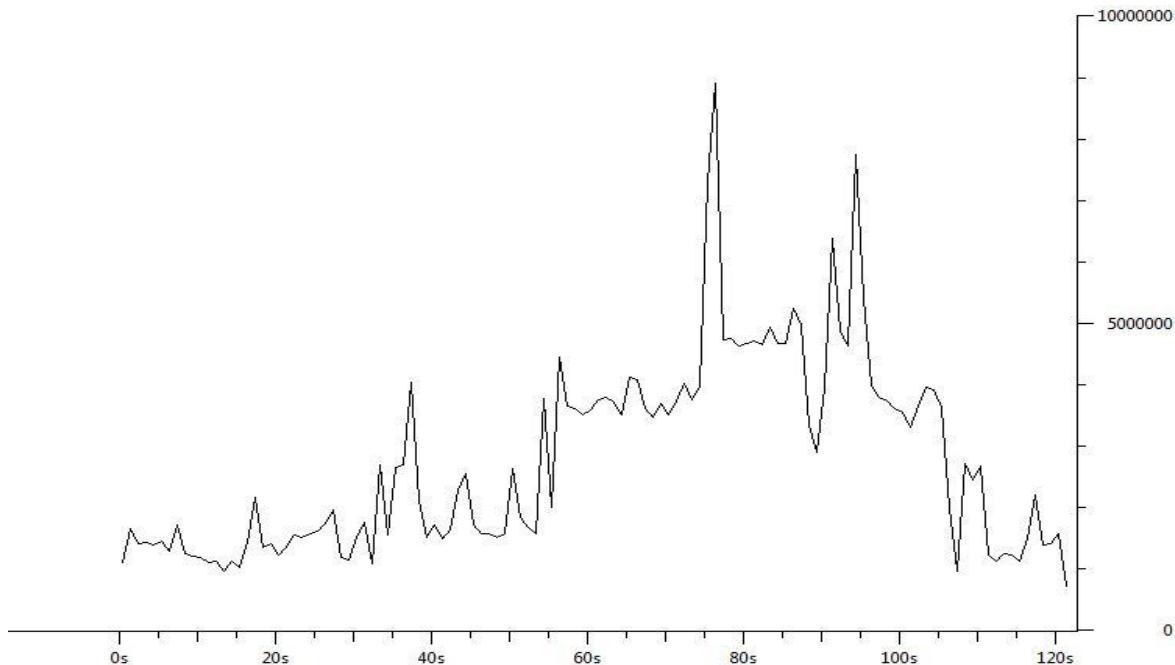


Figura 67: Caudal en el Origen en bits por segundo

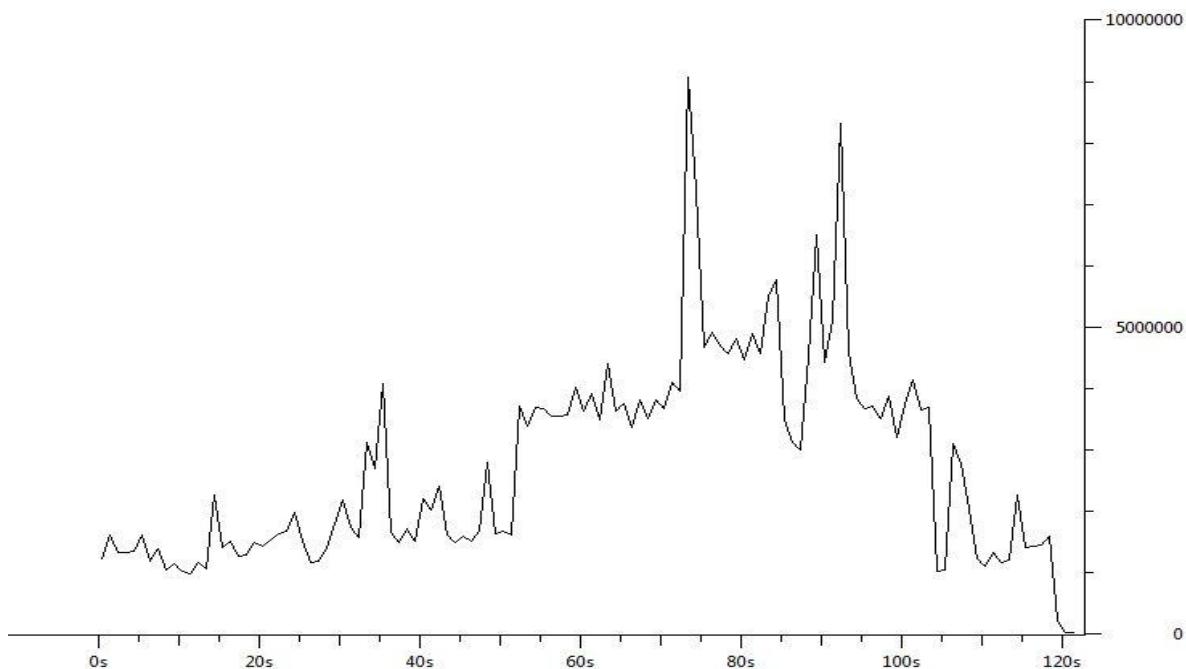
Analizador:

Figura 68: Caudal en el Analizador en bits por segundo

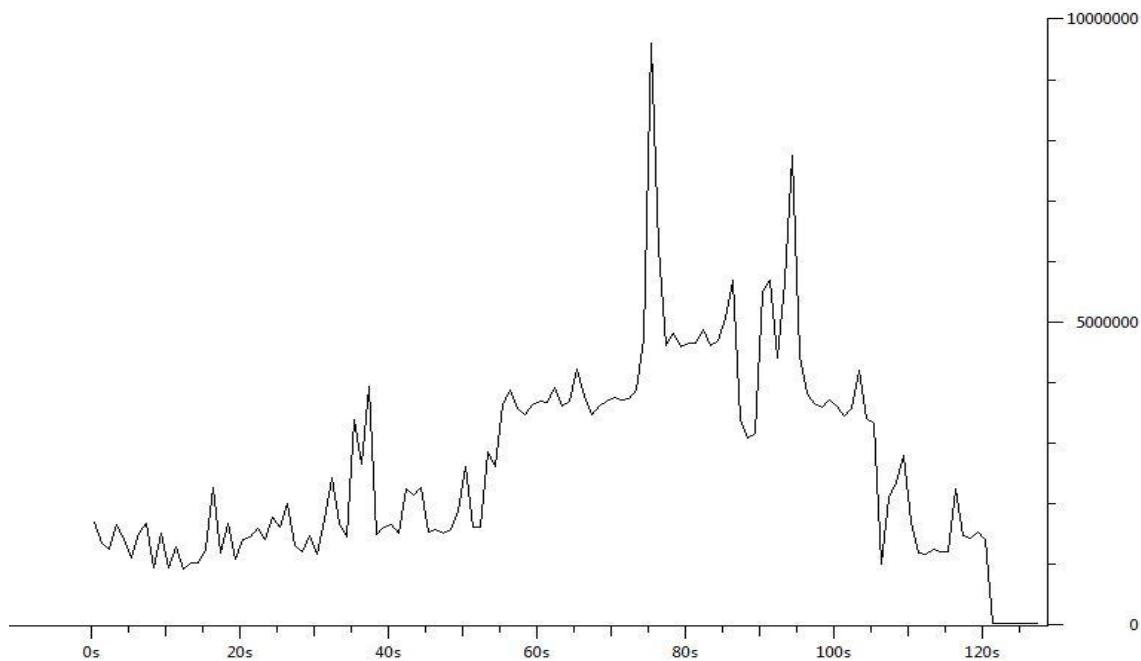
Destino:

Figura 69: Caudal en el Destino en bits por segundo

Y el jitter en este caso es:

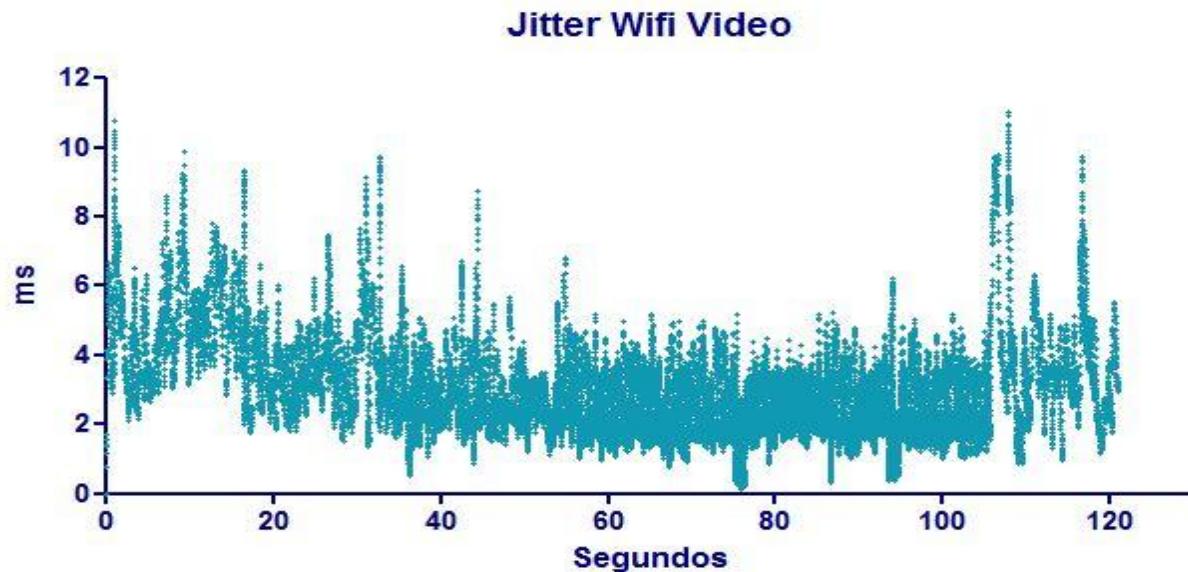


Figura 70: Jitter en el Caso 4.1

Con **valor medio: 2.821 ms**

Las **pérdidas** sin corte son: $1 - (30705/30805) = 0.324\%$

Caso 4.2:

Ahora repetimos la misma transmisión pero desconectando el mismo enlace que en el caso 2, en los mismos instantes.

