UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Escuela Nacional de Estudios Profesionales Aragón

"IMPLEMENTACIÓN DE UNA RED PRIVADA VIRTUAL (VPN); BASADA EN PROTOCOLO MPLS, SOBRE UNA RED IP."

T E S I S

PARA OBTENER EL TITULO DE:
INGENIERO MECÁNICO ELÉCTRICO
ÁREA: ELÉCTRICA ELECTRÓNICA
P R E S E N T A:
SONIA OSORNO SAAVEDRA

ASESOR: ING. PABLO LUNA ESCORZA

MÉXICO 2003

A

PAGINACION

DISCONTINUA

Por si alguna vez soñamos,
y si haremos, pues estamos
en mundo tan singular,
que el vivir solo es soñar;
y la experiencia me enseña,
que el hombre que vive, sueña
lo que es hasta despertar.

Sueño el rico en su riqueza,
que más cuidados le ofrece;
Sueña el pobre que padece
su miseria y su pobreza;
sueña el que a medrar empieza,
sueña el que afana y pretende,
sueña el que agravia y ofende,
y en el mundo, en conclusión,
todos sueñan lo que son,
aunque ninguno lo entiende.

¿ Qué quizá soñando estoy,
aunque despierto me veo?
No sueño pues toco y creo
lo que he sido y lo que soy
y no estoy muy engañado:
pues si ha sido soñado,
lo que vi palpable y cierto,
lo que veo será incierto;
y no es mucho que rendido,
pues veo estando dormido,
que sueño estando despierto.

AGRADECIMIENTOS

A DIOS por todo lo bueno que me ha dado en la vida; por acompañarme y darme fuerza en los momentos difíciles.

A mi PAPÁ, que desde el cielo es la estrella que ilumina mi camino.

A mi MAMÁ, por darme la vida, por la confianza que siempre me has tenido y porque nunca me limitaste en las cosas que quise hacer.

A mis HERMANOS por el apoyo y ayuda que me han brindado durante toda mi vida. Gerardo, Angel, Mauricio y Luis

A Nohemi, por haber compartido conmigo estos años de esfuerzo, por los buenos momentos que pasamos juntas y por ser mi amiga.

A todos MIS AMIGOS por su amistad, su compañía y por todo lo que nos divertimos juntos.

A los Ingenieros; Adriana Porta, Arturo Rodríguez y Javier Ortiz, por sus conocimientos y por toda la ayuda que me prestaron para realizar esta tesis.

A mis SINODALES, por todo lo que aprendí de ustedes en mis años de estudio y por todo el apoyo que me han brindado para concluir este trabajo.

Ing. Raúl Barrón Vera
Ing. Pablo Luna Escorza
Ing. Carlos Ulises Mavridis Tovar
Ing. Alain Morones Camacho
Ing. José Luis García Espinosa

A la Universidad Nacional Autónoma de México y a la Escuela Nacional de Estudios Profesionales, ARAGÓN, por darme la oportunidad de convertirme en una profesionista.

A todos GRACIAS.

ÍNDICE

Páginas

CAPÍTULO II
CONJUNTO DE PROTOCOLOS TCP/IP

CAPÍTULO III
INTRODUCCIÓN A LA ARQUITECTURA MPLS

CAPÍTULO IV
REDES PRIVADAS VIRTUALES (VPN'S) BASADAS EN MPLS

INTRODUCCIÓN

El mundo de las Telecomunicaciones ha cambiado en las ultimas décadas, muchos negocios han dejado de lidiar con intereses locales o regionales para pensar en un mercado global. Muchas compañías tienen facilidades para expandirse a través del país, o incluso del mundo, pero sin duda una de las cosas que muchas de ellas desean es una forma para mantener las comunicaciones rápidas, seguras y confiables, en cualquier lugar en el que se encuentren localizadas sus oficinas.

Hasta hace poco, la confiabilidad de las comunicaciones se había relegado al uso de líneas privadas para mantener una Red de Área Amplia (WAN). Estas líneas dedicadas, van desde un servicio ISDN (144 Kbps) pasando por un, E1 (2.048 Mbps) o un E3 (34 Mbps), hasta un Optical Carrier-3 (OC-3 de 155Mbps) de fibra, proporcionándole a una compañía una forma para expandir su red privada mas allá de su área geográfica inmediata. Una red WAN tienen obvias ventajas sobre una red publica como el Internet, como son el desempeño, la confiabilidad y la seguridad, pero mantener una WAN particularmente cuando se usan líneas dedicadas, puede llegar a ser algo costoso, sobretodo si se toma en cuenta que el costo aumenta en la medida que la distancia entre oficinas aumenta.

Lo que ahora están creando muchas compañías son sus propias Redes Privadas Virtuales (VPN's) para acomodar las necesidades de empleados remotos y oficinas distantes. En donde básicamente una VPN es una red privada que usa una red publica para conectar los sitios de usuarios remotos. La cual en lugar de usar una conexión real, dedicada, como una línea privada, usa una conexión virtual. En donde la red pública comparte su estructura física con varias Redes Privadas Virtuales al mismo tiempo, las cuales tienen la ventaja de disfrutar de los mismos beneficios de una red privada.

Las VPN's entregan conectividad implementada sobre una infraestructura compartida; como puede ser el mismo Internet, o pueden construirse sobre un proveedor de servicio con infraestructura IP, ATM o Frame Relay. El tipo de implementación de una arquitectura VPN puede estar basada en el protocolo IPSec (IP Security) o en el protocolo MPLS (Multiprotocol Label Switching).

El presente trabajo trata a fondo la implementación de redes privadas Virtuales basadas en MPLS, construidas bajo la infraestructura IP de un proveedor de servicio. Este trabajo se encuentra dividido en cinco capítulos que abarcan los siguientes temas:

El capítulo I aborda temas generales relacionados con las redes de área local y de área amplia.
Capítulo II, abarca todo lo relacionado a la arquitectura TCP/IP necesaria para construir una infraestructura IP.
Capítulo III, en este capitulo se discute la operación de MPLS sobre interfaces en donde los paquetes se envían encapsulados en tramas de capa dos.
Capítulo IV, este capitulo introduce el concepto de VPN's y discute los beneficios de las redes privadas virtuales basadas en MPLS, como son el aislamiento, la seguridad, el enrutamiento simplificado y una mejor escalabilidad.
Capitulo V, en este capitulo se muestra la implementación de dos redes privadas virtuales bajo una infraestructura compartida, junto con un análisis completo de su funcionamiento.

CAPÍTULO I

ANTECEDENTES GENERALES

Este capítulo brinda la idea de lo que son las redes de computadoras. El proceso de conectar computadoras y mover datos de una a otra involucra el uso de un conjunto de reglas comunes que gobiernan como las computadoras deben de hablar una con otra, por lo tanto se hablará también de un modelo reconocido internacionalmente, el modelo OSI (Open System Interconnection), que define y estandariza este proceso.

1.1. Modelo OSI, Estandarización de las redes.

En los inicios de las redes, varias compañías, incluyendo IBM, Honeywell y DEC (Digital Equipment Corporation), tenían su propio estándar acerca de como las computadoras podían conectarse. Estos estándares describían los mecanismos necesarios para mover los datos de una computadora a otra. El problema de estos estándares es que no eran compatibles unos con otros. Lo que creó un conflicto cuando los usuarios trataron de comunicar computadoras de diferentes compañías mediante la creación de una red.

Una solución a este problema la brindó, más tarde, la ISO (organización de Estándares Internacionales) ya que desarrollo un modelo básico de referencia de interconexión de sistemas abiertos, mejor conocido como el modelo OSI, para definir las interfaces y protocolos de las redes en una estructura de capas. Este modelo fijó como metas; lograr la comunicación entre equipos construidos por diferentes manufacturas y hacer las aplicaciones independientes del hardware en donde operan.

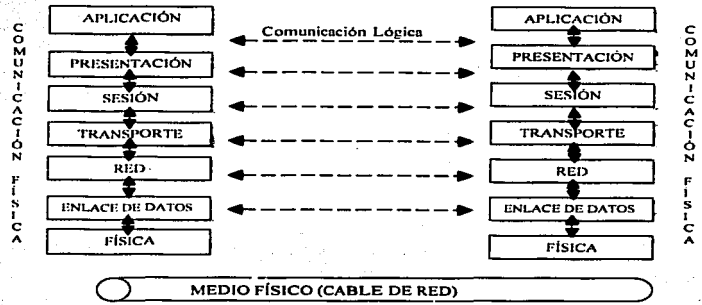


Figura 1.1. Modelo Básico de OSI

A continuación se hace una breve reseña de las 7 capas del modelo de referencia OSI, el cual se puede observar en la figura 1.1.

1.1.1 Capa Física - Capa 1

- Se encarga de las especificaciones mecánicas, eléctricas y procedimientos de funcionamiento de las interfaces de los equipos a conectar, tipos de conector, nivel de las señales y asignación de los pines en el conector.
- Maneja voltajes y pulsos eléctricos.
- Designa cables conectores y componentes.
- Define cual técnica de transmisión se usará para mandar datos sobre el cable de red.
- Define codificación de datos y sincronización de datos.
- Establece las velocidades de transmisión.
- Define cuanto dura cada bit y como es traducido en el apropiado impulso eléctrico o óptico para el cable de red.

1.1.2 Capa de enlace – Capa 2

- Coloca tramas de datos de la red dentro del medio físico.
- Controla el flujo de tramas de modo que los receptores lentos no se vean desbordados por los transmisores rápidos.
- Utiliza el CRC (Chequeo de Redundancia Cíclica) para corrección y verificación de información para asegurar que los datos fueron recibidos correctamente en una transmisión libre de error.
- Cuenta con un mecanismo de retransmisión de tramas para recuperar las tramas perdidas, duplicadas y erróneas.
- Es capaz de ofrecer servicios sin conexión y sin reconocimiento, sin conexión y con reconocimiento y servicio orientado a conexión.

Servicio sin conexión y sin reconocimiento

La máquina origen transmite tramas independientes a la máquina destino, sin que ésta proporcione un reconocimiento. No se establece conexión previa. Ver figura 1.2.

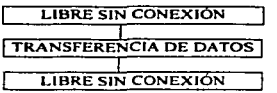


Figura 1.2. Sin reconocimiento, sin control de flujo, sin recuperación de errores

Servicio sin conexión y con reconocimiento.

La máquina origen transmite tramas independientes a la máquina destino, pero cada una de las tramas se reconoce en forma individual. Ver figura 1.3.

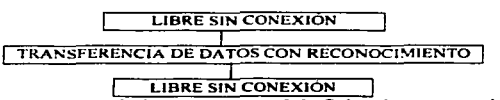


Figura 1.3. Con reconocimiento, sin control de flujo, sin recuperación de errores

Servicio orientado a conexión

Con este tipo de servicio, las máquinas origen y destino establecen una conexión antes de transmitir algún dato. Cada una de las tramas se numera y la capa de enlace garantiza que cada trama transmitida sea recibida. En este servicio se tienen tres fases. En la primera fase la conexión se establece, en la segunda las tramas se transmiten y en la tercera fase la conexión se libera. Ver figura 1.4.

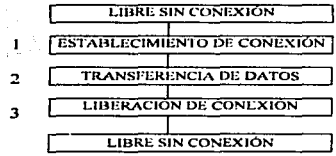


Figura 1.4. Reconocimiento, control de flujo, recuperación de errores

1.1.3 Capa de Red - Capa 3

- Determina el enrutamiento de la máquina fuente a la máquina destino, es decir, determina que camino deben de tomar los datos basados en las condiciones de red, prioridad de servicio y otros factores.
- Maneja problemas de trafico sobre la red, como es conmutar, enrutar y controlar la congestión de datos.
- Si el mensaje es demasiado grande lo puede desensamblar en pequeñas unidades y en la estación original reensamblar los datos al tamaño original.
- Realiza interconexión de red.
- Responsable de las direcciones de los mensajes así como de traducir direcciones y nombres lógicos en direcciones físicas.

1.1.4 Capa de transporte – Capa 4

- Se asegura que los mensajes sean entregados libres de error, en secuencia y sin pérdida o duplicación.
- Divide grandes mensajes en varios paquetes y junta pequeños mensajes en un solo paquete
- Proporciona dos tipos de servicio de transporte orientado a conexión y sin conexión.

El servicio orientado a conexión tiene una fase de establecimiento de conexión, de transferencia de datos y de liberación de conexión, como el de capa 2.

Establecimiento de Conexión

En la figura 1.5. se observan con flechas las unidades de datos de protocolo de transporte (TPDU) que envían y reciben las entidades de transporte, para establecer una conexión en un proceso normal.

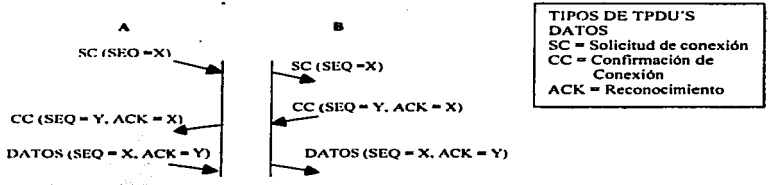


Figura 1.5. Solicitud de Conexión

A selecciona un número de secuencia, X por ejemplo y lo envía a B en una TPDU de solicitud de conexión (SC); B contesta con una TPDU de confirmación de conexión (CC), reconociendo a X y anunciando su propio número de secuencia inicial, Y. Por último A reconoce la elección que hizo B, del número de secuencia inicial, en su primera TPDU de datos.

Liberación de conexión

En la figura 1.6 se presenta el caso normal de liberación de conexión, en el que uno de los usuarios transmite una solicitud de desconexión (SD), para indicar la liberación de la conexión. En el momento que ésta llega, el receptor devuelve una confirmación de desconexión (CD) y arranca un temporizador, por si acaso se pierde la CD. En el momento que llega, el emisor original devuelve una TPDU ACK y elimina la conexión. Por último, cuando llega este ACK, el receptor elimina también la conexión.

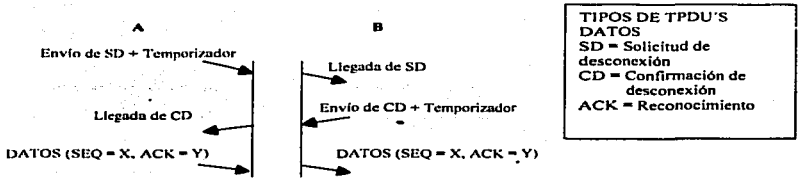


Figura 1.6. Desconexión

1.1.5 Capa de sesión – Capa 5

- Permite a diferentes computadoras establecer, usar y terminar una conexión llamada sesión.
- Desarrolla reconocimiento de nombres y funciones; como seguridad, que se necesitan para permitir a dos aplicaciones comunicarse sobre la red.
- Proporciona la sincronización entre las tareas de los usuarios.
- Implementa un control de dialogo entre procesos de comunicación, regulando cual lado transmite, por cuanto tiempo y como.

1.1.6 Capa de Presentación- capa 6

- Determina el formato usado para intercambiar datos entre computadoras.
- Es conocido como el traductor de red, ya que en el transmisor esta capa traduce datos de un formato que viene de la capa de aplicación en un formato intermedio, adecuado para la transmisión y en el receptor ésta capa traduce éste formato intermedio en el formato usual para la capa de aplicación de la computadora.
- También maneja la seguridad de la red, proporcionando servicios como encriptación conocido también como cifrado de datos. Que es un proceso para poner en clave la información.
- Proporciona reglas para la transferencia de datos para asegurar el entendimiento entre dispositivos diferentes de la red.
- Proporciona compresión de datos para reducir el número de bits que necesitan transmitirse. Quiere decir que asigna un código corto totalmente diferente a la información por transmitir. El transmisor y el receptor deben saber la equivalencia entre el código de información y el de transmisión.

1.1.7 Capa de Aplicación – Capa 7

- Sirve como ventana para que los procesos de aplicación tengan acceso a la red de servicios.
- Representa el servicio que soporta directamente las aplicaciones de usuario como son el software para transferencia de archivos, servicio de terminal virtual, acceso a bases de datos y para servicios de correo electrónico.

El modelo OSI regula el paso de los datos de la aplicación del usuario final hasta el cable de red. Como los datos son pasados desde la aplicación a través de las 7 capas del modelo OSI, cada una de las capas agrega sus datos con una información especifica de capa. Ésta información en la forma de encabezado, es leída después por la capa correspondiente.

Cuando los datos llegan al destino, cada capa desempaqueta el encabezado apropiado es decir, cada capa lee la información que fue agregada por su correspondiente capa en el emisor y responde a éste, y después manda la trama de datos a la siguiente capa alta.

En la figura 1.7. se muestra el modelo OSI, en donde cada capa agrega su propio encabezado.

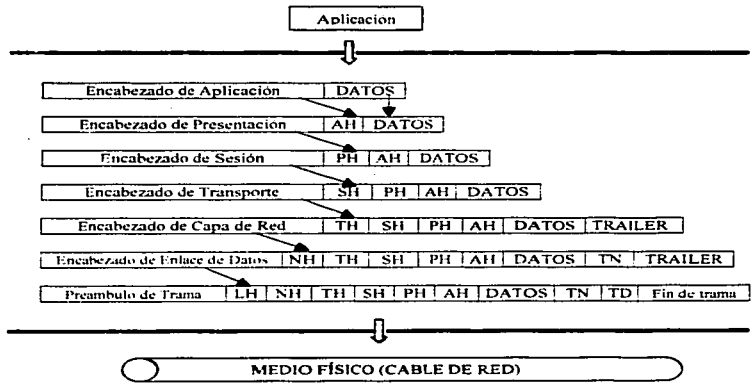


Figura 1.7. Envio de Datos a través de las 7 capas del modelo.

1.2 REDES LAN

Una red de área local, o LAN (Local Area Network), es llamada así porque las computadoras que forman la red están conectadas cerca la una de la otra compartiendo un área relativamente pequeña. Por ejemplo una red de computadoras en un mismo edificio.

1.2.1 Componentes de una LAN

El mínimo de componentes de hardware requerido para construir una LAN se muestran en la figura 1.8 e incluye:

- Al menos una computadora que funcione como servidor para compartir los recursos.
- Al menos una computadora, conocida como estación de trabajo o cliente, que tendrá acceso a los recursos compartidos.
- Un adaptador de red para cada computadora (tarjeta de red).
- Un HUB o un Switch.
- Cable para conectar las computadoras.

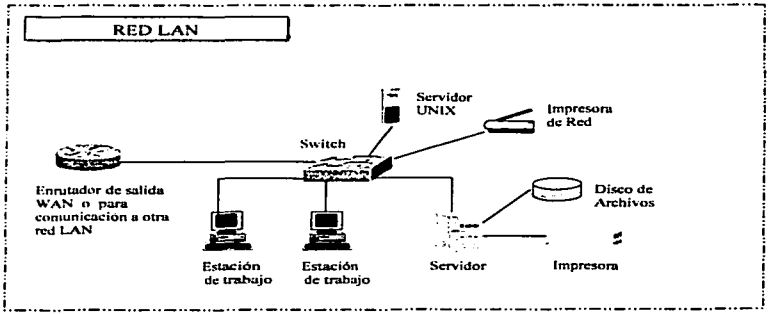


Figura 1.8. Red de Área Local

El mínimo de software requerido para una red LAN incluye:

- Un sistema operativo de red
- Un controlador de protocolo de red apropiado
- Programas de Aplicación de Red

1.2.1.1 Servidor y Estación de Trabajo

El servidor es una máquina de computo cuyos recursos son dedicados para servir a estaciones de trabajo también conocidos como clientes.
El beneficio principal de un servidor dedicado es que proporciona un control central de los recursos compartidos, esto proporciona una mejor seguridad y fácil mantenimiento de programas y datos. Un servidor también puede ser usado si el cliente tiene espacio en disco limitado, el servidor puede tomar el lugar del disco duro y en cada caso, cada usuario tendrá un archivo localizado sobre el directorio del servidor.
Cuando una estación de trabajo quiere hacer uso de estos recursos, tienen que conectarse al sistema de red que el servidor está corriendo.
Al manejar la seguridad de la red los servidores tienen dos medios para otorgar los permisos y permitir a las estaciones de trabajo acceder a los recursos compartidos y son:

Seguridad al compartir
Se le asigna un password a cada recurso compartido. Y el usuario al tratar de entrar al recurso debe teclear el password.

Seguridad al nivel de usuario.
Asigna ciertas reglas básicas usuario por usuario. El servidor checa el nombre de usuario y el password como una combinación única en cada usuario y lo valida comparando esta combinación contra su base de datos.

1.2.1.2 Tarjeta de red

Las tarjetas de red son primordiales para la conexión de PC´s como estaciones de trabajo, así como para la conexión de impresoras en red.

Una tarjeta de red se instala en una PC en una de las ranuras de expansión de la tarjeta madre, una vez que se encuentra instalada se procede a la configuración por software, las tarjetas generalmente tienen tres opciones que deben ser configuradas propiamente para que funcione bien en la computadora y éstos son: el nivel de interrupción (IRQ), el dirección de puerto base I/O y la dirección de memoria base.

1.2.1.3 HUB – Switch

El HUB y el switch son equipos que sirven como nodo central para las redes de cable de par torcido (UTP). Estos equipos tienen un conjunto de puertos jack RJ45 en donde se conectan las estaciones de trabajo, el servidor de archivos y cualquier otro equipo que cumpla con el estándar IEEE 802.3. La mayoría de los HUB y algunos de los switches tienen un puerto AUI (DB15) que es una interfaz adicional para poder incorporar el equipo a una red con medio físico diferente al UTP con la ayuda de un transceiver. La principal diferencia entre el HUB y el switch es que éste ultimo aísla cada uno de los puertos es decir cada uno de los puertos es independiente de los otros y además tiene la opción de asignar varios puertos para formar parte de un grupo, el cual es conocido como VLAN.

1.2.1.4 CABLEADO

La industria LAN ha estandarizado tres medios físicos que pueden ser usados en la capa física de la red: Par torcido, coaxial y fibra óptica. En la tabla 1.1 se hace una comparación de las características de los cables

Cable de Par Torcido UTP (Unshielded Twisted Pair)

Un par de alambres de cobre aislados, torcidos uno con respecto de otro forman un cable de par torcido. El cable se encuentra protegido por una cubierta exterior aislante. El conector más usado para la conexión de UTP es el RJ45 de 8 pines que es el conector que se implementa en la red.

Cable Coaxial

El cable coaxial se forma por un alambre conductor básico y una cubierta formada por una malla de alambre que actúa como tierra. El alambre conductor y la tierra se encuentran separados por un aislante plástico y, finalmente todo el conjunto está protegido por una cubierta exterior aislante. Los cables coaxiales pueden ser de varios anchos los más conocidos son RG8 y RG11que son cable grueso Ethernet de 50 ohms, el RG-58 que es cable delgado Ethernet de 50 ohms.

Fibra Óptica

Es usado para portar la señal de datos digitales en forma de pulsos modulados de luz. Está formado de un cilindro extremadamente delgado de vidrio, que se le da el nombre de núcleo, cubierto por una capa de vidrio, conocida como el revestimiento. A su vez, se encuentra cubierta por una placa aislante y protectora en la parte exterior para darle más integridad al cable. Hay dos fibras por cable una para la transmisión y una para la recepción.

Características	Coaxial delgado	Coaxial grueso	UTP	Fibra Óptica
Longitud de uso	185 metros	500 metros	100 metros	2 kilometros
Rango de transmisión	10Mbps	10Mbps	10 y 100Mbps	100Mbps
flexibilidad	Algo flexible	Poco flexible	Es el más flexible	No es flexible
Facilidad de instalación	Fácil	Fácil	Muy Fácil	Difícil
Susceptibilidad a interferencia	Buena resistencia a interferencia	Buena resistencia a interferencia	Susceptible a interferencia	No susceptible
Características especiales	Componentes electrónicos más baratos que el UTP	Componentes electrónicos más baratos que el UTP	Es como el alambre telefónico: frecuentemente preinstalado en edificios	Soporta voz, datos y video

Tabla 1.1 Características de los cables

1.2.1.5 Sistema operativo de red.

El sistema operativo de red es el alma de la red. El hardware del sistema proporciona las trayectorias de datos y las plataformas en la red, pero el sistema operativo es el encargado de controlar la red; la funcionalidad, la facilidad de uso, el rendimiento, la administración, la seguridad de los datos y la seguridad de acceso, dependen del sistema operativo.
Actualmente los sistemas operativos más usados son: Microsoft Windows NT, Microsoft Windows XP, Microsoft Windows 2000 Server, Appleshare de APPLE y Netware de Novell - UNIX en máquinas HP, SUN, IBM y hasta en PC's

1.2.1.6 Controlador de protocolo o Protocolo de red

Un protocolo es un conjunto de reglas que gobiernan la comunicación entre 2 estaciones. Al igual que dos personas necesitan entender el mismo lenguaje con el objeto de hablar una con la otra, las estaciones necesitan correr el mismo controlador de protocolo para comunicarse. Un protocolo de red, o un controlador de protocolo, es generalmente el responsable del empaquetamiento y enrutamiento de los datos y mensajes de aplicación sobre las capas de red y de transporte.

1.2.1.7 Programas de Aplicación de Red

Cualquier aplicación que tiene la habilidad de aceptar y enviar datos a través de la red es considerada de red, como puede ser programas de mail, programas de manejo de archivos y programas de base de datos.

1.2.2 Estándares de Redes de Área Local

Al haber una proliferación de productos LAN y con ellos una necesidad de consistencia, el instituto de Ingenieros Eléctricos y Electrónicos mejor conocido como IEEE, tomo la tarea de definir los estándares de redes de área local LAN. Este proyecto fue llamado 802 por el año y mes en que comenzó, febrero de 1980.

Las especificaciones del proyecto IEEE 802 relacionan principalmente las capas uno y dos del modelo OSI, aunque estos dos modelos son compatibles una característica distintiva del IEEE 802 es que divide la capa dos en dos subcapas. Una parte define el control de acceso al medio, llamado **subcapa MAC** y la otra el control de enlace lógico, llamada **subcapa LLC.**

Desde 1985 el IEEE ha seguido investigando para agregar más estándares de redes. Ver la tabla 1.2.

IEEE802.1	Administración de sistemas e interconexión
IEEE802.2	Control de enlace lógico
IEEE802.3	Red usando acceso CSMA/CD para ethernet
IEEE802.4	Red de bus, Arnet, usando acceso Token Passing
IEEE802.5	Red de anillo, Token Ring, usando acceso Token Passing
IEEE802.6	Redes de área metropolitana
IEEE802.7	Tecnología de banda ancha
IEEE802.8	Tecnología de fibra óptica
IEEE802.9	Voz integrada y datos
IEEE802.10	Seguridad de LAN
IEEE802.11	Redes inalámbricas
IEEE802.12	Fast ethernet
IEEE802.14	Ethernet 100Base- VG

Tabla 1.2. Estándares de redes LAN

1.2.2.1 Estándar IEEE802.3 y ETHERNET

Ethernet fue inventado en 1970. La versión 1.0 fue liberada por Digital,Intel y Xerox en 1980. La versión 2.0 de Ethernet apareció en 1982 y la especificación IEEE 802.3 surgió en 1985.

Estos estándares cubren los protocolos de capa física y de la subcapa MAC. Además:

1. Definen la estructura de la trama.
2. Definen el control de acceso al medio por CSMA/CD.
3. Definen las características del medio físico.

Para un mejor entendimiento de estos tres puntos los iré explicando uno a uno.

IEEE802.3 está basado en Ethernet, pero éste define opciones de múltiples capas físicas. Hoy en día el término Ethernet es frecuentemente usado para aplicarse a todas las redes LAN que cubren el estándar 802.3 que define el protocolo conocido como CSMA/CD. Además Ethernet ha sobrevivido como la mejor tecnología LAN y es actualmente usada por aproximadamente el 85 por ciento de las redes LAN del mundo.

En redes cumpliendo el estándar IEEE, los enlaces son generalmente proporcionados por la especificación IEEE802.2, también conocida como subcapa LLC.

1. Estructura de las tramas

Campos de la trama Ethernet

Preámbulo	Dirección destino	Dirección fuente	Tipo	datos	Chequeo de redundancia cíclica

Campos de la trama IEEE802.3

Preámbulo	Delimitador De inicio de trama	Dirección destino	Dirección fuente	Longitud	Encabezado 802.2 y Datos	Chequeo de redundancia cíclica

La siguiente es una breve explicación de los campos de cada una de las tramas mostradas.

❖ **Preámbulo y delimitador de inicio de trama**

Para Ethernet el preámbulo es de 8 bytes y para 802.3 es de 7 bytes, está formado por unos y ceros alternativos. El siguiente byte en 802.3 es el delimitador de inicio de trama que es como el preámbulo, excepto que finaliza con dos unos consecutivos. Estos bits indican el inicio que viene una trama y sincronizan a todos los receptores en la LAN.

❖ **Dirección destino. (DA)**

Esta dirección es de seis bytes para Ethernet y de 2 a 6 bytes para IEEE802.3 y es la dirección donde deben ser entregados los datos.

❖ **Dirección fuente. (SA)**

Es de la misma longitud que la dirección destino para cada estándar y es la dirección de quien envía la trama.

Estas direcciones (SA y DA) se encuentran en la ROM de las tarjetas de red en los puertos de los equipos de comunicación, la IEEE es responsable de asignar los tres primeros bytes para cada vendedor y cada vendedor asigna tres bytes adicionales para formar la dirección de seis bytes, conocida también como MAC Address o dirección física. Ver tabla 1.3.

IEEE	Fabricante	
00000C	00000C	CISCO
00001B	00001B	NOVELL
08005A	08005A	IBM

Tabla 1.3. Ejemplos para diferentes vendedores

❖ **Tipo**

La trama Ethernet utilizan este campo de 2 bytes, en lugar del campo de longitud, para indicar el protocolo de capa superior contenido en el campo de datos de la trama.

❖ **Longitud**

El campo de longitud de dos bytes que utiliza 802.3, indica cuantos bytes de datos están en la trama.

❖ **Campo de datos**

Es una secuencia de n bytes de cualquier valor, donde n es menor o igual a 1500. Si la longitud de datos es menor a 46, el campo de datos debe ser rellenado hasta que al menos sea de 46 bytes.

❖ **Encabezado 802.2 subcapa LLC**

La subcapa LLC es responsable de proporcionar una trayectoria de transmisión libre de error a la capa de red. Y estas funciones son proporcionadas por las especificaciones del protocolo IEEE 802.2.

Este encabezado consta de hasta 4 campos, ver figura 1.9. Como todos los protocolos de la capa de enlace, LLC ofrece varios servicios a la capa de red. Estos servicios son obtenidos en lugares llamados puntos de acceso al servicio, SAP. Cada SAP tiene una dirección, para LLC únicamente identifican un proceso de la capa de red y para éstos procesos los SAP son lugares para dejar mensajes acerca del servicio deseado.

Figura 1.9. Subcapa LLC encapsulada en la trama 802.3

Campos del Encabezado LLC

a) **DSAP, Punto de Acceso al Servicio Destino.** Es el primero del encabezado IEEE 802.2 y tienen una longitud de un byte.

b) **SSAP, Punto de Acceso al Servicio Fuente.** Tiene una longitud de un byte. Los DSAP y SSAP son asignados por la oficina de estándares del IEEE, donde el bit menos significativo tiene una función especial. El Bit menos significativo del DSAP indica si el destino es una dirección individual o un grupo. El bit más significativo del SSAP indica si la unidad de datos LLC contiene una solicitud o una respuesta. LLC usa estos bits para indicar como procesar ciertos bits en el campo de control.

c) **Campo de control**. Puede contener uno o dos bytes, dependiendo del tipo de servicio suministrado o solicitado.

Tipos de servicio

Todas las redes 802 pueden proporcionar servicio sin conexión, que no proporciona reconocimientos, control de flujo y recuperación de datos utilizando para estos servicios el apoyo de un protocolo de alguna capa más alta del modelo OSI, por ejemplo de la capa de transporte. Opcionalmente 802 puede proporcionar servicio orientado a conexión brindando control de flujo, reconocimiento y recuperación de tramas.

Comando y Respuestas del Campo de Control

Los comando y respuestas establecidos en el campo de control LLC dependen de si es orientado a conexión o servicio sin conexión.

Redes LAN, Servicio sin conexión

Comandos	Respuestas
UI	XID
XID	TEST
TEST	

UI (unnumber information), información no numerada
XID (Exchange Station Identification), identificación de estación para el intercambio de información.
TEST. Trama de prueba para la trayectoria de información.

Redes LAN, Servicio orientado a conexión

Comandos	Respuestas
I	I
RR	RR
RNR	RNR
REJ	REJ
SABME	UA, FRMR
DISC	UA, DM

I, Trama de información.
RR (Receiver Ready), Receptor listo para recibir tramas de información
RNR (Receiver Not Ready), Receptor no listo, indicando una condición de ocupado.
REJ (Reject), Trama de rechazo de transmisión o solicitud de retransmisión.
SABME (Set Asyncronous Balance Mode Extended), Trama de establecimiento de modo balanceado asíncrono balanceado.
DISC (Disconnect), Trama de Desconexión.
UA (Unnumbered Acknowlegment), Trama de reconocimiento no numerada.
FRMR (Frame Reject), Trama de indicación de rechazo de trama.
DM (Disconnect Mode), Trama de modo desconectado.

❖ **Chequeo de Redundancia Cíclica, CRC.**

Este campo contiene un código de chequeo de redundancia cíclica de 32 bits, calculados por el emisor antes de la transmisión y confirma en la recepción.
El CRC se calcula sobre los campos de dirección fuente, dirección destino, longitud o tipo, datos y campos de relleno de la trama. El receptor calcula el CRC con la información recibida y compara éste con el que dato viene en el campo CRC de la trama que recibió. El receptor descarta cualquier trama con un CRC que no cumpla con ser igual al CRC calculado por el emisor.

2. Método de acceso CSMA/CD (Carrier-Sense Multiple Access with Collision Detection)

El protocolo CSMA/CD fue desarrollado originalmente para permitirle a dos o más estaciones compartir un medio común en una conexión sin switch donde cada MAC determina por si misma cuando se permite mandar una trama. Este método se utiliza en la transmisión Half Duplex.

Las reglas que sigue el protocolo son resumidas por las propias iniciales del protocolo, como se explica a continuación.

- **Carrier Sense** (Monitoreo de portadora). Cada estación hará un monitoreo constante del medio de transmisión para detectar presencia de portadora, un voltaje especifico, con lo que se dará cuenta si otra estación está transmitiendo y utilizando el medio, si no detecta portadora transmitirá en ese momento. Ver figura 1.10.

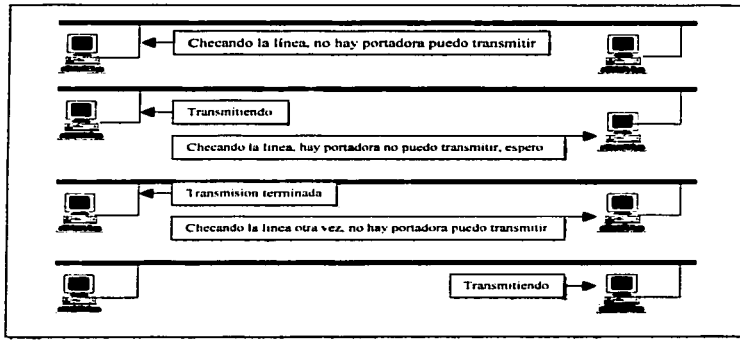


Figura 1.10. Monitoreo de Portadora.

- Múltiple Access (Acceso Múltiple). Las estaciones empiezan a transmitir en cualquier momento que detecten que el medio está libre. Al usar este método es posible que dos estaciones detecten al mismo tiempo el medio libre para transmitir, y al comenzar a transmitir las dos se ocasionaría lo que se conoce como colisión.

Figura 1.11. Acceso Múltiple

- Collision Detect (Detección de colisión). Al ocurrir una colisión cada estación debe ser capaz de darse cuenta y parar la trasmisión, después debe esperar un periodo de tiempo apropiado para comenzar una retransmisión.

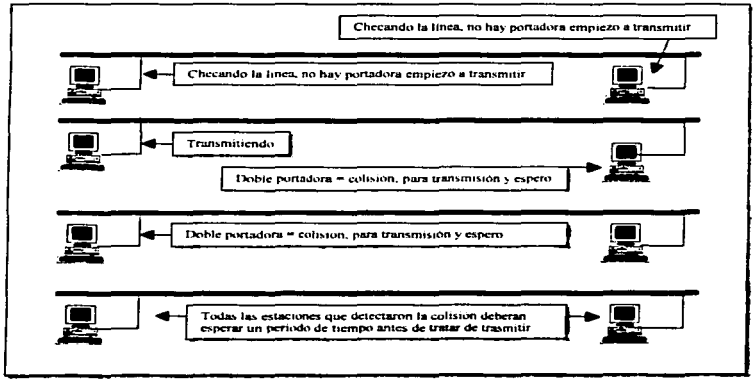


Figura 1.12. Monitoreo de Portadora.