Origen:

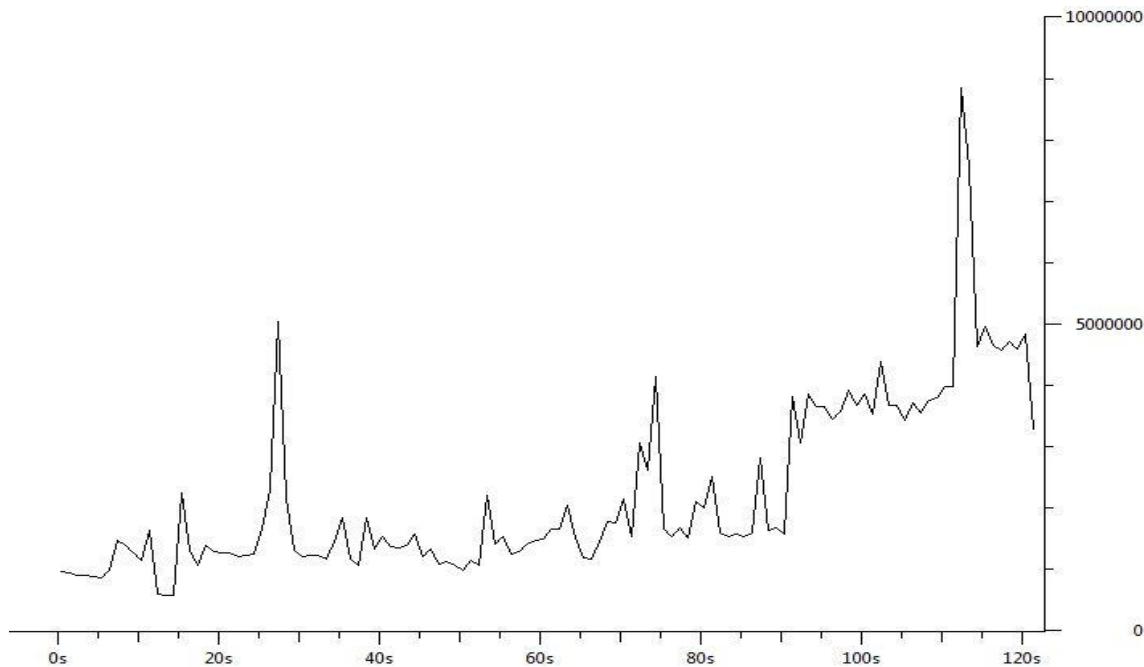


Figura 71: Caudal en el Origen en bits por segundo

Analizador:

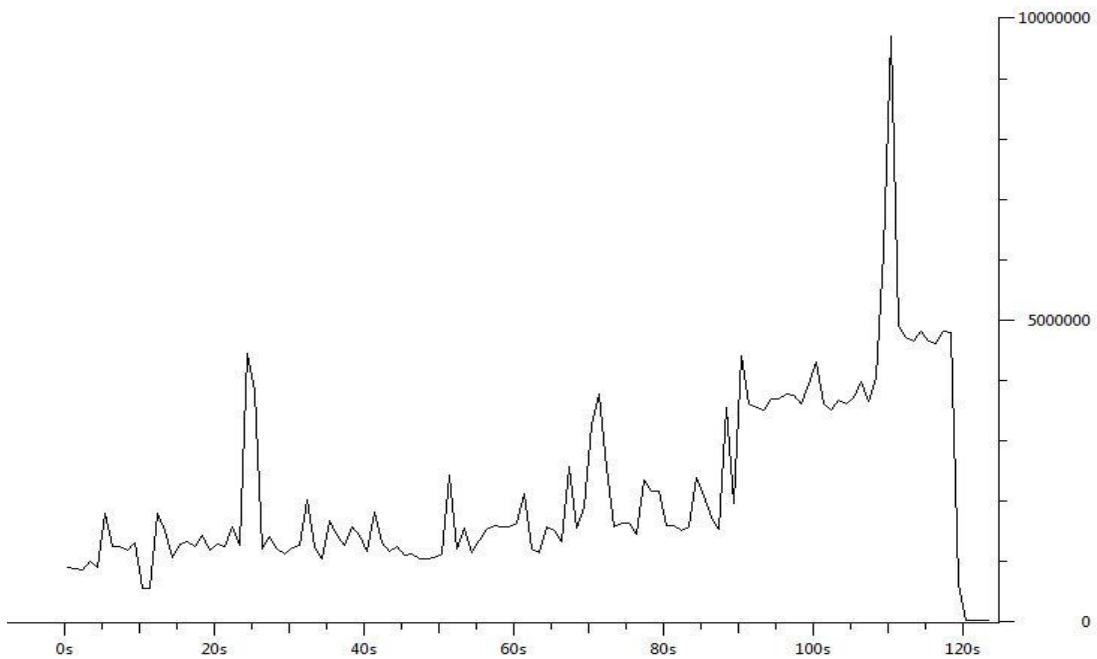


Figura 72: Caudal en el Analizador en bits por segundo

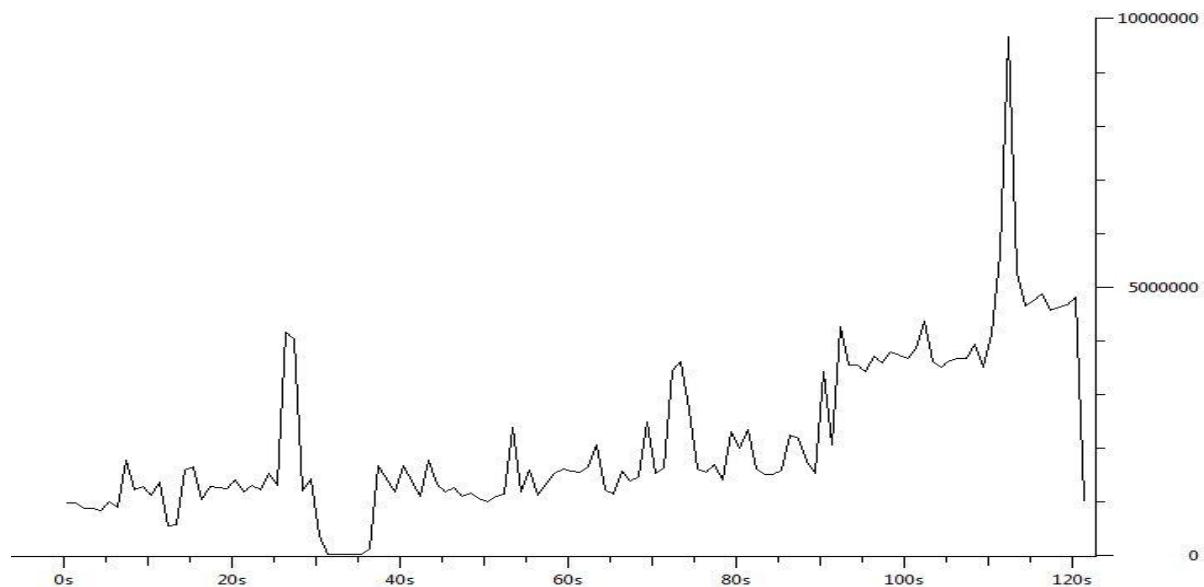
Destino:

Figura 73: Caudal en el Destino en bits por segundo

Como en las pruebas anteriores con Ethernet, si sepáramos el ancho de banda capturado por el “Analizador” en función del destino del paquete, podemos ver claramente cómo actúa el mecanismo de Backup que hemos implementado:

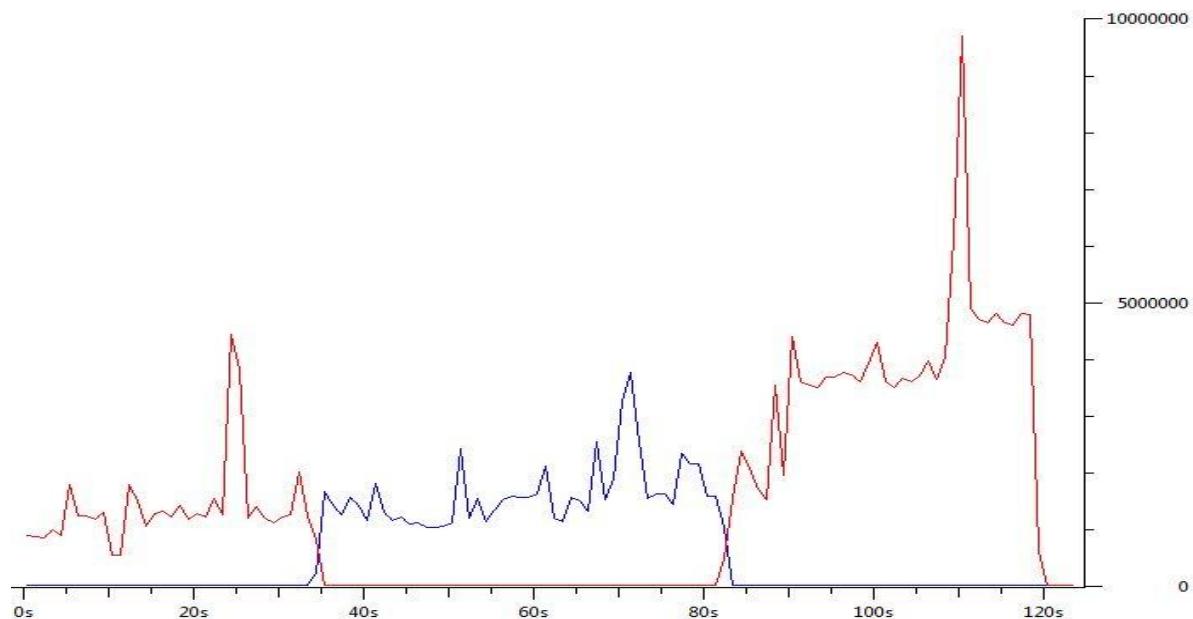


Figura 74: Detalle del destino de los paquetes en el Analizador en bits por segundo

Y con Jitter:

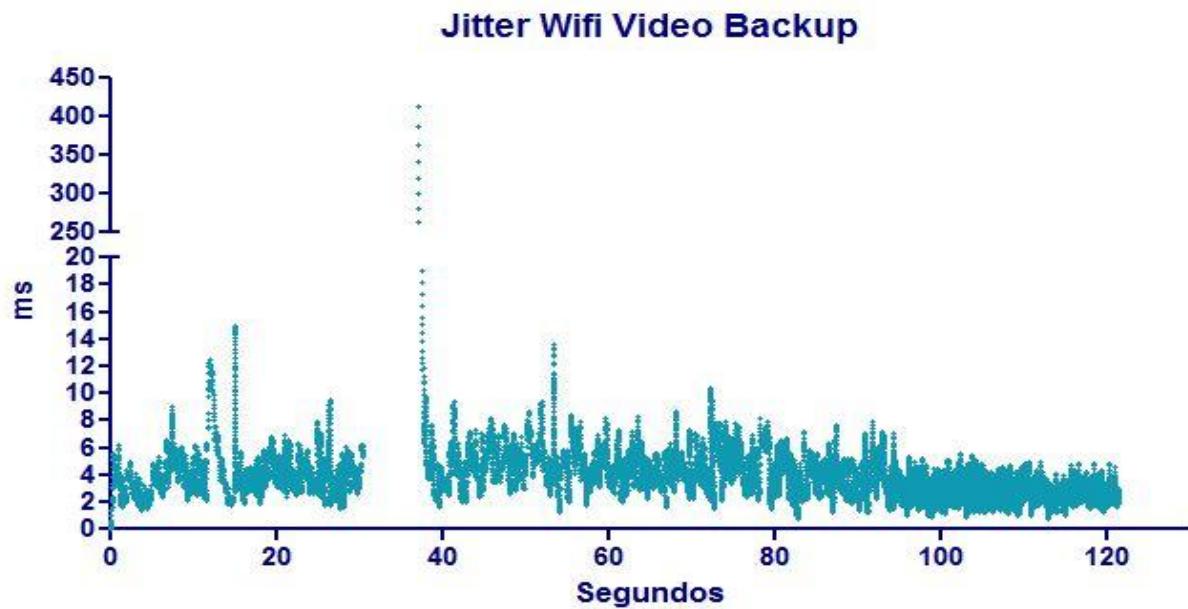


Figura 75: Jitter en el caso 4.2

Con **valor medio: 3.023ms**

Las **pérdidas** con el corte son: $1 - (23781/24632) = 3,45\%$. Aunque el enlace está interrumpido durante **6,5 seg** que es equivalente al 5,4% del tiempo, la interrupción se ha producido durante un periodo con tasa de envío baja, lo que explica las bajas pérdidas.

Caso 4.3:

Ahora analizamos 100 segundos de tráfico real balanceando la carga. Usaremos el método de balanceo “por-paquete”, lo que significa que cada paquete se irá alternando entre los dos caminos disponibles.

Origen:

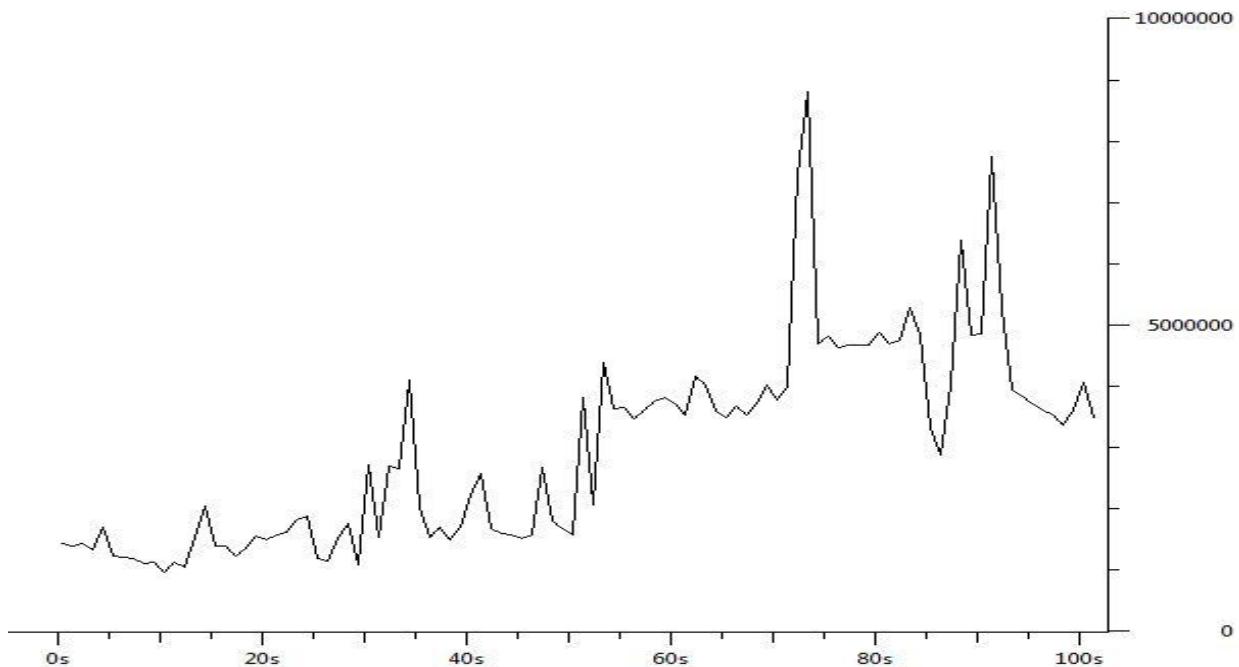


Figura 76: Caudal en el Origen en bits por segundo

Analizador:

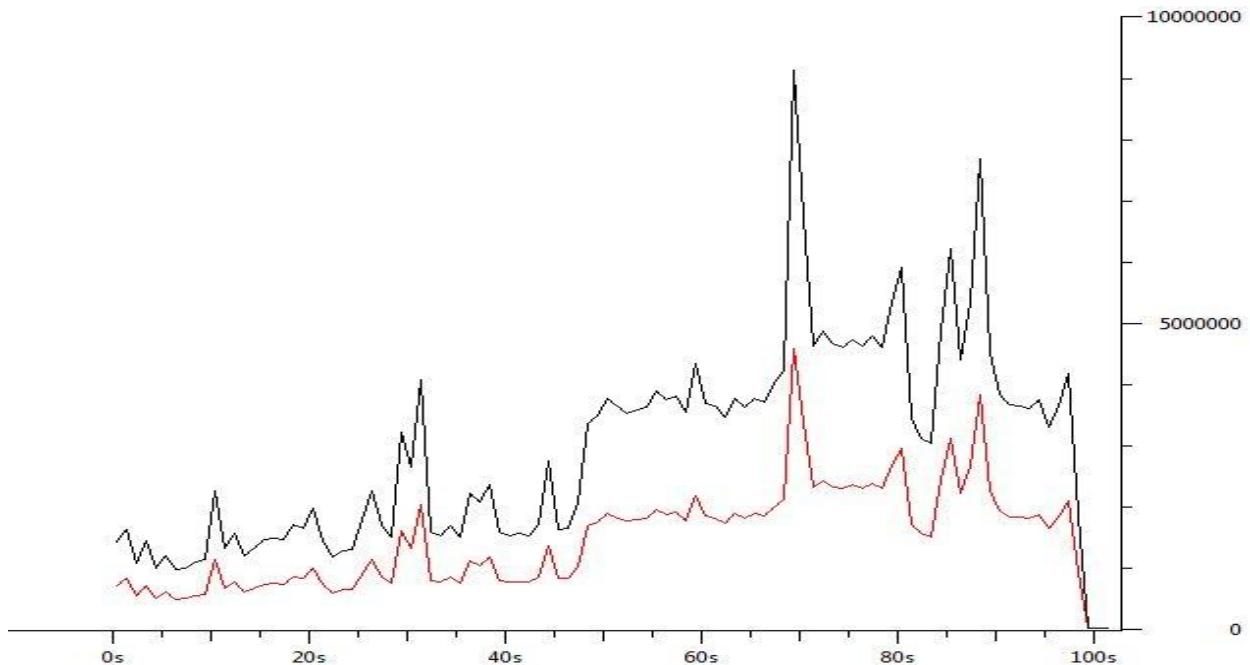


Figura 77: Caudal en el Analizador y con destino al router MPLSCore en bits por segundo

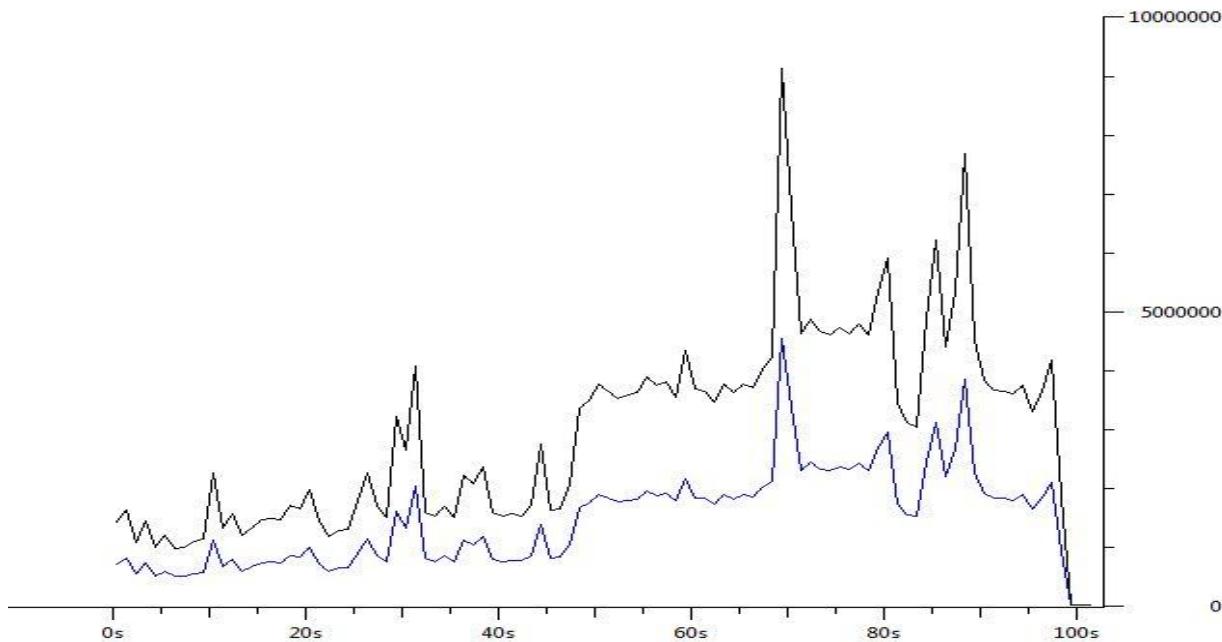


Figura 78: Caudal en el Analizador y con destino al router MPLSB1 en bits por segundo

Donde el tráfico marcado en negro es el tráfico total que pasa por el analizador, el rojo el que se dirige al router MPLSCore y el azul el que se dirige a router de Backup.

Destino:

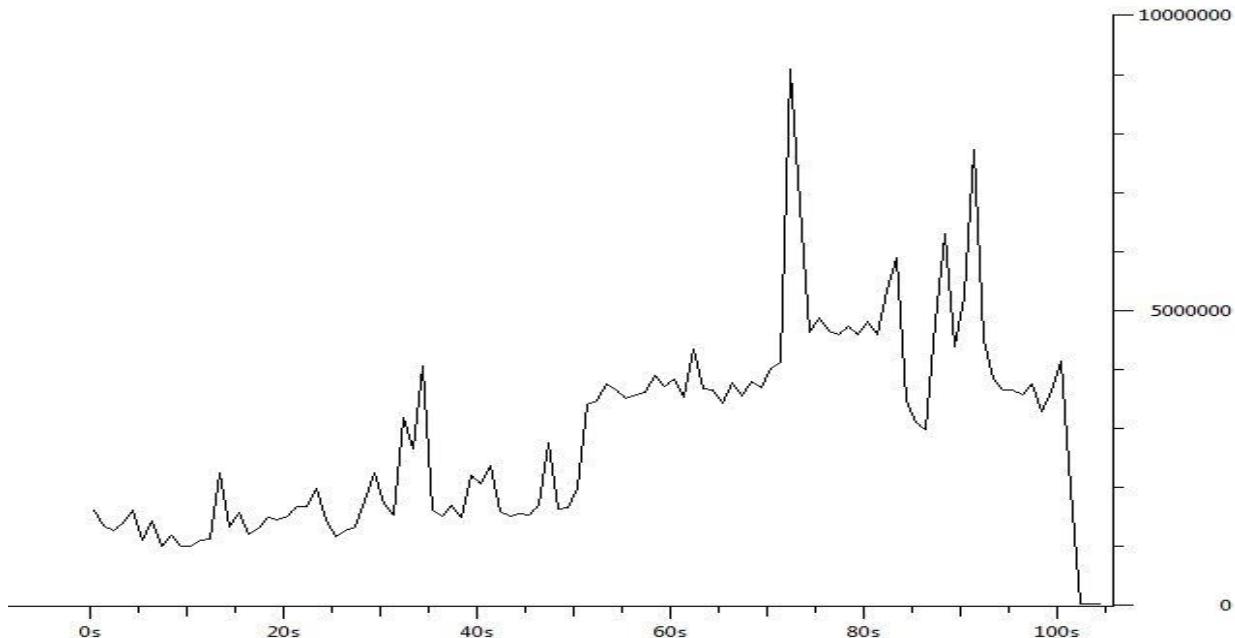


Figura 79: Caudal en e Destino en bits por segundo

Las **pérdidas** para este caso son: $1 - (27586/27740) = 0.55\%$

6.2.1. Conclusiones

Si recogemos los resultados obtenidos en los anteriores casos, obtenemos la siguiente tabla:

Caso	Porcentaje de tiempo		
	sin conexión	Porcentaje de pérdidas	Valor medio del Jitter
Caso 1	0%	0.67%	0.4129 ms
Caso 2	5.49%	6.14%	0.4283 ms
Caso 3	0%	8.24%	-
Caso 4.1	0%	0.33%	2.821 ms
Caso 4.2	5.4%	3.45%	3.023 ms
Caso 4.3	0%	0.55%	-

Tabla 14: Resultados de las pruebas MPLS – Wifi

Como con las pruebas con Ethernet, el caso 1 hemos sometido a la red a su máxima capacidad, que esta vez supera ligeramente los 8 Mbps. Igualmente, los resultados de este caso nos sirven como control para las medidas del resto de casos. De forma análoga a las pruebas anteriores, la red se comporta de forma transparente, ya que la forma de las gráficas del caudal del origen y destino son prácticamente iguales, y solo se pierden el 0.67% de los paquetes. Este valor es ligeramente superior que con Ethernet, debido principalmente a que el caudal en este caso es también ligeramente superior, y por eso, se le ha exigido más a la red, provocando más errores. Por otro lado el jitter oscila principalmente entre 0.2 y 1.3 ms, con el valor medio bastante inferior a 0.5 ms, lo que nos indica que para este parámetro es mejor la red Wifi que la Ethernet.

El caso 2 se comportó exactamente igual que con las pruebas con Ethernet. El enlace tarda prácticamente el mismo tiempo en recuperarse y en total se pierden un 6.14% de los paquetes, ligeramente superior al caso 2 de Ethernet, cosa que concuerda con los resultados obtenidos en el caso 1. Por lo que respecta al jitter, aumenta ligeramente si lo comparamos con el caso 1, debido a la desconexión que ha sufrido la red, pero sigue siendo inferior a 0.5 ms e inferior al caso 1 de Ethernet sin desconexión.

Los resultados del caso 3 son remarcables, ya que sin ninguna desconexión ni pérdida de conectividad, el tanto por ciento de paquetes perdidos supera el 8%. Esto puede deberse a que, como hemos visto con Ethernet, al balancear la carga se pierden más paquetes que sin hacerlo. Además, al emplear una red Wifi al final de la red MPLS, se pueden producir colisiones adicionales al ser el medio físico (el aire) un medio no controlado, y se pierden paquetes adicionales. Esto concuerda con las gráficas de caudal obtenidas, ya que hasta el Analizador todo el tráfico parece correcto, lo que indica que la pérdida de paquetes se produce en la interfaz Wifi.