Transmisión Full-Duplex

La operación full-duplex es una capacidad opcional de la subcapa MAC, que permite transmisión simultánea por dos caminos en enlaces punto a punto (point to point). La transmisión full-duplex es mucho más fácil que la transmisión half-duplex porque ésta no involucra colisiones y no necesita horarios para transmisión. La sustitución del hub por el switch hizo posible la aparición de la transmisión full-duplex ya que el switch aisló e hizo independientes cada uno de sus puertos, logrando que el medio ya no fuera compartido entre ellos y no tuvieran que esperar su turno para transmitir. El resultado no solo fue más tiempo disponible para transmisión, sino también un doble ancho de banda del enlace porque ahora cada enlace puede soportar una tasa de transmisión full duplex, es decir, dos caminos de transmisión.

Esta operación tiene una implementación de control de flujo también opcional que le permite a un receptor que empieza a congestionarse, enviar al transmisor una solicitud para que pare la transmisión por un periodo de tiempo seleccionado.
Ambas opciones, control de flujo y operación full-duplex, son habilitadas enlace por enlace, asumiendo que la capa física asociada es capaz de soportar la operación full-duplex.

Opción de etiquetado VLAN.

El etiquetado VLAN es otra opción de la MAC que proporciona tres capacidades importantes, que en sus inicios no estaban disponibles para usuarios de Ethernet.

- Proporciona un medio para acelerar el tráfico de red de tiempo crítico, proporcionando prioridades de transmisión para las tramas que van de salida.

- Permite que las estaciones sean asignadas a un grupo lógico para comunicarse a través de múltiples LAN como si fueran una sola LAN. Manda las tramas de VLAN solo a los puertos que pertenecen a la VLAN a la que es enviada la información.

- Simplifica el mantenimiento de la red y hace los movimientos y cambios fáciles de administrar.

Una trama etiquetada como VLAN es simplemente una trama básica de datos MAC al que se le ha insertado un encabezado entre el campo de dirección fuente (SA) y el campo de Tipo/Longitud.

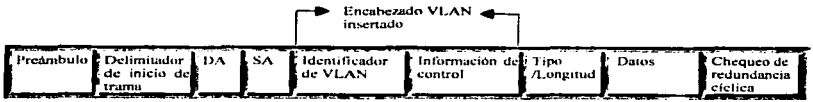


Figura 1.13. Diagrama Trama Ethernet con encabezado VLAN

El encabezado VLAN consiste de los dos campos siguientes, ver figura 1.13.

- **Identificador de VLAN**. Un valor reservado de dos bytes. Indicando que la trama es una trama VLAN.

- **Campo de Control de Etiqueta**. Campo de dos bytes que contiene la prioridad de transmisión (0 a 7, 7 es el más alto) y el valor que identifica la VLAN a la cual las tramas deben ser enviadas.

El etiquetado VLAN requiere que todos los nodos involucrados con un grupo de VLAN estén equipados con la opción de VLAN. Los equipos que manejan las VLAN son los switches y los enrutadores.

3. Topologías y Estructuras de Red Ethernet

Las redes LAN consisten de nodos de red y medios de conexión. Los nodos de red entran dentro de los siguientes dos tipos.

- Equipo terminal de Datos. DTE (Data Terminal Equipment). Equipo que puede ser la fuente o el destino de las tramas de datos. Los DTEs son equipos como PCs, estaciones de trabajo, servidores o una impresora.

- Equipo de comunicación de datos, DCE (Data Communication Equipment). Equipo intermedio de red que recibe y manda tramas a través de la red. Los DCE pueden ser hubs, switches y enrutadores, incluso los módems.

Los actuales medios de conexión incluyen el UTP (Unshielded Twisted Pair) cable de par trenzado no blindado y los diferentes tipos de cable de fibra óptica.

Las LAN toman varios tipos de topologías, pero a pesar de su tamaño, todas son una combinación de tres estructuras básicas de conexión que a continuación se mencionan.

- Conexión Point to point (Punto a punto). Es la más simple de todas ya que solo dos equipos de red están conectados y la conexión puede ser DTE a DTE, DTE a DCE o DCE a DCE. La longitud máxima depende del tipo de cable y el método de transmisión que es usado.

- Conexión tipo bus. La red Ethernet original fue una implementación de una estructura en bus con cable coaxial, donde un solo cable conectaba las estaciones de trabajo sin brincos y sin múltiples trayectorias. Hoy en día ya es difícil de encontrar implementadas este tipo de topologías.

- Conexión de estrella. Es la más usada desde 1990, el nodo central es un switch o un repetidor multipuerto conocido también como hub. Todas las conexiones en una red estrella son enlaces point to point de par trenzado o fibra óptica.

1.2.2.2 Especificaciones físicas de las Topologías LAN

IEEE802.3 especifica varias capas físicas y cada uno tiene un nombre que resume sus características. Para facilitar la identificación se utilizo en los inicios de Ethernet una abreviación formada por los tres parámetros básicos de la red.

Estos parámetros son los siguientes:

1. Su velocidad de transmisión en Mbps
2. La técnica de codificación, Banda Base y Banda Ancha.
3. El tamaño del segmento en unidades de cientos de metros

Por ejemplo

Velocidad	Banda Base	Longitud del segmento(m)	Tipo de cable
10	base	5	coaxial grueso
10	base	2	coaxial delgado
10	base	T	UTP de 10 Mbps

A través de los años y con el desarrollo de diferentes capas físicas en redes Ethernet, el tercer parámetro de esta combinación fue sustituido quedando como sigue:

1. Taza de transmisión en Mbps
2. Método de transmisión, sobreviviendo solo el bandabase.
3. Tipo de medio de transmisión

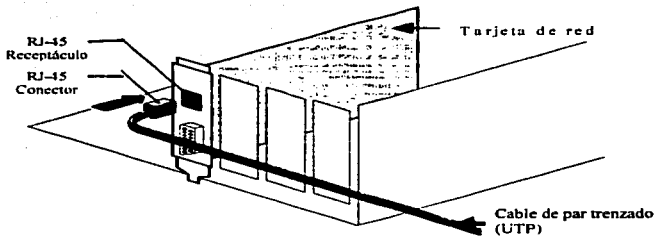
Por ejemplo.

Taza de transmisión	Método de transmisión	Tipo de medio de transmisión	Descripción
10	base	T	10 Mbps, bandabase, sobre UTP
100	base	T	100 Mbps, bandabase, sobre UTP
1000	base	T	1000Mbps, bandabase, sobre UTP
1000	base	LX	1000Mbps, bandabase, sobre fibra óptica

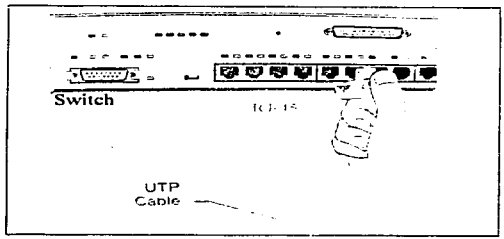
En la siguiente tabla se hace una comparación de cada especificación física en cuanto a límites de operación.

Parámetros	**10 Mbps** (Ethernet)	**100 Mbps** (Fast Ethernet)	**1000 Mbps** (Gigabit Ethernet)
Tamaño de trama mínimo	64 bytes	64 bytes	520 bytes (Agregando el campo de extensión)
Diámetro máximo, DTE a DTE	100 metros UTP	100 metros UTP 412 metros fibra	100 metros UTP 316 metros fibra
Diámetro máximo con repetidores	2500 metros	205 metros	200 metros
Número máximo de repetidores	5	2	1

La figura 1.14 hace un acercamiento a la conexión física Ethernet, desde la tarjeta de red de una computadora hacia un nodo de red (un Switch), mediante cable UTP usando un conector RJ-45. La figura a) muestra la conexión en un extremo del cable UTP, la b) muestra la conexión al otro lado del cable y la figura c) hace un acercamiento al conector RJ-45.



a). Conexión a una PC, utilizando conector RJ-45 y cable UTP



b). Conexión a un Switch, utilizando conector RJ-45 y cable UTP

c). Conector RJ-45

Figura 1.14. Conexión física para redes 10baseT, 100baseT y 1000baseT.

1.3. Redes WAN

Una red WAN (Wide Area Network), red de área amplia, es una red de comunicaciones que como su nombre lo indica cubre un área geográfica grande y que a menudo usa facilidades de transmisión proporcionadas por un proveedor del servicio, como una compañía telefónica. Las WAN unen LANs por medio de enlaces de alta velocidad, como un cable dedicado o una línea telefónica, a la entrada de cada LAN uno o más equipos actúan como el enlace entre la LAN y la WAN; éstos se llaman gateway (equipos de puerta de enlace) y hay de varios tipos en donde el más común es el enrutador. Ver figura 1.15.

Las tecnologías WAN generalmente funcionan en las tres capas más bajas del modelo OSI, la capa física, la capa de enlace de datos y la capa de red.

1.3.1 Protocolos Asociados a las Redes de Área Amplia

Los protocolos WAN que se asocian a cada una de las capas se resumen a continuación:

En la capa física

- **EIA/TIA-232:** Esta norma fue definida como una interfaz estándar para conectar un DTE a un DCE. La interfaz RS-232 especifica un cable de 25 alambres con un conector compatible, DB-25 (25 pines).

- **EIA/TIA-449:** Junto a la RS-422 y RS-423, que son las que especifican las normas eléctricas, forman la norma para transmisión en serie que extienden las distancias y velocidades de transmisión más allá de la norma RS-232. Especifica un cable con 37 alambres, manejando una velocidad de hasta 10 Mbps en una distancia máxima de 15 metros y una velocidad de 90 Kbps para una distancia máxima de 1200m.

- **V.35:** Según su definición original, serviría para conectar un DTE a un DCE síncrono de banda ancha (analógico) que operara en el intervalo de 48 a 168 kbps.

- **X.21:** Estándar CCITT para redes de conmutación de circuitos. Conecta un DTE al DCE.La recomendación X.21 resume una conexión digital directa a una red de telefonía digital, la velocidad de transmisión para X.21 es de 64Kbps.

- **G.703:** Recomendaciones del ITU-T, antiguamente CCITT, relativas a los aspectos generales de una interfaz utilizando Multiplexaje por División de Tiempo.

- **EIA-530:** Presenta el mismo conjunto de señales que la EIA/TIA-232.

- **HSSI** (High-Speed Serial Interface): Estándar de red para las conexiones seriales de alta velocidad (hasta 52Mbps) sobre conexiones WAN.

En la capa de enlace de datos

- **HDLC** (High-Level Data Link Control). Es uno de los protocolos de capa de enlace de datos que se utilizan con más frecuencia, publicado por ISO en los estándares 3309, 4335, 6154 y 6256. HDLC soporta transmisión Half Dúplex y Full Dúplex. Utiliza los comandos y respuestas que se mencionaron en el modelo OSI para la capa de enlace.

- **PPP** (Point to Point Protocol). El protocolo Punto a Punto es un método estándar para encapsular en enlaces seriales. Se utiliza para accesar a la red telefónica y está especificado en la RFC 1332 y RFC 1661. Este protocolo puede entre otras cosas, checar la calidad del enlace durante el establecimiento de la conexión. Además brinda apoyo de autenticación a través de los protocolos PAP (Password Authentication Protocol) y CHAP (Challenge Handshake Autnentication Protocol).

- **Frame relay** (LAP-F). Protocolo característicos de las redes Frame Relay la cual es una de las redes de conmutación de paquetes más común. Esta red fue diseñada para ser más simple, operar a altas velocidades y con enlaces más confiables que la red X.25. Define conexiones entre Cliente DTE y proveedores DCE, en el nivel de la capa de enlace de datos; el acceso FR, puede ser a 56 Kbps, 64Kbps o 1.544Mbps.

- **LAPB** (Link Access Procedure Balanced), derivado de HDLC. Es un protocolo especificado por la arquitectura X.25 para la capa de enlace de Datos. Proporciona comunicación full duplex entre DTE y DCE.

- **MPLS** (Multi-Protocol Label Switching, Conmutación de Etiquetas Multiprotocolo). Es un estándar del IETF. La arquitectura MPLS fue creada para combinar los beneficios de envío de paquetes basados en conmutación de capa 2 con los beneficios de enrutamiento de capa 3, es decir, integra en uno solo los niveles de enlace de datos y de red. MPLS asigna etiquetas a los paquetes para transportarlos a través de la red.

En la capa de red

- **Protocolo X.25**. Este protocolo da origen a la red X.25, la cual también es una red de conmutación de paquetes. Fue desarrollada cuando los enlaces WAN manejaban bajas velocidades y poca confiabilidad.

- **Protocolo IP**. Es el protocolo principal en las redes IP, define la unidad básica de transferencia de datos usada para las redes TCP/IP. Es responsable de mover los paquetes de datos ensamblados, a través de las redes. Usa un conjunto de direcciones únicas para cada dispositivo en la red, a fin de determinar los destinos de los paquetes. Los protocolos TCP/IP se verán con más detalle en el capítulo siguiente.

- **ATM**. El Modo de Transferencia Asíncrona es un estándar de la ITU-T. Es capaz de transferir voz, video y datos a través de redes privadas y redes públicas. Los datos son transportados en unidades conocidas como celdas, cada celda consiste de 5 octetos para la información del encabezado y 48 octetos para la información del usuario (payload). La arquitectura ATM es un sistema de conmutación de paquetes que soporta velocidades de hasta 1,2 Gbps.

- **Protocolo MPLS**. Protocolo que integra la capa 2 y la capa 3.
  Nota: MPLS no puede ni debe ser considerado exclusivamente de capa 2 o de capa 3, sino como un protocolo especial que integra en uno solo los dos niveles.

El ambiente WAN también tienen especificados diferentes tipos de equipos (Hardware). Los siguientes son algunos de estos:

- Switches X.25 y switches Frame Relay
- Access Server (servidor de acceso)
- Módems
- Adaptadores para terminales de ISDN
- Enrutadores
- Multiplexores
- Switches ATM

Switch WAN

Es un mecanismo multipuerto usado en las redes de los proveedores de red. Este equipo conmuta tráfico como el de Frame Relay, X.25 y opera en la capa de enlace de datos.

Access Server

Permite el acceso al servicio de red a través de una línea telefónica, se le pueden configurar números telefónicos. El servicio que proporciona es conocido como RAS (Remote Access Server), servicio de acceso remoto.

Módem

Un módem es un mecanismo que interpreta señales digitales y analógicas, habilitando los datos para ser transmitidos sobre líneas telefónicas. En la fuente la señal digital es convertida en una forma analógica. En el destino, estas señales analógicas son regresadas a su forma digital. Utilizado en los sitios que solo cuentan con líneas telefónicas como medio de trasmisión.

Adaptador de terminal ISDN

Es un mecanismo usado para conectar interfaces BRI de ISDN, con otras interfaces como la RS-232 de un enrutador.

Enrutador.

Los enrutadores interconectan redes en la capa de red del modelo OSI, la cual normalmente incluye lo que se conoce como dirección lógica y siempre será asignada por el administrador de red. Esta dirección lógica se encontrara en el contenido de los paquetes de datos.

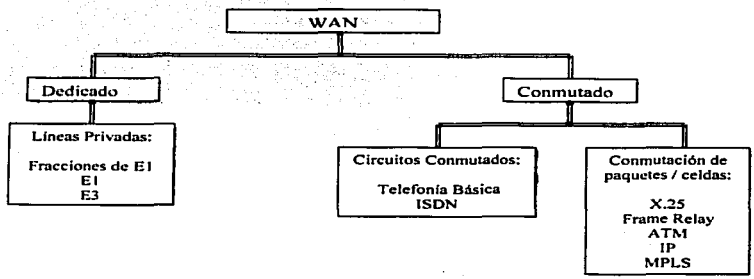
Los enrutadores pueden enviar paquetes sobre diferentes trayectorias en la red, dependiendo de las prioridades del usuario.

Los enrutadores son multiprotocolo es decir, brindan servicio de enrutamiento para diferentes protocolos.

Utilizan protocolos de enrutamiento para calcular la mejor trayectoria a través de la red. Algunos de estos protocolos son: RIP, IGRP, EIGRP y OSPF.

1.3.2 Opciones de Tecnologías WAN

En general tenemos dos tipos de opciones disponibles para las redes de área amplia: líneas dedicadas (Point to Point) o conexiones conmutadas. Las conexiones conmutadas pueden ser mediante conmutación de circuitos o mediante conmutación de paquetes. El siguiente diagrama ilustra esta distribución.



Enlace Punto a Punto (Point to Point).

Un enlace punto a punto proporciona un único y preestablecido camino de comunicación WAN desde el lugar del cliente pasando a través del proveedor de red, como una compañía telefónica, hasta la red remota. Para una línea punto a punto el proveedor asigna pares de alambre y hardware que no son compartidos con nadie más. El precio de este enlace está basado generalmente en el ancho de banda requerido y la distancia entre los dos puntos conectados.

Conmutación de circuitos

Los circuitos conmutados permiten la comunicación de datos, que pueden ser iniciados cuando se necesita y terminados cuando se completa la comunicación. Trabaja como una línea telefónica normal.

Conmutación de paquetes

Esta es una tecnología WAN en la cual los usuarios comparten los recursos de un proveedor. Y como el proveedor hace un uso más eficiente de la infraestructura, el costo para el cliente es mejor en comparación con el uso de enlaces punto apunto. El proveedor crea circuitos virtuales entre los sitios de los clientes enviando paquetes de datos de un lado a otro a través de la red. Esta red del proveedor a menudo es mostrada como una nube.

En donde las conexiones entre los sitios del cliente también son conocidas como circuitos virtuales.

Circuitos virtuales WAN.

Un circuito virtual es una conexión lógica creada dentro de una red compartida entre dos equipos de red. Existen dos tipos de circuitos virtuales: circuitos virtuales conmutados (SVC) y circuitos virtuales permanentes (PVC)

Los SVCs son circuitos virtuales que se establecen basados en una solicitud y terminados cuando la transmisión se completa. Es decir, la comunicación sobre SVC consiste de tres partes: establecimiento del circuito, transferencia de datos y terminación del circuito. Los SVC's son usados cuando la transmisión entre equipos es esporádica.

El PVC es un circuito virtual establecido permanentemente y solo consiste de una parte: la transferencia de datos. Los PVCs son usados en situaciones donde la transferencia de datos es constante. Además son configurados por el proveedor del servicio cuando es solicitado en una orden. Su costo es más alto debido a su disponibilidad constante.

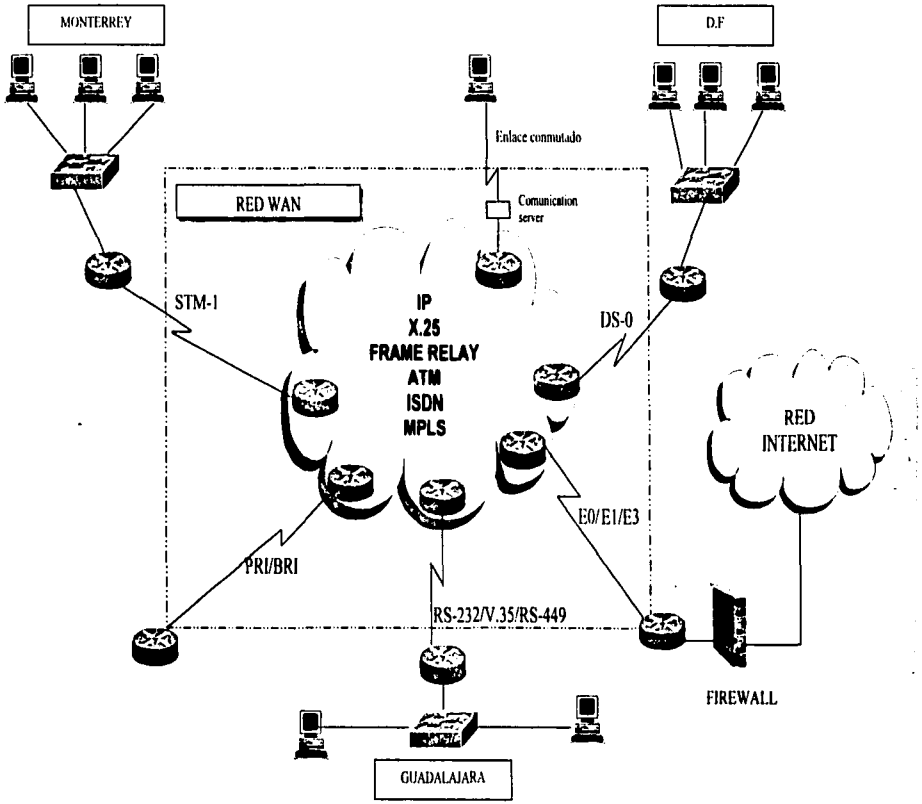


Figura 1.15. Representación de las posibles Redes de Área Amplia

CAPÍTULO II

CONJUNTO DE PROTOCOLOS TCP/IP

2.1 Arquitectura TCP/IP

En la segunda mitad de los 70's, el departamento de defensa de los Estados Unidos de Norteamérica (DoD), propuso un proyecto el cual debía generar el desarrollo de un protocolo para comunicaciones entre nodos de computadoras, de una extensa red. Tal protocolo debía cumplir con ciertos requisitos para ser instalado. Así en 1979, se publico el documento que contenía todo el marco teórico del protocolo TCP/IP.

Aunque TCP/IP indica que es una combinación de dos protocolos Protocolo de Control de Transmisión y Protocolo de Internet, el término TCP/IP no se refiere a una entidad única que combina dos protocolos, sino a un conjunto de protocolos que proporcionan servicios de red como son registros remotos, transferencias remotas de archivos y correo electrónico. Ver figura 2.1.

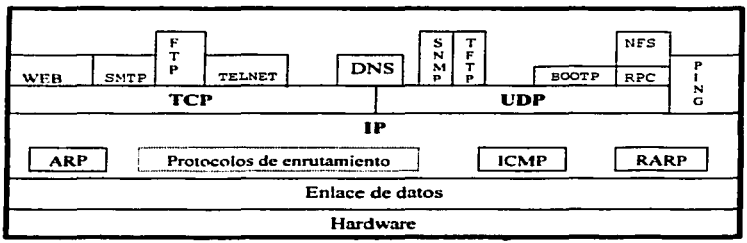


Figura 2.1. Diagrama de los protocolos TCP/IP

Para una mejor comprensión del los protocolos TCP/IP, la información siguiente resume las aplicaciones de usuario que soportan los protocolos TCP/IP.

Telnet.
El programa Telnet proporciona la capacidad de acceso de manera remota. Lo que permite al usuario de una máquina accesar en otra y actuar como si estuviera directamente enfrente de la segunda máquina. La conexión puede estar en cualquier parte de la red local o en otra red en cualquier parte del mundo, siempre que el usuario tenga autorización para accesar en el sistema.

Protocolo de transferencia de archivos, FTP (File Transfer Protocol).
El protocolo FTP permite a un archivo de un sistema copiarse a otro sistema. En realidad el usuario no se registra como un usuario completo en la máquina a la que se desea tener acceso como con Telnet, en su lugar usa el programa FTP para permitir el acceso y poder copiar uno o más archivos a su máquina. El servidor FTP necesita brindar las autorizaciones correctas para que el cliente FTP tenga acceso a los archivos. El termino transferir implica que el archivo se mueve de un sistema a otro, pero en realidad solo se hace una copia y el archivo original no se afecta.

Protocolo de Transferencia de Mail Simple, SMTP (Simple Mail Tranfer Protocol).
El protocolo SMTP se usa para transferir correo electrónico. SMTP es transparente por completo para el usuario, es decir, los usuarios casi nunca se dan cuenta del trabajo de SMTP. Se conecta con máquinas remotas y transfiere mensajes de correo de modo parecido a como FTP transfiere archivos. Además, es un protocolo libre de problemas y muy usado.

Servicio de Nombres de Dominio, DNS (Domain Name Service).
Este protocolo permite resolver la dirección de una máquina destino, utilizando el nombre de esa máquina destino. Es decir, hace una traducción de nombre a dirección de máquina.

Sistema de archivo de red, NFS (Network File System)

Es un conjunto de protocolos desarrollados por Sun Microsystems para permitir a múltiples máquinas, tener acceso a los directorios de cada una de las otras, de manera transparente, lo anterior se logran al usar un esquema de sistemas de archivo distribuido. Los sistemas NFS son comunes en ambientes corporativos grandes, en especial aquellos que usan estaciones de trabajo UNIX.

Protocolo Simple de Administración de Red, SNMP (Simple Network Management Protocol).

Proporciona mensajes de estado (up, down) de los equipos o puertos y reporta problemas a través de una red hacia un administrador. Usa el protocolo UDP (User Datagram Protocol) como un mecanismo de transporte.

Llamada de Procedimiento Remoto, RPC (Remote Procedure Call)

El protocolo RPC es un conjunto de funciones que permite a una aplicación comunicarse con otra máquina (el servidor).

Protocolo Trivial de Transferencia de Archivos, TFTP (Trivial FileTransfer Protocol).

TFTP es un protocolo de transferencia de archivos más sencillo que FTP y carece de seguridad. Usa el protocolo UDP como transporte.

Protocolo de control de transmisión, TCP (Transmisión Control Protocol).

Es un protocolo de comunicaciones que proporciona una transferencia confiable de datos. Es responsable de ensamblar los datos pasados de aplicaciones de capas superiores hacia paquetes estándar y asegurar que los datos se transfieran en forma correcta.

Protocolo de Internet, IP (Internet Protocol).

Es responsable de mover los paquetes de datos ensamblados, ya sea por el TCP o el UDP a través de las redes. Usa un conjunto de direcciones únicas para cada dispositivo en la red, a fin de determinar el enrutamiento y los destinos.

Protocolo de datagrama de usuario, UDP (User Datagram Protocol).
Es un protocolo no orientado a conexión, lo que significa que no atiende la retransmisión de datagramas, UDP no es confiable, pero tiene propósitos especializados. Si las aplicaciones que usa el UDP tienen incorporada la revisión de la confiabilidad se superan los defectos del UDP.

Protocolo de Internet para Mensajes de Control, ICMP (Internet Control Message Protocol).
Es responsable de revisar y generar mensajes sobre el estado de dispositivos en una red. Puede usarse para informar a otros dispositivos de una falla en una máquina particular. Por lo general, ICMP e IP funcionan juntos.

En siguiente sección se hace un análisis más detallado de los dos elementos más importantes del TCP/IP. Empezando con el Protocolo de Internet, la parte IP. Después el TCP y UDP.

2.2. Protocolo de capa de red, IP.

El protocolo de Internet, IP, es un mecanismo de entrega de datagramas sin conexión. IP proporciona tres importantes funciones:

- El protocolo define la unidad básica de transferencia de datos usada para las redes TCP/IP.
- El software IP realiza la función de enrutamiento (encaminamiento), eligiendo una trayectoria sobre la cual los datos pueden ser enviados.
- IP incluye un conjunto de reglas acerca de como los enrutadores y las máquinas (hosts) deben procesar los datagramas, como y cuando los mensajes de error deben ser generados, y las condiciones bajo las cuales los datagramas deben ser descartados.

IP es un protocolo de capa de red que contiene información de direccionamiento y alguna información de control que habilita los paquetes para ser enrutados.

IP está documentado en la RFC 791 y es el protocolo principal de la capa de red dentro del conjunto de protocolos TCP/IP. Además, tiene dos responsabilidades principales:

- Entrega de datagramas realizando el mejor esfuerzo a través de la red proporcionando un servicio sin conexión.
- Proporcionar fragmentación y reensamble de datagramas para soportar enlace de datos con diferentes tamaños de Unidades de Transmisión Máximo, (MTU).

2.2.1. Datagrama IP

El datagrama IP está formado por campos de varias longitudes como se muestran en la figura 2.2.

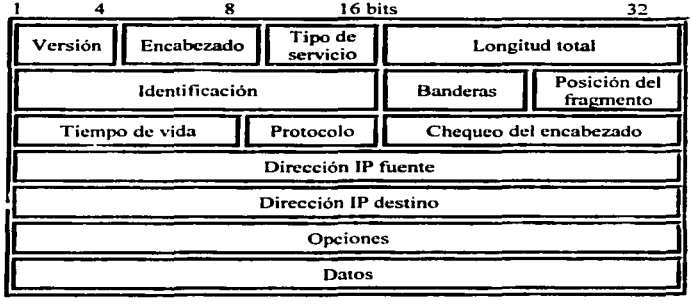
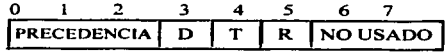


Figura.2.2 Diagrama de campos

2.2.2. Campos del datagrama IP:

- **Versión:** Indica la versión IP actualmente usada. En un campo de cuatro bits.
- **Longitud del encabezado:** Indica la longitud del encabezado del datagrama en palabras de 32 bits. Campo de cuatro bits.

- **Tipo de servicio:** Le especifica a un protocolo de capa superior como le gustaría al datagrama actual ser manejado por éste y asigna al datagrama varios niveles de importancia. Es un campo de un byte, el cual está dividido por bits con una función en particular como se observa en la siguiente figura.



Precedencia = Importancia del datagrama
Este campo de tres bits le dice al enrutador receptor, que tan importantes son los datos.

Los valores posibles de precedencia (importancia) son los siguientes:

Precedencia	Valor
Control de red	111
Control de interconexión entre redes	110
Critico	101
Predominantemente urgente	100
Urgente	011
Inmediato	010
Prioridad	001
Rutinarios	000

D = Retardo
Este bit permite que algunas aplicaciones soliciten rutas con la menor cantidad de retardo de propagación. Para solicitar el mínimo retardo este bit es establecido a 1.

T = Capacidad de transmisión útil
Si este bit es establecido a 1, los enrutadores soportando los datos utilizarán las trayectorias de comunicación con más alta capacidad de transmisión de datos útil.

R = Confiabilidad
Este campo de un bit permite que las aplicaciones soliciten que los datos viajen a través de la ruta con menos oportunidad de perdida de datos, cuando son establecidos en uno.

Estos últimos tres bits son mutuamente excluyentes. Por lo tanto se puede poner únicamente uno de los tres.

- **Longitud total:** Campo de dos bytes, indica la longitud en bytes del paquete entero incluyendo el encabezado, el tamaño de datagrama máximo es de 65535 bytes.

- **Identificación:** Campo de dos bytes, que tiene un número entero que identifica el datagrama actual, además, ayuda a juntar fragmentos de datagrama.

- **Banderas:** Es un campo de tres bits que controlan la fragmentación; el primero especifica si el paquete puede ser fragmentado, el de en medio indica si el paquete es el último fragmento y el tercero no se usa.

- **Posición del fragmento (offset):** Indica la posición de los fragmentos con respecto al datagrama original para que el receptor los ponga en orden.

- **Tiempo de vida:** Campo de un byte, tiene un contador que va decrementando hasta cero en donde el paquete es descartado.

- **Protocolo:** Indica cual protocolo de capa superior recibe los paquetes después del proceso IP.

- **Verificación del encabezado:** Ayuda a mantener la integridad del encabezado IP ya que proporciona un chequeo de error en el encabezado.

- **Dirección IP fuente:** Es una dirección de 32 bits del emisor.

- **Dirección IP destino:** Es una dirección de 32 bits del receptor.

- **Opciones:** Le permite a IP soportar pruebas de enrutamiento y depuración de la red, como seguridad.

- **Datos:** Contienen información de las capas superiores.

2.2.3. Encapsulado IP

Todos los datagramas deben ser transportados por una trama de la capa de enlace de datos, como se muestra en la figura 2.3. El datagrama es vaciado en el campo de datos de la trama y esto se conoce como encapsulado del datagrama IP.

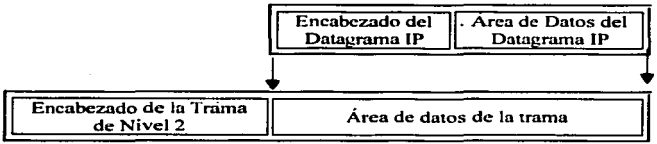


Figura 2.3. Datagrama IP encapsulado en la trama de nivel de enlace de datos.

2.2.4. Tamaño del datagrama, Unidad de Transferencia Máximo (MTU) de la red y fragmentación.

En el caso ideal, un datagrama IP siempre podría ser alojado en el campo de datos de una trama, haciendo la transmisión a través de la red muy eficiente. Sin embargo, el datagrama debe viajar por redes muy diversas con diferentes tamaños de trama o diferentes unidades de transferencia máxima. El protocolo IP se diseño de tal manera que en lugar de tratar de cumplir con un tamaño estándar de unidad de transferencia máxima para las diferentes redes, se elige un tamaño inicial de datagrama y se arregla una forma de dividir grandes datagramas dentro de pequeñas piezas cuando el datagrama necesita atravesar una red que tienen una trama pequeña. Las pequeñas piezas en las cuales el datagrama es dividido son llamadas fragmentos y el proceso de dividir el datagrama es conocido como fragmentación.

La fragmentación usualmente ocurre en un enrutador. El enrutador usualmente recibe en datagrama de una red con una unidad de transferencia y debe enrutar ésta sobre una red en la cual la unidad de transferencia es menor que el tamaño del datagrama.

En la figura 2.4, dos host se conectan directamente a redes Ethernet las cuales tienen una unidad de transferencia de 1500 octetos. Los dos host pueden enviase datagramas hasta 1500 octetos. Sin embargo, la trayectoria entre ellos incluye una red con una unidad de transferencia máxima de 620 octetos. Si el host A envía un datagrama al host B mayor de 620 octetos, el enrutador R1 fragmentara el datagrama. Similarmente R2 fragmentara un datagrama grande del host B que se envíe al host A.

Figura 2.4 Fragmentación de una trama Ethernet.

El tamaño del fragmento es elegido de modo que cada fragmento pueda ser alojado en una sola trama. El protocolo IP no limita el tamaño de los datagramas, ni garantiza que los datagramas sean entregados sin fragmentación. La fuente puede elegir cualquier tamaño de datagrama que piense sea apropiada; la fragmentación y reensamble ocurren automáticamente, la especificación IP señala que los enrutadores deben aceptar datagramas de nasta unidades de transferencia máxima de las redes donde están conectados. Además, los enrutadores deben siempre manejar datagramas de hasta 576 octetos. Los hosts también deben ser capaces de aceptar y reensamblar datagramas de al menos 576 octetos.

La fragmentación de un datagrama significa dividirlo en diversas piezas, como se puede ver en la figura 2.5. Cada pieza tienen el mismo formato que el datagrama original. Cada fragmento contienen un encabezado de datagrama que duplica muchos de los datos del encabezado del datagrama original (excepto un bit en el campo de banderas), seguido por los datos hasta un límite del tamaño de la trama.

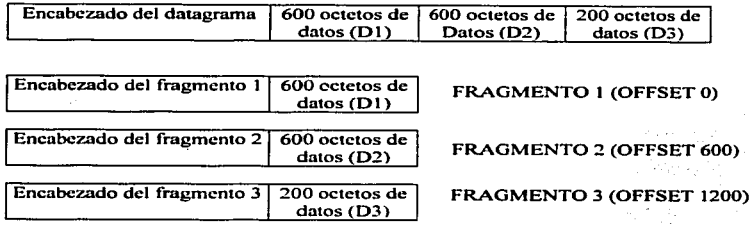


Figura 2.5. Paquete Fragmentado en Tres.

Reensamble de fragmentos.

Una vez que un datagrama es fragmentado, todos los fragmentos viajan a través de los enrutadores hasta alcanzar su destino final donde son reensamblados.

2.2.5. Direccionamiento IP

El direccionamiento IP es lo más importante para el proceso de enrutamiento de datagramas a través de la red. Cada dirección IP tiene componentes específicos y sigue un formato básico. Las direcciones IP pueden ser subdivididas y usadas para crear direcciones para subredes.

A cada host sobre una red TCP/IP se le asigna una dirección lógica de 32 bits que es dividida en dos partes principales: una parte de red y una parte de host, como lo indica la figura 2.6a. La parte de la red, identifica a la red y debe ser asignado por el Centro de Información de red Internet, (InterNIC), si la red va a formar parte de Internet. El número de host, identifica a cada host sobre una red y es asignado por el administrador local.

Como host se entiende cualquier computadora personal, mini computadora, enrutador, servidor, o cualquier maquina o equipo de cómputo con CPU.

2.2.5.1. Formato de la dirección IP

Los 32 bits de una dirección IP son agrupados en 4 octetos de bits separados por puntos y representados en un formato decimal, ver figura 2.6b. Cada bit en el octeto tiene un peso binario (128, 64, 32, 16, 8, 4, 2, 1). El mínimo valor para un octeto es 0, y el máximo valor para el octeto es 255.

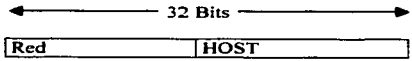


Figura. 2.6a. Dirección lógica de 32 bits

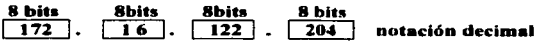


Figura. 2.6b. Formato decimal de la dirección lógica.

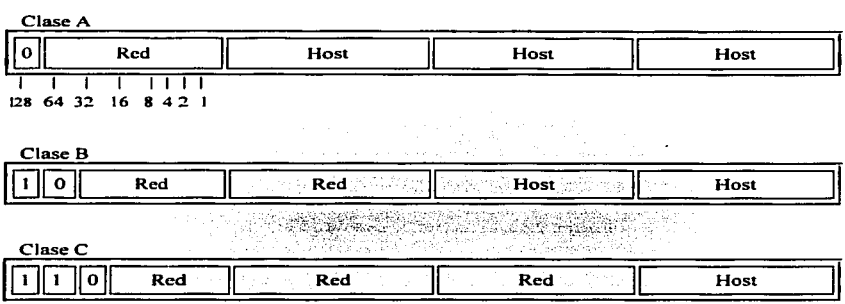
2.2.5.2. Clases de direcciones IP

El direccionamiento IP se divide en cinco clases diferentes: A, B, C, D y E. De las cuales solo la clase A, B y C son para uso comercial. El primer octeto de la izquierda indica la clase de red. Ver la siguiente tabla.

Clase de dirección IP	Formato	Proposito	Rango de direcciones	Bits utilizados Red / host	Numero máximo de host
A	R.H.H.H	Organizaciones grandes	1.0.0.0 a 126.0.0.0	7 /24	16,777, 214
B	R.R.H.H	Organizaciones medianas	128.1.0.0 a 191.254.0.0	14/16	65.534
C	R.R.R.H	Organizaciones algo pequeñas	192.0.1.0 a 223.255.254.0	22/8	254
D	---	Grupo de multicast	224.0.0.0 a 239.255.255.255	---	---
E	---	Experimentales	240.0.0.0 a 254.255.255.255	---	---

R = Número que corresponde a la red H = Número que corresponde al host

Formato de las clases A, B y C de las direcciones IP.



Para determinar a que clase pertenece una dirección lo más fácil es examinar el primer octeto de la dirección y compararla con la información de la tabla anterior.

2.2.5.3. Direccionamiento de subredes IP.

Las redes IP pueden ser divididas en pequeñas redes llamadas subredes. La subdivisión proporciona un uso más eficiente de las direcciones de red y la capacidad de contener tráfico de Broadcast.

Las subredes están bajo administración local. De una dirección de red se pueden sacar varias subredes. Por ejemplo, 172.16.1.0, 172.16.2.0, 172.16.3.0 y 172.16.4.0 son subredes de la red 171.16.0.0, que es una dirección de clase B.

2.2.5.4. Mascaras de Subred.

Las direcciones de subred se crean tomando bits prestados de la parte del host; el número de bits prestados varia y es especificado por la mascara de subred.
Las mascaras de subred usan el mismo formato que la dirección IP. Sin embargo, coloca 0's binarios en todos los bits que especifican la parte del host y tiene 1's en todos los bits que especifican la parte de red y subred.

La mascara estándar para una dirección como la anterior de clase B es 255.255.0.0; la cual especifica que los dos primeros octetos son de red y los restantes son para identificar el host, como lo indica el formato para las direcciones de clase B. Para crear las subredes se debe tomar una parte que corresponde al host, la cual se especifica con la mascara de red como se indica el ejemplo siguiente.

Red	Red	Subred	Host	
11111111	11111111	11111111	00000000	Binario
255 .	**255** .	**255** .	**0**	Decimal

Para una dirección 172.16.4.2 utilizando la mascara anterior, se está indicando que los primeros 2 octetos identifican la red, el tercer octeto la subred y el último identifica al host. Cabe aclarar que con esta máscara, esta dirección de subred puede tener 253 direcciones de host diferentes.

2.3. Protocolo de Resolución de Direcciones. ARP (Address Resolution Protocol).

Para que dos máquinas se comuniquen sobre la red, cada una de ellas debe conocer la dirección física (MAC) de la otra máquina.

Con la ayuda del ARP una máquina puede dinámicamente descubrir la dirección física de otra máquina correspondiente a una dirección IP particular. Ver figura 2.7.

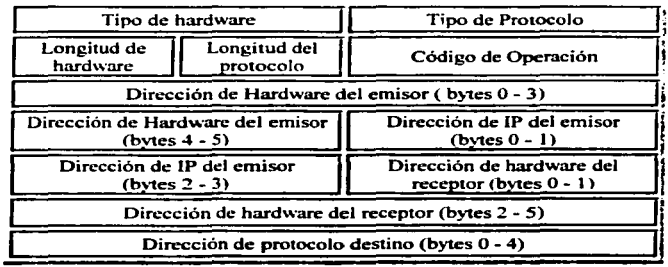


Figura 2.7. Formato del Mensaje ARP.

2.3.1. Campos del mensaje ARP

- **Tipo de Hardware:** Indica el tipo de interfaz del hardware utilizado en la capa física.
- **Tipo de Protocolo**: Indica el tipo de protocolo que está usando el transmisor.
- **Longitud de la Dirección de hardware:** Usado para la longitud de cada dirección de hardware. dado en bytes.
- **Longitud del Protocolo:** Contiene la longitud en bytes de la dirección de protocolo de capa 3. En el caso de IP es establecida a 4.
- **Código de Operación:** Indica si el datagrama es una solicitud ARP o una respuesta ARP. 0001 para una Solicitud (request) y 0002 para una respuesta (reply).
- **Dirección de Hardware del Emisor:** Dirección de la capa física del equipo que envía el mensaje.
- **Dirección de IP del Emisor:** Dirección de capa 3 de la estación que envía el mensaje.
- **Dirección de Hardware del Receptor:** la dirección del hardware del dispositivo receptor.
- **Dirección de IP Destino:** Dirección de capa 3 de la estación destino que recibe el mensaje.

2.3.2. Funcionamiento del protocolo ARP.

Si una máquina A que quiere comunicarse con la máquina B, pero no conoce su dirección física sólo conoce su dirección IP; envía una trama con la dirección física destino en Broadcast (FFFFFFFFFFFF) y la dirección IP de la máquina B. Todas las máquinas en el segmento, incluyendo B reciben el paquete, pero únicamente la máquina B reconoce su dirección IP y envía una respuesta que contiene su dirección física.

Cuando A recibe la respuesta de B, ésta usa la dirección física de B para enviar paquetes IP directamente a B, además, guarda la dirección en la memoria para futuras comunicaciones.

La mejor manera de observar el funcionamiento de ARP es usando otra aplicación de TCP/IP, el comando PING.

Cuando se usa este comando y se observa con un analizador de protocolos se obtienen los dos mensajes ARP, uno de solicitud y otro de respuesta. En el mensaje de solicitud se observa como se usa una dirección de broadcast para asegurarse que la estación destino escuchará el mensaje de solicitud y lo contestará poniendo en la respuesta su dirección física.

Un caso especial de ARP es el conocido como **Proxy ARP**.

Proxy ARP se aplica cuando una máquina se quiere comunicar con otra máquina; cada una conectada en una red diferente, las cuales están comunicadas por un enrutador.

Proxy ARP consiste en responder con la dirección física del enrutador, la solicitud ARP de una máquina en una red física a otra máquina en una red física diferente. Ver figura 2.8.

El enrutador responde con su dirección física porque reconoce la dirección que se busca en su tabla de enrutamiento y al recibir información con ese destino el se encargara de hacerla llegar.

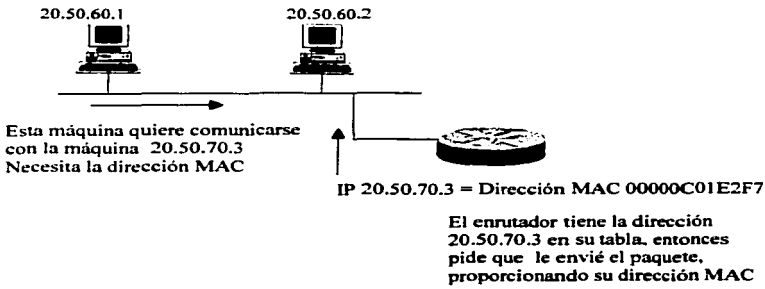


Figura 2.8. Solicitud ARP

También existe un protocolo que funciona al revés de ARP y es el protocolo RARP.

RARP es usado para encontrar la dirección IP, basados en una dirección física de la máquina destino. RARP es importante para nodos sin disco duro, los cuales no conocen su dirección de red cuando ellos se inicializan (boot).

2.4. Protocolos de enrutamiento.

Los enrutadores son los equipos responsables de la transmisión de información de una red local a otra. Un enrutador por lo general tiene más de dos interfaces que pueden corresponden redes o subredes diferentes si así se desea, el trabajo del enrutador es enviar (enrutar), paquetes de una red a otra. En el caso más simple la función de un enrutador es básicamente la de aceptar un paquete por una interfaz y enrutarlo por otra.

Con el objeto de desempeñar su función los enrutadores mantienen tablas de rutas IP de la red (Tablas de enrutamiento). Éstos tienen la habilidad para determinar el camino que el paquete debe seguir para alcanzar su destino. Los protocolos de enrutamiento son el lenguaje por medio del cual los enrutadores intercambian actualizaciones uno con otro acerca de la localización y condición de todos los enlaces posibles en la red.

Existe un número variado de protocolos de enrutamiento, cada uno con ventajas y desventajas. Los tipos de protocolos están agrupados por los siguientes puntos

- Por la función del protocolo de enrutamiento.

  Descubrir a su vecino
  Enrutamiento interior
  Enrutamiento exterior

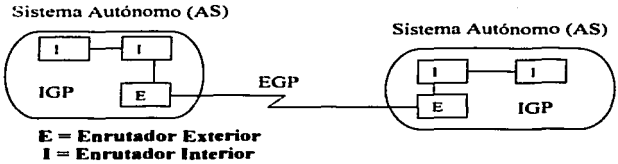
- Basados en la determinación del camino.

  Vector a distancia
  Estado del enlace

- Basados en la modificación del camino
  Estáticos
  Dinámicos.

2.4.1. Protocolos de enrutamiento Exterior e interior.

Algunos enrutadores son usados para mover información a través de un grupo particular de redes bajo el mismo control administrativo. Un grupo de redes y enrutadores bajo el mismo control administrativo es conocido como sistema autónomo (AS, Autonomus System). Los enrutadores usados con un único sistema autónomo son llamados enrutadores interiores y la comunicación se logra a través de enrutamiento interno, ellos usan una variedad de Protocolos de gateway internos (IGPs), como son: RIP, IGRP, EIGRP y OSPF. Los enrutadores que mueven información entre sistemas autónomos diferentes son conocidos como enrutadores exteriores, y manejan protocolos de enrutamiento exteriores, conocidos como Protocolos de gateway exterior (EGP), como es BGP.



2.4.2. Protocolo de Enrutamiento Vector a Distancia

El termino vector a distancia se refiere a la clase de algoritmos que los enrutadores usan para propagar información de enrutamiento. Se asume que cada enrutador comienza con un conjunto de rutas de aquellas redes a las cuales está conectado. Ejemplos de estos protocolos son RIP, IGRP y EIGRP.

Al utilizar este tipo de protocolo cada nodo mantiene la distancia desde el mismo hacia cada posible destino, las distancias son procesadas usando la información recibida de los nodos vecinos.

Los enrutadores mantienen una lista de las rutas en una tabla, donde cada entrada identifica una red destino y proporciona la distancia a esa red medida en saltos, como lo hace el protocolo RIP, o calculando una métrica, como lo hace IGRP y EIGRP.

Periódicamente cada enrutador envía una copia de su tabla de enrutamiento a cualquier otro enrutador que pueda alcanzar directamente. RIP manda actualizaciones cada 30 segundos, IGRP en intervalos de 90 segundos y EIGRP manda actualizaciones sólo cuando la topología de la red cambia.

El termino vector a distancia viene de la información enviada en mensajes periódicos. Un mensaje contiene una lista de pares (V, D), donde V identifica un destino (vector) y D identifica la distancia al destino.

Nota: el protocolo EIGRP es un protocolo Vector a Distancia mejorado, ya que también combina características de Estado del enlace.

2.4.3. Protocolos de Estado del Enlace

En los protocolos del estado del enlace cada nodo es responsable de aprender todo acerca de su vecinos, y poner la información en un paquete de estado del enlace (LSP). Los LSP son transmitidos a todos los nodos y usando la información de todos los nodos, se crea un mapa completo de la topología; y una vez que se obtiene el mapa se corre un algoritmo para encontrar el mejor camino al destino. Estos tipos de protocolos no intercambian una tabla de los destinos, en vez de esto propaga periódicamente información acerca del estado del enlace a todos los enrutadores.

Para informar a todos los enrutadores, cada nodo envía un mensaje en broadcast que lista el estado de cada uno de sus enlaces. El mensaje de estado no reporta rutas, simplemente indica si la comunicación es posible entre pares de enrutadores.
Un ejemplo de este tipo de protocolo es el OSPF.

2.4.4. Diferencias entre los protocolos Vector a Distancia y Estado del Enlace

Estado del Enlace

- Solo manda actualizaciones en el momento que ocurre un cambio
- No manda todas las rutas solo manda el estado del enlace que cambio.
- Se ayuda de mensajes Hellos para ver si el vecino está vivo.

Vector a Distancia

- El enrutador manda una actualización cada cierto tiempo, exista o no un cambio en los enlaces, la cual lleva la tabla de enrutamiento completa.
- En cuanto ocurre un cambio en la estructura de la red, manda a sus vecinos la tabla completa.
- No existen los mensajes de Hellos

Nota: El cambio en la red incluye, la perdida de un enlace o que se levante un enlace o simplemente se pierda una ruta porque alguien quito la dirección o apago el quipo.

2.4.5. Enrutamiento dinámico y estático

Los protocolos de enrutamiento IP son dinámicos. En el enrutamiento dinámico las rutas son calculadas a intervalos regulares por el software en el equipo y van aprendiendo de sus vecinos los hosts remotos que sus vecinos le anuncian. En cambio en el enrutamiento estático, las rutas son establecidas por el administrador de red y no son cambiadas hasta que el administrador de red lo hace.

El proceso de enrutamiento consiste solo de paquetes enviados basados en la información interna, sin importar si los paquetes llegan a su destino final. En otras palabras IP no proporciona retorno de reporte de error a la fuente cuando anomalías en el enrutamiento ocurren. Esta tarea es dejada a otro protocolo de Internet, el Protocolo de Control de Mensajes de Internet (ICMP).

2.5. Protocolo de Control de Mensajes de Internet. (ICMP).

ICMP es responsable de revisar y generar mensajes sobre el estado de dispositivos en una red. Puede usarse para informar a otros dispositivos de una falla en una máquina particular. Por lo general, ICMP e IP funcionan juntos.

ICMP proporciona mensajes de ayuda como las siguientes:

- Mensajes de Solicitud de Eco y respuesta para pruebas de alcance de nodos a través de la red (PING).
- Mensajes de redireccionamiento para estimular un enrutamiento más eficiente.
- Mensajes de tiempo excedido para informar a las fuentes que los datagramas han excedido su tiempo asignado para existir en la red.
- Mensajes de anuncios de enrutador y de solicitud de enrutador, para determinar las direcciones de enrutadores sobre redes directamente conectadas.

2.5.1. Encapsulado del mensaje ICMP.

Los mensajes ICMP requieren ser encapsulados en el campo de datos de un datagrama IP. Los datos conduciendo mensajes ICMP son enrutados exactamente como los datagramas conduciendo información de usuario. La figura 2.9 muestra el mensaje ICMP encapsulado.

Los mensajes ICMP pueden ser extraviados o descartados, pero no se generan mensajes de reporte de error que resulten de datagramas conduciendo mensajes de error ICMP.

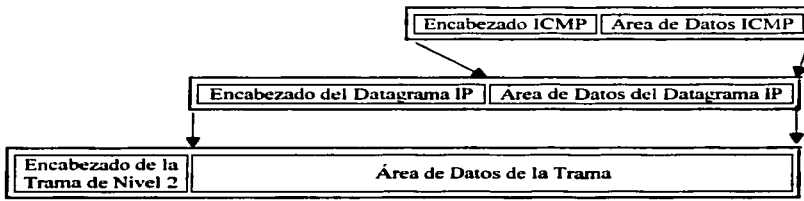


Figura 2.9. Mensaje ICMP Encapsulado en un paquete IP

2.5.2. Formato del mensaje ICMP

Aunque cada mensaje ICMP tiene su propio formato, todos ellos comienzan con los mismos tres campos:

- Un campo de tipo de mensaje, el cual contiene un entero de 8 bits. Indicando si el mensaje es una solicitud o una respuesta.
- Un campo de código de 8 bits que proporciona información adicional acerca del tipo de mensaje.
- Un campo de Checksum (verificación) de 16 bits para el encabezado ICMP.

Adicionalmente, los mensajes ICMP que reportan errores siempre incluyen el encabezado y los primeros 64 bits de datos del datagrama con problemas.

El campo de código de mensaje de 8 bits puede contener uno de los valores mostrados en la tabla siguiente:

Valor	Descripción
0	Eco de Respuesta
3	Destino no Alcanzable
4	Fuente agotada
5	Se requiere redireccionamiento
8	Eco de solicitud
11	Tiempo de vida excedido
12	Problema con los parámetros
13	Solicitud de marcador de tiempo
14	Respuesta de marcador de tiempo
15	Solicitud de información
16	Respuesta de información
17	Solicitud de máscara de dirección
18	Respuesta de mascara de dirección

2.6. Protocolo de Control de Transmisión TCP.

TCP proporciona transmisión de datos confiable en un ambiente IP. TCP corresponde a la capa de transporte, capa 4 del modelo OSI. Entre los servicios que proporciona el TCP están: transferencia de datos fluido, confiabilidad, control de flujo eficiente, operación full-duplex y múltiplexaje.

El protocolo también permite especificar como el software distingue entre múltiples destinos en una máquina dada, y como las máquinas en comunicación se recuperan de errores como pérdida o duplicación de paquetes.

El TCP no es un protocolo sencillo, pero una de las razones por las que se utiliza este protocolo de trasporte complejo, es la falta de confiabilidad del IP, ya que IP no garantiza la llegada de un datagrama; es un sistema no orientado a conexión y no confiable. El IP tan sólo maneja el enrutamiento (encaminamiento) de los datagramas y si ocurren problemas, IP desecha el paquete; generando en el proceso un mensaje de error ICMP que regresa al emisor. Entonces la tarea de comprobar el estado del datagrama enviado a través de la red y manejar la retransmisión de la información si se han desechado algunos datos recae en el protocolo TCP.

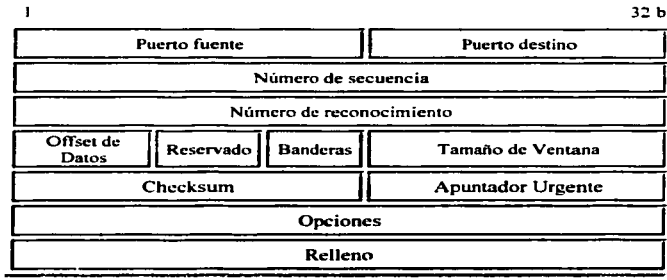


Figura 2.10. Formato del paquete TCP.

2.6.1. Formato del paquete TCP.

Los campos que describen al paquete TCP se pueden observar en la figura 2.10 y se describen a continuación.

- **Puerto fuente:** Número de 16 bits que identifica un programa de aplicación de capa superior que usa la conexión TCP. Puerto TCP de la sesión fuente.
- **Puerto destino:** Campo de 16 bits con el número de puerto TCP de la sesión destino.
- **Número de secuencia:** Campo de 4 bytes que identifica la posición del primer octeto de datos en el mensaje actual. En la fase de establecimiento de conexión, este campo también puede ser usado para identificar un número de secuencia inicial usado en una transmisión entrante.
- **Número de reconocimiento:** Este campo de 4 bytes muestra el número del siguiente byte de datos que el destino espera recibir.
- **Posición de Datos (Offset):** Indica el número de palabras de 32 bits que están en el encabezado TCP.
- **Reservado:** Campo de 6 bits reservado para uso futuro.
- **Banderas:** Porta una variedad de información de control, incluyendo los bits SYN y ACK usados para establecer la conexión, y el bit FIN usado para terminar una conexión.

Estos bits se representan en la siguiente figura y se describen a continuación:

URG	ACK	PSH	RST	SYN	FIN

Bit urgente (URG). Si está activa (un valor de 1), indica que los datos en este paquete son urgentes y deben ser procesados antes de los demás datos. Los datos más frecuentemente usados de esta naturaleza son comandos para cancelar la sesión o hacer cambios en el estado de la sesión.

Reconocimiento Valido (ACK). Este bit es puesto en 1, si los datos encontrados en el campo de reconocimiento son un número valido. Durante el inicio de una sesión el número de este campo será establecido a 0, mostrando que los datos todavía no han sido intercambiados.

Solicitud de empuje (PSH). Este bit es usado para solicitar que los datos de un usuario normal sean procesados inmediatamente. Se da cuando un usuario está frente al teclado y el tiempo de respuesta debe ser rápido, como en sesiones TELNET o FTP.

Reseteo de conexión (RST). Indica que la conexión debe reiniciarse.

Secuencia de sincronización (SYN). Si está activa, indica que los números de secuencia deben sincronizarse. Esta bandera se usa cuando se está estableciendo una conexión.

Envió de Datos Final (FIN). Cuando es establecido a 1 el proceso destino sabe que el final de datos para esta sesión ha sido enviado por el emisor.

- **Tamaño de Ventana.** Número de dos octetos que indica cuantos octetos adicionales de datos está preparado para recibir.
- **Checksum (Verificación).** Contienen un entero de 16 bits usado para checar el encabezado TCP y los datos.
- **Apuntador Urgente.** Usado si se estableció la bandera de URG. Se utiliza para indicar un desplazamiento en octetos a partir del número de secuencia actual, en el que se encuentran datos urgentes.
- **Opciones. Especifica las opciones de TCP.** Cada opción consta de un número de opción, el numero de bytes en ésta y los valores de la opción. Los valores van desde 64 a 4096 bytes.
- **Relleno.** Campo con bits en cero para completar un múltiplo de palabras de 32 bits.

2.6.2. Encapsulado del segmento TCP

El contenido del segmento TCP se encapsula en el área de datos de un datagrama de IP como se muestra en la figura 2.11.

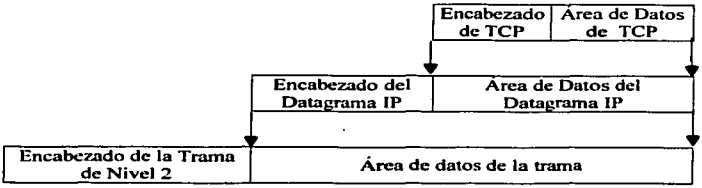


Figura 2.11. Paquete TCP encapsulado en IP

2.6.3. Puertos y Sockets

Todas las aplicaciones de capa superior que usan el TCP (o UDP) tienen un número de puerto que identifica a la aplicación. Esto permite a una máquina por medio del número de puerto identificar el tipo de servicio que le está solicitando un sistema TCP. Los números de puerto pueden cambiarse, aunque esto puede causar dificultades. La mayor parte de los sistemas mantienen un archivo de números de puerto y su correspondiente servicio.

Hay dos formas de asignar los puertos. La primera, una autoridad central asigna números de puertos bien conocidos para diversos servicios y publica una tabla. Cada puerto identifica el destino final, la aplicación, dentro de la máquina. Estos valores bien conocidos toman valores bajos. Desde 0 hasta 1024. Ver tabla 2.2.

La segunda, un software de red asigna dinámicamente los puertos los cuales se asignan de maneras aleatoria, y toman valores grandes desde 1025 hasta 65535.

Número de Puerto	Nombre del Proceso	Descripción
7	Eco	Eco
9	DISCARD	Descartar
11	USERS	Usuarios Activos
13	DAYTIME	Hora del Día
17	QUOTE	Cita del Día
20	FTP-DATA	Datos default de la Transferencia de Archivos
21	FTP	Protocolo de Transferencia de Archivos
23	TELNET	Telnet (conexión remota)
25	SMTP	Transferencia de Correo Simple
37	TIME	Tiempo
42	NAMESERV	Servidor de Nombre de Host
43	NICNAME	Quién es
53	DOMAIN	Servidor de Nombres de Dominio
67	BOOTPS	Servidor de Protocolo de Arranque
68	BOOTPC	Cliente de Protocolo de Arranque
69	TFTP	Transferencia de Archivos Trivial
80	WWW	WEB
161	SNMP	SNMP
179	BGP	Protocolo de Gateway Fronterizo
520	RIP	Protocolo de Información de Enrutamiento

Tabla 2.2. Puertos para las aplicaciones TCP y UDP

Cada circuito de comunicación que entra y sale de la capa TCP se identifica de manera única por una combinación de dos números, los cuales juntos, se llaman un socket (conector). El socket está compuesto por la dirección IP de la maquina y el número de puerto usado por el software TCP. Tanto la máquina transmisora como la receptora tienen sockets. Debido a que la dirección IP es única a lo largo de la red y los números de puerto son únicos para la máquina individual, los números de socket también son únicos a lo largo de la red entera. Esto permite que un proceso hable con otro proceso a través de la red, basado por completo en el número de socket.

Debido a que TCP identifica una conexión con un par de puntos finales, un número de puerto TCP dado puede ser compartido y usar varias conexiones al mismo tiempo (multiplexaje).

Las aplicaciones accesan a la red vía puertos TCP. La razón de tener puertos es que los servicios pueden ser solicitados por su bien conocida identidad y el cliente que está solicitando el servicio puede usar un puerto aleatorio, permitiendo que más de una sesión corra con ese servicio desde la misma dirección IP. Ver figura 2.12.

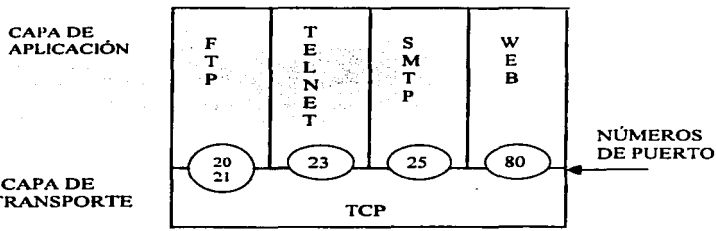


Figura 2.12. Puertos usados por la capa de trasporte para accesar a las aplicaciones TCP.

2.7. El protocolo de usuario UDP.

El protocolo de datagrama de usuario UDP proporciona el mecanismo primario que los programas de aplicación usan para enviar datagramas a otros programas de aplicación. UDP proporciona puertos de protocolo para distinguir entre múltiples programas ejecutándose en una máquina. Cada mensaje contienen un número de puerto destino y fuente, haciendo posible que el software UDP en el destino entregue el mensaje al recipiente correcto. Ver figura 2.13.

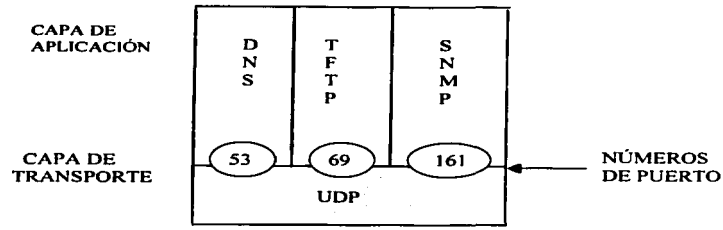


Figura 2.13. Puertos usados por la capa de trasporte para accesar a las aplicaciones UDP.

2.7.1. Encabezado UDP.

UDP es mucho más simple que el protocolo TCP y es muy útil en situaciones donde la confiabilidad de TCP no es necesaria. El encabezado UDP tienen solo 4 campos. Ver figura 2.14.

- **Puerto fuente**. Campo de 16 bits el cual tiene la misma función que en el encabezado TCP. El puerto fuente es un puerto aleatorio mayor a 1024.

- **Puerto destino**. Campo de 16 bis realizando la misma función que en TCP. El puerto destino es un puerto bien conocido y se asocia con una aplicación, por ejemplo el 69 UDP, para la aplicación TFTP.

- **Longitud**. De 16 bits especifica la longitud del encabezado UDP y los datos.

- **Chequeo (Checksum)**. Campo de 16 bits permite revisar la integridad del paquete; el chequeo es opcional, un valor de 0 significa que el chequeo no ha sido registrado.

Puerto fuente	Puerto destino
Longitud	Chequeo(Checksum)
Datos	

Figura 2.14. Campos del encabezado UDP.

UDP utiliza el protocolo IP para transportar mensajes de una máquina a otra, y proporciona la misma entrega de datagramas sin conexión no confiable que IP. No usa reconocimientos para asegurar que los mensajes lleguen, no ordena mensajes y no proporciona control de flujo.

En la figura 2.15 se muestra el encapsulado del datagrama UDP en el área de datos de IP.

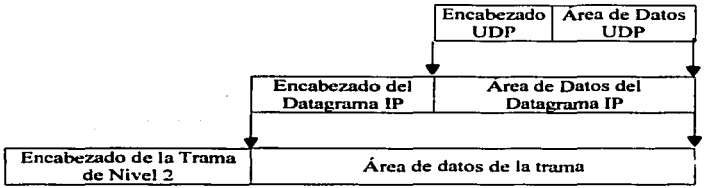


Figura 2.15 Encapsulado del datagrama UDP.

CAPÍTULO III

INTRODUCCIÓN A LA ARQUITECTURA MPLS

Uno de los factores de éxito de la Internet actual está en la aceptación de los protocolos TCP/IP como estándar para todo tipo de servicios y aplicaciones; la cual ha desplazado a las redes de datos tradicionales y ha llegado a ser el modelo de red pública. Cabe aclarar que el enrutamiento IP tradicional tiene varias limitaciones bien conocidas, que van desde temas de escalabilidad hasta un pobre soporte de ingeniería de tráfico.

Si se toma en cuenta el crecimiento imparable de Internet, así como la demanda de nuevos y más sofisticados servicios de red, y las desventajas de enrutamiento IP que comienzan a ser más obvias: se observa la necesidad de cambios tecnológicos fundamentales respecto a las prácticas habituales desarrolladas en la implementación de redes.

Es en este momento cuando toma fuerza una nueva arquitectura de red de reciente aparición, conocida como: MPLS (Multi-Protocol Label Switching, Conmutación de Etiquetas Multiprotocolo)

MPLS fue creada para combinar los beneficios de envío de paquetes basados en conmutación de capa 2 con los beneficios de enrutamiento de capa 3. Similar a redes de capa 2, por ejemplo Frame Relay o ATM, MPLS asigna etiquetas a los paquetes para transporte a través de las redes. El mecanismo de trasporte a lo largo de la red es el intercambio de etiquetas (Label swapping), en el cual los paquetes portan una pequeña etiqueta de longitud arreglada que les dice a los nodos de conmutación a lo largo de la trayectoria del paquete, como procesar y transportar los datos.

Los miembros de la comunidad IETF trabajaron extensivamente para proporcionar un conjunto de estándares para comercializar y evolucionar las ideas de varios proveedores y particulares en el área de conmutación de etiquetas

El objetivo principal del grupo de trabajo MPLS es estandarizar una tecnología base, que integra el transporte por conmutación de etiquetas con el enrutamiento de capa de red.
Se espera que esta tecnología base mejore el precio y desempeño del enrutamiento de capa tres, mejore la escalabilidad de la capa de red y proporcionar una mayor flexibilidad en la entrega de (nuevos) servicios de enrutamiento, gracias a un nuevo servicio de enrutamiento que es agregado sin cambiar el concepto de envío.

La principal diferencia entre MPLS y las tecnologías tradicionales WAN es la forma en que las etiqueta son asignadas y la capacidad para portar una pila de etiquetas (label stack) adjuntas al paquete. El concepto de pila de etiquetas habilita nuevas aplicaciones, como son, ingeniería de trafico, redes privadas virtuales, rápido reenrutamiento alrededor de un nodo o enlace con falla y más.

La arquitectura MPLS describe los mecanismos para realizar la conmutación de etiquetas. MPLS separa claramente el Plano de Control, donde los protocolos de enrutamiento de capa 3 establecen los caminos usados para el transporte de paquetes y el Plano de Datos, en donde se envían paquetes de datos a través de la estructura MPLS, por medio de los LSP (Trayectorias Conmutadas por Etiquetas).

3.1 Arquitectura MPLS

La arquitectura es dividida en dos componentes separadas:

- La componente de control (Plano de control)
- La componente de transporte (Plano de datos)

La componente de control es responsable de crear y mantener la información de la tabla de transporte y para esto utiliza los protocolos estándar de enrutamiento (OSPF, EIGRP, IS-IS y BGP-4) para el intercambio de información con los otros enrutadores. En la figura 3.1 se muestra la arquitectura básica de un nodo MPLS desempeñando enrutamiento IP.

La componente de transporte busca en la tabla de transporte de etiquetas, la cual mantiene la componente de control mediante la conmutación de etiquetas, para tomar la decisión de encaminamiento de cada paquete.

En concreto, la componente de transporte examina la información del encabezado del paquete, busca en la tabla de transporte la entrada correspondiente y dirige el paquete desde la interfaz de entrada del enrutador a la de salida, basado en la etiqueta encontrada en la tabla de trasporte.

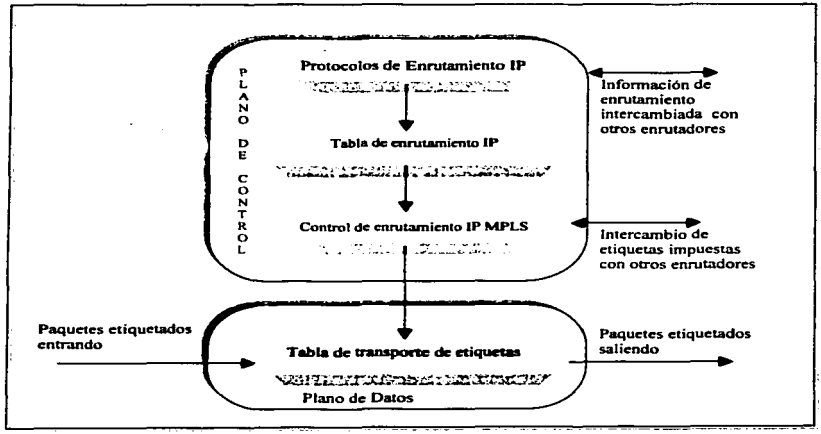


Figura 3.1 Arquitectura básica de un nodo MPLS

Cada nodo MPLS debe correr uno o más protocolos de enrutamiento (o utilizar enrutamiento estático) para intercambiar información de enrutamiento con otros nodos MPLS. En este sentido cada nodo MPLS es un enrutador IP en el plano de control.

Los protocolos de enrutamiento construyen la tabla de enrutamiento, las cuales son usadas para construir la tabla de trasporte IP.

En un nodo MPLS la tabla de enrutamiento se usa para determinar el intercambio de etiquetas impuestas, donde cada nodo MPLS intercambia con sus vecinos una etiqueta por cada subred contenida dentro su tabla. El intercambio de etiquetas por cada destino unicast es realizado por el Protocolo de Distribución de Etiquetas, TDP/LDP.

El proceso de control de enrutamiento IP MPLS, usa las etiquetas intercambiadas con los nodos para construir la Tabla de Transporte de Etiquetas, la cual a su vez es utilizada para enviar los paquetes etiquetados a través de la red MPLS.

Como en todas las nuevas arquitecturas, es necesario describir los términos nuevos que describen los dispositivos que hacen posible la arquitectura MPLS. Estos términos se trataran de aquí en delante.

El primer dispositivo es el enrutador conmutador de etiquetas. LSR (Label Switch Router) que se refiere a cualquier enrutador o switch que implementa procedimientos de distribución de etiquetas y realiza el envío de paquetes basándose en estas etiquetas MPLS. Este procedimiento la permite a un LSR distribuir sus etiquetas impuesta a otros LSR dentro de la red MPLS

Si un LSR impone o dispone de las etiquetas en el borde de la red MPLS este es conocido como un LSR-Frontera (Edge-LSR). La imposición de etiqueta (conocida como acción push), es la acción de colocar una etiqueta o una pila de etiquetas a un paquete en el punto de ingreso del dominio MPLS. La disposición de etiquetas es lo contrario (conocida también como acción pop), es el acto de remover la ultima etiqueta de un paquete en el punto de salida antes de que este sea enviado a un vecino que está fuera del dominio MPLS. Cualquier enrutador que tenga vecinos no MPLS es considerado un Edge-LSR.