Finalmente, cuando repetimos las medidas pero con tráfico real (caso 4.1), vemos que el porcentaje de paquetes perdidos disminuye, igual como ocurría con Ethernet. Como hemos visto, al no tener el vídeo un caudal constante, el jitter aumenta (2.821 ms de media) y fluctúa entre 0 y 11 ms, pero como los instantes transmitidos no son los mismos en cada caso, no son comparables.

En el caso 4.2, ocurre exactamente como con Ethernet, pero con unas pérdidas menores. Las pérdidas dependen principalmente del uso que se le dé a la red, que en este caso oscila entre solo 2 y 3 Mbps en media. Al ser las pérdidas tan bajas, un observador en el destino que esté viendo el vídeo no se percata de la desconexión, debido como hemos explicado, al buffer del VLC.

El jitter para este caso aumenta ligeramente en relación al caso 1 y 2, pero los instantes de vídeo transmitidos en los casos 4.1 y 4.2 no son exactamente los mismos, y por lo tanto no es comparable.

Para terminar, cuando realizamos el balanceo del tráfico real, vemos que esta vez se comporta correctamente, ya que no se producen errores remarcables en la interfaz Wifi. Simplemente aumentan ligeramente las pérdidas con referencia al caso 4.1, como ya hemos visto siempre que se ha realizado el balanceo.

Con esta prueba se puede concluir que la red troncal MPLS terminada con un interfaz Wifi baja el valor del jitter. Por otro lado, se pueden producir grandes pérdidas de paquetes en el interfaz Wifi si este falla, pero si no lo hace, no aumentan significativamente las pérdidas. Por lo que respecta al balanceo, la red se comporta igual que con Ethernet.

6.3. Pruebas a través de los routers IP-MPLS - QoS

Medidas: Caudal

Estas medidas se tomarán mediante el envío de un vídeo en streaming (duración 100 segundos), capturando mediante Wireshark las trazas de todo el tráfico emitido y recibido, para los siguientes casos de uso:

- Limitando el tráfico que entra a la red MPLS, con destino a una red Ethernet a 3 Mbps.

Escenario: Como usaremos una red Ethernet, el escenario será el siguiente:

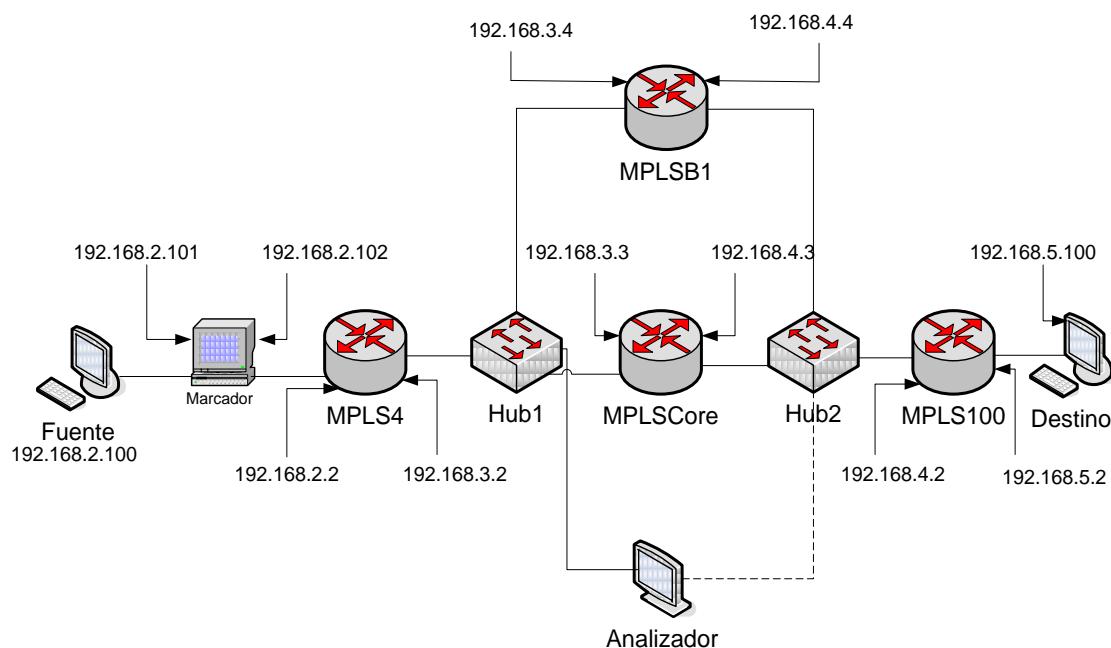


Figura 80: Escenario básico con Backup

Obtenemos los siguientes resultados, en los distintos puntos de observación:

Origen:

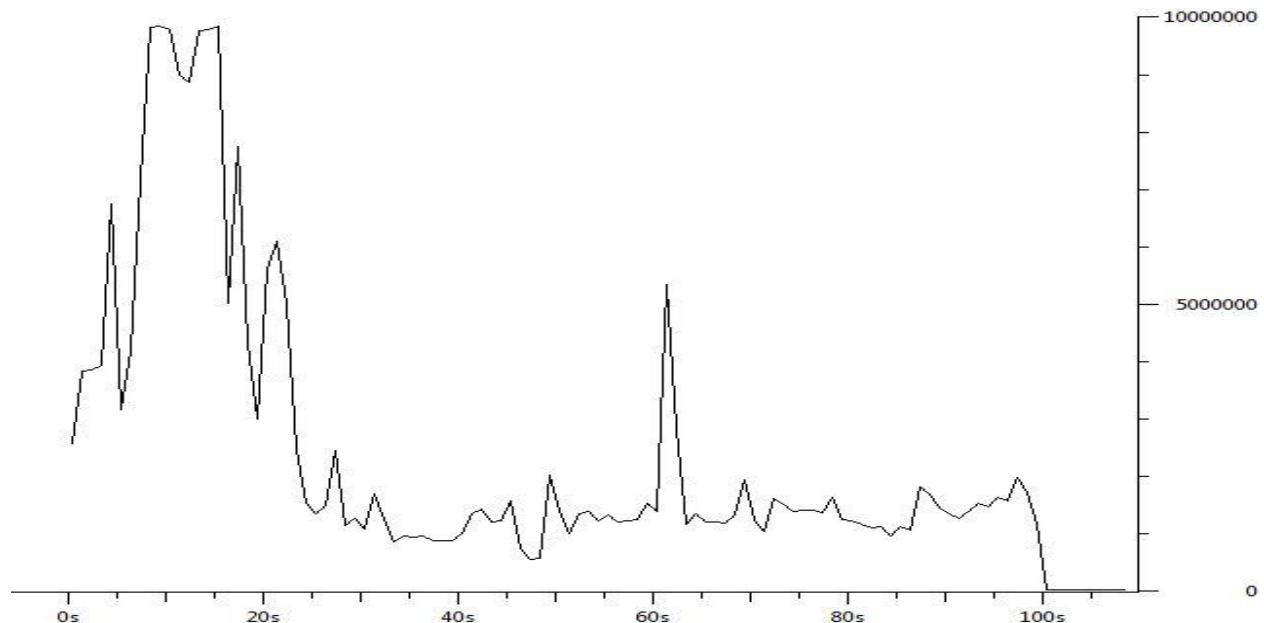


Figura 81: Caudal en el Origen en bits por segundo

Analizador:

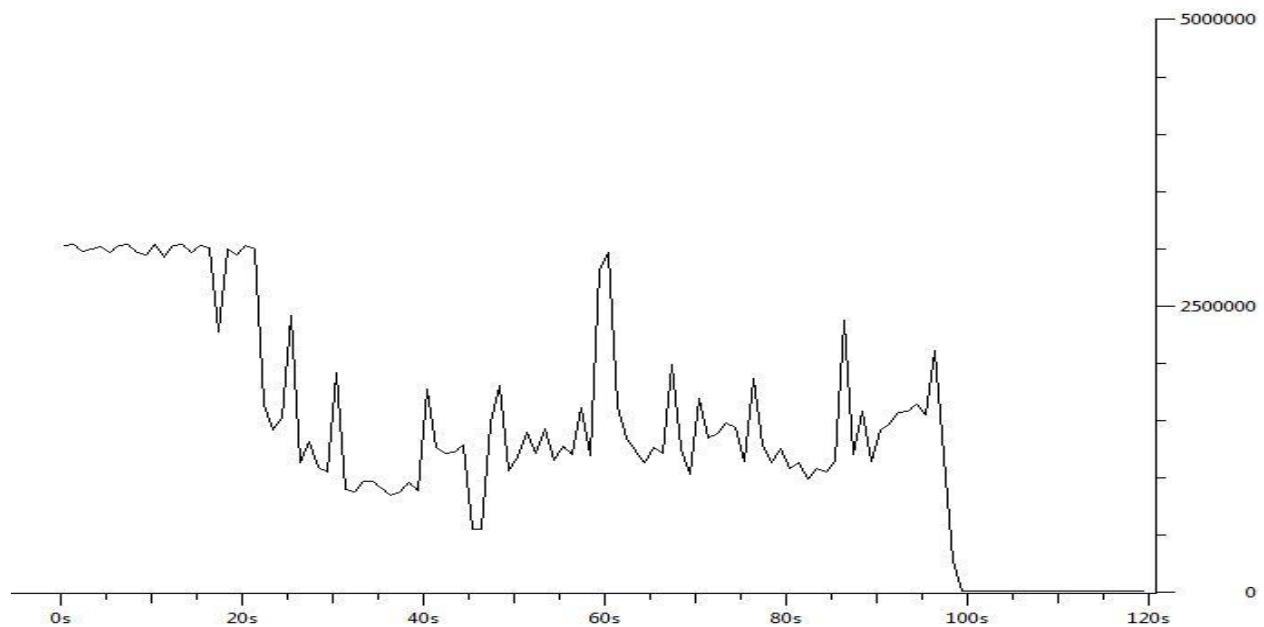


Figura 82: Caudal en el Analizador en bits por segundo

Destino:

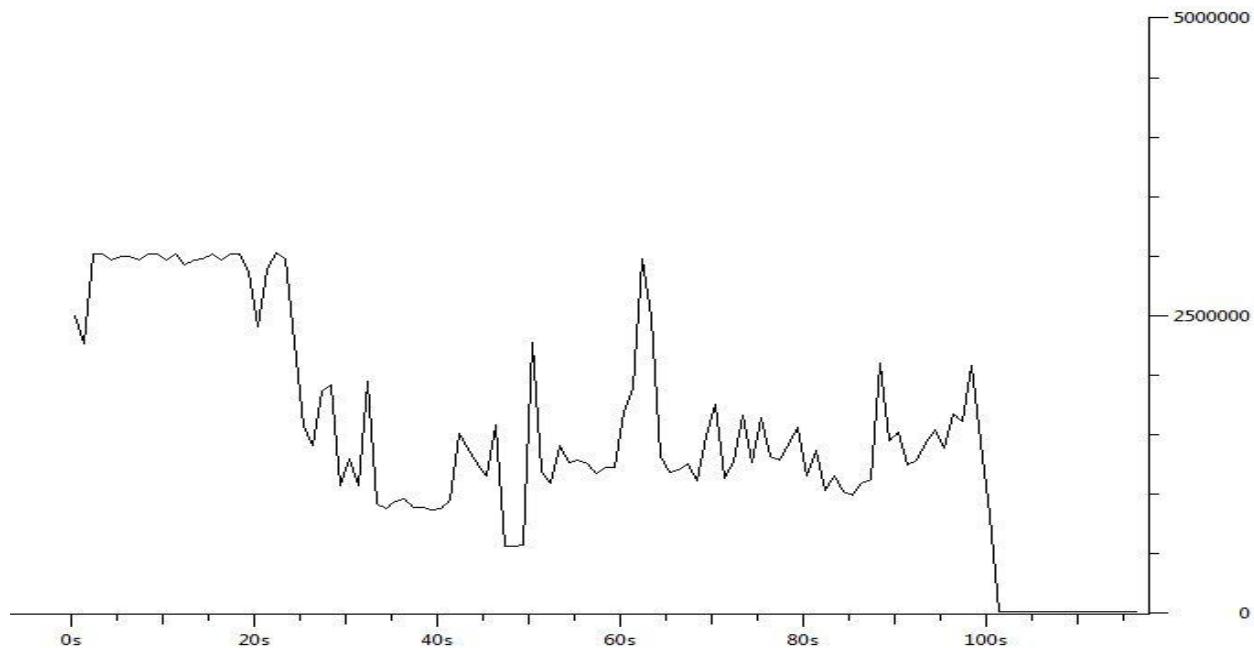


Figura 83: Caudal en el Destino en bits por segundo

6.3.1. Conclusiones

Para esta prueba simplemente se ha limitado el tráfico entrante a la red a 3 Mbps con la política “shape” que se ha explicado en el capítulo 5.9.

Como se observa en las gráficas del caudal, la política funciona perfectamente, eliminando todos los picos que superan los 3 Mbps. Esta política nos puede ser especialmente útil para reservar un ancho de banda determinado a cada flujo que pasa a través de la red. Dependiendo de qué destino tenga el tráfico (Ethernet, Wifi, u otros), podemos marcar dicho tráfico con el “marcador”, y limitarlo con una política a un cierto ancho de banda. Así nos aseguremos que cada flujo no afecta al otro con lo que respecta al ancho de banda.

Cabe destacar que para que se apreciaran bien los resultados de esta prueba, se ha limitado excesivamente el ancho de banda, cosa que ha provocado que se pierdan muchos paquetes. Esto a su vez, ha afectado a la correcta visualización del vídeo en el Destino, provocado numerosos cortes.

Otra política que nos podría ser útil es la de “bandwidth”, que hace lo mismo que la de “shape” pero solo si se está utilizando todo el ancho de banda disponible. Lo que significa que aunque limitemos que tráfico a 3 Mbps, si hay ancho de banda libre, no se limitaría el flujo. No hemos podido realizar pruebas con esta política porque no disponíamos en el laboratorio de suficientes equipos para poder tener los flujos para ocupar los 100 Mbps de los que dispone la red MPLS.

6.4. Pruebas a través de los routers IP-MPLS – Túnel IP-IP

Para finalizar las pruebas, se realizaron las siguientes medias con la Universidad Politécnica de Valencia con tráfico SIP a través de Internet, mediante dos túneles bidireccionales IP-IP.

Medidas: Caudal, retardo extremo-extremo, jitter extremo-extremo, pérdidas

Estas medidas se tomarán mediante ping e Jperf (duración 2 minutos), capturando mediante Wireshark las trazas de todo el tráfico emitido y recibido, para los siguientes casos de uso:

Caso 1: Medidas sin afectación de ningún tipo (medidas libres)

Caso 2: Medidas con rotura de enlace y activación de Backup. La rotura se emulará mediante la simple desconexión de uno de los enlaces.

Caso 3: Medidas con balanceo de carga.

Caso 4: Transmisión de tráfico real enviando tráfico SIP, repitiendo los casos 1, 2 y 3.

Escenario: Como usaremos túneles, el escenario será el siguiente:

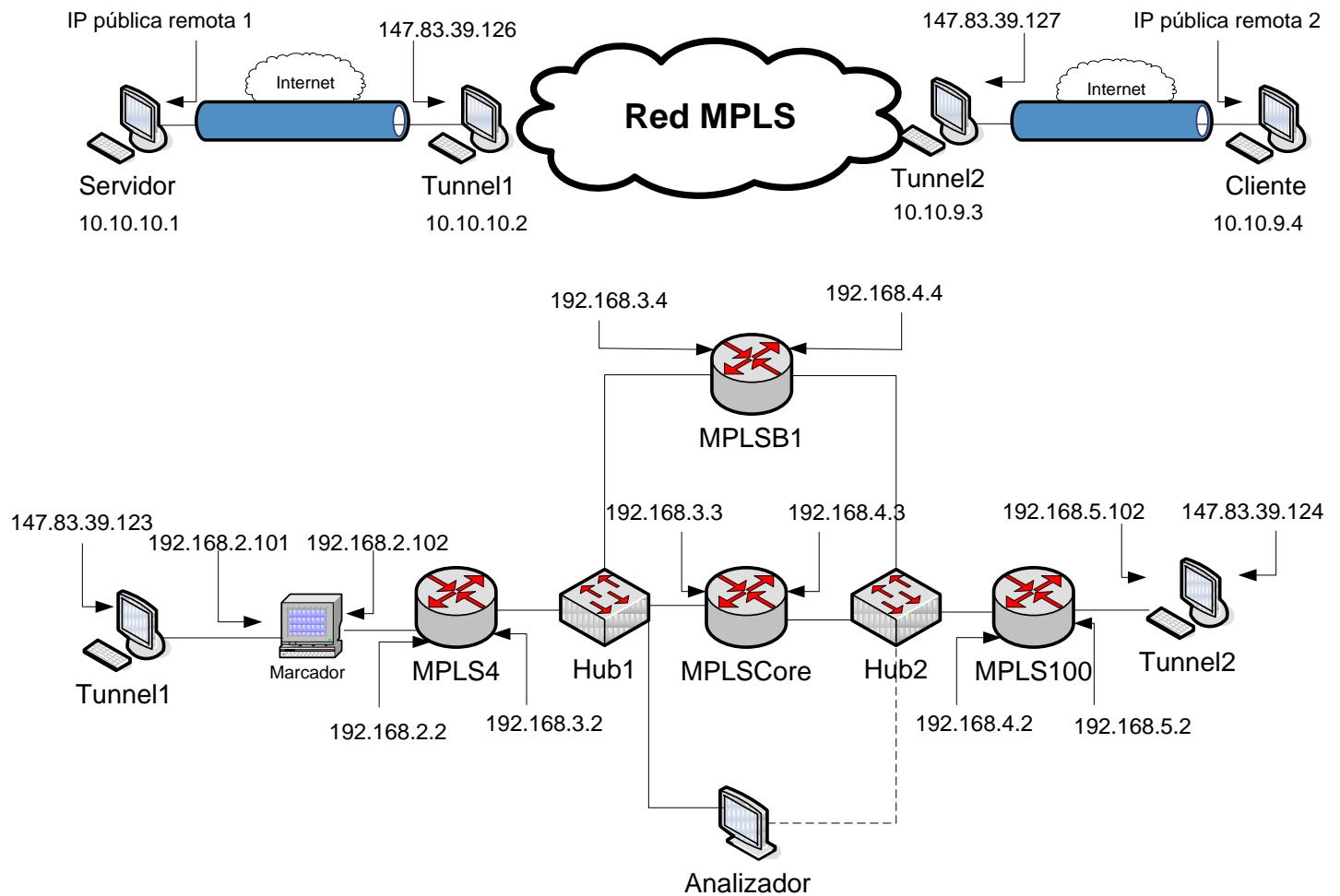


Figura 84: Escenario con Túneles

Y se han usado las siguientes IPs:

IP pública remota 1: **158.42.129.29**

IP pública remota 2: **158.42.128.125**

Lamentablemente, a causa de problemas con los firewalls de Valencia, solo se pudo efectuar el caso 1, con los siguientes resultados:

Podemos observar los siguientes paquetes a lo largo de la red:

Entre el cliente y la red MPLS:

```
Frame 9893 (1514 bytes on wire, 1514 bytes captured)
Ethernet II, Src: Dell_5a:af:5c (b8:ac:6f:5a:af:5c), Dst: NortelNe_25:7e:03 (00:01:81:25:7e:03)
Internet Protocol, Src: 158.42.128.125 (158.42.128.125), Dst: 147.83.39.127 (147.83.39.127)
Internet Protocol, Src: 10.10.9.4 (10.10.9.4), Dst: 10.10.10.1 (10.10.10.1)
Transmission Control Protocol, Src Port: 56579 (56579), Dst Port: commplex-link (5001), Seq: 687
Data (1428 bytes)
```

Figura 85: Paquete entre el cliente y la red MPLS

En medio de la red MPLS:

```
Frame 819 (1498 bytes on wire, 1498 bytes captured)
Ethernet II, Src: Cisco_a4:9a:78 (00:13:80:a4:9a:78), Dst: Cisco_f7:d3:e1 (00:d0:bb:f7:d3:e1)
MultiProtocol Label Switching Header, Label: 22, Exp: 0, S: 1, TTL: 61
Internet Protocol, Src: 10.10.9.4 (10.10.9.4), Dst: 10.10.10.1 (10.10.10.1)
Transmission Control Protocol, Src Port: 56579 (56579), Dst Port: commplex-link (5001), Seq: 687
Data (1428 bytes)
```

Figura 86: Paquete a través de la red MPLS

Y entre la red MPLS y el servidor:

```
Frame 39868 (1514 bytes on wire, 1514 bytes captured)
Ethernet II, Src: NortelNe_25:7e:03 (00:01:81:25:7e:03), Dst: Intel_19:54:f9 (00:19:d1:19:54:f9)
Internet Protocol, Src: 147.83.39.126 (147.83.39.126), Dst: 158.42.129.29 (158.42.129.29)
Internet Protocol, Src: 10.10.9.4 (10.10.9.4), Dst: 10.10.10.1 (10.10.10.1)
Transmission Control Protocol, Src Port: 56579 (56579), Dst Port: commplex-link (5001), Seq: 2214
Data (1428 bytes)
```

Figura 87: Paquete entre la red MPLS y el servidor

Como se usó el protocolo TCP con el programa Iperf, se generaron paquetes de reconocimiento ACK en sentido contrario:

Entre el servidor y la red:

```
Frame 31300 (86 bytes on wire, 86 bytes captured)
Ethernet II, Src: Intel_19:54:f9 (00:19:d1:19:54:f9), Dst: NortelNe_25:7e:03 (00:01:81:25:7e:03)
Internet Protocol, Src: 158.42.129.29 (158.42.129.29), Dst: 147.83.39.126 (147.83.39.126)
Internet Protocol, Src: 10.10.10.1 (10.10.10.1), Dst: 10.10.9.4 (10.10.9.4)
Transmission Control Protocol, Src Port: commplex-link (5001), Dst Port: 56579 (56579), Seq: 1,
```

Figura 88: ACK entre el servidor y la red MPLS

En el medio de la red:

```
Frame 817 (70 bytes on wire, 70 bytes captured)
Ethernet II, Src: Cisco_f7:d3:e1 (00:d0:bb:f7:d3:e1), Dst: Cisco_a4:9a:78 (00:13:80:a4:9a:78)
MultiProtocol Label Switching Header, Label: 26, Exp: 0, S: 1, TTL: 61
Internet Protocol, Src: 10.10.10.1 (10.10.10.1), Dst: 10.10.9.4 (10.10.9.4)
Transmission Control Protocol, Src Port: complex-link (5001), Dst Port: 56579 (56579), Seq: 1,
```

Figura 89: ACK a través de la red MPLS

Y finalmente entre la red MPLS y el cliente:

```
Frame 9894 (86 bytes on wire, 86 bytes captured)
Ethernet II, Src: NortelNe_25:7e:03 (00:01:81:25:7e:03), Dst: Dell_5a:af:5c (b8:ac:6f:5a:af:5c)
Internet Protocol, Src: 147.83.39.127 (147.83.39.127), Dst: 158.42.128.125 (158.42.128.125)
Internet Protocol, Src: 10.10.10.1 (10.10.10.1), Dst: 10.10.9.4 (10.10.9.4)
Transmission Control Protocol, Src Port: complex-link (5001), Dst Port: 56579 (56579), Seq: 1,
```

Figura 90: ACK entre la red MPLS y el cliente

Y los siguientes caudales:

Cliente:

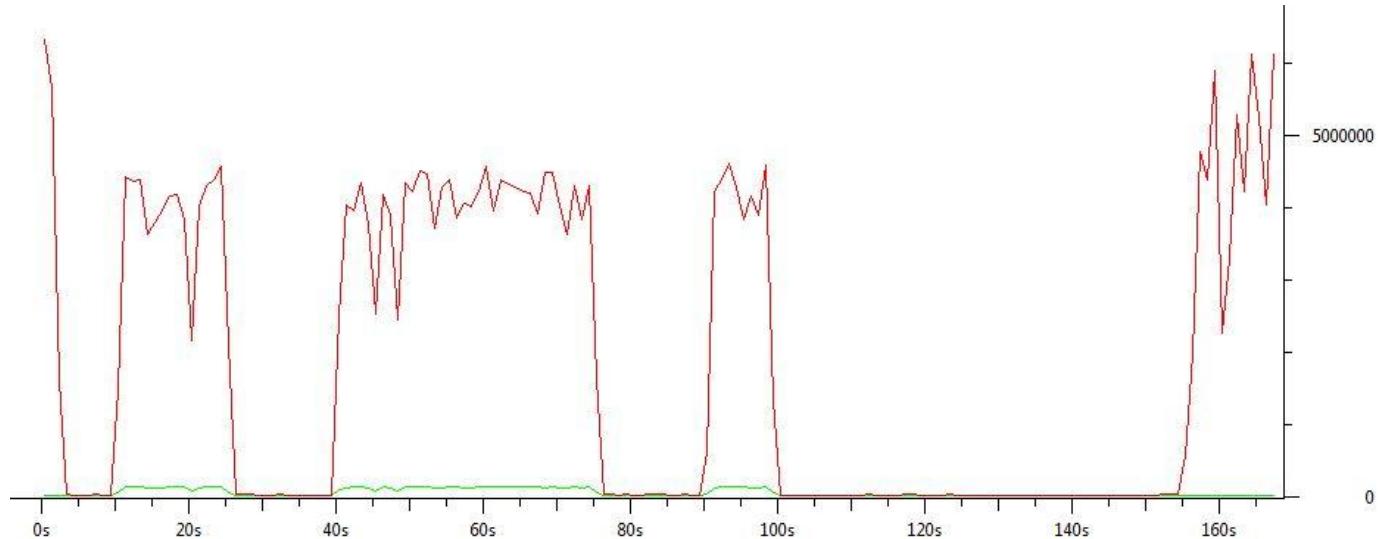


Figura 91: Caudal en el Cliente en bits por segundo

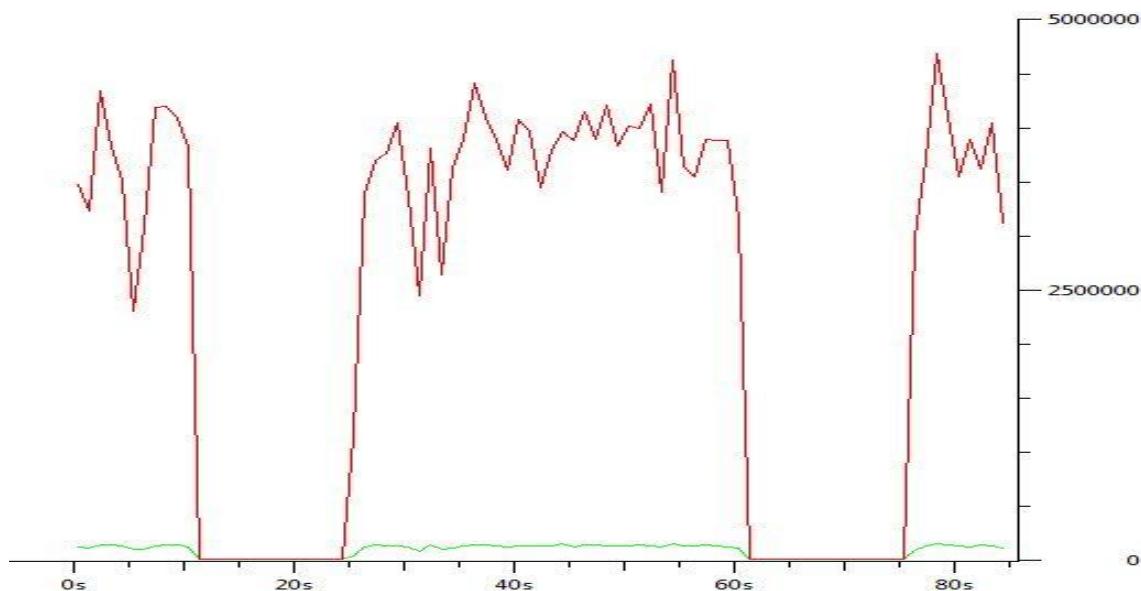
Servidor:

Figura 92: Caudal en el Servidor en bits por segundo

El tráfico fue capturado por la gente de la Universitat Politècnica de Valencia, y no tienen la misma duración. La gráfica roja corresponde a los datos transmitidos, y la gráfica en verde corresponde a los ACKs que envía el servidor de regreso.

Por otro lado, como solo se pudo hacer el caso 1, y limitado por el firewall de la Universitat Politècnica de Valencia, no hemos podido obtener ninguna conclusión de esta prueba.

7. Conclusiones y línea futuras de trabajo

7.1. Conclusiones

En este Proyecto se ha configurado una red troncal MPLS y se ha interconectado con distintas redes. Si analizamos los objetivos que nos propusimos en el capítulo 2, vemos que estos se han cumplido ampliamente.

Se ha estudiado la arquitectura y el funcionamiento del protocolo MPLS, y se ha aprendido a usar los comandos básicos para la configuración de una pequeña red.

Con esta pequeña red configurada, se ha establecido una conexión extremo-extremo y se ha verificar su correcto funcionamiento.

Hemos añadido un camino de backup para dotar de redundancia a la red, y de balanceo de carga para repartir el ancho de banda. Además, hemos configurado la red para que actúen estos mecanismos de forma automática.

Una vez configurados estos mecanismos, hemos realizado diversas pruebas para comprobar cómo responde la red troncal MPLS al conectarle distintas redes heterogéneas y con tráfico de video real. También se ha intentado conectar la red MPLS con equipos remotos de la Universitat Politècnica de Valencia y realizar nuevas pruebas, pero por problemas ajenos a nosotros, no se han podido realizar correctamente.

Además, se ha estudiado como implementar mecanismos de QoS a través de la red MPLS y probar su funcionamiento. En concreto se ha probado cómo funciona la política “shape”.

Finalmente se han elaborado análisis y conclusiones sobre los resultados obtenidos con las pruebas llevadas a cabo, especificando los detalles de cada una de las redes que han intervenido.

Con referencia a las pruebas realizadas, se ha podido constatar que los routers utilizan la técnica de extracción de etiquetas en el penúltimo salto (**Penultimate Hop Popping**). Este hecho se ha observado al analizar el proceso de gestión de etiquetas de los routers, que concuerda con su tabla de “forwarding”.

Sobre el mecanismo de Backup, se ha comprobado que cuando se ha producido un fallo en un enlace, la red se ha recuperado a partir de los 5 segundos configurados más 1,5 segundos necesarios para recalcular la nueva ruta. Luego, ha vuelto a redirigir el tráfico al camino principal cuando este ha estado disponible. Todo ello de forma automática, y sin que un observador que estuviese viendo el video transmitido se percata de nada. Además, según las necesidades del tipo de tráfico o el tamaño del buffer en el Destino, podemos configurar un tiempo menor para la recuperación del enlace, sacrificando un poco el rendimiento de la red, ya que se necesitan más paquetes de control. Cabe destacar también, que todo el proceso se ha realizado sin el mecanismo que introduce MPLS, **Fast ReRoute**, con el cual es de esperar que los resultados sean aun mejores.

El mecanismo de balanceo de carga (load balancing) también ha funcionado adecuadamente, enviando cada paquete por un camino distinto, siempre que tuvieran un destino común. Con este mecanismo se esperaba que al repartir la carga entre dos enlaces y no exigirle tanto a la red, bajara la pérdida de paquetes. Pero en lugar eso, la pérdidas de paquetes aumentado en torno al 0.2%. Aunque en esta conclusión no hemos tenido en cuenta las pérdidas de la prueba realizada con una red Wifi y Jperf, ya que esas pérdidas se deben principalmente a un fallo en el enlace inalámbrico.

Gracias al “marcador” y al software Xmarker, también hemos podido comprobar el funcionamiento de la técnica de marcado de tráfico con DiffServ para dotar a nuestra red de cierto QoS. Con una pequeña prueba con la política “shape” (para delimitar el tráfico enviado a cierto caudal máximo) se ha concluido la correcta configuración de las clases y políticas de QoS, y nos abre amplias posibilidades para la obtención de la QoS deseada en cada flujo de tráfico enviado.

Finalmente, con las pruebas con las distintas redes heterogéneas, se ha podido observar cómo el retardo (delay) introducido por la red MPLS es muy bajo; tan solo de 22,3 μ s en media. Lo que concuerda con la teoría explicada en el capítulo 3, ya que MPLS enruta los paquetes más rápidamente que un router IP convencional. Además, este valor de delay se obtiene después de que un paquete atravesie tres routers MPLS, el “marcador” y dos Hubs. Si consideramos que para una red como Internet, un retardo inferior a 50 ms es aceptable, podríamos tener más de 2000 mil veces nuestra red

como red troncal para llegar a ese valor. Por supuesto, tenemos que tener en cuenta que los valores de las pruebas han sido obtenidos bajo un sistema controlado y limitado, y por lo tanto sin todas las interferencias que contiene Internet. Igualmente, los resultados nos indican que nuestra red sería funcional a gran escala.

Por lo que respecta a la pérdida de paquetes, también se ha mantenido en una tasa muy baja (inferior al 1%) tanto en Ethernet como con Wifi. Los valores del jitter son además inferiores a 1 ms para Ethernet y aun mejores para Wifi (inferiores a 0.5 ms). Este nos indica que la red MPLS juntamente con estas redes es muy adecuada para la transmisión de tráfico con altas demandas de QoS, como la videoconferencia o la transmisión de video en streaming.

Con todo ello, en nuestra opinión no cabe duda que MPLS representa el futuro de las redes de transporte. Su condición de multiprotocolo y su capacidad para ofrecer mecanismos de QoS convierten en MPLS en el mejor sistema para conseguir la mejor eficiencia de una red.

7.2. Líneas futuras de trabajo

Finalmente se plantean posibles líneas futuras de trabajo para complementar o continuar con el Proyecto.

- Repetir las pruebas con otras redes conectadas a la troncal MPLS, como Bluetooth o redes 3G.
- Poder realizar las pruebas a través de los túneles IPIP, conectando la red MPLS a otras remotas.
- Realizar más pruebas de QoS con más flujos simultáneos, para poder implementar nuevas políticas.
- Disponer de más routers para poder configurar más caminos, y redes más complejas.

8. Bibliografía

[ALW2002] Vivek Alwayn. Advanced MPLS Design and Implementation. Cisco Press. Copyright © 2002 Cisco Press.

[BLACK2002] Uyless Black. MPLS and label switching networks. Prentice Hall. Copyright © 2002 Uyless Black.

[FAUC2003] F. Le Faucheur, "Requirements for Support of Differentiated Services-Aware MPLS Traffic Engineering", RFC 3564, June 2003.

[HESS2010] Xavier Hesselbach, Joan Antoni García-Espín, Miquel González, Javier Gonzalo, Sergi Figuerola "E3MS: A traffic engineering prototype for autoprovisioning services in IP/DiffServ/MPLS networks", Jitel 2010.

[RFC 1349] P. Almquist. "Type of Service in the Internet Protocol Suite", July 1992

[RFC2205] RSVP sobre MPLS. RFC2205

[RFC 2474] K. Nichols; S. Blake. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", December 1998

[RFC 2475] S. Blake. "An Architecture for Differentiated Services", December 1998

[RFC3031] E.Rosen, A.Viswanathan and R.Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001

[RFC3032] ROSEN, E.; TAPPAN, D.; FEDORKOW, G.; REKHTER, Y.; FARINACCI, D.; LI, T.; CONTA, A. § RFC3032: MPLS Label Stack Encoding § Internet Engineering Task Force (IETF), January 2001
§ <http://www.ietf.org/rfc/rfc3032.txt?number=3032>

[RFC 3036] L. Andersson. "LDP Specification", January 2001

[RFC 3168] K. Ramakrishnan; S. Floyd. "The Addition of Explicit Congestion Notification (ECN) to IP"