Un edge-LSR usa una tabla IP tradicional aumentada con información de etiquetado, así los paquetes pueden ser enviados como paquetes etiquetados a otro nodo MPLS o como paquetes IP puros a nodos no MPLS, en donde la etiqueta es removida y un chequeo de capa 3 (un chequeo del encabezado IP) es realizado para encontrar el destino no MPLS (dirección IP).

3.1.1. Imposición de etiquetas en la frontera de la red.

La imposición de etiquetas es una función al borde de la red, el cual significa que los paquetes son etiquetados antes de que sean enviados al dominio MPLS. Para realizar esta función, un edge-LSR necesita entender donde se debe ubicar el encabezado del paquete y que etiqueta, o pila de etiquetas, debe asignar al paquete.

En el transporte IP de capa 3 tradicional, cada hop en la red realiza un chequeo en la tabla de transporte IP para la dirección destino que el paquete IP contiene dentro del encabezado de capa 3 (conocido como chequeo de capa 3). Éste selecciona la dirección del próximo salto (next hop) para la el paquete y lo envía por una interfaz hacia su destino.

Para escoger el próximo salto del paquete IP, el enrutador realiza una combinación de dos funciones. La primera función es una separación de un conjunto de posibles paquetes en un conjunto de prefijos IP destino. La segunda función es igualar cada prefijo IP destino a una dirección IP del próximo salto.

MPLS también realiza la primera función haciendo una clasificación para el envió, para lo cual utiliza lo que conocemos como FEC (Forwarding Equivalence Classes), Clasificación Equivalente de Envío. El FEC se refiere a la agrupación de paquetes IP para ser enviados en la misma forma, sobre el mismo camino y con el mismo trato de transporte.
La segunda función equivale a igualar una etiqueta local a una dirección IP destino, con un etiqueta que identifica al próximo salto (next hop), de alguna manera esto quiere decir que los prefijos IP ya no serán utilizados para la envío de los paquetes, aunque sabemos que para construir la tablas de transporte siempre serán necesarios.

En una red basada en transporte IP, se hace en cada nodo un análisis del encabezado IP de un paquete enviado y se busca en la tabla de enrutamiento la trayectoria que deberá seguir el paquete en la red hasta su destino. En cuanto se introduce MPLS en la red, el paquete es asignado a un FEC particular solo una vez, y así es como entra el paquete a la red por el edge-LSR. Es entonces cuando el FEC al cual el paquete es asignado, es codificado con un identificador de longitud arreglada, que conocemos como etiqueta.

Cuando el paquete es enviado a su próximo nodo, la etiqueta está ya agregada al paquete IP, así que el próximo equipo en el camino del paquete puede entonces enviarlo a su destino basado en la etiqueta, evitando así un análisis de la información del encabezado de IP del paquete.

3.1.2. Transporte de Paquetes MPLS y Trayectorias Conmutadas por Etiquetas (LSP)

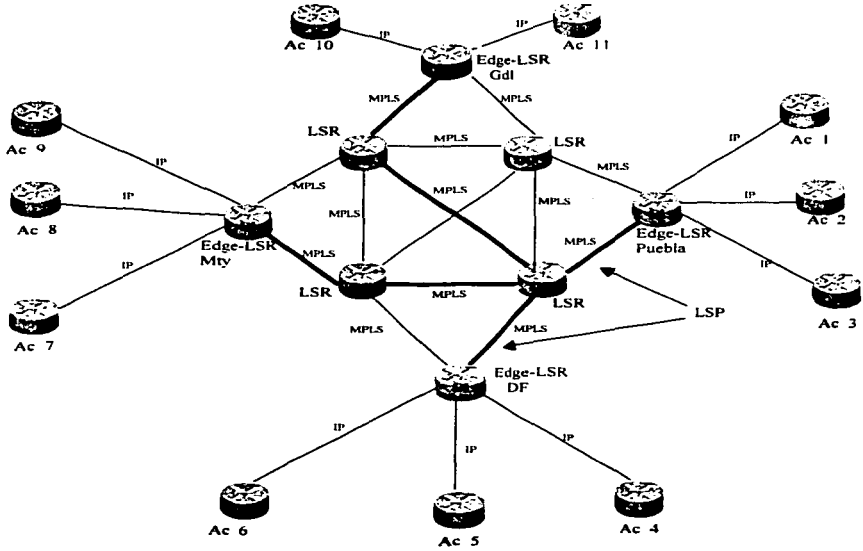
Cada paquete entra a la red MPLS por un LSR de ingreso y sale de la red por un LSR de egreso, este mecanismos crea lo que se conoce como Trayectoria Conmutada por Etiqueta, Label Switched Path (LSP), el cual esencialmente describe un conjunto de LSRs a través de los cuales un paquete debe atravesar para alcanzar un FEC particular en un LSR de egreso. El LSP es unidireccional lo que significa que un LSP diferente es usado para el tráfico de regreso de un FEC particular.

Al momento que el paquete atraviesa una red MPLS, cada LSR cambia la etiqueta entrante por una etiqueta de salida, lo cual continua hasta que el último LSR (LSR de egreso) es alcanzado.

Cada LSR mantienen 2 tablas, las cuales mantienen información que es relevante para la componente de transporte MPLS. La primera es conocida como Base de Información de Etiquetas (TIB/LIB), la cual contiene todas las etiquetas asignadas por el propio enrutador y la asociación de estas etiquetas con las etiquetas recibidas de sus vecinos para cada prefijo IP. Como lo mencionamos estas etiquetas son distribuidas a través del protocolo de distribución de etiquetas (TDP/LDP)

La segunda tabla es conocida como Base de información de transporte por etiquetas (TFIB/TLIB)), la cual es usada durante el transporte actual de paquetes y sostiene solo las etiquetas que están siendo usadas por la componente de transporte MPLS. Es decir, es una asociación de la etiqueta local con la etiqueta del vecino.

En la siguiente figura aparece una red habilitada con MPLS para establecer Trayectorias Conmutadas por Etiquetas, mediante las cuales se lograra la comunicación entre los enrutadores que se encuentran en los sitios de acceso. Por ejemplo el ac_3 de Puebla con el ac_8 de Monterrey o el ac_4 del D.F con el ac_11 de Guadalajara.



Indica algunos de los posibles LSP que crean por las etiquetas MPLS

3.1.3. Aplicaciones MPLS

El verdadero poder de MPLS recae en las aplicaciones que hace posibles, en las que se encuentran Ingeniería de Tráfico y Redes Privadas Virtuales (VPN's). Todas las aplicaciones utilizan la estructura de plano de control mostrada en la figura 3.2. En la figura 3.3 se muestran los diferentes tipos de aplicaciones y su interacción con el plano de datos.

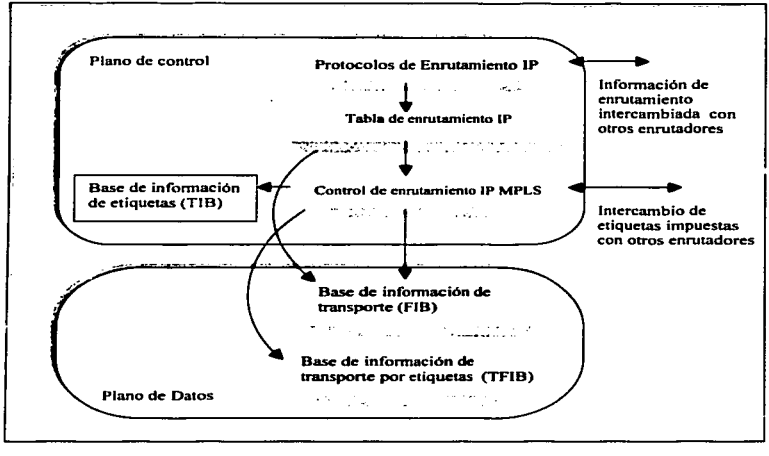


Figura 3.2 Arquitectura de un Edge-LSR usando terminología nueva

Ingeniería de trafico

El objetivo básico de la ingeniería de trafico es adaptar los flujos de tráfico a los recursos físicos de la red, la idea es equilibrar de forma optima la utilización de esos recursos, de manera que no haya algunos que estén sobre utilizados, mientras otros puedan estar desaprovechados.

A comienzos de los 90 los esquemas para adaptar de forma efectiva los flujos de trafico a la topología física de las redes IP eran bastante rudimentarios. Los flujos de tráfico siguen el camino más corto (con menos saltos) calculado por el algoritmo IGP correspondiente. En casos de congestión de algunos enlaces, el problema se resolvía a base de añadir más capacidad a los enlaces. La ingeniería de trafico consiste en trasladar determinados flujos seleccionados por el algoritmo IGP sobre enlaces más cogestionados a otros enlaces más descargados, aunque estén fuera de la ruta más corta. MPLS es una herramienta efectiva para esta aplicación en grandes backbones (columna vertebral de la red) debido a que:

- Permite al administrador de la red el establecimiento de rutas explicitas, especificando el camino físico exacto de un LSP.
- Permite obtener estadísticas de uso LSP, que se pueden utilizar en la planificación de la red y como herramientas de análisis de cuellos de botella y carga de los enlaces, lo que resulta bastante útil para planes de expansión futura.
- Permite hacer encaminamiento restringido de modo que el administrador de la red pueda seleccionar determinadas rutas para servicios especiales (con distintos niveles de calidad).

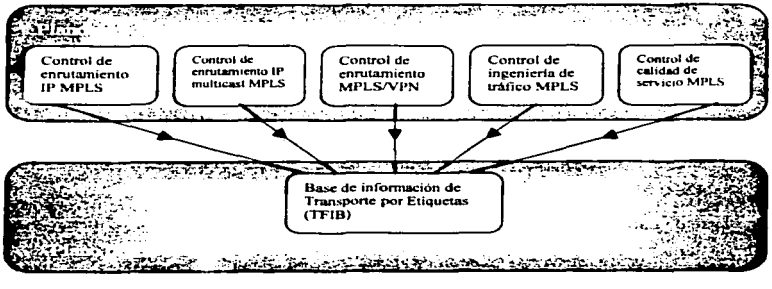


Figura 3.3 Aplicaciones MPLS

Clases de Servicio (CoS)

MPLS está diseñado para poder transportar servicios diferenciados, en donde se definen una variedad de mecanismos para poder clasificar el trafico en un número reducido de clases de servicio, con diferentes prioridades. Según los requisitos de los usuarios, este modelo permite diferenciar servicios tradicionales como el WWW o el correo electrónico (para los que el retardo no es critico), de otras aplicaciones mucho más dependientes del retardo y de la variación del mismo, como son las de video y voz interactiva. Para ello se emplea el campo de la trama de datos conocido como Type of Service (tipo de servicio). Esta es la técnica de QoS (quality of Service) de marcar los paquetes que se envían a la red.

MPLS se adapta perfectamente a ese modelo. Ya que las etiquetas MPLS tienen el campo EXP para poder propagar la clase de servicio CoS en el correspondiente LSP, más adelante se analiza el encabezado MPLS. De este modo, una red MPLS puede transportar distintas clases de trafico, ya que:

- El trafico que fluye a través de un determinado LSP se puede asignar en los diferentes saltos, de acuerdo con la información contenida en el campo EXP.
- Entre cada par de LSR exteriores se pueden proporcionar múltiples LSP's, cada uno de ellos con distintas prestaciones y con diferentes garantías de ancho de banda.

Redes Privadas Virtuales

Una red privada virtual (VPN) se construye a base de conexiones realizadas sobre una infraestructura compartida, con funcionalidades de red y de seguridad equivalente a las que se obtienen con una red privada. Por ser este el tema que más nos interesa, en el capitulo IV se analiza afondo el concepto de redes privadas virtuales.

Cada aplicación MPLS tienen el mismo conjunto de componentes:

- Una base de datos definiendo la tabla de Clases Equivalentes de Transporte (FECs) para la aplicación.
- Protocolos de control para el intercambio de contenidos de la tabla FEC entre los LSR (Protocolos de enrutamiento IP o enrutamiento estático).

- Procesos de control que desempeñan la imposición de etiquetas para FECs y un protocolo para intercambiar etiquetas impuestas entre los LSRs (Protocolo de distribución de etiquetas. TDP/LDP).
- Una base de datos comparando FEC y Etiqueta, FIB (Base de información de Etiqueta). La cual es usada en el LSR para ingresar paquetes y enviarlos dentro de la red MPLS.

3.1.4. Operación del plano de datos MPLS

Para describir la propagación de un paquete IP a través de la red MPLS, tenemos tres pasos principales:

- El LSR de ingreso recibe un paquete IP, clasifica el paquete dentro de su clase equivalente de transporte (FEC), y etiqueta el paquete con la etiqueta de salida correspondiente a la FEC.
- El LSR del centro recibe este paquete etiquetado y usa la tabla de transporte por etiquetas para intercambiar la etiqueta de entrada del paquete que llega, por la etiqueta correspondiente de salida para la misma FEC (en este caso la misma subred).
- Cuando el LSR de egreso recibe el paquete etiquetado para esta particular FEC, remueve la etiqueta y desempeña un chequeo tradicional de capa 3 sobre el paquete IP.

La figura 3.4 muestra estos tres pasos, para un paquete atravesando la red desde un punto llamado Toluca hacia un cliente conectado en el punto Cd. Juárez. Aquí el enrutador Toluca recibe un paquete IP con la dirección destino 10.20.1.1 y realiza el chequeo tradicional de capa 3 a través de la tabla de transporte IP (también conocida como Base de Información de Transporte, FIB). Entonces el enrutador Toluca impone la etiqueta 30 dentro del paquete antes de que sea enviado a el enrutador DF. Para saber que el paquete que le está llegando es un paquete etiquetado y no un paquete IP puro. DF utiliza la información MPLS que le fue insertada al paquete como encabezado MPLS. Debido a que el mecanismo de conmutación rápida de Cisco CEF (Cisco Expresss Forwarding), es el único mecanismo de conmutación de capa 3 que usa la tabla FIB, es importante que CEF se establezca en todos los enrutadores corriendo MPLS y en todas las interfaces de ingreso recibiendo paquetes IP no etiquetados que son propagados como paquetes etiquetados a través de la columna vertebral MPLS.

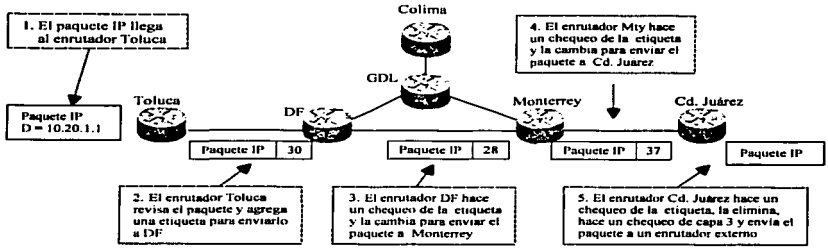


Figura 3.4 Diagrama de transporte de paquetes

3.2. Encabezado MPLS

La etiqueta MPLS es insertada entre el encabezado de capa 2 y el contenido de capa 3 en la trama de capa 2, como se puede observar en la figura 3.5b

Encabezado de capa de Enlace	Encabezado de capa de Red

Figura 3.5a. Paquete no etiquetado en la trama de capa 2

Encabezado de capa 2	Encabezado MPLS	Encabezado de capa 3

Figura 3.5b. Paquete etiquetado en la trama de capa 2

El encabezado MPLS que se muestra en la figura 3.6 contiene:

- Un campo de 20 bits para la **etiqueta**.
- Un campo **EXP** (Esperimental) de 3 bits que identifican la clase de servicio.
- Un campo **S** (Bottom of Stack) de un bit que sirve para apilar etiquetas de forma jerárquica. Si el valor de este bit es 1, significa que solo tienen una etiqueta almacenada y si el valor es 0 significa que hay más de una, es decir, hay otra etiqueta después de ella.

- Un campo **TTL** (Time to Live) el cual tienen la misma función en detección de loops que el campo TTL de IP.

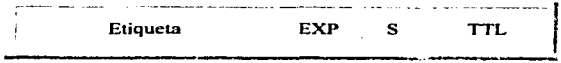


Figura 3.6. Encabezado MPLS

El bit bottom of stack implementa un depósito de etiquetas MPLS, el cual está definido como una combinación de una o más etiquetas adjuntas a un solo paquete. El enrutamiento de direcciones IP simple no usa más de una etiqueta, pero otras aplicaciones, incluyendo las Redes Privadas Virtuales (VPN's) basadas en MPLS o ingeniería de trafico, realmente hacen uso de esta capacidad.

Como el encabezado de etiquetas MPLS es insertado entre el encabezado de capa 2 y el encabezado de capa 3, el enrutador transmisor debe indicarle al enrutador receptor que el paquete que está siendo transmitido no es un datagrama IP puro sino un paquete etiquetado, es decir, un paquete MPLS. Para facilitar esto, nuevos valores de protocolo fueron definidos para portarse sobre el encabezado de capa 2 en el campo de Tipo de protocolo y así poder identificar un paquete manejando protocolo MPLS.

Para ambientes LAN, los paquetes etiquetados portando paquetes unicast (un solo destino) y paquetes multicast (varios destinos) de capa 3, usan un valor en el campo de Tipo de 8847 y 8848. Estos valores pueden ser usados directamente en medios Ethernet, incluyendo Fast Ethernet y Gigabit Ethernet.

En el ejemplo 3.1 se muestra un paquete MPLS visto desde un analizador de protocolos donde se puede observar el valor del campo Tipo de Protocolo, en la parte donde se encuentran los valores en Hexadecimal. Además, se muestra el encabezado y el valor de cada uno de sus campos MPLS.

```
Record #1 (From Hub To Node) Captured on 12.06.02 at 15:01:18.564962000 Length = 66
Runtime Frame# 1

---------- ETHER Header ----------
ETHER: Destination: 00-60-3E-33-32-60
ETHER: Source: 00-03-31-B2-70-68
ETHER: Protocol: MPLS_UCAST
ETHER: FCS: 3D7D37CC

---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x000131FC
MPLS:    0000 0000 0000 0001 0011 .... .... .... Label = 19
MPLS:    .... .... .... .... .... 000. .... .... Experimental Use = 0
MPLS:    .... .... .... .... .... ...1 .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:    .... .... .... .... .... .... 1111 1100 Time To Live = 252

---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:    0000 00.. DS Codepoint = Default PHB (0)
IP:    .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:    .0.. .... .... .... Don't Fragment Bit = FALSE
IP:    ..0. .... .... .... More Fragments Bit = FALSE
IP:    ...0 0000 0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 2F73
IP: Source address = 192.169.8.5
IP: Destination address = 192.169.4.1

---------- TCP Header ----------
TCP: Source port = 62484
TCP: Destination port = telnet (23)
TCP: Sequence number = 347664028
TCP: Ack number = 0
TCP: Data offset = 24
TCP: Flags = 0x02
TCP:    ..0. .... URGENT Flag = FALSE
TCP:    ...0 .... ACK Flag = FALSE
TCP:    .... 0... PUSH Flag = FALSE
TCP:    .... .0.. RST Flag = FALSE
TCP:    .... ..1. SYN Flag = TRUE
TCP:    .... ...0 FIN Flag = FALSE
TCP: Window = 2144
TCP: Checksum = F187
TCP: Urgent pointer = 00000000
TCP: Options = (mss 536)

Record #1 (From Hub To Node) Captured on 12.06.02 at 15:01:18.564962000 Length = 66

00 60 3e 33 32 60 00 03   31 b2 70 68 88 47 00 01   .'>32'.. 1.ph.G..
31 fc 45 00 00 2c 00 00   00 00 fc 06 2f 73 c0 a9   1.E.,... ..../s..
08 05 c0 a9 04 01 f4 14   00 17 cf 39 51 1c 00 00   ........ ...9Q...
00 00 60 02 08 60 f1 87   00 00 02 04 02 18 3d 7d   ..'..'.. ......=}
37 cc                                               7.
```

Ejemplo. 3.1 Trama completa de un paquete con encabezado MPLS

3.3. Actividades de un Enrutador Conmutador de Etiquetas (LSR)

Un enrutador operando como LSR-MPLS puede desempeñar varias acciones sobre un paquete etiquetado, las cuales se describen a continuación:

- **Pop Tag** (quitar etiqueta). Remueve la etiqueta de arriba en el conjunto de etiquetas del paquete y propaga el paquete restante ya sea como paquete etiquetado (si el bit Bottom stack es cero) o como un paquete IP no etiquetado (si el bit Bottom of stack es 1 o si el campo Tag Stack en la tabla LFIB está vacío).
- **Swap tag** (cambiar etiqueta). Remplaza la primera etiqueta de la pila de etiquetas con otro valor.
- **Push Tag** (desplazar etiqueta). Remplaza la etiqueta de arriba en la pila de etiquetas por un conjunto de etiquetas.
- **Aggregate** (agregado). Remueve la etiqueta de arriba del conjunto de etiquetas y hace un chequeo de capa 3, para enrutar el paquete IP. La etiqueta removida debe ser la última del paquete, de otra manera el paquete será descartado.
- **Untag** (quitar etiquetado). Remueve la primera etiqueta de la pila de etiquetas y envía el paquete IP existente a la dirección IP destino. La etiqueta removida es la única etiqueta en el paquete; de otra forma el paquete es descartado.

3.4 Conmutación de Etiquetas

Sin tomar en cuenta si el paquete etiquetado contienen una sola etiqueta o varias etiquetas (label stack), la conmutación de etiquetas se realiza en la misma forma. En ambos casos el enrutador conmuta el paquete actuando solo sobre la primera etiqueta de la pila de etiquetas (label stack), ignorando las otras etiquetas contenidas en el paquete.

Cada enrutador le asignara a cada dirección IP una etiqueta local; la cual anunciará a cada uno de sus vecinos, para que los vecinos con esta información sepan que esta etiqueta es la que el enrutador desea se le imponga al paquete que va dirigido a esa dirección, es decir, como etiqueta de salida cuando se dirigen hacia éstos. Así sucede en cada nodo hasta el paquete alcance su destino.

Por ejemplo en la figura 3.7, los enrutadores Toluca y Cd. Juárez soportan Redes Privadas Virtuales y están de acuerdo en asignar una etiqueta con valor de 28 a una red 10.20.0.0/16 la cual es alcanzable a través del enrutador en Cd. Juárez. Los enrutadores del Core (núcleo) como Monterrey y DF no necesitan estar consientes de esto.

El enrutador Cd.juarez para enviar paquetes a ese destino construye un pila de etiquetas. La etiqueta del fondo en la pila es la etiqueta 28, que en común acuerdo asignaron los enrutadores y la primera etiqueta es la etiqueta que el enrutador DF asigno a la dirección del enrutador Cd. Juárez. Cuando la red propaga el paquete, la primera etiqueta es conmutada exactamente como se muestra en la figura, en donde cada enrutador indica que etiqueta de salida desea que se le ponga al paquete para alcanzarlo y la segunda etiqueta en la pila llega hasta el enrutador Cd. Juárez intacta, es decir, no cambia durante el trayecto.

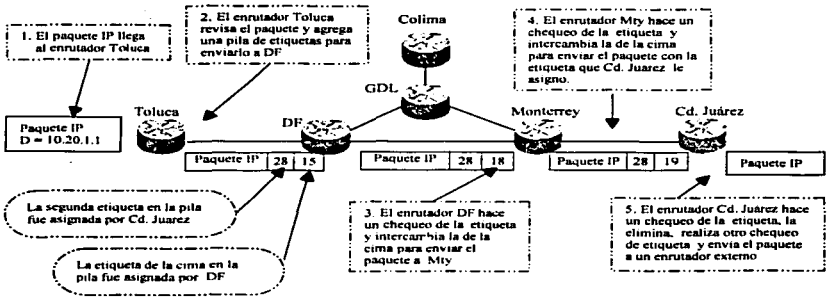


Figura 3.7 Conmutación de etiquetas utilizando la pila de etiquetas (label stack)

3.5. Propagación de etiquetas impuestas. Protocolo TDP

Sabiendo que las etiquetas deben ser propagadas entre los LSRs a través de las interfaces, es necesario conocer que lo que hace posible esta propagación.

Los software de IOS (Sistema Operativo) de Cisco, implementan dos protocolos de imposición de etiquetas que pueden ser usados para asociar subredes IP con etiquetas MPLS (o lo que es lo mismo asociación de FEC a etiquetas) para el propósito de enrutamiento de destinos unicast. Los cuales son:

- Tag Distribution Protocol, **TDP** (Protocolo de Distribución de Etiquetas). El cual es un protocolo propietario de cisco.
- Label Distribution Protocol, **LDP** (Protocolo de Distribución de Etiquetas). El cual es un estándar de IETF.

Funcionalmente los protocolos son equivalentes y pueden ser usados en cualquier interfaz dentro de la red.

3.5.1. Establecimiento de sesiones LDP/TDP

Cuando habilitas MPLS sobre la primer interfaz, el proceso TDP/LDP se inicia y se crea la estructura de la base de información de etiquetas (TIB/LIB). El enrutador también trata de descubrir otros LSRs sobre las interfaces que están corriendo MPLS a través de paquetes que llevan mensajes Hello (Hola) TDP, utilizando para enviar los mensajes el protocolo de transporte UDP(User Datagram Protocol). Los mensajes Hello son enviados como broadcast (a todos los vecinos) o como paquetes multicast (solo a algunos), descubriendo al vecino LSR automáticamente.

Después de que el proceso Hello de TDP descubre un vecino, una sesión TDP se establece con el vecino. Las sesiones TDP corren sobre el puerto 711 de TCP (Transfer Control Protocol), mientras que LDP usa el puerto TCP 646. TCP es usado como protocolo de transporte para asegurar una entrega de información confiable.

Después de que una sesión se establece, se monitorea constantemente con paquetes keepalive para asegurar que siga en operación.

Este monitoreo identifica el enrutador local y el remoto (vecino), la dirección IP y el puerto TCP en el cual la sesión es establecida, el tiempo que lleva arriba la conexión y la interfaz a través de la cual descubrió al vecino.

Distribución de Etiquetas

Tan pronto como la base de información de etiquetas (TIB/LIB) es creada, una etiqueta es asignada a cada FEC. En enrutamiento basado en destinos unicast, el FEC es equivalente a un prefijo IP de la tabla de enrutamiento. De esta forma una etiqueta es asignada a cada prefijo IP contenido dentro de la tabla de enrutamiento y la combinación de los dos es almacenada en la tabla TIB/LIB.

La base de información de etiquetas (TIB/LIB) se mantienen siempre sincronizada a la tabla de enrutamiento, tan pronto como la nueva ruta aparece en la tabla de enrutamiento, una etiqueta nueva es asignada y ligada a la nueva ruta.

Los LSR asignan una etiqueta a cada prefijo IP en su tabla de enrutamiento tan pronto como éste aparece en la tabla de enrutamiento. El propósito de la asignación es para que otros LSR utilicen la etiqueta para enviar paquetes etiquetados hacia el LSR que impuso la etiqueta para esa dirección IP.

Este método de asignación y distribución de etiqueta es conocido como. *control independiente* de asignación de etiquetas con distribución de etiquetas *no solicitada hacia atrás*. El cual se resume en los siguientes puntos.

- La asignación de etiquetas en enrutadores se hace sin tomar en cuenta si el enrutador ha recibido ya una etiqueta para el mismo prefijo de su enrutador next-hop (next-hop se refiere al próximo salto para el paquete con un destino en particular) o no. Por lo que la asignación de la etiqueta es llamada control independiente.

- El método de distribución es *no solicitado* porque el LSR asigna la etiqueta y anuncia la combinación prefijo-etiqueta a su siguiente vecino sin importar si este lo necesita la etiqueta o no. La distribución sobre demanda es otra posibilidad. Un LSR asigna solo una etiqueta a un prefijo IP y lo distribuye a su próximo vecino cuando éste lo solicita.

Todas las etiquetas son anunciadas inmediatamente a los otros enrutadores a través de sesiones TDP. Los LSRs adyacentes reciben la combinación prefijo-etiqueta, la almacenan en su tabla TIB/LIB, y la usan para su tabla FIB o TFIB/LFIB, solo si la combinación fue recibida del vecino que está en dirección a ese destino, el cual se convierte en ese momento en el next- hop para esa FEC particular. Este método de almacenamiento es conocido como *modo de retención liberal*.

3.5.2. Convergencia en la red MPLS

Un aspecto importante en el diseño de una red MPLS es el tiempo de convergencia de la red. El tiempo de convergencia se refiere al tiempo que tarda la red en propagar alguna falla en los enlaces o algún cambio en la estructura de la red; por lo que es de suma importancia que este tiempo de propagación sea muy pequeño. Algunas aplicaciones MPLS, como MPLS/VPN, no trabajan correctamente a menos que el paquete etiquetado sea enviado por todo el camino hasta el LSR de egreso. En estas aplicaciones el tiempo de convergencia puede verse incrementado por el retardo en la propagación de etiquetas.

En una red MPLS, que usa el modo de retención liberal en combinación con el control de etiquetas independiente y distribución de etiquetas no solicitado, explicados anteriormente, minimiza el retardo de convergencia TDP/LDP. Cada enrutador usando modo de retención liberal tiene todas las etiquetas asignadas por sus vecinos TDP, para sus prefijos IP, de manera que siempre podrá encontrar una etiqueta de salida adecuada, siguiendo la convergencia de la tabla de enrutamiento sin tener que preguntar a su nuevo enrutador Next-Hop por la etiqueta asignada.

3.6. Eliminación de la etiqueta en el penúltimo salto (Penultimate hop Popping).

En una red MPLS un LSR de egreso puede tener que realizar dos chequeos sobre un paquete recibido desde un vecino MPLS y con destino a una red fuera del dominio MPLS. Éste debe examinar la pila de etiquetas en el encabezado y debe realizar un chequeo de la etiqueta solo para darse cuenta que la etiqueta tiene que ser removida y inspeccionado el paquete IP resultante; por lo tanto un análisis de capa 3 debe ser realizado sobre el paquete IP antes de poder ser enviado a su destino final. Ver figura 3.8.

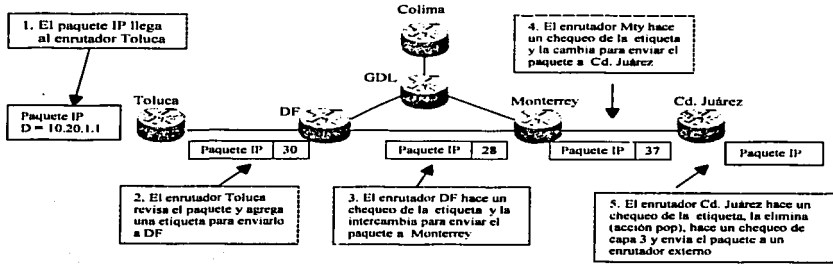


Figura 3.8 Doble chequeo de encabezados en CD. Juárez

Como el doble chequeo en el enrutador puede reducir el desempeño de ese nodo. se introdujo a la arquitectura MPLS lo que conocemos como Penultimate hop Popping. Este concepto le permite a un enrutador solicitar la eliminación de la etiqueta de un paquete, al enrutador que se encuentra justo antes que éste, consiguiendo de esta forma que solo le envíe un paquete IP puro. Ver figura 3.9.

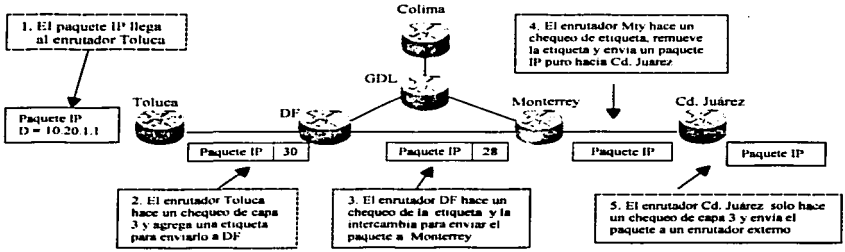


Figura 3.9 Anulación de etiqueta en el penúltimo salto

Penultimate hop Popping es usado solamente para las subredes conectadas directamente o rutas agregadas. En el caso de una interfaz directamente conectada, es necesario un análisis de capa 3 para obtener la información correcta del próximo salto para un paquete que será enviado a un destino conectado directamente. En el caso de que el prefijo sea un agregado, también se necesitara el análisis de capa 3 para encontrar una ruta más específica que se utilizara para encaminar al paquete hacia el destino correcto.

El Penultimate hop Popping es solicitado a través de TDP, colocando una etiqueta de valor especial, 1 para TDP, que también es conocida como valor implicit-null (valor nulo).

Cuando un LSR de egreso solicita un Penultimate hop Popping, para un prefijo, la entrada local LIB en el LSR de egreso y la entrada LIB remota en el LSR de adelante indican el valor imp-null y la entrada TFIB en el penúltimo LSR indica la operación pop tag (eliminación de etiqueta).

3.7. Interacción de MPLS con enlaces Ethernet

La implementación de enlaces Ethernet (Ethernet, Fast Ethernet o Gigabit Ethernet) adentro de la topología es también un tema importante ya que MPLS requerirá de un espacio disponible al encapsularse dentro de la trama.
Cada tipo de medio tiene un tamaño máximo de trama de 1518 octetos, no incluyendo el preámbulo, con una carga de datos que van desde los 46 bytes hasta los 1500 bytes.

Como ya se menciono el uso de MPLS dentro de la red incrementa el tamaño del paquete, debido a la agregación de etiquetas dentro del campo de etiquetas. Como el encabezado de la etiqueta es de 4 bytes, esto significa que si un paquete de 1500 bytes de carga se recibe y un encabezado de etiqueta se agrega, entonces se necesitaría enviar la trama con una carga de 1504 bytes. Lo cual puede causar un problema porque el MTU sobre estos enlaces es menor que el paquete que se quiere enviar.

La mayoría de los nodos IP de hoy en día soportan el uso de un mecanismo para descubrir el MTU en la trayectoria, el cual está documentado en el RFC 1191, este mecanismo le permite a un nodo IP descubrir dinámicamente el tamaño MTU máximo disponible a lo largo de la trayectoria, desde la fuente al destino.

La idea básica del mecanismo es que la fuente asume que la trayectoria tendrá el valor MTU que corresponde a su primer salto y así envía todos los paquetes sobre esa trayectoria con el bit de no fragmentar, activo. No se envía ningún datagrama mayor que el MTU del primer salto. Los nodos que no utilicen este procedimiento no deben mandar paquetes que sean mayores a 576 bytes.

Cuando un enrutador recibe un paquete que es mayor que el MTU de la interfaz de salida dirigida al destino contenido dentro del paquete, con el bit de no fragmentar activado, el enrutador debe enviar un mensaje ICMP con destino no alcanzable y un código indicando la necesidad de fragmentación. El proceso para descubrir el MTU confía en la información contenida en este mensaje para determinar el tamaño de paquete máximo que puede ser enviado a través de la trayectoria para un destino particular. En la figura 3.10 se puede observar un ejemplo.

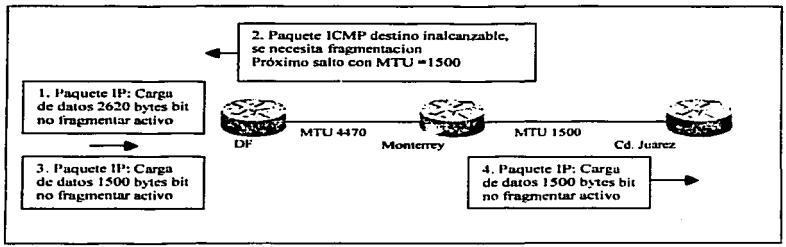


Figura 3.10 Mecanismo para descubrir el MTU en la trayectoria.

Cuando se implementa este procedimiento, los paquetes pueden ser enviados satisfactoriamente a través de la estructura MPLS sin fragmentación. Sin embargo, cada LSR puede fragmentar paquetes etiquetados o no etiquetados si ellos son mayores que el MTU de salida, siempre y cuando el bit de no fragmentar no este activo.

En la vida real podemos encontrar equipos que no usan este método para descubrir el MTU y además mandan paquetes de más 576 bytes. Incluso algunos Firewalls tiran los mensajes ICMP de destino inalcanzable, lo cual interrumpe el proceso de descubrimiento MTU. Por tales causas se vio la necesidad de implementar un mecanismo que permitiera a las tramas con una carga de más de 1500 bytes poder ser enviadas a través de la red.

La compañía Cisco System introdujo una solución que permitió a los puertos Ethernet sobre un enrutador soportar paquetes MPLS que tienen una carga de datos mayor de 1500 bytes. Lo cual se logro incrementando el MTU de los puertos Ethernet a 1526 bytes, que constituye el tamaño estándar máximo de trama Ethernet de 1518 más 8 octetos para dos niveles de etiquetas MPLS. Esta cantidad de etiquetas es el adecuado para estos tiempos y soporta la introducción de MPLS y VPN habilitadas mediante MPLS.

Cada LSR debe soportar la configuración de un parámetro conocido como, "Máximo Tamaño Inicial de Datagrama IP Etiquetado". Este parámetro es usado en el ingreso del dominio MPLS, así que el paquete puede ser fragmentado en la orilla de la red si éste es mayor que el tamaño MTU configurado. El tamaño de MTU necesita ser configurado en todos los enlaces de la estructura de la red. La ventaja de esto es que el paquete es fragmentado antes de entrar dentro del dominio de la red MPLS y no requiere más fragmentaciones dentro de la estructura MPLS.

3.8. Como prevenir y detectar un Loop MPLS

Lo primero que hay que entender es el termino Loop, se puede decir que un Loop es un ciclo repetitivo en el que un paquete se ve envuelto, es decir, el paquete cae en un circulo vicioso y no sale de éste. Para que se entienda mejor este término Loop pongamos un ejemplo: si un paquete de información sale del DF con destino a Monterrey pasando antes por Querétaro y Durango, tomando un camino incorrecto puede pasarse mucho tiempo dando saltos de nodo en nodo sin poder alcanzar su destino e incluso regresar al lugar que lo originó, a esta serie de saltos sin razón lo conocemos como Loop.

Al momento de implementar una arquitectura MPLS, un tema importante que debe considerarse es la capacidad para detectar y prevenir el transporte de Loops dentro de la red. Como ya se mencionó el transporte de un Loop en una red IP es el proceso por el cual un enrutador envía un paquete con un destino en particular por un camino incorrecto, lo cual puede ocurrir si un enrutador no está bien configurado o no está configurado, de manera que el primer enrutador está apuntando a otro que no es el próximo salto para ese destino en particular. Ahora que sabemos que es un Loop debemos entender como detectar y lidiar con el transporte de loops.

Como sabemos las etiquetas son asignadas a un FEC particular utilizando el modo de control independiente y al utilizar esta asignación de etiquetas se establece un camino conmutado por etiqueta (LSP) a través de la red, nos basaremos en estos términos para explicar como cada LSR detecta y previene un Loop.

En una red IP tradicional la detección de Loop puede lograrse examinando el campo TTL (tiempo de vida) de un paquete entrante, al usar este campo cada enrutador decrementa su valor en 1 cada vez que realiza un salto; si el campo alcanza el valor 0, el paquete se tira y el Loop se elimina. De esta manera se evita que el paquete ande saltando de un lado a otro por toda la red de manera indefinida.

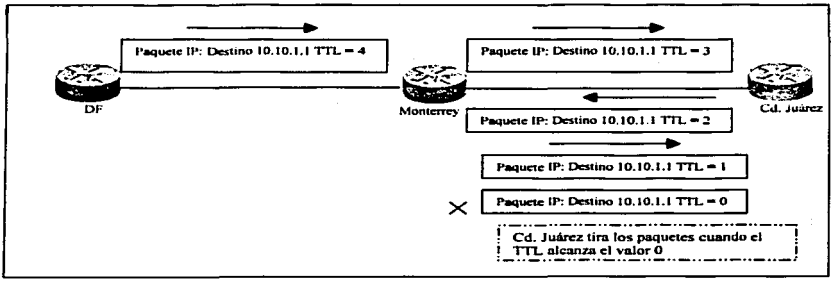


Figura 3.11 detección de Loop usando el campo TTL

El mismo mecanismo se usa dentro del plano de datos en una implementación MPLS. Cada LSR a lo largo de un camino conmutado (LSP), decrementa el campo TTL del encabezado MPLS al momento que éste envía una trama MPLS y tira cualquier paquete que alcanza un TTL con valor 0. De alguna manera podemos entender que si un paquete alcanzo un valor de 0 sin alcanzar su destino es porque ya se había creado un Loop.

La detección de Loops obviamente es muy necesaria, sin embargo, también es necesario que cada LSR sea capaz de prevenirlos antes de que ocurran. La prevención debe ser lograda dentro del plano de control porque es donde los LSPs son creados.

En una red IP estándar, la prevención de Loops es trabajo del protocolo de enrutamiento. Como todos los LSRs usan el mismo protocolo de enrutamiento para llenar sus tablas, esta información también es usada en MPLS para crear los LSPs dentro de la red. Por esta razón MPLS confía en los protocolos de enrutamiento para asegurarse de que la información de las tablas de enrutamiento están libres de Loops, de la misma manera que un enrutador de una red IP.

CAPÍTULO IV

REDES PRIVADAS VIRTUALES (VPNS), BASADAS EN MPLS

4.1 Introducción a las redes privadas virtuales (VPN's)

El mundo ha cambiado en las ultimas décadas, muchos negocios han tenido que pasar de los intereses locales o regionales a pensar en un mercado global, es decir han obtenido la facilidad para expandirse alrededor de su país o incluso alrededor del mundo. Pero hay una cosa que todas las compañías necesitan; una forma para mantener la comunicación rápida, segura y confiable sin importar en donde estén localizadas sus oficinas.

Las redes WAN tradicionales ofrecen líneas alquiladas para garantizar la confiabilidad, el desempeño y la seguridad en las comunicaciones; las cuales son obvias ventajas sobre una red pública como es el Internet, pero también es cierto que mantener una WAN utilizando líneas alquiladas resulta bastante caro y mientras más lejos estén las oficinas más incrementará el costo. Esta es una de las razones por las que muchas compañías están creando sus propias redes privadas virtuales (VPN's) para acomodar las necesidades de sus empleados remotos y oficinas distantes e incluso ofrecer el servicio VPN a distintos clientes a un menor costo.

Una VPN es una red en la cual dos sitos pueden comunicarse sobre la red del proveedor de una manera privada, esto significa que ningún sitio fuera de la VPN puede interceptar sus paquetes o inyectar nuevos paquetes a la red privada. La red del proveedor es configurada de tal forma que solo los paquetes de la VPN pueden ser transmitidos a través de esa VPN, lo que quiere decir que ningún dato entra o sale de la VPN a menos que esto sea especificado a través de la configuración. El objetivo del servicio VPN es proporcionar la conectividad de los clientes a través de una infraestructura compartida, disfrutando de las mismas políticas que en una red privada.

Los siguientes son los atributos esenciales de una red VPN.

Seguridad

Es importante que las redes protejan los datos delicados de manera que permanezcan confidenciales, el mecanismo de seguridad utilizado en las VPN's basadas en MPLS es la separación del tráfico, MPLS mantienen separado el tráfico que cada VPN maneja, utilizando por cada VPN, un identificador de rutas conocido como Route Distinguisher. La separación del trafico tambien brinda la oportunidad de utilizar, el mismo direccionamiento IP en todas las VPN's sin tener el problema de direcciones duplicadas, ya que cada una de las VPN's que comparten la infraestructura física, está aislada de las otras.

Confiabilidad y Redundancia

Las VPN's también son capaces de entregar un servicio con alta disponibilidad, según lo requiera el cliente. Una combinación de confiabilidad y redundancia es la llave para mantener la continuidad del negocio y recuperarse de fallas, el mecanismo para lograr esto es la instalación de sitios de respaldo.

Escalabilidad

Las VPN's deben adaptarse a las necesidades de conectividad del cliente, el cual puede empezar desde una pequeña oficina y crecer hasta expandirse al rededor del país. Por lo tanto el proveedor de servicio debe proporcionar e implementar rápidamente las solicitudes del cliente para extender sus servicios. El proveedor de servicio requiere por lo tanto de la habilidad para escalar la VPN y así acomodar un crecimiento no planeado y manejar los cambios basados en la demanda del cliente.

4.2 Introducción a la arquitectura VPN-MPLS

Con la introducción de MPLS el cual combina los beneficios de la conmutación de capa 2 con el enrutamiento de capa 3, es posible construir una tecnología que combine algunos beneficios como son; la seguridad, el aislamiento y el enrutamiento simplificado.

Esta nueva tecnología que mencionamos es la que se conoce como VPN-MPLS (Red privada virtual basada en MPLS). Esta tecnología brinda un enrutamiento simple para el cliente, además hace posible la implementación de varias topologías que antes no eran posibles de implementar. MPLS también agrega los beneficios del método orientado a conexión, al enrutamiento IP.

Una red VPN basada en MPLS es una estructura de red IP entregando servicios de red privada sobre una estructura compartida, la cual utiliza la conmutación por etiquetas para enrutar los paquetes de datos a su destino. Se conocen como redes privadas virtuales (VPNs); porque aunque comparten una sola red física, son totalmente independientes unas de otras. Esta red física puede manejar una gran cantidad de VPNs diferentes, aún cuando éstas estén manejando los mismos prefijos de direcciones, es decir, el mismo segmento de direcciones IP.

4.2.1 Estructura básica de una topología VPN-MPLS y su terminología

En la figura 4.1, se muestra una topología típica de una red VPN-MPLS, utilizando la tecnología de red IP. Como se puede observar en esta figura, la solución para la implem ntación de escenario relacionado con VPN's tiene varios componentes los cuales describimos a continuación.

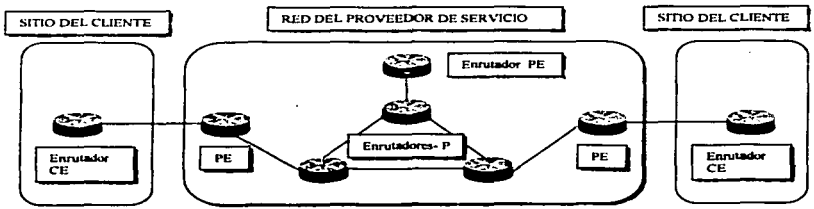


Figura 4.1. Estructura típica de una red VPN

- El Proveedor de Servicio es la organización que es propietaria de la infraestructura es decir, los equipos y los medios de transmisión, que proporcionan una simulación de líneas privadas a sus clientes. En este escenario de VPN, el Proveedor de Servicio ofrece a los clientes servicios de Redes Privadas Virtuales.

- El cliente se conecta a la red del proveedor de servicio a través de un equipo conocido como **Customer Edge (CE)** o Frontera del Cliente, este equipo proporciona la conectividad final del cliente normalmente es un enrutador.

- El CE se conecta a través de un medio de transmisión como puede ser una línea alquilada, hacia un equipo en la frontera del proveedor de servicio el cual también es un enrutador. El equipo en la frontera del proveedor se conoce como **Provider Edge (PE)** o Frontera del Proveedor.

- El proveedor de servicio normalmente tienen equipo adicional en el núcleo (core) de su red, estos equipos son conocidos como **Enrutadores - P.**

- Un equipo conocido como **Route Reflector (RR),** usado para reflejar las rutas BGP entre los enrutadores Provider Edge (PE).

4.3 Conceptos básicos de una red VPN/MPLS

Para explicar los conceptos de la arquitectura VPN/MPLS usaremos el esquema de red que se muestra en la figura 4.2. En este diseño tendremos a un proveedor de servicio, el cual ofrece servicios de VPN's basados en la arquitectura VPN/MPLS. El proveedor tiene dos clientes VPN, uno que conoceremos como autos y el otro como planetas. Los dos clientes tienen sucursales en Guadalajara (gdl), Monterrey (mty) y México DF (mex).

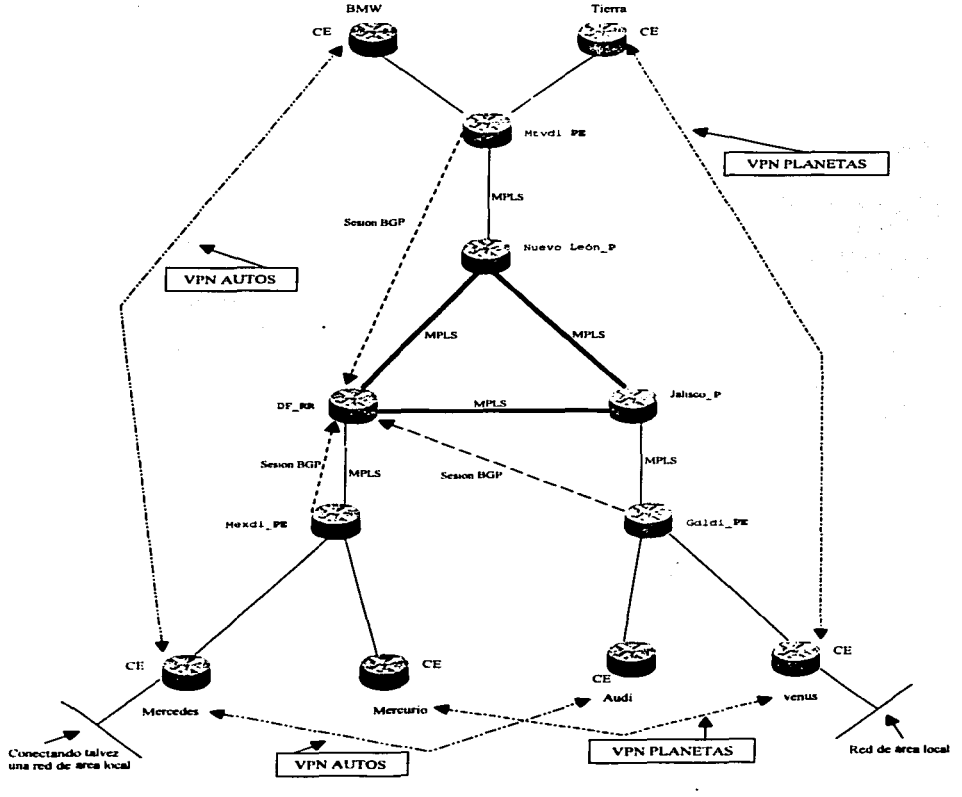


Figura 4.2. Topología de una red VPN-MPLS con sitios en Mty, DF y Gdl.

De acuerdo a la terminología utilizada para implementar una red VPN, los enrutadores que encontramos en la figura tienen una función específica y se menciona a continuación:

- Los enrutadores **DF**, **Jalisco** y **Nuevo León** no tienen ninguna conexión de clientes, por lo tanto son enrutadores del proveedor (**P**). Aunque tomaremos prestado el enrutador DF para que sea también el **RR**.
- Los enrutadores **mexdi**, **gdldi** y **mtydi** enlazan la red con sus clientes. Por lo tanto ellos son enrutadores en la frontera del proveedor (**PE**).
- Los enrutadores del cliente Planetas es decir, **mercurio**, **tierra** y **venus** así como los del cliente Autos; **mercedes**, **bmw** y **audi** que están conectados a la red del proveedor son enrutadores en la frontera del cliente (**CE**)

Suponiendo que los dos clientes utilizaran el mismo espacio de direcciones, por ejemplo la red 192.169.0.0, se ocasionaría lo que en las redes IP tradicionales se conoce como traslape de direcciones, es decir, tendríamos direcciones duplicadas al momento de integrarlos a la red, desde el punto de vista de una red VPN el hecho de que existan direcciones duplicadas con diferentes clientes no tiene la misma repercusión que para una red tradicional, veamos porque en el siguiente punto.

4.3.1 Tablas de transporte y enrutamiento VPN

El traslape de direcciones, resultado de utilizar direcciones IP privadas, iguales, dentro de las redes de los clientes, es uno de los mayores obstáculos en la implementación de una red.

A este problema VPN/MPLS proporciona una solución; la cual indica que cada VPN dentro del enrutador debe tener su propia tabla de transporte y de enrutamiento, así que a cualquier cliente o sitio que pertenezca a una VPN se le brinda acceso sólo al conjunto de rutas que estén contenidas dentro de su tabla. De esta manera los enrutadores PE en una red VPN/MPLS tendrán una tabla de enrutamiento por cada VPN para alcanzar a todos los vecinos que pertenezcan a esa VPN y una tabla de enrutamiento global que será usada para alcanzar a otros destinos dentro de la red del proveedor que no pertenecen a una VPN, o también para alcanzar destinos externos a la red (como el Internet). Esto también indica que en un solo enrutador físico se crea un enrutador virtual, por cada VPN. Ver figura 4.3.

El concepto de enrutador virtual le permite al cliente usar direcciones IP privadas o direcciones IP globales en cada VPN, por lo tanto las VPN pueden usar el mismo espacio de direcciones sin ningún problema a menos que, dos VPN's utilizando el mismo direccionamiento quisieran comunicarse.

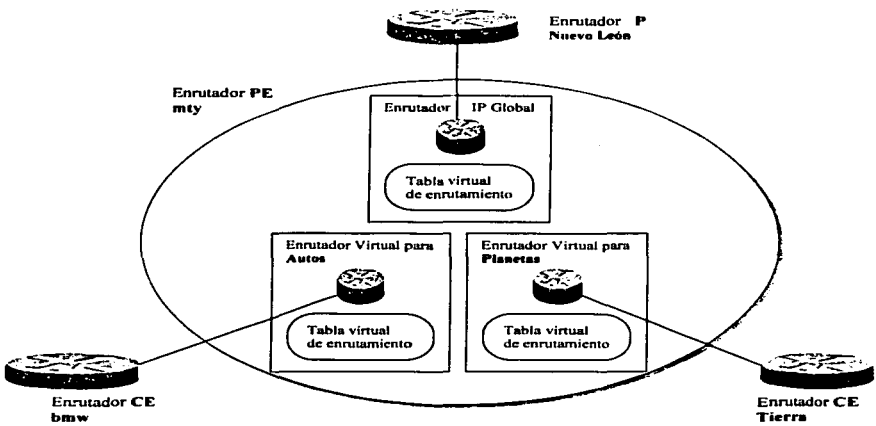


Figura 4.3. Enrutadores virtuales creados dentro de un enrutador físico PE

Al hablar de VPN no solo encontramos las tablas de enrutamiento virtuales asociadas a los enrutadores virtuales, sino que existen otras estructuras como las que mencionamos a continuación.

- Una tabla de transporte que está basada en la tecnología CEF (Cisco Express Forwarding) y maneja todas las rutas que existen.
- Un conjunto de interfaces que usan la tabla de trasporte.

- Reglas que controlan la importación y exportación de rutas dentro y fuera de la tabla de enrutamiento VPN, estas rutas fueron introducidas para soportar el traslape de direcciones de las VPN's.
- Un conjunto de protocolos de enrutamiento, incluyendo rutas estáticas, los cuales inyectan información dentro de la tabla de transporte VPN.

La utilidad y operación de estas estructuras serán detalladas poco a poco.

Otro concepto importante en la arquitectura es el que se conoce como **VRF** (VPN routing and forwarding instance) Instancia de Transporte y Enrutamiento VPN, termino derivado de la combinación de la tabla de enrutamiento IP-VPN y la tabla de transporte IP-VPN.

En las redes tradicionales no existe diferencia entre estas dos tablas, pero en el ambiente MPLS aparece una sola diferencia y es que la tabla de transporte IP también contiene información de encapsulado MPLS. Esta tabla de trasporte utiliza la tecnología CEF que logra que la conmutación de paquetes sea más rápida.

Una VPN es asociada a una VRF en el enrutador PE, aunque también puede existir el caso en donde un cliente necesite conectarse con otro cliente dentro de una VPN diferente, si esto ocurre se requerirá más de una VRF por VPN. Tambien es importante recordar que cada interfaz que conecte a un cliente VPN desde el enrutador PE al CE debe ser insertada dentro de la VRF que le corresponde a ese cliente.

Para no confundir más adelante lo que es una VRF y una VPN los dos puntos siguientes definen claramente estos dos conceptos:

- La VRF es simplemente una colección de rutas que deben estar disponibles para un sitio particular o un conjunto de sitios conectados a un enrutador PE. Estas rutas pueden pertenecer a más de una VPN.

- Una VPN se forma esencialmente de un conjunto de sitios que comparten información de enrutamiento común, lo cual significa también que un sitio puede pertenecer a más de una VPN si este sostiene rutas de VPN's separadas. Una VPN tambien puede ser un grupo de usuarios cercanos.

La relación entre las VPN's, sitios y VRF's puede ser resumida en la siguiente regla, la cual tambien puede ser usada como la definición básica de una VRF.

Todos los sitios que comparten la misma información de enrutamiento, que tienen permitido comunicarse unos con otros y que están conectados al enrutador PE, pueden ser colocados en una VRF en común.

Si seguimos esta regla para la red que propusimos, tendremos que establecer dos VRF ya que tenemos dos VPN y debemos asignar los clientes que corresponden a cada una, como lo indica en la tabla 4.1.

Enrutador PE	VRF	Sitios dentro de la VRF	VPN a la que pertenecen	Route Distinguisher asociado
Mexdi Gdldi Mtydi	Autos	Mercedes Bmw Audi	Autos	192.168.0.0:1
	Planetas	Mercurio Venus Planetas	Planetas	192.169.0.0:1

Tabla 4.1 VRF's creadas en los enrutadores PE con su correspondiente Route Distinguisher

En este momento surge la duda de como le hace el enrutador para enviar los paquetes de un sitio hacia su destino sin invadir otra VPN, es decir como sabe que rutas necesitan ser insertadas dentro de cada VRF.

Para resolver esta duda necesitamos introducir el concepto Route Target, el cual es de alguna manera un identificador de VPN. Cada ruta VPN es etiquetada con este Route Target cuando ésta es exportada dentro de la VRF, con el fin de ser enviada hacia otras VRF's. Cada VRF debe especificar un Route Target para importar y exportar sus rutas dentro y fuera de esta VRF, de esta forma el enrutador PE identifica que ruta pertenece a que VRF y por lo tanto a que VPN con solo checar el Route Target que la ruta trae consigo.

En nuestro ejemplo tenemos dos VPN, por lo tanto requerimos de dos Route Targets como se observa en la tabla anterior.

Tanto las VRF como los Route Target deben ser definidos en los enrutadores PE de la red, retomando de nuevo el esquema de red de la figura 4.2, observamos que tenemos tres PE's los cuales necesitan establecer comunicación.

En este momento es necesario explicar dos puntos muy importante que son los siguientes:

- El intercambio de información y rutas de los clientes de la VPN entre enrutadores PE.
- La forma en la que los enrutadores PE transportan los paquetes originados en los clientes de la VPN.

4.3.2 Intercambio de rutas VPN entre enrutadores PE's

Los PE's corren sobre la red un único protocolo de enrutamiento para intercambiar entre ellos todas las rutas VPN, este protocolo es el Multiprotocolo BGP (MP-BGP). Para soportar el traslape de direcciones de los clientes VPN, las direcciones IP de los clientes deben ser aumentadas con información adicional para hacerlas únicas, es decir las subredes IP anunciadas por el enrutador CE hacia el enrutador PE deben ser aumentadas por un prefijo de 64 bits, a este prefijo se le conoce como **Route Distinguisher.**

Las direcciones que ahora resultan de 96 bit (64 del Route distinguisher y 32 de la dirección IP) se intercambian entre los enrutadores PE usando un atributo especial del Multiprotocolo BGP conocido como **Address-Family** (familia de direcciones). Las razones para escoger el BGP como el protocolo de enrutamiento para transportar las rutas VPN son las siguientes:

- BGP es el único protocolo de enrutamiento que puede soportar una gran cantidad de rutas y en una red VPN el número de rutas puede llegar a ser bastante grande, debido a que es proporcional al número de sitios o clientes que conecta la VPN.

- BGP, EIGRP y IS-IS son los únicos protocolos de enrutamiento que son multiprotocolo por diseño, es decir todos ellos pueden portar información de enrutamiento para varias familias de direcciones diferentes. Además BGP está diseñado para intercambiar información entre enrutadores que no están directamente conectados, característica que le permite mantener la información de enrutamiento VPN fuera de los enrutadores del core (núcleo de la red) del proveedor o sea los enrutadores P.

- BGP puede portar cualquier información adjunta a la ruta como un atributo opcional de BGP; de hecho se pueden definir atributos adicionales que serán enviados de manera transparente, sin tomarlos en cuenta, por cualquier enrutador que no entiende de ellos. Esta propiedad de BGP hace la propagación de Route Targets entre enrutadores PE realmente simple.

Como ya mencionamos el multiprotocolo BGP por diseño será un requisito en la red del proveedor, sin embargo, es importante mencionar que los clientes pueden usar cualquier protocolo de enrutamiento o incluso utilizar rutas estáticas para intercambiar información de enrutamiento con el enrutador PE. Por lo tanto queda claro que debe existir una interacción entre el protocolo de enrutamiento de cada VPN (enlace PE-CE) y el protocolo BGP. Las rutas generadas por varios protocolos de enrutamiento (entre el CE y el PE) así como las rutas estáticas configuradas, deben ser redistribuidas (insertadas) dentro de BGP. El Route Distinguisher se agrega a las direcciones de la VPN al momento de la redistribución. El Route Target de exportación tambien se le agrega a la ruta. De esta forma la información resultante de 96 bits se propaga por medio del protocolo BGP hacia los demás PE's. Ver figura 4.4.

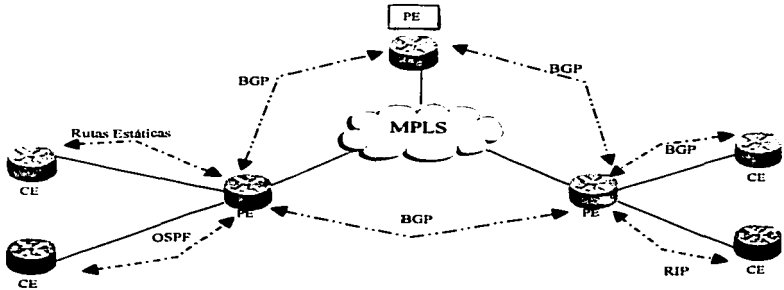


Figura 4.4 Protocolos de enrutamiento utilizados dentro de la VPN

Después de que el enrutador PE recibe las rutas BGP, inserta las rutas recibidas dentro de las diferentes tablas VRF basándose en el Route Target, el cual como mencionamos es un atributo que se le agrega a cada ruta. El Route Distinguisher es eliminado cuando la ruta es importada dentro de la VRF, quedando de nuevo una ruta IP tradicional.

BGP es una herramienta necesaria para propagar la información de los clientes entre los PE's, por lo tanto debe quedarnos claro que es necesario crear un arreglo completo de sesiones BGP entre todos los PE's para que a todos les llegue la información. Además a medida que los clientes y por lo tanto los PE aumenten, el número de sesiones entre PE a través de la red también aumentará, desde el punto de vista administrativo esto llegaría a ser una pesadilla, de manera que si consideramos la escalabilidad de nuestro diseño de red, debemos mantener en mente que el número de sesiones entre los enrutadores PE podría llegar a ser bastante grande, estamos hablando de que tendríamos que construir una por cada vecino PE, además un vecino BGP no puede soportar más de 100 sesiones.

Lo anterior significa que necesitamos emplear una técnica que acorte el número de sesiones BGP que son requeridas y además que maneje la distribución de la información de enrutamiento solo a los puntos de la red en donde es necesario. Esta técnica es conocida como jerarquía **Route reflector**

El termino Route Reflection (Reflejo de Rutas) se usa para describir la operación de un nodo BGP remoto, anunciando una ruta que fue aprendida a través de una sesión BGP hacia otro vecino BGP. El nodo remoto BGP que propaga las rutas BGP a otros vecinos en este caso los PE's, recibe el nombre de **Route Reflector (RR)** y no es más que otro enrutador que se encarga de reflejar las rutas de PE a PE, y aunque estas rutas son de la VPN el Route Reflector no presta atención a lo que está transportando es decir, la información para él es transparente.

Con la introducción de un Route Reflector, los PE's solo necesitan establecer una sola sesión con el RR y éste se encarga de pasar todas las rutas que él reciba a sus vecinos PE. Las sesiones que deben establecerse se muestran en la figura 4.2.

4.3.3. Trasporte de paquetes de la VPN.

Ya revisamos que las direcciones IP dentro de la VPN están aumentadas con un Route Distinguisher para hacer de ellas unas direcciones únicas. De esta misma forma, cuando los paquetes que se originan en la VPN viajan a través de la columna vertebral de la red del proveedor (los enrutadores P), deben de ser reconocidas de manera individual.

Para lograr manejar una dirección de 96 bits podríamos hacer dos cosas la primera sería rescribir el encabezado IP para transportar una dirección de este tamaño o se podría crear un túnel IP sobre la red VPN para transportar los paquetes sin que se tome en cuenta la información que lleva adentro, estas dos soluciones harían compleja la red por lo que se optó por utilizar la arquitectura MPLS para el transporte de paquetes a través de la red del proveedor.

Gracias a MPLS obtuvimos una tercera opción, cada paquete VPN es etiquetado por el PE de ingreso con una etiqueta MPLS exclusiva que identifica por cierto al enrutador PE de egreso y se envía a través de la red. Todos los enrutadores en la red de manera subsecuente conmutan por medio de las etiquetas sin tener que checar dentro del paquete mismo buscando una dirección IP destino.

Para explicar como se realiza este proceso utilizaremos la figura 4.5. Cada enrutador PE necesita una dirección que lo identifique exclusivamente, usualmente utilizamos una dirección IP conocida como Loopback, dirección que por diseño permanecerá siempre activa desde el momento de configurarla, esta dirección se propaga a través de la red del proveedor por medio del protocolo de enrutamiento interno, también se usa como el **atributo de BGP conocido como Next Hop** para todas las rutas VPN anunciadas por el enrutador PE. Cada enrutador P asigna una etiqueta para esa Loopback, la cual se propaga a cada uno de los vecinos. Finalmente todos los enrutadores PE reciben una etiqueta asociada con el enrutador PE de egreso a través del proceso de distribución de etiquetas MPLS, después de que el enrutador de ingreso recibió la etiqueta del PE de egreso se puede empezar el intercambio de paquetes VPN.

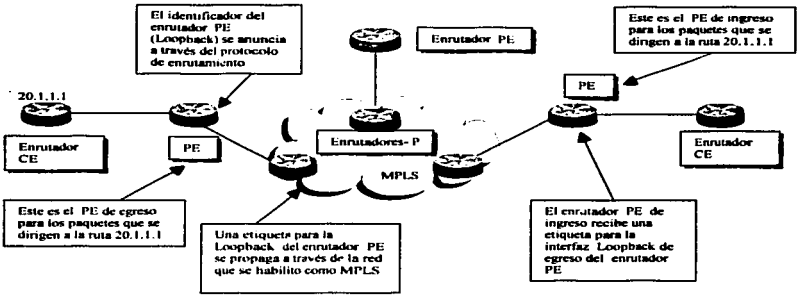


Figura 4.5. Pasos preparatorios para el transporte de paquetes

Sin embargo, cuando el PE recibe el paquete VPN no tiene la información necesaria para decidir a cual VPN está destinado el paquete. Para hacer exclusiva la comunicación entre los sitios de la VPN se introdujo un segundo conjunto de etiquetas que identifican a la VPN. Por lo que ahora el paquete debe manejar dos etiquetas una para la VPN y otra para la conmutación MPLS.

Veamos la figura 4.6, para ver como se realiza este proceso paso a paso, cada enrutador PE coloca una etiqueta por cada ruta dentro de la VRF (paso 1), etiqueta que identifica a la VPN. Esta etiqueta y la de MPLS se propagan junto a su correspondiente ruta a través de BGP hacia todos los enrutadores PE (paso 2). El enrutador PE recibe la actualización de BGP e instala en su tabla VRF las rutas que recibió, con su respectiva etiqueta asignada por el enrutador de egreso.

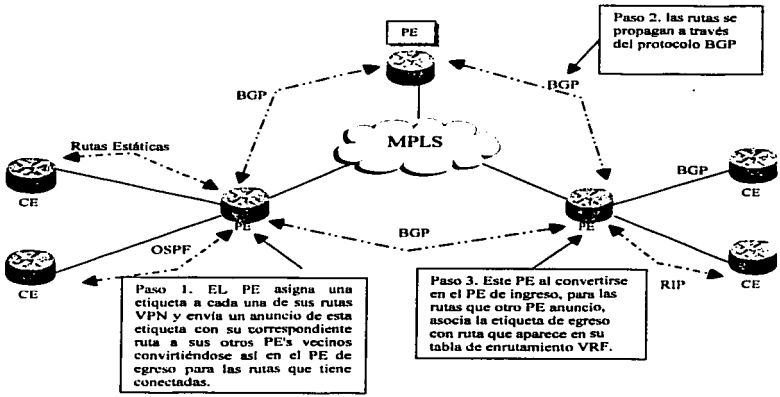


Figura 4.6. Asignación y distribución de etiquetas VPN

Cuando el enrutador PE de ingreso recibe un paquete VPN examina la VRF correspondiente y se extrae la etiqueta de la tabla virtual de la VPN, para después insertarla dentro del paquete, esta etiqueta está asociada con la dirección destino del paquete (etiqueta VPN), la cual fue asignada por el enrutador PE de egreso. Otra etiqueta (etiqueta MPLS) que apunta hacia el enrutador del próximo salto, se obtiene de la tabla de trasporte. Ambas etiquetas se envían dentro del encabezado MPLS (label stack) hacia el enrutador PE de egreso.

Todos los enrutadores P dentro de la red conmutan los paquetes VPN basándose solo en la primera etiqueta, es decir, la etiqueta MPLS la cual apunta hacia el enrutador de egreso. Debido a las reglas MPLS, el enrutador P nunca revisa más allá de la primera etiqueta de esta manera la segunda etiqueta y por lo tanto el hecho de que transportan un paquete VPN siempre pasa inadvertido para estos enrutadores.

El enrutador PE de egreso recibe el paquete etiquetado VPN, ya que la primera etiqueta fue eliminada previamente por un pop tag y desempeña un chequeo sobre la segunda etiqueta, la cual únicamente identifica la VRF designada y algunas veces la interfaz de salida del paqueta sobre el enrutador PE, finalmente el paquete es enviado al enrutador CE.

El enrutador PE asigna etiquetas para las rutas de tal forma que minimiza la necesidad de un chequeo adicional del encabezado IP en la VRF designada. Como lo vimos el en capitulo de MPLS el enrutador que está justo antes del enrutador PE de egreso puede remover la primera etiqueta de la localidad de etiquetas (label stack), y enviar el paquete hacia el enrutador PE sólo con la etiqueta de la VPN de esta forma el enrutador PE se ahorra un chequeo de etiquetas.

4.4 Operación de la arquitectura VPN/MPLS

Hasta este momento hemos visto varios conceptos veamos como se entrelazan todos ellos, al mismo tiempo que resumimos las necesidades básicas de una red VPN, en los siguientes puntos.

- Las redes privadas virtuales basadas en MPLS combinan los beneficios del modelo superpuesto VPN, como son el aislamiento y la seguridad con los beneficios del modelo acoplado VPN, como son el enrutamiento simplificado y mejor escalabilidad.

- Cada VPN necesita una VRF separada en cada enrutador PE para garantizar el aislamiento y habilitar el uso de direcciones IP duplicadas. Las VRF sirven para establecer el servicio VPN dentro de la red y en ella se coloca la información de enrutamiento del cliente de la VPN.

- Se necesita atributo conocido como Route Target para identificar a las rutas que pertenecen a la VPN en la cual la VRF participa. Un conjunto de Route Targets puede ser asociado con una VRF o rutas VPN.

- Las direcciones IP-VPN deben ser aumentadas con un Route Distinguisher de 64 bits para hacer a las direcciones VPN globalmente únicas. Estas direcciones de 96 bits se intercambian entre los enrutadores PE a través del protocolo BGP, el cual puede incluso portar información de atributos adicionales para las rutas (por ejemplo el route target), por medio de atributos opcionales de BGP conocido como **Extended Communities** (Comunidades Extendidas).

- Cada enrutador PE necesita una identificación propia, generalmente se utiliza una dirección IP que es conocida como loopback en el ambiente de configuraciones, la cual se usa para asignarle una etiqueta y habilitar el transporte de paquetes VPN a través de la red.

- Cada enrutador PE asigna una etiqueta única a cada ruta en cada VRF y propaga estas etiquetas junto con las direcciones VPN de 96 bits a través de BGP. Cabe aclarar que BGP es un formato estándar que puede manejar solo direcciones IPv4 (direcciones IP versión 4).

- El enrutador PE usa dos etiquetas dentro de la localidad de etiquetas MPLS (label stack), una para identificar al enrutador PE la cual se asigna a través del mecanismo de distribución de etiquetas MPLS normal (protocolo TDP o UDP) y la otra etiqueta para etiquetar los paquetes VPN, esta etiqueta es asignada por el enrutador PE de egreso. Es por eso que la primera es conocida como etiqueta MPLS y la segunda como etiqueta de la VPN.

- El label stack se inserta dentro del paquete VPN y el paquete MPLS resultante se envía a través de los enrutadores P. La etiqueta VPN se mantendrá constante hasta alcanzar el enrutador PE de egreso y a la VPN a la cual pertenece el paquete, la etiqueta MPLS se utilizara para transportar el paquete a través de la red e ira cambiando (conmutando) en cada uno de los nodos.

Los conceptos de la arquitectura **VPN/MPLS** se describen completamente en la **RFC2547bis** y se llama BGP/MPLS/VPN's.

4.5. Términos de diseño y configuración para establecer un servicio VPN-MPLS.

Después de repasar los conceptos básicos de MPLS, es más fácil entender como se implementará esta arquitectura en términos de diseño y a través de la configuración de la infraestructura del proveedor de servicio. Para la configuración seguiremos una serie de pasos básicos que son necesarios en toda la implementación.

Una de las topologías más simples que se pueden implementar usando la arquitectura VPN/MPLS es una intranet entre múltiples sitios que pertenecen a la organización. A través de este capitulo hemos usado una topología como ésta, red que tomaremos otra vez para explicar los pasos necesarios para la implementación. Ver figura 4.2.

Como podemos ver la red tienen dos clientes VPN: Autos y Planetas. La dos organizaciones tiene sitios en México, Guadalajara y Monterrey compartiendo el enrutador PE entre los dos CE. El proveedor de servicio aprende rutas desde ambos clientes VPN a través de protocolos como RIP, BGP, OSPF y rutas estáticas.

La siguiente tabla muestra las direcciones IP para ambos clientes VPN así como las direcciones loopback que identifican cada PE y que serán usadas por BGP.

Organización	Sitio	Subred
Autos	Mty	192.168.8.0/30
	Gdlo	192.168.4.0/30
	Mex	192.168.0.0/30
Planetas	Mty	192.169.8.0/30
	Gdlo	192.169.4.0/30
	Mex	192.169.0.0/30
Red del proveedor	Mty (Loopback)	10.30.1.1
	Gdlo	10.20.1.1
	Mex	10.10.1.1

Nota: la dirección 192.168.8.0 indica la red IP y el /30 indica la mascara de subred que están utilizando

Para lograr este servicio de VPN a través de la estructura VPN/MPLS se tienen que seguir los siguientes pasos:

- Definir y configurar las VRF's
- Definir y configurar los Route Distinguishers
- Definir y configurar las políticas de importación y de exportación
- Configurar el multiprotocolo BGP en la red del proveedor
- Asociar las interfaces del enlace PE - CE a las VRF's previamente definidas
- Configurar el enlace PE- CE, asociándoles un protocolo de enrutamiento

Ahora para examinar con detalle cada uno de estos pasos vamos a utilizar una rama de la red de la figura 4.1 y seguiremos la configuración de manera que quede establecida la arquitectura VPN/MPLS.

4.5.1 Configuración de las VRF's

El primer paso es definir y configurar las VRF's para las VPN's autos y planetas, ya que las dos VPN están presentes en cada uno de los sitios de la red, la configuración de estas dos VRF's para estos dos clientes específicos deben existir en todos los enrutadores PE
La siguiente es la configuración en el enrutador PE (mex), para la VRF autos y la VRF planetas:

```
ip vrf autos
 rd 192.168.0.0:1
 route-target export 192.168.0.0:1
 route-target import 192.168.0.0:1
```

```
ip vrf planetas
 rd 192.169.0.0:1
 route-target export 192.169.0.0:1
 route-target import 192.169.0.0:1
```

Cuando se crean las VRF se puede empezar a configurar las variables asociadas con las VRF, como son el Route Distinguisher y las políticas de importación y de exportación que también mostramos en los cuadros anteriores.

4.5.2 Como definir el Route Distinguisher

Debido a que en la arquitectura MPLS cada VPN debe ser capaz de utilizar los mismos prefijos IP que otra VPN esté utilizando; es necesario colocar a las direcciones un Route Distinguisher, estas rutas necesitan ser únicas para que BGP pueda manejar el mismo prefijo desde dos VPN's separadas y las considere como rutas no idénticas. Este Route Distinguisher debe ser diferente en cada VPN. La arquitectura VPN/MPLS restringe la comunicación entre VPN's que utilizan el mismo espacio de direcciones.

Cada VRF necesita tener asociado más de un Route Distinguisher, pero la recomendación es utilizar un solo para la VPN. El diseñador de red puede asignar un valor en particular para el Route Distinguisher para cada VRF dentro del enrutador PE. La estructura de este valor puede ser como sigue:

ASN:nn = Número de sistema autónomo: número de red
IP-address:nn = Dirección IP: número de red

Es importante saber que el Route Distinguisher no tienen ningún significado para el protocolo BGP y es interpretado solo como una secuencia de bits que es parte de la dirección VPN-IP para funciones de administración y que distingue a la VPN.

El valor del Route Distinguisher debe ser el mismo para cada VRF, dentro de los enrutadores PE, que pertenece a una VPN en particular. La tabla siguiente enlista los valores asignados para cada VPN, en este ejemplo como podemos darnos cuenta las direcciones IP asociadas al Route Distinguisher de cada cliente son diferentes, por lo tanto no existe problema al utilizar un valor de red igual ya que de todas formas el Route Distinguisher resultante es diferente para las dos VPN.

Si existiera el caso donde el administrador usará las direcciones iguales por ejemplo la 192.168.0.0 el valor de red que se le asignaría a Autos sería **1**, con Route Distinguisher de 192.168.0.0:1 y el valor de planetas sería **2**, con Route Distinguisher de 192.168.0.0:2.

Clientes VPN	Red IP	Valor Único	Route Distinguisher
Autos	192.168.0.0	1	192.168.0.0:1
Planetas	192.169.0.0	1	192.169.0.0:1

La misma configuración se necesita en los enrutadores PE de gdl y de mty.

4.5.3. Como definir las políticas de importación y exportación

Como hemos visto los enrutadores PE aprenden rutas desde otros PE's a través de la red y desde los sitios de los clientes adjuntos a el. Estas rutas deben ser insertadas dentro de las tablas de enrutamiento de una VPN especifica. Cualquier ruta aprendida desde los clientes es anunciada a través de la red VPN/MPLS utilizando el protocolo BGP y cualquier ruta aprendida vía BGP es colocada dentro de la VRF de interés. Para lograr esto cada enrutador necesita información que le indique como procesar cualquier ruta recibida, esta información no solo le dice al enrutador PE dentro de cual VRF debe importar la ruta, sino también que información debe agregar a la ruta cuando la anuncie a otro enrutador PE.
La entidad conocida como Route Target determina cual VRF y al mismo tiempo, cual sitio VPN debe recibir la ruta.

Los atributos opcionales de BGP contienen un conjunto de **Extended Communities** que definen el sitio desde donde fueron aprendidas las direcciones IPv4-VPN (ruta de origen - **Route Origin**) y un conjunto de enrutadores a los cuales la ruta debe ser exportada (ruta de destino - el **Route Target**).

Estamos hablando de que BGP define dos **Communities** el route target y el route origin. El route origin es conocido también como SOO y sirve para prevenir loops de enrutamiento entre los sitios, evitando que un enrutador PE reciba el anuncio, proveniente desde otro PE, de una ruta que el previamente anunció. El atributo de BGP conocido como route target, debe ser configurado en los enrutadores PE para definir las políticas de importación y exportación de rutas sobre una VRF por cada sitio.

La figura 4.7 ayudara a entender mejor como se utilizan las políticas para importar y exportar las rutas y en la tabla se muestra la configuración de estas políticas en el enrutador PE mty.

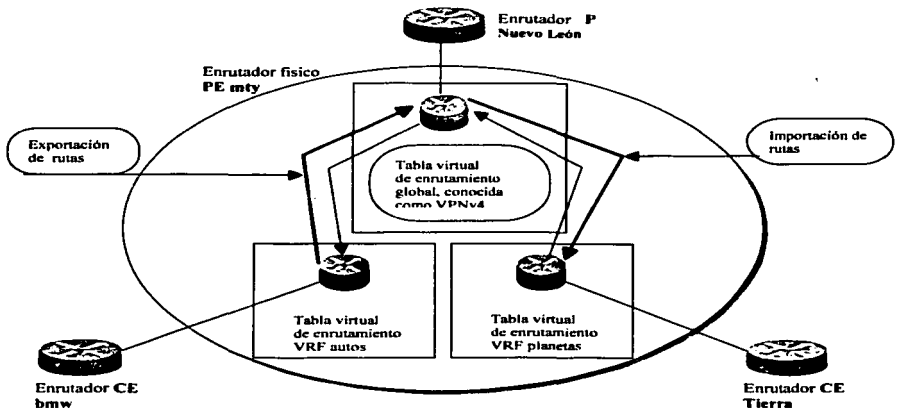


Figura 4.7. Manejo de políticas de importación y exportación dentro de un PE

El último paso en la configuración de una VRF son precisamente las políticas de importación y exportación que la VRF va a utilizar. Estas políticas serán usadas para anunciar rutas dentro de la VRF. Las tablas siguientes muestran el enrutador PE (mty) con la configuración de estas políticas para cada una de las VRF que estamos usando.

```
ip vrf autos
route-target export 192.168.0.0:1
route-target import 192.168.0.0:1
```

```
ip vrf planetas
 route-target export 192.169.0.0:1
 route-target import 192.169.0.0:1
```

El route target import especifica las rutas que la VRF desea importar, originadas desde las VRF's de otros PE's y el route target export, exporta las rutas que la VRF desea dentro de las actualizaciones BGP para ser anunciadas a otros PE.

4.5.4. Asociación de interfaces a las VRF's

Después de que se definen todas las VRF relevantes sobre el enrutador PE, se debe asociar la interfaz del enrutador PE que se enlaza con el enrutador CE, con la VRF a la que pertenece el sitio que conecta, ya que a través de esta interfaz se inyectaran las rutas del cliente dentro de la VPN a la que pertenece. Más de una interfaz puede pertenecer a la misma VRF.

La siguiente tabla muestra como se asocian estas interfaces a las diferentes VRF's.

```
interface Serial3/2
 ip vrf forwarding autos
 ip address 192.168.8.6 255.255.255.252

interface Serial3/3
 ip vrf forwarding planetas
 ip address 192.169.8.6 255.255.255.252
```

Cuando se asocian las interfaces con una VRF particular, sus direcciones IP deben ser instaladas como parte de la VRF.

Solo las interfaces que están corriendo la conmutación CEF pueden ser asociadas con VRF's porque el mecanismo de conmutación CEF es un requisito necesario para implementar satisfactoriamente el transporte de datos VPN/MPLS ya que la imposición de etiquetas se logra a través de trayectorias conmutadas por CEF.

4.5.5. Implementación y uso del multiprotocolo BGP.

Ya sabemos que MP-BGP se utiliza para anunciar las rutas de los clientes VPN entre los enrutadores PE, rutas que fueron aprendidas desde los enrutadores CE. Estas rutas pudieron haber sido aprendidas por el CE a través de protocolos como BGP-4, RIP versión 2, OSPF, EIGRP o rutas estáticas.

MP-BGP solamente se requiere dentro de la columna vertebral del proveedor (backbone), específicamente en los enrutadores PE y el RR. Además todas las sesiones son internas de BGP, internas porque las sesiones son entre enrutadores que pertenecen al mismo sistema autónomo.

MP-BGP es un requisito porque las actualizaciones de BGP necesitan portar más información aparte de la dirección IP, es decir, las actualizaciones MP-BGP contienen rutas VPN-IPv4, las cuales portan información de etiquetas MPLS y de comunidades extendidas (extended communities) de BGP.

Los enrutador PE envían actualizaciones MP-BGP a otros enrutador PE a través del RR; estas actualizaciones contienen información VPN/MPLS la cual está comprendida por los siguientes puntos:

- Address Family Information (AFI)
- Next-hop Information
- NLRI (Network Layer Reachability Information)

La información **Address-family** identifica el protocolo de capa de red que está siendo portado dentro de la actualización. Este es colocado en AFI=1 y sub - AFI =128 en el caso de VPN/MPLS.

La información de **next hop** es la dirección del próximo enrutador sobre la trayectoria hacia el destino. En el caso de VPN/MPLS esta es la dirección del enrutador PE anunciante, es decir el que anuncio la ruta destino, con su respectiva etiqueta.

La NLRI (información de alcanzabilidad de capa de red) está codificada con el siguiente formato de información para MPLS.

Longitud: es la longitud total de la etiqueta más el prefijo IP (incluyendo el Route Distinguisher)

Etiqueta: porta una o más etiquetas en una localidad. Este campo porta las partes del encabezado MPLS como son el valor de la etiqueta, el bit experimental y el bit bottom of stack.

Prefijo: Route Distinguisher más el prefijo IP.

4.5.6. Establecimiento de la comunicación mediante BGP

La configuración de BGP requiere de varios puntos que tienen que ser configurados para cualquier sesión BGP-interior de PE a RR a través de la estructura VPN/MPLS.

Como parte de las especificaciones de MP-BGP, una extensión de BGP conocida como **address-Family** debe ser creada para permitirle a BGP portar un protocolo diferente a **IPv4**; hablando de la arquitectura VPN/MPLS, esta address-family será utilizada para portar las direcciónes VPN-IPv4, BGP es usado en este tipo de arquitectura solamente para portar este tipo de direcciones.

El siguiente paso en la configuración de MP-BGP es definir y activar las sesiones entre los enrutadores PE. El tipo de sesión y las especificaciones de que tipo de rutas (VPN-IPv4 o IPv4) portara la sesión, son controladas por las address-families dentro de la configuración de BGP, para las rutas que serán inyectadas dentro y fuera de las VRF's , es decir, rutas VPN-IPv4 y a través del proceso de configuración BGP normal, para las rutas que pertenecen a la tabla de enrutamiento global, rutas IP puras.

Se debe configurar en BGP una address-family por cada VRF configurada sobre el enrutador PE y aparte otra address-family para portar rutas VPN-IPv4 entre los enrutadores PE, esta última address family también debe estar dentro del RR, ya que él es el que refleja las rutas hacia los otros PE's. Todos los vecinos BGP que pertenezcan a la VPN están definidos o establecidos bajo su address-family asociada. El proceso BGP con ninguna address-family especificada, es la addres-family default para las sesiones que no están asociadas con ninguna VRF o solo portan rutas IPv4 desde la tabla de enrutamiento global.

Las configuraciones de las sesiones BGP, que portan rutas IPv4 son exactamente las mismas que una configuración estándar de BGP, con la excepción de que las sesiones deben ser activadas. Para activar las sesiones se necesita especificar la dirección del vecino, su sistema autónomo y activarlo. Ver la siguiente tabla.

```
router bgp 100

 neighbor 10.1.1.1 remote-as 100
 neighbor 10.1.1.1 update-source Loopback0
 neighbor 10.1.1.1 activate
```

El proceso BGP activa las sesiones MP-BGP que portan los prefijos VPN-IPv4 a través del uso de las address-family propietarias de BGP. Estas address-family crean un subprocesos de enrutamiento de BGP para intercambiar prefijos VPN-IPv4. La siguiente tabla muestra la configuración en el enrutador PE mty colocando el RR como vecino, la configuración del RR debe ser la misma, pero activando a cada uno de sus vecinos PE.

```
mtydi_PE

address-family vpnv4
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community both
 exit-address-family
```

En el ejemplo se puede observar que dentro del address-family VPNv4 solo necesitamos activar a los vecinos para el uso de los prefijos VPN-IPv4; ya que los vecinos BGP deben ser configurados primero bajo el proceso BGP global. Además es necesario instruir a BGP para anunciar los atributos de comunidad extendida (extended community) que ya conocemos, al activar send community both le permitimos a BGP utilizar sus atributos estándar y extendidos. Cabe aclarar que los PE's deben especificar siempre como vecino al RR, utilizando para esto su dirección loopback.

Después de que las rutas de los clientes llenan las VRF's, estas rutas tienen que ser anunciadas a través de la red VPN/MPLS. BGP desempeña este trabajo llevando estas rutas como prefijos VPN-IPv4 a través de las sesiones entre los enrutadores PE. Para permitir que esto suceda, un subproceso de enrutamiento tiene que ser agregado dentro del proceso BGP para indicarle a BGP cuales rutas VRF anunciar.

Otra vez para lograr esto necesitamos usar una address-family, pero esta vez necesitaremos una para cada VRF. Las rutas que pertenecen a las VRF y que están asociadas a estas address-family necesitan ser redistribuidas dentro de BGP para poder ser anunciadas a otros PE's. En la siguiente tabla se muestra la configuración de las address family para cada VRF del enrutador PE (mty).

```
address-family ipv4 vrf planetas
 redistribute static
 exit-address-family
 !
 address-family ipv4 vrf autos
 redistribute static
 exit-address-family
```

El Route Target comunidad extendida de BGP y el Route Distinguisher controlan la selección de rutas VPN-IPv4, esto sucede después de que las rutas fueron aprendidas desde otros enrutadores PE a través de las sesiones BGP y antes de que estas rutas sean importadas dentro de cualquier VRF.

El primer paso en el proceso de decisión BGP es agrupar todos las rutas relevantes para que se puedan comparar. Antes de que el enrutador PE pueda seleccionar rutas, tienen que saber cuales rutas de VPN existen y cual de estas rutas debe ser comparada con las demás por el proceso de selección. Cada VRF está configurada con informes que le indican al enrutador PE que rutas deben ser importadas dentro de la VRF. Nosotros ya sabemos que el Route Target controla este proceso de importación, así que el enrutador PE debe hacer lo siguiente:

- Tomar todas las rutas con el mismo Route Target que indican los informes de importación, contenidos dentro de la VRF.
- Considerar todas las rutas que tienen el mismo Route Distinguisher, que está asignado a la VRF que está procesando.
- Crear nuevas trayectorias BGP con un Route Distinguisher que es igual al Route Distinguisher configurado para la VRF que está procesando.

Todas las rutas después de esto son comparables y el proceso de selección se puede realizar. En la figura 4.8 se entrelazan todos estos conceptos para un mejor entendimiento y detalla paso a paso como van sucediendo los hechos a partir de alguna configuración dentro del PE.

4.5.7. Opciones para establecer el enlace Cliente-Proveedor.

Hay cuatro caminos separados para que una estructura VPN-MPLS pueda recibir las rutas desde el cliente de la VPN (CE) y estas son los protocolos de enrutamiento; **BGP**, **RIP** y **OSPF** además de las **rutas estáticas**.

Sin importar que protocolo se utilice entre el PE y el CE, las rutas de los clientes deben ser insertadas dentro de la VRF que está asociada a la interfaz a la cual el CE está conectado, esto requiere que el proceso de enrutamiento utilizado en el PE sea configurado de tal forma que cualquier ruta aprendida sea relacionada con la VRF, en el capitulo 5 se detallara los pasos a seguir por cada uno de estos protocolos. Cuando las tablas VRF tienen alguna ruta, estas deben ser introducidas dentro de BGP (proceso de redistribución), para poder ser anunciadas a otros PE's como rutas VPN-IP.

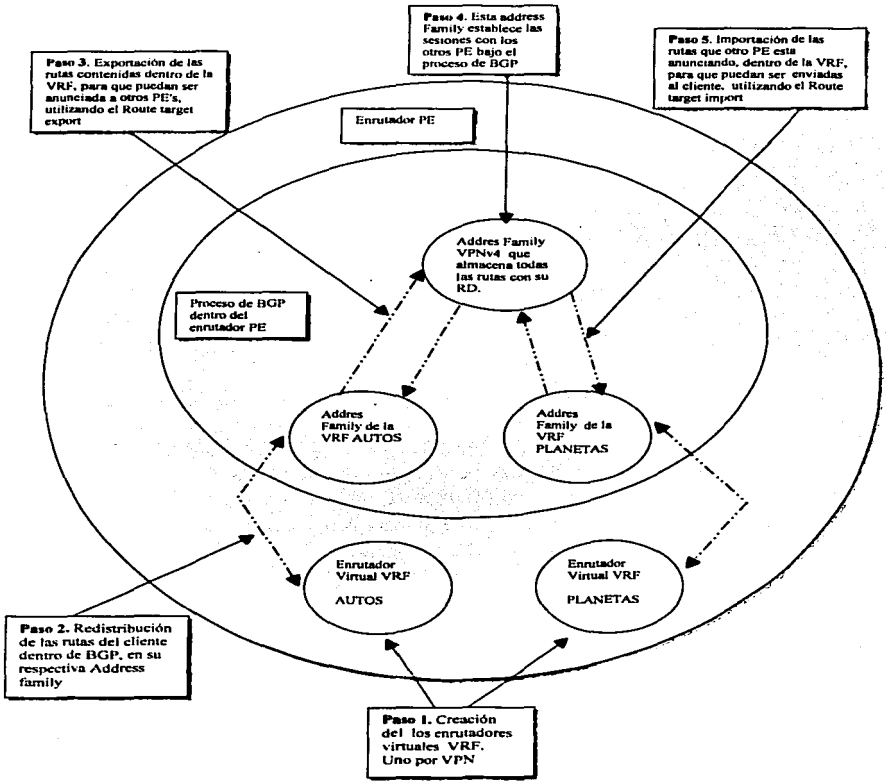


Figura 4.8 Funcionamiento interno de un Enrutador PE

CAPÍTULO V

IMPLEMENTACIÓN PRÁCTICA DE UNA RED VPN/MPLS

5.1 Guía para implementar la red VPN/MPLS

Este capítulo está basado en los capítulos previos. con el fin de corroborar el funcionamiento y la eficacia de la de la arquitectura MPLS.

Cuando se introduce una nueva tecnología, siempre se tienen que conocer los temas relacionados y hacer una buena elección del diseño para que la implementación resulte satisfactoria.

En los capítulos anteriores ya se introdujo toda la información relacionada a esta arquitectura, así que nos toca el turno de construir la red VPN-MPLS y analizar su funcionamiento, el diseño de red que utilizaremos será muy sencillo pero cubriendo la estructura básica de una arquitectura VPN-MPLS. Para la implementación de la red VPN-MPLS, utilizaremos nueve enrutadores CISCO, los cuales se distribuyen como lo muestra la figura 5.1. En este diagrama se pueden observar los enrutadores que utilizamos, las interfaces a las cuales está conectado cada uno de los equipos con su respectivo puerto (# de interfaz), la dirección IP que cada puerto debe tener y la dirección IP (loopback) que identifica a cada enrutador.

El diagrama maneja la siguiente nomenclatura como esta para las interfaces:

E0 ⟶ **Interfaz Ethernet # de puerto 0**
S0 ⟶ **Interfaz serial # 0**
F 0/0 ⟶ **Interfaz Fast Ethernet # 0/0**
P 3/1/0 ⟶ **Interfaz de fibra óptica POS (Packet over Sonet) # 3/1/0**

Nodo	Loopback 0
Gdldi_PE	10.1.1.1
Gdldo_P	10.2.1.1
Nube_P	10.2.2.1
Mexdo_P	10.2.3.1
Mexdi_PE	10.3.3.3
Mty_RR	10.4.4.4

Nodo	loopback
Tigger_CE	L0 20.1.1.1
Igor_CE	L0 20.3.3.3
Vegueta_CE	L0 20.3.3.3
Goku_CE	L0 20.1.1.1

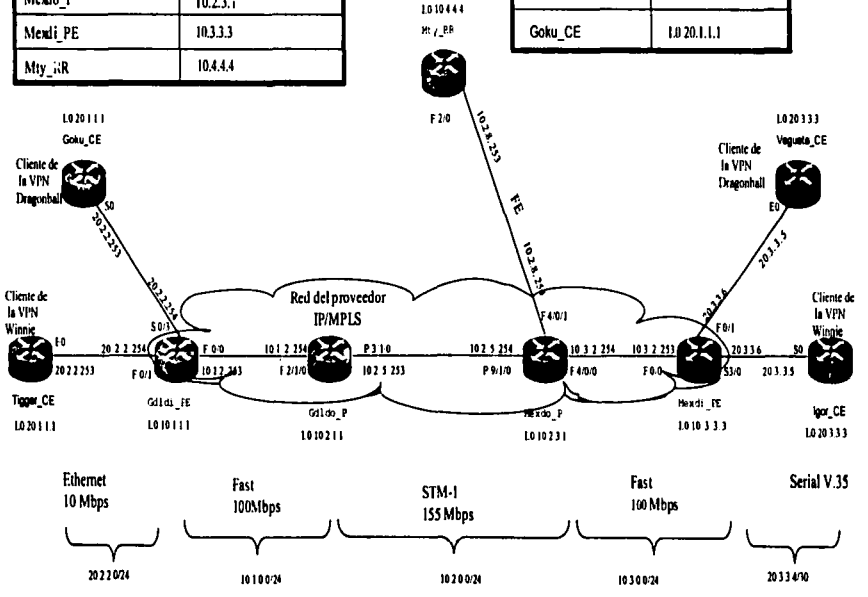


Figura 5.1 Diagrama de la red a implementar

La estructura de la red esta formada por:

- Dos enrutadores CE que representan los sitios de los clientes de una VPN que llamaremos WINNIE, los cuales son, Igor y Tigger
- Dos enrutadores CE que representan los sitios de los clientes de una VPN que llamaremos DRAGONBALL, los cuales son, Goku y Vegueta.
- Dos enrutadores PE, uno conocido como Gdldi_PE y el otro como Mexdi_PE
- Dos enrutadores P, que son Gdldo_P y el otro Mexdo_P
- Un enrutador que funcionara como Route Reflector para las rutas BGP, que llamaremos Mty_RR

La tabla 5.1, muestra el número de VPN's que utilizaremos, las VRF's que debemos crear y el Route Distinguisher de cada VRF, en esta implementación tenemos solamente dos enrutadores PE que deben manejar las VRF como se puede observar a continuación.

Enrutador PE	VPN a la que pertenecen	Sitios dentro de la VPN	VRF	Route Distinguisher asociado
Mexdi Gdldi	Winnie	Tigger Igor	Winnie	20.0.0.0:1
	Dragonball	Goku Vegueta	Dragonball	20.0.0.0:2

Tabla 5.1 Información asignada a las VPN's Winnie y Dragonball

Las dos VPN's estarán manejando el mismo direccionamiento, lo que las hará diferentes será el Route Distinguisher.

En el capítulo anterior se introdujeron los pasos necesarios para implementar una red VPN-MPLS, estos pasos serán los que seguiremos uno a uno hasta armar toda la red, sabemos que en cada enrutador se tiene configurar una serie de comandos, lo cual no es otra cosa mas que introducir información al enrutador para que éste desempeñe una tarea. La información que cada enrutador necesita, es decir la configuración, se muestra paso a paso junto con una breve explicación de porque se necesita dentro de la red.

5.2. Implementación de la infraestructura de red IP

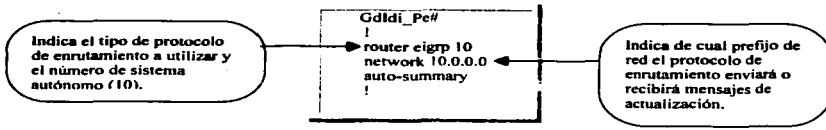
Lo primero que hay que implementar es la estructura IP en toda la red, en la figura 5.1 se muestra cada una de las interfaces con su respectiva direcciones IP. las cuales son necesarias para que cada uno de los nodos se pueda comunicar con otro. El ejemplo 5.1 muestra la configuración de las interfaces físicas y de la loopback que se introdujo dentro del enrutador Gdldi_PE, todos los enrutadores se configuran de la misma manera por lo que no es necesario mostrar cada una de las interfaces de la red. Lo que sí debe quedar claro es que todas y cada una de las configuraciones se hicieron respetando totalmente la información proporcionada en el diagrama de la figura 5.1. como son las direcciones IP de las interfaces de cada enrutador

```
Gdldi_Pe#

interface FastEthernet0/0
 ip address 10.1.2.253 255.255.255.0
 speed 100
 full-duplex
!
interface FastEthernet0/1
 ip address 20.2.2.254 255.255.255.0
 speed 10
half-duplex
!
interface Loopback0
ip address 10.1.1.1 255.255.255.0
!
```

Ejemplo 5.1. Configuración de las direcciones IP de uno de los enrutadores

Para que todos los enrutadores conozcan cada una de las rutas para poder alcanzar a cada uno de los nodos de la red, es importante utilizar un protocolo de enrutamiento, para esta red se implemento el protocolo EIGRP. el ejemplo 5.2 muestra la configuración que se introdujo dentro del enrutador Gdldi_PE y sólo se muestra una sola configuración debido a que cada enrutador debe utiliza el mismo protocolo de enrutamiento y pertenecer al mismo sistema autónomo; por lo tanto lleva la misma configuración. La configuración de un protocolo de enrutamiento es muy simple. como lo indica el ejemplo.



Ejemplo 5.2. Protocolo de enrutamiento utilizado en la red IP del proveedor

Después de que todos los enrutadores tienen sus interfaces con su respectiva dirección IP y esta habilitado el protocolo de enrutamiento se debe revisar que efectivamente todos los enrutadores saben como llegar a cada nodo en la red, para lo cual se debe desplegar la tabla de enrutamiento y si ésta tiene registradas todas las rutas contenidas en la red quiere decir que todo está bien, el ejemplo 5.3 muestra la tabla de enrutamiento que construyó el enrutador Mexdi_PE a partir de los anuncios de sus vecinos en donde aparecen todas las rutas de la red 10.0.0.0, las que se aprendieron vía EIGRP, se identifican por la letra D, y las que el mismo enrutador tiene directamente conectadas, letra C, cada una de estas rutas pertenecen a diferentes nodos dentro de la red del proveedor de servicio, comparar con la figura 5.1.

```
Mexdi_PE#sh ip route
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, 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

    10.0.0.0/24 is subnetted, 11 subnets
D      10.2.8.0 [90/30720] via 10.3.2.254, 03:22:34, FastEthernet0/0
D      10.2.1.0 [90/161280] via 10.3.2.254, 03:22:34, FastEthernet0/0
D      10.1.2.0 [90/35840] via 10.3.2.254, 03:22:34, FastEthernet0/0
D      10.4.4.0 [90/158720] via 10.3.2.254, 03:22:34, FastEthernet0/0
C      10.3.3.0 is directly connected, Loopback0
D      10.2.2.0 [90/158720] via 10.3.2.254, 03:22:34, FastEthernet0/0
D      10.1.1.0 [90/163840] via 10.3.2.254, 03:22:34, FastEthernet0/0
C      10.3.2.0 is directly connected, FastEthernet0/0
D      10.2.3.0 [90/156160] via 10.3.2.254, 03:22:35, FastEthernet0/0
D      10.2.5.0 [90/33280] via 10.3.2.254, 03:22:35, FastEthernet0/0
```

Ejemplo 5.3. Tabla de enrutamiento construida por Mexdi_PE.

5.3. Implementación de la red VPN-MPLS

Una vez que se implemento totalmente la red IP, comienza la implementación de la parte de MPLS y VPN's.

Los siguientes son los pasos a seguir para implementar la conmutación por etiquetas y las redes privadas virtuales, en el capítulo anterior se explicó cada uno de estos por lo que aquí debe ser más fácil de entender y seguir la configuración de los enrutadores de la red, además con esta implementación ofrezco un análisis detallado de lo que va sucediendo con cada configuración, para aclarar cualquier duda.

- Habilitar MPLS y CEF en todos los enrutadores de la red del proveedor (enrutadores P y PE).
- Definir y configurar las VRF's, Route Distinguishers y políticas de importación y de exportación
- Asociar las interfaces del enlace PE - CE a las VRF's previamente definidas
- Configurar el multiprotocolo BGP en la red del proveedor
- Configurar el enlace PE- CE, asociándoles un protocolo de enrutamiento

5.3.1 Habilitando MPLS y CEF en todos los enrutadores de la red del proveedor.

Como sabemos al momento de introducir MPLS en la red se dejará de establecer un enrutamiento basado en direcciones IP y se establecerá un enrutamiento basado en etiquetas MPLS. Dentro de los enrutadores PE y P se configuraran los siguientes comandos:

```
Mexdi_PE#
!
ip cef
tag-switching tdp router-id Loopback0
```

- El **ip cef** es un requisito para MPLS ya que proporciona una forma de conmutar paquetes más rápida.
- El comando **tag-switching tdp router-id Loopback 0**, se pone para solicitarle al enrutador que tome la dirección loopback como la dirección que lo identificará de aquí en adelante.

La activación del protocolo MPLS debe ser interfaz por interfaz en toda la red, a excepción de la interfaz que conecta con el sitio del cliente, por ejemplo dentro del enrutador Mexdi_PE, dentro de la interfaz FastEthernet0/0 que lo conecta al enrutador Mexdo_P se introduce el comando tag-switching ip, con el cual se habilita la conmutación de paquetes por etiquetas MPLS, como se muestra a continuación:

```
Mexdi_PE# interface FastEthernet0/0
Mexdi_PE# tag-switching ip
```

Este comando debe habilitarse en todas las interfaces que enlazan los enrutadores PE y P, además las que enlazan los enrutadores P y P. Ver figura 5.1 para identificar estas interfaces.

Al establecer MPLS por interfaz, comienza el intercambio de mensajes entre enrutadores vecinos, con las etiquetas que cada uno le agrega a la ruta, en la siguiente sección veremos un análisis práctico de como comienza este intercambio de rutas para formar las tablas MPLS.

5.3.1.1 Análisis de la Imposición y Distribución de Etiquetas

Para llevar acabo este análisis, fue necesario establecer un orden en cuanto a la manera de habilitar el protocolo MPLS en toda la red, en la figura 5.2 se detallan los pasos que se siguieron en esta práctica, para habilitar el protocolo MPLS en la red, como se puede observar se comienza ha habilitar MPLS de derecha a izquierda en cada una de las interfaces de los enrutadores PE y P, el objetivo de realizar estos pasos es la de ir analizando enrutador por enrutador la imposición y distribución de sus etiquetas.

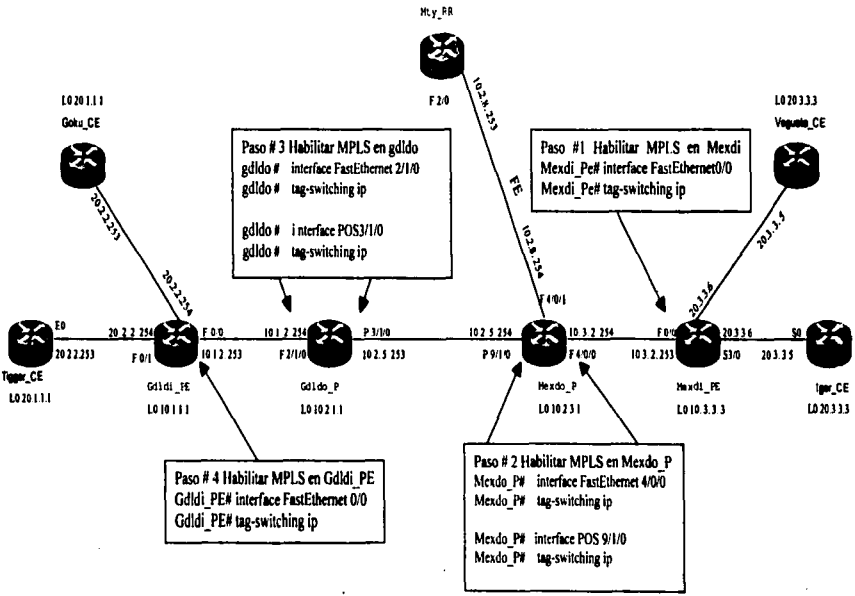


Figura 5.2. Pasos para habilitar MPLS en toda la red.

El análisis de la propagación de etiquetas se hizo con los mensajes reales que se intercambiaron entre enrutadores, para una sola ruta la 10.1.1.1, debo aclarar que se originan mensajes al mismo tiempo para cada una de las rutas contenidas dentro de la tabla, como los mensajes son exactamente iguales me permito escoger una sola ruta y analizar la imposición y distribución de etiquetas desde Mexdi_PE hasta Gdldi_PE y así demostrar como se origina la TIB/LIB (Base de información de etiquetas), y la tabla TFIB/LFIB (Base de información de transporte de etiquetas), las cuales se mencionan en el capitulo III.

El análisis comienza después de realizar el primer paso, los mensajes que origina cada enrutador están enumerados para después hacer una descripción ordenada de cada uno de éstos.

Paso # 1: Activamos MPLS en la interfaz Fast Ethernet 0/0 del enrutador Mexdi_PE como lo vimos en la figura 5.2, cuando esta interfaz ya está habilitada con MPLS, el enrutador origina los siguientes mensajes:

```
Mexdi_PE#

1. Jun 26 17:40:56: tib: find route tags: 10.1.1.0/255.255.255.0, FastEthernet0/0, nh 10.3.2.254
2. Jun 26 17:40:56: tagcon: tibent(10.1.1.0/24): created; find route tags request
3. Jun 26 17:40:56: tagcon: tibent(10.1.1.0/24): label 21 (#14) assigned
4. Jun 26 17:40:56: tagcon: route_tag_change for: 10.1.1.0/24 inlabel 21, outlabel unknown,
                    nexthop lsr 10.3.2.254:0, reason response to find_route_tags

5. Jun 26 17:40:56: LFIB: finish res:inc tag=21,outg=Unkn.next_hop=10.3.2.254,FastEthernet0/0
```

Descripción:

1. Indica que encontró la ruta 10.1.1.0/255.255.255.0, que le llego por su interfaz Fast 0/0 y el next hop para esa ruta es la dirección 10.3.2.254 (interfaz del vecino Mexdo_P).
2. Indica que creó una entrada para esta ruta dentro de la tabla TIB y solicita etiquetas para la ruta encontrada.
3. Indica que el enrutador Mexdi_PE impuso la etiqueta 21 para esta ruta.

4. Hubo un cambio en cuanto a las etiquetas: la etiqueta de entrada es 21, desconoce la etiqueta de salida, pero espera que se la envié su next hop (próximo salto para alcanzar esa dirección) es decir, enrutador conmutador de etiquetas (LSR) 10.3.2.254.
5. Se crea la entrada en la tabla TFIB/LFIB, con etiqueta de entrada 21, de salida desconocida, con next hop 10.3.2.254 el cual puede alcanzar saliendo por la interfaz Fast Ethernet 0/0.

En este momento, solamente el enrutador Mexdi_PE tiene habilitado MPLS, por lo cual no existe vecino alguno al cual enviar o recibir mensajes de las etiquetas impuestas, por lo tanto cuando crea una entrada dentro de la tabla TFIB/LFIB para cada una de las rutas, éstas sólo aparecen con las etiquetas de entrada, para corroborar esta información el ejemplo 5.4 muestra el desplegado de la tabla TFIB/LFIB que este enrutador creó.

```
Mexdi_PE#sh tag-switching forwarding-table

Local  Outgoing    Prefix           Bytes tag   Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched    interface

16     Untagged    10.2.8.0/24      0           Fa0/0       10.3.2.254
18     Untagged    10.2.1.0/24      0           Fa0/0       10.3.2.254
19     Untagged    10.1.2.0/24      0           Fa0/0       10.3.2.254
20     Untagged    10.4.4.0/24      0           Fa0/0       10.3.2.254
21     Untagged    10.1.1.0/24      0           Fa0/0       10.3.2.254
22     Untagged    10.2.3.0/24      0           Fa0/0       10.3.2.254
23     Untagged    10.2.5.0/24      0           Fa0/0       10.3.2.254
```

Ejemplo 5.4. Desplegado de la tabla TFIB/LFIB dentro del enrutador Mexdi_PE

Como se puede observar efectivamente se crea la tabla con una entrada para cada una de las rutas y el enrutador les asigna una etiqueta local, como se puede comprobar con la ruta 10.1.1.1, que como vimos en el desplegado de los mensajes le impone la etiqueta 21, pero en el campo de salida (outgoing) no tiene etiqueta alguna asignada (estado untagged), la causa de este estado es porque aún no existe un vecino que le anuncie etiquetas para estas rutas y así poder colocarlas como etiquetas de salida para esos destinos.

Al no encontrar vecinos de MPLS Mexdi_PE considera que la red MPLS termina con el y por lo tanto tiene conectada una red IP, de acuerdo a esto el pone untagged en el campo de salida de todas las rutas, para poder enviar paquetes IP puros.

Con la creación de esta tabla también podemos comprobar el control independiente que tiene el enrutador, de su tabla de etiquetas ya que impone etiquetas a las rutas sin importar si algún vecino le anuncia o le va a anunciar etiquetas para cada uno de sus destinos.

Paso # 2: En este paso se habilita MPLS dentro de dos interfaces del enrutador Mexdo_P, vecino de Mexdi_PE. Ver la figura 5.2.

De igual forma al momento de habilitar MPLS el enrutador Mexdo_P asigna etiquetas para cada una de las rutas contenidas dentro de su tabla, los siguientes son los mensajes que origina para el destino 10.1.1.1:

```
Mexdo_P#
1. Jun 26 17:44:26: tib: find route tags: 10.1.1.0/255.255.255.0, POS9/1/0, nh 10.2.5.253, res nh 10.2.5.253
2. Jun 26 17:44:26: tagcon: tibent(10.1.1.0/24): created; find route tags request
3. Jun 26 17:44:26: tagcon: tibent(10.1.1.0/24): lcl tag 30 (#12) assigned
4. Jun 26 17:44:26: tagcon: route_tag_change for: 10.1.1.0/24 intag 30, outtag unknown,
                    nexthop tsr 10.2.5.253, reason response to find_route_tags

5. Jun 26 17:44:26: TFIB: finish res:inc tag=30,outg=Unkn,next_hop=10.2.5.253,POS9/1/0
```

Descripción:

1. Indica que encontró la ruta 10.1.1.0/255.255.255.0, que le llego por su interfaz POS 9/1/0 y el next hop para esa ruta es la dirección 10.2.5.253 (interfaz del vecino Mexdo_P).
2. Indica que creó una entrada para esta ruta dentro de la tabla TIB y solicita etiquetas para la ruta encontrada.
3. Indica que el enrutador le asigno la etiqueta local 30.
4. Indica la etiqueta de entrada como 30 y que desconoce la etiqueta de salida pero espera respuesta de su next hop (próximo salto para alcanzar esa dirección), enviándole una etiqueta de salida.
5. La tabla TFIB/LFIB crea una entrada para el destino 10.1.1.1, le asigna la etiqueta 30, la etiqueta de salida desconocida, con next hop 10.2.5.253 por la interfaz POS 9/1/0.

El enrutador siempre solicita una etiqueta al vecino marcado como next hop y no a cualquier vecino, ya que a través del next hop alcanzara el destino deseado.

Habiéndose percatado el enrutador Mexdo_P de que tiene un vecino con dirección 10.3.3.3, procederá a enviar enseguida las etiquetas que asigno de manera local, por medio de mensajes iguales al siguiente:

```
Mexdo_P#

1. Jun 26 17:44:40: tagcon: adj 10.3.3.3:0 (pp 0x61E241AC): advertise 10.1.1.0/24, tag 30 (#12)
```

Descripción:

1. Este mensaje lo origina el enrutador Mexdo_P para el vecino 10.3.3.3 es decir, Mexdi_PE, en donde se le anuncia la imposición local de la etiqueta 30 para la ruta 10.1.1.0/24.

Por su parte el enrutador Mexdi_PE recibe y procesa el anuncio de la siguiente forma.

```
Mexdi_PE#

1. Jun 26 17:44:51: tagcon: tibent(10.1.1.0/24): label 30 from 10.2.3.1:0 added
2. Jun 26 17:44:51: tagcon: route_tag_change for: 10.1.1.0/24 inlabel 21, outlabel 30,
                    nexthop lsr 10.2.3.1:0, reason response to find_route_tags

3. Jun 26 17:44:51: LFIB: finish res:inc tag=21,outg=30,next_hop=10.3.2.254,FastEthernet0/0
```

Descripción:

1. Indica que para la entrada 10.1.1.0/24, de la tabla TIB/LIB recibe una etiqueta 30 procedente de la dirección 10.2.3.1, dirección que identifica a su vecino Mexdo_P.
2. Indica que la ruta 10.1.1.0/24 ha tendido un cambio al agregarse la etiqueta 30 como etiqueta de salida para ese destino. Anuncio que recibió de su next hop LSR. 10.2.3.1.
3. Agrega la siguiente información en la tabla TFIB/LFIB para el destino 10.1.1.1, etiqueta de salida 30, next hop (siguiente salto) 10.3.2.254 por la interfaz Fast Ethernet 0/0.

El ejemplo 5.5a se muestra las modificaciones que la tabla TFIB/LFIB del enrutador Mexdi_PE realizó después de este anuncio y anuncios similares que Mexdo_P le envió, por cada una de las rutas conteni[illegible]en la tabla. Ahora si Mexdi_PE tiene una etiqueta de salida para la red 10.1.1.1 y puede enviar los paquetes con sólo conmutar la etiqueta 21 por la 30 y enviar el paquete por la interfaz Fast Ethernet 0/0 para que su vecino haga lo mismo.

El ejemplo 5.5b. muestra un fragmento de la tabla TIB/LIB del enrutador Mexdi_PE, después del anuncio de etiqueta de su next hop LSR 10.2.3.1; para la ruta 10.1.1.1.

```
Mexdi_PE#sh tag-switching forwarding-table

Local  Outgoing    Prefix           Bytes tag   Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched    interface
16     Pop tag     10.2.8.0/24      0           Fa0/0       10.3.2.254
18     28          10.2.1.0/24      0           Fa0/0       10.3.2.254
19     27          10.1.2.0/24      0           Fa0/0       10.3.2.254
20     31          10.4.4.0/24      0           Fa0/0       10.3.2.254
21     30          10.1.1.0/24      0           Fa0/0       10.3.2.254
22     Pop tag     10.2.3.0/24      0           Fa0/0       10.3.2.254
23     Pop tag     10.2.5.0/24      0           Fa0/0       10.3.2.254
```

Ejemplo 5.5a. Tabla TFIB/LFIB dentro del enrutador Mexdi_PE

```
mexdi_pe#sh tag-switching tdp bindings 10.1.1.0 255.255.255.0
  tib entry: 10.1.1.0/24, rev 2
        local binding:  tag: 21
        remote binding: tsr: 10.?.3.1:0, tag: 30
```

Ejemplo 5.5b. Tabla TIB/LIB dentro del enrutador Mexdi_PE.

Además de recibir el anuncio para la red 10.1.1.0 y los restantes por cada una de las rutas, al enrutador Mexdi_PE, también le corresponde anunciar las etiquetas que agregó a las rutas, dado que acaba de aparecer un vecino. Con el anuncio que Mexdi_PE le hace a su vecino Mexdo_P de la etiqueta que impuso a la ruta 10.1.1.0 ocurre algo curioso, que sucede sólo en casos particulares, como ya vimos cada enrutador espera un mensaje de una etiqueta para las rutas contenidas dentro de su tabla pero especifica que solamente aceptará este mensaje proveniente de la dirección marcada como next hop, en este caso Mexdo_P no espera un anuncio de etiqueta para la ruta 10.1.1.1 proveniente de Mexdi_PE debido a que no es el next hop, sin embargo Mexdi_PE hace el anuncio.

A continuación se puede ver el desplegado de dos anuncios que envía Mexdi_PE a Mexdo_P, uno de una ruta para la cual si puede hacer un anuncio y otro para la red 10.1.1.1 para la cual no puede anunciar una etiqueta, esto es para que quede clara la diferencia.

```
Mexdi_PE#

1. Jun 26 17:44:51: tagcon: peer 10.2.3.1:0 (pp 0x63C72D1C): advertise 10.1.1.0/24, label 21 (#14)
2. Jun 26 17:44:51: tagcon: peer 10.2.3.1:0 (pp 0x63C72D1C): advertise 10.3.3.0/24, label 1(imp-null) (#12)
```

Descripción:

1. Mexdi_PE anuncia a su vecino Mexdo_P (10.2.3.1), la etiqueta 21 que asigno a la ruta 10.1.1.0.
2. Mexdi_PE anuncia a su vecino Mexdo_P (10.2.3.1), la etiqueta 1 (imp-null) que asigno a la ruta 10.3.3.0/24, con la cual esta solicitando un POP TAG para los paquetes con este destino.

Mexdi_PE no es el next hop para la ruta 10.1.1.0/24 porque no esta en la trayectoria para alcanzar ese destino, por lo tanto Mexdo_P no debe tomar en cuenta la etiqueta que mexdi le está anunciando, ya que no le corresponde hacer el anuncio, analizando los mensajes que le llagan a Mexdo_P con respecto a los anuncios que Mexdi_PE le acaba de hacer, podemos corroborar esta información.

```
Mexdo_P#

1. Jun 26 17:44:40: tagcon: tibent(10.1.1.0/24): rem tag 21 from 10.3.3.3:0 added
2. Jun 26 17:44:40: tib:addr bound check: negative, gateway=10.2.5.253
3. Jun 26 17:44:40: tagcon: Omit route_tag_change for: 10.1.1.0/24
                tsr 10.3.3.3:0: next hop = 10.2.5.253
```

Este es el mensaje que recibe Mexdo_P el cual indica lo siguiente:

1. Recibe una etiqueta 21 procedente de la dirección 10.3.3.3, Mexdi_PE
2. Checa la dirección de procedencia y rechaza el anuncio porque el next hop marcado para esa ruta es el 10.2.5.253.
3. Omite el cambio para esa ruta porque espera una etiqueta procedente de su next hop.

Como puede verse aunque Mexdi_PE envía un anuncio con una etiqueta para ese destino el enrutador Mexdo_P lo rechaza, en cambio veamos en los siguientes mensajes lo que sucede con una ruta para la cual si está asignado como próximo salto, la ruta 10.3.3.3.

Mexdi_PE anunció una etiqueta 1, la cual tiene el significado implicit-null, como se explica en el capitulo III con esta etiqueta se solicita al enrutador anterior en este caso a Mexdo_P que se aplique un pop tag (anulación de etiqueta) al paquete con este destino en particular.

Enseguida se muestra como procesa Mexdo_P la información que recibe de la ruta 10.3.3.0.

```
Mexdo_P#
1. Jun 26 17:44:40: tagcon: tibent(10.3.3.0/24): rem tag 1 from 10.3.3.3:0 added
2. Jun 26 17:44:40: tagcon: route_tag_change for: 10.3.3.0/24 intag 29, outtag imp-null,
                    nexthop tsr 10.3.3.3:0, reason response to find_route_tags

3. Jun 26 17:44:40: TFIB: finish res:inc tag=29,outg=Imp_null,next_hop=10.3.2.253,FastEthernet4/0/0
```

Descripción

1. Recibe una etiqueta 1 procedente de la dirección 10.3.3.3, Mexdi_PE.
2. Realiza el ajuste colocando en el campo de salida la descripción imp-null (implicit null) ya que recibió una etiqueta 1 del next hop para esa ruta; quine además es el último salto que el paquete debe dar.
3. Información que almacena en la tabla TFIB/LFIB

Después de recibir la etiqueta para la ruta 10.3.3.0 el enrutador Mexdo_P establece las tablas TFIB/LFIB y TIB/LIB como se muestra en los ejemplos 5.6a y 5.6b.

```
Mexdo_P#sh tag-switching forwarding-table

Local  Outgoing    Prefix           Bytes tag   Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched    interface
27     Untagged    10.1.2.0/24      0           PO9/1/0     point2point
28     Untagged    10.2.1.0/24      0           PO9/1/0     point2point
29     Pop tag     10.3.3.0/24      0           Fa4/0/0     10.3.2.253
30     Untagged    10.1.1.0/24      0           PO9/1/0     point2point
31     Untagged    10.4.4.0/24      381         Fa4/0/1     10.2.8.253
```

Ejemplo 5.6a. Tabla TFIB/LFIB dentro del enrutador Mexdo_P.

```
mexdo_p#sh tag-switching tdp bindings 10.3.3.0 255.255.255.0
  tib entry: 10.3.3.0/24, rev 24
        local binding:  tag: 29
        remote binding: tsr: 10.3.3.3:0, tag: imp-null
```

Ejemplo 5.6b. Tabla TIB/LIB dentro del enrutador Mexdo_P.

Con esto dos ejemplos se puede concluir que sólo se aceptan los anuncios de etiquetas para las rutas, cuando el enrutador que los hace es el next hop de esas rutas.

Paso #3: Para poder continuar con la formación de las tablas en todos los enrutadores y que Mexdo_P pueda switchear paquetes, se activa MPLS en el enrutador Gdldo_P vecino de Mexdo_P, éste también tienen dos interfaces que deben manejar conmutación por etiquetas como se muestra en la figura 5.2.

Lo primero que hace el enrutador cuando se habilita con MPLS es imponer sus etiquetas veamos cual le corresponde a la ruta 10.1.1.0.

```
Gdldo_P#
1. Jun 26 17:48:59: tib: find route tags: 10.1.1.0/255.255.255.0, FastEthernet2/1/0, nh 10.1.2.253, res nh 10.1.2.253

2. Jun 26 17:48:59: tagcon: tibent(10.1.1.0/24): created; find route tags request
3. Jun 26 17:48:59: tagcon: tibent(10.1.1.0/24): lcl tag 30 (#14) assigned
4. Jun 26 17:48:59: tagcon: route_tag_change for: 10.1.1.0/24 intag 30, outtag unknown,
                    nexthop tsr 10.1.2.253, reason response to find_route_tags

5. Jun 26 17:48:59: TFIB: finish res:inc tag=30,outg=Unkn,next_hop=10.1.2.253,FastEthernet2/1/0
```

Descripción:

1. Indica que se dispone a encontrar etiquetas para la ruta 10.1.1.0/255.255.255.0, que recibió por la interfaz fast 2/1/0 procedente de la dirección 10.1.2.253.
2. Indica que se creó una entrada para esta ruta dentro de la tabla TIB y que se está haciendo una solicitud de etiquetas para esa ruta.
3. Indica que el enrutador le asignó la etiqueta 30 como etiqueta local.
4. Indica la etiqueta de entrada como 30 y que desconoce la etiqueta de salida pero espera respuesta de su next hop (próximo salto para alcanzar esa dirección), enviándole una etiqueta de salida.
5. Se crea la entrada en la tabla TFIB/LFIB, con etiqueta de entrada 30, de salida desconocida, con next hop 10.1.2.253, el cual puede alcanzar saliendo por la interfaz Fast Ethernet 2/1/0.

Al mismo tiempo se percata de que tiene un vecino con dirección 10.2.3.1, por lo que procede a enviar el siguiente anuncio:

```
Gdldo_P#
Jun 26 17:49:13: tagcon: adj 10.2.3.1:0 (pp 0x61850D2C): advertise 10.1.1.0/24, tag 30 (#14)
```

Descripción:

1. Este mensaje lo origina el enrutador Gdldo_P para el vecino 10.2.3.1 es decir, Mexdo_P, en donde se le anuncia la imposición local de la etiqueta 30 para la ruta 10.1.1.0/24.

Por su parte el enrutador Mexdo_P recibe y procesa el anuncio de esta forma:

```
Mexdo_P#
1. Jun 26 17:49:03: tagcon: tibent(10.1.1.0/24): rem tag 30 from 10.2.1.1:0 added
2. Jun 26 17:49:03: tagcon: route_tag_change for: 10.1.1.0/24 intag 30, outtag 30,
                    nexthop tsr 10.2.1.1:0, reason response to find_route_tags

3. Jun 26 17:49:03: TFIB: finish res:inc tag=30,outg=30,next_hop=10.2.1.1,POS9/1/0
```

Descripción:

1. Recibe una etiqueta 30 procedente de la dirección 10.2.1.1. Gdldo_P, para la ruta 10.1.1.0/24.
2. Realiza el ajuste colocando en el campo de salida la etiqueta 30 y en el campo de entrada la etiqueta 30 que el mismo impuso.
3. Información que almacena en la tabla TFIB/LFIB, que incluye las etiquetas local y de salida, el next hop 10.2.1.1 con la interfaz de salida POS 9/1/0.

Mexdo_P realiza los cambios correspondientes en sus tablas TFIB/LFIB (ejemplo 5.7a) y TIB/LIB (ejemplo 5.7b) después de recibir la información, para esta ruta en particular coincidieron las etiquetas, esto no significa que siempre deban ser las mismas simplemente fue casualidad.

```
Mexdo_P#sh tag-switching for

Local   Outgoing    Prefix          Bytes tag   Outgoing    Next Hop
tag     tag or VC   or Tunnel Id    switched    interface
27      Pop tag     10.1.2.0/24     0           PO9/1/0     point2point
28      Pop tag     10.2.1.0/24     0           PO9/1/0     point2point
29      Pop tag     10.3.3.0/24     0           Fa4/0/0     10.3.2.253
30      30          10.1.1.0/24     0           PO9/1/0     point2point
31      Untagged    10.4.4.0/24     1016        Fa4/0/1     10.2.8.253
```

Ejemplo 5.7a Tabla TFIB/LFIB dentro de Mexdo_P

```
Mexdo_P#sh tag-switching tdp bindings 10.1.1.0 255.255.255.0
  tib entry: 10.1.1.0/24, rev 20
      local binding:  tag: 30
      remote binding: tsr: 10.2.1.1:0, tag: 30
```

Ejemplo 5.7b Tabla TIB/LIB dentro de Mexdo_P

De esta forma se va formando el camino por donde los paquetes con destino 10.1.1.1 pasarán y las etiquetas que los enrutadores tendrán que switchar para que el paquete alcance su destino.

Paso #4 Habilitar MPLS en el último enrutador de la trayectoria, el enrutador Gdldi_PE; éste tienen una interfaz dentro de la red MPLS y una que corresponde a la conexión con el cliente, en la figura 5.2 se muestra la activación del protocolo MPLS.

De igual forma al momento de habilitar MPLS el enrutador Gdldi_P realiza el mismo procedo al asignar etiquetas para cada una de las rutas contenidas dentro de su tabla, el cuadro siguiente muestra los mensajes que origina para el destino 10.1.1.1:

```
Gdldi_PE#
1. Jun 26 17:53:57: tib: find route tags: 10.1.1.0/255.255.255.0, Loopback0, nh 0.0.0.0, res nh 0.0.0.0
2. Jun 26 17:53:57: tagcon: tibent(10.1.1.0/24): created: find route tags request
3. Jun 26 17:53:57: tagcon: tibent(10.1.1.0/24): label 1 (#14) assigned
4 .Jun 26 17:53:57: tagcon: route_tag_change for: 10.1.1.0/24 inlabel imp-null, outlabel unknown,
                    nexthop lsr 0.0.0.0:0, reason response to find_route_tags
```

Descripción:

1. Indica que se dispone a encontrar etiquetas para la ruta 10.1.1.0/255.255.255.0, que pertenece a su loopback.
2. Indica que se creó una entrada para esta ruta dentro de la tabla TIB/LIB (ejemplo 5.8b) y que se está haciendo una solicitud para encontrar etiquetas asignadas a esa ruta.
3. Indica que el enrutador le asigno la etiqueta local 1, debido a que tiene directamente conectada esta ruta, como una dirección IP en una interfaz loopback.
4. Impone a la entrada un imp-null para solicitar al penúltimo salto que realice un pop tag a los paquetes que se dirijan a este destino.

Como el enrutador Gdldi_PE tiene esta dirección destino no crea una entrada en la tabla TFIB/LFIB (ejemplo 5.8a) ya que no necesitará una etiqueta de salida para transportar el paquete y por lo tanto no tampoco necesita una interfaz de salida.

```
gdldi_pe#sh tag for
Local   Outgoing    Prefix          Bytes tag   Outgoing    Next Hop
tag     tag or VC   or Tunnel Id    switched    interface
16      26          10.2.8.0/24     0           Fa0/0       10.1.2.254
17      Pop tag     10.2.1.0/24     0           Fa0/0       10.1.2.254
18      29          10.4.4.0/24     0           Fa0/0       10.1.2.254
19      28          10.3.3.0/24     0           Fa0/0       10.1.2.254
20      31          10.3.2.0/24     0           Fa0/0       10.1.2.254
21      32          10.2.3.0/24     0           Fa0/0       10.1.2.254
22      Pop tag     10.2.5.0/24     0           Fa0/0       10.1.2.254
```

Ejemplo 5.8a Tabla TFIB/LFIB dentro de Gdldi_PE

```
gdldi_pe#sh tag-switching tdp bindings
  tib entry: 10.1.1.0/24, rev 16
        local binding:  tag: imp-null
        remote binding: tsr: 10.2.1.1:0, tag: 30
```

Ejemplo5.8b Tabla TIB/LIB, para la ruta 10.1.1.1 dentro del enrutador Gdldi_PE

El siguiente es el anuncio que Gdldi_PE, envía a su vecino Gdldo_P, para que inserte la solicitud de pop tag en el campo de salida de esa ruta.

```
Gdldi_PE
Jun 26 17:54:02: tagcon: peer 10.2.1.1:0 (pp 0x63B39384): advertise 10.1.1.0/24, label 1 (imp-null) (#14)
```

Gdldo_P recibe y procesa la información guardándola en la tabla TFIB/LFIB como se muestra a continuación:

```
Gdldo_P#
1. Jun 26 17:54:02: tagcon: tibent(10.1.1.0/24): rem tag 1 from 10.1.1.1:0 added
2 Jun 26 17:54:02: tagcon: route_tag_change for: 10.1.1.0/24 intag 30, outtag imp-null,
                   nexthop tsr 10.1.1.1:0, reason response to find_route_tags

3. Jun 26 17:54:02: TFIB: finish res:inc tag=30,outg=Imp_null, next_hop=10.1.2.253,FastEthernet2/1/0
```

Descripción:

1. Recibe una etiqueta 1 procedente de la dirección 10.1.1.1, Gdldi_PE, para la ruta 10.1.1.0/24.
2. Realiza el ajuste colocando en el campo de salida la etiqueta imp-null y en el campo de entrada la etiqueta 30 que el mismo impuso.
3. Información que almacena en la tabla TFIB/LFIB, que incluye las etiqueta local y de salida, el next hop 10.1.2.253 con la interfaz de salida Fast 2/1/0.

Gdldo_P recibió una etiqueta 1 de Gdldi_PE para la ruta 10.1.1.0/24, la registro como imp_null en la tabla TIB/LIB y en la tabla TFIB/LFIB a parece con la indicación de pop tag, de esta forma al momento de que Gdldo_P reciba un paquete con una etiqueta 30, procederá a eliminarla y enviar el paquete sin etiqueta MPLS. Ver ejemplo 5.9a y 5.9b.

```
Gdldo_P#sh tag-switching forwarding-table
Local  Outgoing     Prefix            Bytes tag   Outgoing    Next Hop
tag    tag or VC    or Tunnel Id      switched    interface
26     Pop tag      10.2.8.0/24       0           PO3/1/0     point2point
28     29           10.3.3.0/24       0           PO3/1/0     point2point
29     31           10.4.4.0/24       0           PO3/1/0     point2point
30     Pop tag      10.1.1.0/24       381         Fa2/1/0     10.1.2.253
31     Pop tag      10.3.2.0/24       0           PO3/1/0     point2point
32     Pop tag      10.2.3.0/24       0           PO3/1/0     point2point
```

Ejemplo 5.9a. Tabla TFIB/LFIB dentro de Gdldo_P.

```
gdldo_p#sh tag-switching tdp bindings 10.1.1.0 255.255.255.0
  tib entry: 10.1.1.0/24, rev 20
      local binding:  tag: 30
      remote binding: tsr: 10.1.1.1:0, tag: imp-null
```

Ejemplo 5.9b. Tabla TIB/LIB dentro de Gdldo_P.

Hasta este momento hemos establecido la comunicación mediante MPLS ahora vienen la parte importante de las VPN.

5.3.2 Definir y configurar las VRF's

Como ya sabemos por cada VPN que deseamos utilizar, necesitamos establecer la configuración de una VRF dentro de los enrutadores PE y el protocolo BGP para la propagación de rutas VPN entre los PE, lo cual se logra estableciendo una sesión BGP con el enrutador Route reflector que hará posible la propagación de rutas VPN, es decir, estos son los únicos enrutadores que manejan las VPNs. por lo tanto los enrutadores P no necesitan ningún tipo de información relacionada con las VPN's.

Dos VPN's son las que utilizaremos en esta red por lo tanto es necesario configurar dos VRF's dentro de los dos enrutadores PE que tenemos, al mismo tiempo hay que asociarles un Route Distinguisher para diferenciarlas ya que necesitamos utilizar el mismo espacio de direcciones IP para demostrar que no existe problema alguno al utilizar direcciones IP duplicadas con clientes diferentes, en los siguiente cuadros puede verse que la configuración de las dos VRF's es exactamente la misma pero con un RD diferente. Cada PE debe tener dentro de su configuración las mismas VRF para poder lograr la comunicación:

```
Mexdi_Pe#
ip vrf dragonball
 rd 20.0.0.0:2
 route-target export 20.0.0.0:2
 route-target import 20.0.0.0:2
!
ip vrf winnie
 rd 20.0.0.0:1
 route-target export 20.0.0.0:1
 route-target import 20.0.0.0:1
```

```
Gdldi_Pe#
ip vrf dragonball
 rd 20.0.0.0:2
 route-target export 20.0.0.0:2
 route-target import 20.0.0.0:2
!
ip vrf winnie
 rd 20.0.0.0:1
 route-target export 20.0.0.0:1
 route-target import 20.0.0.0:1
```

En estos cuadros también se puede ver como se declaran las políticas de importación, por ejemplo la VRF Winnie pide que le importen y le exporte los paquetes que contengan el Route Distinguisher 20.0.0.0:1 y la VRF Dragonball los paquetes que porten el Route Distinguisher 20.0.0.0:2.

5.3.3. Configuración del multiprotocolo BGP en la red del proveedor

Para implementar BGP en toda la red se estableció una sesión BGP con el Route Reflector (RR) por cada enrutador PE, sesión que manejará las direcciones VPN-IPv4 (dirección IP + el RD) y direcciones IP puras, al mismo tiempo.

Como se explicó en el capitulo IV, BGP es por diseño una necesidad dentro de la red VPN-MPLS; en los cuadros siguientes aparece la configuración que se utilizó dentro de los dos enrutadores PE y el enrutador RR, en donde se puede observar que primero de manera global se activaron todos los vecinos BGP.

Dentro de los PE's sólo se necesita activar al RR el cual se identifica por la dirección 10.4.4.4, el RR por su parte debe activar a los enrutadores PE que necesiten conocer las rutas de la VPN, en este caso Mexdi_PE que lo identifica la dirección 10.3.3.3 y el enrutador Gdldi_PE con la dirección 10.1.1.1, especificando además que son clientes del RR. Al activar los vecinos de manera global sólo se garantiza el manejo de rutas IP, por lo que para propagar rutas VPN, es necesario recurrir al subproceso de BGP conocido cono Address Family VPNv4, para que los PE's y el RR puedan transportar rutas VPN-IPv4.

Mexdi_PE# (IP 10.3.3.3)

```
!
router bgp 100
 synchronization
 bgp log-neighbor-changes
 neighbor 10.4.4.4 remote-as 100
 neighbor 10.4.4.4 update-source
 Loopback0
 no auto-summary
 !
 address-family vpnv4
 neighbor 10.4.4.4 activate
 neighbor 10.4.4.4 send-community
 extended
 no auto-summary
 exit-address-family
 !
 address-family ipv4 vrf winnie
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
 !
address-family ipv4 vrf dragonball
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
!
```

Gdldi_PE# (IP 10.1.1.1)

```
!
router bgp 100
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.4.4.4 remote-as 100
 neighbor 10.4.4.4 update-source
 Loopback0
 no auto-summary
 !
 address-family vpnv4
 neighbor 10.4.4.4 activate
 neighbor 10.4.4.4 send-community
 extended
 no auto-summary
 exit-address-family
 !
 address-family ipv4 vrf winnie
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
 !
address-family ipv4 vrf dragonball
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
!
```

Mty_RR# (IP 10.4.4.4)

```
!
router bgp 100
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 10.1.1.1 remote-as 100
 neighbor 10.1.1.1 update-source Loopback0
 neighbor 10.1.1.1 route-reflector-client

 neighbor 10.3.3.3 remote-as 100
 neighbor 10.3.3.3 update-source Loopback0
 neighbor 10.3.3.3 route-reflector-client
 !
 address-family vpnv4
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 route-reflector-client
 neighbor 10.1.1.1 send-community extended

 neighbor 10.3.3.3 activate
 neighbor 10.3.3.3 route-reflector-client
 neighbor 10.3.3.3 send-community extended
 exit-address-family
```

La **address-family VPNv4** debe tener las siguientes características:

- Manejar de manera global todas las rutas de todas las VPN's que existan en la red al mismo tiempo y aunque estén revueltas siempre se distinguirán por su Route Distinguisher.
- A través de esta se recibirán y enviaran los anuncios de rutas pertenecientes a las VPN's, cumpliendo en todo momento con las políticas de importación y exportación de cada VRF.

- Como los enrutadores PE's y RR deben manejar las direcciones VPN-IPv4 es importante que este presente dentro de los tres equipos.

Para poder insertar las rutas VPN dentro del proceso de BGP, es necesario un subproceso de BGP, conocido como address-family VRF por cada VPN existente en la red, en este caso son dos una para Winnie y otra para Dragonball, dentro de estos subprocesos de BGP se insertan las rutas procedentes del cliente VPN mediante un proceso de redistribución de rutas, vamos adelantando que entre el PE y el CE se establecerá enrutamiento estático por lo que dentro de BGP haremos una redistribución de rutas estáticas; como se puede observar en los cuadros anteriores, estas address-family vrf no están dentro del enrutador Route reflector lo cual queda claro ya que los únicos que tienen conectados los sitios VPN son los enrutadores PE y el Route Reflector sólo tiene la función de reflejar las rutas VPN.

5.3.3.1 Análisis del establecimiento de las sesiones BGP

Cuando se termina la configuración de los enrutadores PE's y el RR, comienza un intercambio de mensajes entre los enrutadores PE y el RR para establecer las sesiones BGP. El siguiente es un análisis de los mensajes reales que intercambiaron los enrutadores Mexdi_PE y Gdldi_PE con el Route Reflector para establecer una sesión con éste.

Mensajes desplegados por el RR

```
Mty_RR#
1. Jun 26 20:25:53: BGP: Performing BGP general scanning
   Jun 26 20:25:53: BGP(0): scanning IPv4 Unicast routing tables
   Jun 26 20:25:53: BGP(1): scanning VPNv4 Unicast routing tables
2. Jun 26 20:26:14: %BGP-5-ADJCHANGE: neighbor 10.1.1.1 Up
   Jun 26 20:26:14: BGP(0): 10.1.1.1 computing updates, afi 0, neighbor version 0, table version 1,
                    starting at 0.0.0.0

3. Jun 26 20:28:26: %BGP-5-ADJCHANGE: neighbor 10.3.3.3 Up
   Jun 26 20:28:26: BGP(1): 10.3.3.3 computing updates, afi 1, neighbor version 0, table version 3,
                               starting at 0.0.0.0
```

Descripción:

1. El RR está ejecutando una búsqueda general para encontrar vecinos IP o VPN.
2. El RR está levantando una sesión con el vecino 10.1.1.1, Gdldi_PE.
3. El RR está levantando una sesión con el vecino 10.3.3.3, Mexdi_PE.

Mensajes desplegados en Gdldi_PE

```
Gdldi_PE#
1. Jun 26 20:25:09: BGP: Performing BGP general scanning
   Jun 26 20:25:09: BGP(0): scanning IPv4 Unicast routing tables
   Jun 26 20:25:09: BGP(1): scanning VPNv4 Unicast routing tables
2. Jun 26 20:26:14: %BGP-5-ADJCHANGE: neighbor 10.4.4.4 Up
   Jun 26 20:26:14: BGP(0): 10.4.4.4 computing updates, afi 0, neighbor version 0, table version 1,
                    starting at 0.0.0.0
```

Descripción:

1. Gdldi_PE está ejecutando una búsqueda general para encontrar vecinos IP o VPN.
2. Gdldi_PE está levantando una sesión con el vecino 10.4.4.4, Mty_RR.

Mensajes desplegados por Mexdi_PE

```
Mexdi_PE#
1. Jun 26 20:26:12: BGP: Performing BGP general scanning
   Jun 26 20:26:12: BGP(0): scanning IPv4 Unicast routing tables
   Jun 26 20:26:12: BGP(1): scanning VPNv4 Unicast routing tables
2. Jun 26 20:28:26: %BGP-5-ADJCHANGE: neighbor 10.4.4.4 Up
   Jun 26 20:28:26: BGP(2): 10.4.4.4 computing updates, afi 2, neighbor version 0, table version 5,
                    starting at 0.0.0.0
```

Descripción:

1. Mexdi_PE está ejecutando una búsqueda general para encontrar vecinos IP o VPN.
2. Mexdi_PE está levantando una sesión con el vecino 10.4.4.4, Mty_RR.

En este momento el Route reflector ya levantó una sesión con cada enrutador PE, por lo que esta listo para recibir y propagar las rutas VPN.

5.3.4 Asociación de las interfaces del PE a las VRF's previamente definidas

Con la configuración de BGP se completo la implementación de la red del proveedor, sin embargo todavía no existe la comunicación entre los sitios del cliente VPN porque falta establecer la comunicación entre los enrutadores PE y CE.

Para que un enrutador PE tenga rutas que distribuir a otros PE's es necesario que tenga al menos un cliente VPN que le envié estas rutas. En esta red existen dos enrutadores PE los cuales deben manejar dos VPN que son Winnie y Dragonball, cada uno de ellos tiene conectado un enrutador (CE) por cada sitio de cliente VPN.

Para establecer un enlace PE a CE y mantener separada la información de los dos clientes VPN que están conectados al enrutador PE, la interfaz que conecta a cada sitio VPN se asocia con la VRF a la que pertenece el sitio, para que la información de rutas que entran por esta interfaz sea almacenada en una tabla de transporte y enrutamiento separada.

En los cuadros siguientes aparece la forma de asociar las interfaces de los enrutadores PE que conectan a los enrutadores CE, enrutadores que representan los sitios de los clientes de las VPN's Winnie y Dragonball.

```
Mexdi_Pe#
!
interface FastEthernet 0/1
 ip vrf forwarding dragonball
 ip address 20.3.3.6 255.255.255.252
!
interface Serial 3/0
 ip vrf forwarding winnie
 ip address 20.3.3.6 255.255.255.252
!
```

```
Gdldi_Pe#
!
interface Serial 0/3
 ip vrf forwarding dragonball
 ip address 20.2.2.254 255.255.255.0
!
interface FastEthernet 0/1
 ip vrf forwarding winnie
 ip address 20.2.2.254 255.255.255.0
!
```

Para corroborar que interfaz es la que conecta a cada enrutador CE hay que ver el diagrama 5.1, para relacionar una interfaz con la VRF que les corresponde se requiere la información contenida en la tabla 5.1, ya que cada sitio que conecta una interfaz pertenece a una VPN.

Después de esta configuración las interfaces están listas para recibir información de rutas VPN y el enrutador PE está listo para guardar esta información dentro de tablas de enrutamiento separadas por VPN; es el momento de establecer un protocolo de enrutamiento entre el PE y el CE para que el cliente CE envié los anuncios de las rutas que tiene dentro de su tabla de enrutamiento, hacia el enrutador PE.

5.3.5. Configuración del enlace PE-CE, asociándole un protocolo de enrutamiento

Lo primero que hice para establecer el enlace PE-CE fue la asignación de las interfaces del enrutador PE a su VRF respectiva, como se vio en la sección anterior. El siguiente paso es configurar la parte IP de los sitios del cliente.

Como se ha mencionado la configuración del lado del cliente es la parte más sencilla en la implementación de la red, lo cual es una de las ventajas que goza el cliente ya que no debe preocuparse por configurar algo demasiado complicado y engorroso, además de que no necesita hacer ningún cambio en la configuración que este manejando actualmente.

Los enrutadores CE, es decir los enrutadores que conectan los sitios del cliente, manejan solamente enrutamiento basado en IP, de modo que sus interfaces sólo requieren de una dirección IP y para comunicarse con el enrutador PE requieren de un protocolo de enrutamiento dinámico o una sola ruta estática, en los cuadros siguientes se muestra la configuración IP de los enrutadores CE.

```
Tigger_CE#
interface Loopback0
 ip address 20.1.1.1 255.255.255.255
!
interface Ethernet0
 ip address 20.2.2.253 255.255.255.0
!
```

```
Goku_CE#
interface Loopback0
 ip address 20.1.1.1 255.255.255.255
!
interface Serial0/3
 ip address 20.2.2.253 255.255.255.0
!
```

```
Igor_CE#
interface Loopback0
 ip address 20.3.3.3 255.255.255.255
!
interface Serial0
 ip address 20.3.3.5 255.255.255.252
```

```
Vegueta_CE#
interface Loopback0
 ip address 20.3.3.3 255.255.255.255
!
interface Ethernet0
 ip address 20.3.3.5 255.255.255.252
```

Como se puede ver la configuración de ambas VPN's es exactamente la misma, al permitirnos poder establecer la implementación con las mismas direcciones IP en las dos VPN estamos comprobando lo que vimos en la teoría de VPN's, es decir la posibilidad de usar direcciones IP duplicadas, lo cual como sabemos es una de las principales ventajas de las VPN's.

La forma más fácil y segura de establecer el enlace PE-CE es utilizando rutas estáticas de los dos lados, ya que de esta forma ninguno de las dos partes inundaría las tablas de enrutamiento con información de rutas innecesarias o confidenciales que sólo deben ser conocidas por la red del cliente o por la red del proveedor de servicio según sea el caso.

Para las dos VPN's utilice enrutamiento estático, en los cuadros siguientes se muestra la configuración de una de ellas, en donde veremos la configuración necesaria del lado del cliente y del lado del proveedor, es decir en los enrutadores CE y PE para poder establecer la comunicación de los sitios de los clientes de la VPN Winnie, desde Tigger hasta Igor.

tigger_ce#
ip route 20.0.0.0 255.0.0.0 Ethernet 0

Gdldi_Pe#
ip route vrf winnie 20.1.1.0 255.255.255.0 20.2.2.253

Como se puede observar sólo basta con configurar una ruta estática de ambos lados, aunque cada uno maneja un formato diferente la causa principal de esta diferencia es que el enrutador CE sólo maneja rutas IP mientras que el enrutador PE debe manejar las rutas que entran de cada VPN y mantenerlas separadas en su respectiva tabla de enrutamiento, para después poder enviar los paquetes al sitio VPN al que pertenecen, por su correspondiente interfaz.

igor_ce#
ip route 20.0.0.0 255.0.0.0 Serial 0

Mexdi_Pe#
ip route vrf winnie 20.3.3.0 255.255.255.0 Serial3/0

Dentro de los enrutadores CE, la ruta estática nos indica que cualquier paquete que se dirija a la red 20.0.0.0 debe ser enviado o sacado por la interfaz Ethernet 0, según la configuración de Tigger y por la interfaz serial 0 según la configuración de Igor.

Dentro de los enrutadores PE el comando para establecer la ruta estática debe llevar la VRF a la que pertenece la interfaz de salida, en este caso debe ser la VRF Winnie como lo muestran los cuadros, ya que en el caso de Mexdi_PE cualquier paquete con destino a la red 20.3.3.0 /24 debe ser enviado por la interfaz serial 3/0 y esta interfaz fue asociada previamente a la VRF Winnie.

En el caso de Gdldi_PE cualquier paquete con destino a la red 20.1.1.0 /24 debe ser enviado hacia la dirección 20.2.2.253; como puede verse tambien es posible colocar una dirección IP en lugar de una interfaz de salida, el comportamiento es el mismo y mientras se le agregue la VRF a la que pertenece, el enrutador enviará los paquetes por la interfaz correspondiente.

La mejor manera de comprobar que realmente se estableció comunicación entre el PE y CE es revisando que las tablas de enrutamiento de cada VPN estén llenas con las rutas que el enrutador CE tienen directamente conectadas.

Las siguientes tablas muestran estas tablas con las etiquetas que cada enrutador PE impuso a estas rutas, cabe aclarar que estas etiquetas son etiquetas VPN.

```
Mexdi_PE#sh ip bgp vpnv4 vrf winnie tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)
   20.1.1.0/24      10.1.1.1        notag/23
   20.2.2.0/24      10.1.1.1        notag/24
   20.3.3.0/24      0.0.0.0         24/notag
   20.3.3.4/30      0.0.0.0         25/aggregate(winnie)
```

Ejemplo 5.10. Tabla de Transporte y enrutamiento de la VPN winnie.

En este cuadro se pueden ver claramente las dos rutas que pertenecen a este enrutador, estoy hablando del dirección loopback 20.3.3.3 que pertenece a la red 20.3.3.0 y las direcciones 20.3.3.5 y 20.3.3.6 que pertenecen a la red 20.3.3.4/30, a estas rutas el enrutador Mexdi_PE les ha impuesto las etiquetas locales 24 y 25 respectivamente, estas etiquetas son anunciadas vía BGP hacia el otro enrutador PE, junto con su next hop es decir la dirección IP de quien anuncia las rutas, en este caso Mexdi_PE (10.3.3.3).

Aquí tambien podemos ver dos rutas que fueron anunciadas por la dirección IP 10.1.1.1, que identifica al enrutador Gdldi_PE, la ruta 20.1.1.0 a la que Gdldi_PE le impuso la etiqueta 23 y la ruta 20.2.2.0 con etiqueta 24, Mexdi_PE al recibir la información coloca las etiquetas recibidas en el campo de salida para esas rutas, así como la dirección IP del enrutador que se anuncio como next hop para estas rutas.

Dentro del enrutador Gdldi_PE se puede corroborar la información. En el ejemplo 5.11 se pueden ver las rutas que Mexdi_PE le anuncia a Gdldi_PE con etiquetas 24 y 25, las cuales son colocadas en el campo de salida como corresponde.

```
Gdldi_PE#sh ip bgp vpnv4 vrf winnie tags

   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)
   20.1.1.0/24      20.2.2.253      23/notag
   20.2.2.0/24      0.0.0.0         24/aggregate(winnie)
   20.3.3.0/24      10.3.3.3        notag/24
   20.3.3.4/30      10.3.3.3        notag/25
```

Ejemplo 5.11. Tabla de transporte y enrutamiento de la VRF Winnie dentro de Gdldi_PE.

En la teoría se menciona la creación de una tabla por cada VPN dentro de los PE's para demostrar que de verdad se crean tablas de transporte separadas por cada VPN, en los ejemplos 5.12 y 5.13 se muestra la tabla que se creo en cada enrutador PE para la otra VPN dragonball.

```
Mexdi_PE#sh ip bgp vpnv4 vrf dragonball tag
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:2 (dragonball)
   20.1.1.0/24      10.1.1.1        notag/26
   20.2.2.0/24      10.1.1.1        notag/25
   20.3.3.0/24      20.3.3.5        26/notag
   20.3.3.4/30      0.0.0.0         27/aggregate(dragonball)
```

Ejemplo 5.12. Tabla de transporte y enrutamiento de la VRF Dragonball dentro de Mexdi_PE.

```
Gdldi_PE#sh ip bgp vpnv4 vrf dragonball tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:2 (dragonball)
   20.1.1.0/24      0.0.0.0         26/notag
   20.2.2.0/24      0.0.0.0         25/aggregate(dragonball)
   20.3.3.0/24      10.3.3.3        notag/26
   20.3.3.4/30      10.3.3.3        notag/27
```

Ejemplo 5.13. Tabla de transporte y enrutamiento de la VRF Dragonball dentro de Gdldi_PE.

Las tablas de la VPN Dragonball se formaron de la misma forma que las de la VPN Winnie, las rutas que otros PE's anuncian son colocadas en la tabla y las etiquetas que se recibieron, en el campo de salida para esa ruta, así mismo aparecen la rutas locales a las que se les impone una etiqueta la cual es almacenada en el campo de entrada y que serán anunciadas a otros PE's.

Una comparación que se puede hacer entre las tablas de la VRF Winnie y las tablas de la VRF Dragonball, es de las rutas que manejan, en donde a simple vista podría decirse que contienen las mismas, pero no son las mismas ya que están debidamente identificadas por su Route Distinguisher diferente; ya que la VRF Winnie maneja el Route Distinguisher 20.0.0.0:1 que le asignamos y la VRF Dragonball maneja el 20.0.0.0:2. Otra comparación que se puede hacer es de las etiquetas VPN diferentes que cada tabla maneja.

5.3.6 Análisis de la propagación de rutas VPN a través de BGP

Desde el momento que los enrutadores PE recibieron rutas de los sitios del cliente, éstos las almacenaron en la tabla que les correspondía, según la interfaz por donde las recibieron. Inmediatamente comenzó la propagación de estas rutas mediante BGP, desde un PE hasta otro PE, cada uno inserto las rutas en la VPN a la que pertenecían según el Route Distinguisher que traían. En los cuadros que aparecen en la sección anterior aparecen estas rutas dentro de las tablas de transporte y enrutamiento, en esta sección ofrezco un análisis de los mensajes que se intercambiaron para anunciar estas rutas.

Para que las rutas VPN fueran insertadas dentro de estas tablas VRF debieron haber sido anunciadas previamente por los enrutadores PE a través del Route Reflector por medio del protocolo BGP, el siguiente es un análisis de la forma en que las rutas para la VPN winnie fueron propagadas de PE a PE.

```
Mexdi_PE#
1. Jun 26 20:28:26: BGP(2): 10.4.4.4 send UPDATE (format) 20.0.0.0:1:20.3.3.0/24, next 10.3.3.3,
                            metric 0, path , extended community RT:20.0.0.0:1
   Jun 26 20:28:26: BGP(2): 10.4.4.4 send UPDATE (prepend, chgflags: 0x0) 20.0.0.0:1:20.3.3.4/30,
                            next 10.3.3.3, metric 0, path , extended community RT:20.0.0.0:1
Jun 26 20:28:26: BGP(2): 10.4.4.4 initial update completed
```

Descripción:

1. Mexdi_PE envía dos actualizaciones al Route Reflector 10.4.4.4, con las rutas 20.3.3.0/24 y 20.3.3.4/30, colocando su dirección IP, 10.3.3.3, como next hop para esas rutas y las envía utilizando el route target 20.0.0.0:1

```
Gdldi_PE#
1. Jun 26 20:26:14: BGP(2): 10.4.4.4 send UPDATE (format) 20.0.0.0:1:20.1.1.0/24, next 10.1.1.1,
                            metric 0, path , extended community RT:20.0.0.0:1
   Jun 26 20:26:14: BGP(2): 10.4.4.4 send UPDATE (prepend, chgflags: 0x0) 20.0.0.0:1:20.2.2.0/24,
                            next 10.1.1.1, metric 0, path , extended community RT:20.0.0.0:1
   Jun 26 20:26:14: BGP(2): 10.4.4.4 1 updates enqueued (average=106, maximum=106)
   Jun 26 20:26:14: BGP(2): 10.4.4.4 update run completed, afi 2, ran for 0ms, neighbor version 0,
                            start version 5, throttled to 5
   Jun 26 20:26:14: BGP(2): 10.4.4.4 initial update completed
```

Descripción:

1. Gdldi_PE envía dos actualizaciones al Route Reflector 10.4.4.4, con las rutas 20.1.1.0/24 y 20.2.2.0/240, colocando su dirección IP, 10.1.1.1, como next hop para esas rutas y las envía utilizando el route target 20.0.0.0:1

De esta forma recibe el Route Reflector las rutas

```
Mty_RR#

1. Jun 26 20:26:14: BGP(1): 10.1.1.1 rcvd UPDATE w/ attr: nexthop 10.1.1.1, origin ?, localpref 100,
                            metric 0, extended community RT:20.0.0.0:1
   Jun 26 20:26:14: BGP(1): 10.1.1.1 rcvd 20.0.0.0:1:20.1.1.0/24
   Jun 26 20:26:14: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.1.1.0/24
   Jun 26 20:26:14: BGP(1): 10.1.1.1 rcvd 20.0.0.0:1:20.2.2.0/24
   Jun 26 20:26:14: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.2.2.0/24

2. Jun 26 20:28:26: BGP(1): 10.3.3.3 rcvd UPDATE w/ attr: nexthop 10.3.3.3, origin ?, localpref 100,
                            metric 0, extended community RT:20.0.0.0:1
   Jun 26 20:28:26: BGP(1): 10.3.3.3 rcvd 20.0.0.0:1:20.3.3.0/24
   Jun 26 20:28:26: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.3.3.0/24
   Jun 26 20:28:26: BGP(1): 10.3.3.3 rcvd 20.0.0.0:1:20.3.3.4/30
   Jun 26 20:28:26: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.3.3.4/30
```

1. El route reflector recibe dos actualizaciones de rutas VPN, procedentes del PE 10.1.1.1, con Route Distinguisher 20.0.0.0:1, las cuales tienen asignado el next hop 10.1.1.1
2. El Route Reflector recibe dos actualizaciones de rutas VPN, procedentes del PE 10.3.3.3, con Route Distinguisher 20.0.0.0:1, las cuales tienen asignado el next hop 10.3.3.3

Como es obvio el Route Reflector envía estas actualizaciones a los PE's

```
Mty_RR#
1. Jun 26 20:28:26: BGP(1): 10.1.1.1 send UPDATE (format) 20.0.0.0:1:20.3.3.0/24, next 10.3.3.3,
                    metric 0, path , extended community RT:20.0.0.0:1
Jun 26 20:28:26: BGP(1): 10.1.1.1 send UPDATE (prepend, chgflags: 0x208)
          20.0.0.0:1:20.3.3.4/30, next 10.3.3.3, metric 0, path , extended community RT:20.0.0.0:1
Jun 26 20:28:26: BGP(1): 10.1.1.1 1 updates enqueued (average=120, maximum=120)
Jun 26 20:28:26: BGP(1): 10.1.1.1 update run completed, afi 1, ran for 0ms, neighbor version 3,
                         start version 5, throttled to 5

2. Jun 26 20:28:26: BGP(1): 10.3.3.3 send UPDATE (format) 20.0.0.0:1:20.1.1.0/24,
                                next 10.1.1.1, metric 0, path , extended community RT:20.0.0.0:1
   Jun 26 20:28:26: BGP(1):  10.3.3.3 send UPDATE (prepend, chgflags: 0x0) 20.0.0.0:1:20.2.2.0/24,
                             next 10.1.1.1, metric 0, path , extended community RT:20.0.0.0:1
   Jun 26 20:28:26: BGP(1): 10.3.3.3 1 updates enqueued (average=119, maximum=119)
   Jun 26 20:28:26: BGP(1): 10.3.3.3 update run completed, afi 1, ran for 0ms, neighbor version 0,
                            start version 3, throttled to 3
   Jun 26 20:28:26: BGP: 10.3.3.3 initial update completed
```

Descripción:

1. La actualización que recibió el RR de la dirección 10.3.3.3, las envía hacia el otro enrutador PE con el que tiene una sesión, es decir el 10.1.1.1, tal como las recibió.

2. La actualización que recibió el RR de la dirección 10.1.1.1, las envía hacia el otro enrutador PE con el que tiene una sesión, es decir el 10.3.3.3, tal como la recibió.

El enrutador Mexdi_PE lo recibe de la siguiente forma:

```
Mexdi_PE#
Jun 26 20:28:26: BGP: Incoming path from 10.4.4.4
Jun 26 20:28:26: BGP(2): 10.4.4.4 rcvd UPDATE w/ attr: nexthop 10.1.1.1, origin?, localpref 100,
metric 0, originator 10.1.1.1, clusterlist 10.4.4.4, extended community RT:20.0.0.0:1

1. Jun 26 20:28:26: BGP(2): 10.4.4.4 rcvd 20.0.0.0:1:20.1.1.0/24
   Jun 26 20:28:26: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.1.1.0/24
   Jun 26 20:28:26: BGP: Accepted path from 10.4.4.4
   Jun 26 20:28:26: BGP(2): 10.4.4.4 rcvd 20.0.0.0:1:20.2.2.0/24
   Jun 26 20:28:26: vpn: bgp_vpnv4_bnetinit: 20.0.0.0:1:20.2.2.0/24
   Jun 26 20:28:26: BGP: Accepted path from 10.4.4.4
2. Jun 26 20:28:27: BGP: Import walker start version 5, end version 7
   Jun 26 20:28:27: BGP: ... start import cfg version = 7
   Jun 26 20:28:27: BGP: Import timer expired. Walking from 5 to 7
   Jun 26 20:28:27: vpn: winnie same RD import, do best path
```

```
3. Jun 26 20:28:27: BGP(2): Revise route installing 1 of 1 route for 20.1.1.0/24 -> 10.1.1.1
                            to winnie IP table
   Jun 26 20:28:27: BGP(2): Revise route installing 1 of 1 route for 20.2.2.0/24 -> 10.1.1.1
                            to winnie IP table
4. Jun 26 20:28:28: vpn: tag_vpn_find_route_tags: 20.0.0.0:1:20.2.2.0
   Jun 26 20:28:28: vpn: intag=vpn-route, outtag=24, outtag owner=BGP
   Jun 26 20:28:28: vpn: tag_vpn_find_route_tags: 20.0.0.0:1:20.1.1.0
   Jun 26 20:28:28: vpn: intag=vpn-route, outtag=23, outtag owner=BGP

5. Jun 26 20:28:31: BGP(2): 10.4.4.4 computing updates, afi 2, neighbor version 5, table version 9,
                            starting at 0.0.0.0
   Jun 26 20:28:31: BGP(2): 10.4.4.4 update run completed, afi 2, ran for 0ms, neighbor version 5,
                            start version 9, throttled to 9
```

Descripción:

1. Están llegando dos rutas de la procedentes del RR 10.4.4.4, las rutas 20.1.1.0 y la 20.2.2.0, acepta las dos.
2. Procede a realizar la importación a la VRF, revisando el Route Distinguisher de importación que encuentra en la ruta es el mismo de la VRF winnie.
3. Instala las dos rutas dentro de la tabla de enrutamiento winnie, con su next hop 10.1.1.1
4. Encuentra las etiquetas VPN de las rutas. Para la 20.2.2.0 el campo de la etiqueta de entrada indica que es una ruta VPN (intag=vpn-route) y la etiqueta de salida es 24 (outtag=24). Para la 20.1.1.0 el campo de la etiqueta de entrada indica que es una ruta VPN (intag=vpn-route) y la etiqueta de salida es 23 (outtag=23).
5. Termina la actualización.

Así es como acaba el intercambio de rutas, y así es como se forman las tablas de enrutamiento y transporte de las VPN. Cualquier ruta que se añada a la red del cliente será distribuida por los PE's de la misma forma que acabamos de ver.

La tabla de enrutamiento de la VPN Dragonball se forma con el intercambio de mensajes similares entre enrutadores PE, claro que se distinguen por llevar un Route Distinguisher diferente.

5.3.7 Análisis del funcionamiento de la red VPN-MPLS

La configuración de la red VPN-MPLS se ha terminado satisfactoriamente, hemos analizado como se van formando las tablas de enrutamiento y transporte MPLS y VPN, la siguiente es una demostración de que la conmutación y el transporte de paquetes dentro de la red funcionan de manera adecuada, tal y como la teoría lo indica para una red VPN-MPLS.

Una manera de probar que realmente existe comunicación entre los sitios de la red, es con el envío de un PING, el cual nos sirve para probar conexión, si el host remoto nos responde una solicitud de ECO, es decir el PING, se puede tener la seguridad de que el mensaje fue recibido y por lo tanto los paquetes están viajando correctamente y el host remoto también esta funcionando correctamente.

Otra manera de comprobar el funcionamiento es enviando un Telnet (solicitud de conexión de terminal remota) de un extremo a otro, para establecer una conexión de terminal virtual con el equipo remoto. La aplicación Telnet esta identificada por el puerto 23 de TCP.

La prueba que he escogido para comprobar el funcionamiento de la red que implemente, es la solicitud de una conexión de terminal remota, Telnet, desde Igor (20.3.3.3) hasta Tigger (20.1.1.1). Al parejo de esta prueba de conexión, realizaré un análisis de los paquetes reales que viajan a través de las interfaces que conectan los nodos de la red MPLS, para lo cual recurriré a otro equipo conocido como analizador de protocolos, el cual es sólo un auxiliar en la demostración del funcionamiento de la red y no interfiere en su desempeño ya que solamente realiza mediciones.

En total voy a utilizar tres analizadores de protocolos, distribuidos como se muestra en la figura 5.3, con los cuales iré capturando en cada enlace de la red MPLS los paquetes que viajan a través de éstos, para después hacer un análisis de como se va dando la conmutación de etiquetas hasta que el paquete alcance su destino. El análisis ofrece la oportunidad de ver cada uno de los protocolos que conforman el paquete, es decir la trama completa y así respaldar la teoría introducida en los capítulos anteriores.

Al momento de realizar la prueba para comprobar el funcionamiento de la red, los analizadores de protocolos conectados en los enlaces Mexdi_PE - Mexdo_P, Mexdo_P-Gdldo_P y Gdldo_P-Gdldi_PE, realizan la captura de las tramas.

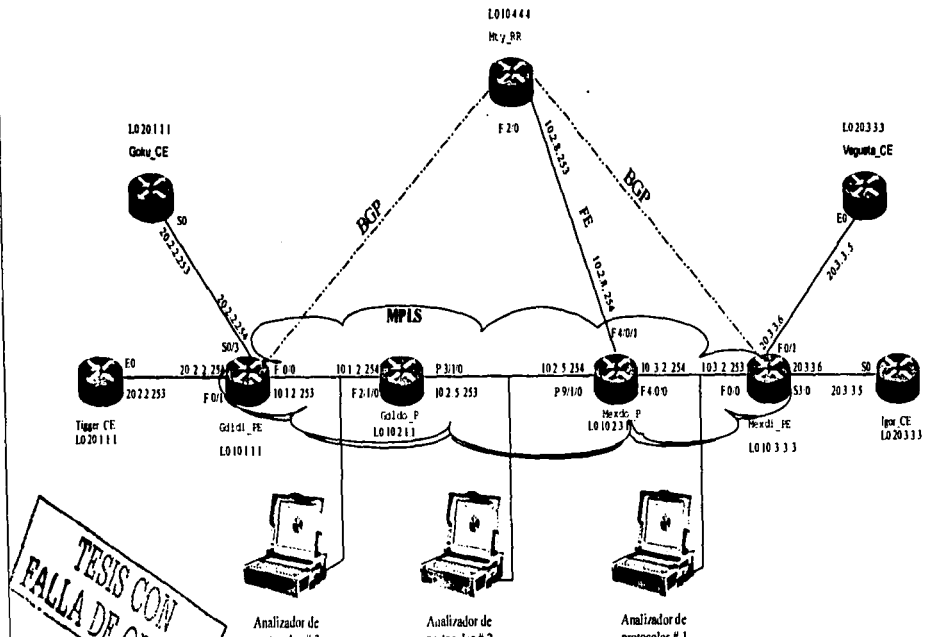
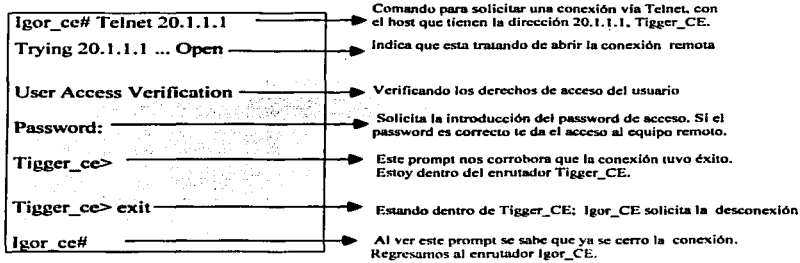


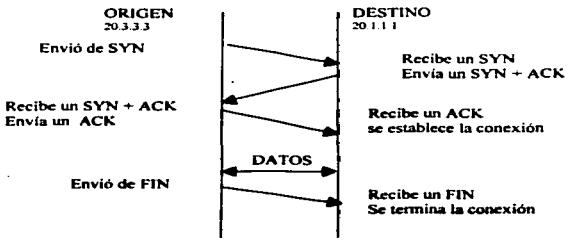
Figura 5.3 Diagrama con la conexión simultánea de tres analizadores de protocolos

Solicitud de conexión

Para establecer la conexión al sitio remoto es necesario utilizar un comando Telnet y la dirección IP del host remoto al que queremos conectarnos. El cuadro siguiente muestra la información que se desplegó en la pantalla de la PC, al momento que solicité la conexión con el sitio Tigger_CE desde el sitio Igor_CE.



La conexión tuvo éxito, quiere decir que la red funciona correctamente. Como lo mencionamos al momento de realizar la conexión los tres analizadores de protocolos hicieron la captura de los paquetes que pasaron por la red; como la aplicación Telnet es reconocida a través del puerto 23 de TCP, la secuencia de la conexión debió haberse establecido por medio de TCP como sigue:



Como se puede observar en esta secuencia, existió un intercambio de paquetes entre el origen y el destino; generando paquetes de solicitud (de Igor a Tigger) y de respuesta (de Tigger a Igor).

Los ejemplos 5.14, 5.15 y 5.16 muestran la captura de la trama de datos que viajó a través de los enlaces MPLS portando la solicitud de conexión SYN del cliente Igor para Tigger; los ejemplos 5.17, 5.18 y 5.19 muestran el paquete de datos que Tigger genero respondiendo a Igor con un syn + ack, el cual viajó a través de los nodos MPLS y fué capturado por los analizadores de protocolos. Teniendo ya la captura de las tramas que se generaron en ambos sentidos de la red, me permito hacer el siguiente análisis de las etiquetas que cada paquete portó hasta alcanzar su destino.

Revisión de las tramas de datos que pasaron por los enlaces de la red MPLS

El análisis de las tramas capturadas por los analizadores de protocolos nos ayudara a comprobar que los paquetes efectivamente conmutaron basándose en las etiquetas que se encuentran en las tablas MPLS y VPN. Ver Diagrama 5.4 para seguir la conmutación de etiquetas que utilizaron estos paquetes.

Siguiendo la secuencia antes vista sabemos que el primer paquete lo originó Igor _CE, por ser el enrutador que solicita la conexión, sabemos que al salir de Igor este paquete es IP puro, con dirección origen 20.3.3.3 y destino 20.1.1.1.

Al entrar al enrutador Mexdi_PE# éste debe agregar al paquete dos etiquetas la primera es la de VPN y la segunda la de MPLS, haciendo el análisis de la trayectoria con las tablas obtenemos la siguiente información:

```
Mexdi_Pe# sh ip bgp vpnv4 vrf winnie tags

   Network          Next Hop        In tag/Out tag
   20.1.1.0/24      10.1.1.1            notag/23
```

En esta primera tabla se muestra la etiqueta VPN 23 que deberá ser insertada en el paquete para alcanzar el destino 20.1.1.1, aquí es importante recordar que esta etiqueta debe mantenerse constante al pasar por cada nodo en la red, hasta llegar al next hop marcado. Revisando los ejemplos 5.14, 5.15 y 5.16 puede comprobarse que esta información cumple con la información insertada dentro de la trama de datos.

En la tabla anterior se observa que la dirección IP 10.1.1.1 es el next hop para alcanzar el destino 20.1.1.1, así que para saber como llegar a la dirección del next hop el enrutador recurre a la tabla TFIB/LFIB, y le asigna la etiqueta MPLS, la cual ira conmutando al pasar por cada nodo de la red hasta llegar a la dirección 10.1.1.1, esta etiqueta será la 30 como se muestra en la tabla siguiente:

```
Mexdi_Pe#
Local  Outgoing     Prefix          Bytes tag  Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched   interface
21      30          10.1.1.0/24       0         Fa0/0      10.3.2.254
```

En el ejemplo 5.14 podemos comprobar que efectivamente esta es la otra etiqueta que se inserta dentro del paquete, el cual es enviado al next hop 10.3.2.254 (nodo Mexdo_P).

El nodo Mexdo_P debe conmutar la etiqueta de entrada por la de salida como lo muestra el siguiente cuadro y se comprueba con la trama del ejemplo 5.15.

```
Mexdo_P#
Local  Outgoing     Prefix          Bytes tag  Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched   interface

30     30           10.1.1.0/24       224        PO9/1/0    point2point
```

El nodo Gdldo_P , debe conmuta también la etiqueta de entrada por la de salida, al tratar de hacerlo se encontrará con una solicitud de pop tag, como vemos en el cuadro y eliminará la etiqueta MPLS del paquete, enviándole a Gdldi_PE únicamente la etiqueta VPN dentro del paquete, esta información también cumple con la información contenida en la trama del ejemplo 5.16.

```
Gdldo_P#
Local  Outgoing     Prefix          Bytes tag  Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched   interface

30     Pop tag      10.1.1.0/24     14367      Fa2/1/0     10.1.2.253
```

Al momento de llegar el paquete a Gdldi_PE, revisa la etiqueta VPN que trae consigo, la elimina como se lo indica la tabla con un notag y realiza un chequeo del encabezado IP para enviar el paquete hacia su destino Tigger,, con dirección IP 20.1.1.1.

```
  Gdldi_PE#sh ip bgp vpnv4 vrf winnie tags
    Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)
    20.1.1.0/24      20.2.2.253      23/notag
```

Retomando la secuencia de conexión TCP sabemos que el cliente Tigger responde a Igor con un syn+ ack generando de esta forma el segundo paquete IP con dirección origen 20.1.1.1 y destino 20.3.3.3, como sabemos las trayectorias conmutadas por etiquetas son unidireccionales por lo que este paquete tomara otras etiquetas para conmutar hacia hasta su destino, ver diagrama 5.5, el siguiente es el análisis.

Empezamos con la etiqueta VPN que Gdldi_PE le debe insertar al paquete.

```
gdldi_pe#sh ip bgp vpnv4 vrf winnie tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)

 20.3.3.0/24      10.3.3.3        notag/24
```

Como vemos la etiqueta que el paquete debe llevar es la 24, lo cual podemos comprobar fácilmente revisando la trama del ejemplo 5.17, en donde efectivamente se utilizo la etiqueta VPN 24.

En la tabla anterior encontramos que este paquete debe llegar al next hop 10.3.3.3, por lo cual debe buscar en la tabla TFIB que etiqueta MPLS le corresponde a este destino.

```
gdldi_pe#sh tag for
Local   Outgoing     Prefix          Bytes tag   Outgoing    Next Hop
tag     tag or VC    or Tunnel Id    switched    interface

19      28           10.3.3.0/24      0           Fa0/0       10.1.2.254
```

Como podemos observar la etiqueta que debe insertar en el paquete es la 28, esta información también es corroborada por el ejemplo 5.17.

El nodo Gdldo_P debe conmutar la etiqueta de entrada por la de salida como lo muestra el siguiente cuadro y se comprueba con la trama del ejemplo 5.18.

```
gdldo_p#sh tag for
Local   Outgoing     Prefix           Bytes tag   Outgoing    Next Hop
tag     tag or VC    or Tunnel Id     switched    interface

28      29           10.3.3.0/24      224         PO3/1/0     point2point
```

El nodo Mexdo_P , debe conmuta también la etiqueta de entrada por la de salida, al tratar de hacerlo se encuentra con una solicitud de pop tag y eliminará la etiqueta MPLS del paquete, enviándole a Mexdi_PE únicamente la etiqueta VPN dentro del paquete, esta información tambien cumple con la información contenida en la trama del ejemplo 5.16.

```
mexdo_p#sh tag forwarding-table
Local   Outgoing     Prefix           Bytes tag   Outgoing    Next Hop
tag     tag or VC    or Tunnel Id     switched    interface

29      Pop tag      10.3.3.0/24      4858        Fa4/0/0     10.3.2.253
```

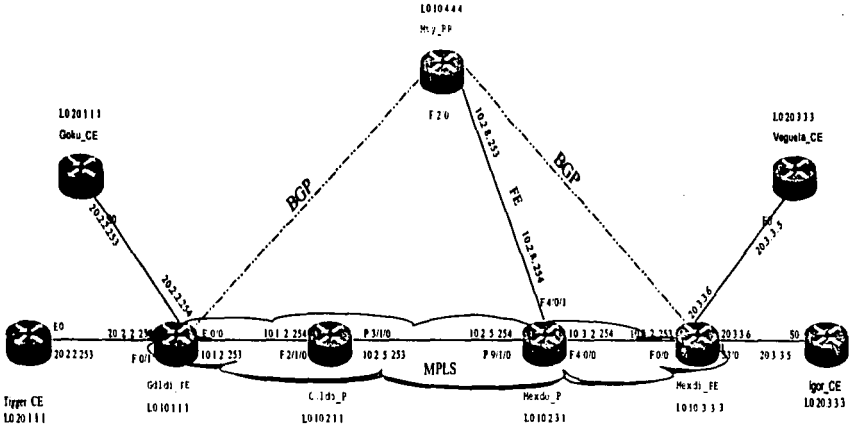
Finalmente al llegar al ultimo nodo MPLS, Mexdi_PE , verifica en su tabla de VPN que debe hacer con esta etiqueta, como podemos observar en la siguiente tabla. éste enrutador debe eliminar la etiqueta y realizar un chequeo del encabezado IP para enviar el paquete a su destino 20.3.3.3. En el ejemplo 5.19 se puede comprobar que el paquete que se dirige a Mexdi_PE solo lleva la etiqueta VPN.

```
mexdi_pe#sh ip bgp vpnv4 vrf winnie tags

   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)

   20.3.3.0/24      0.0.0.0         24/notag
```

En la figura siguiente mostramos el análisis del paquete viajando a través de la red, de Igor a Tigger, con la conmutación de etiquetas que cada enrutador realiza para ese paquete y comprobándolo después con la información obtenida de los analizadores de protocolos.



Encabezado IP Origen 20.3.3.3 Destino 20.1.1.1	23 | Encabezado IP Next hop (MPLS) 10.1.2.253	30 | 23 | Encabezado IP Next hop (MPLS) 10.2.5.253	30 | 23 | Encabezado IP Next hop (VPN) 10.1.1.1 Next hop (MPLS) 10.3.2.254	Encabezado IP Origen 20.3.3.3 Destino 20.1.1.1
Disposición de la Etiqueta MPLS (acción Untag)	Eliminación de etiqueta MPLS (acción Pop Tag)	Conmutación de etiquetas (acción Swap)	Imposición de etiquetas (acción Push)	

Figura 5.4 Paquete MPLS viajando de Igor_CE a Tigger_CE.

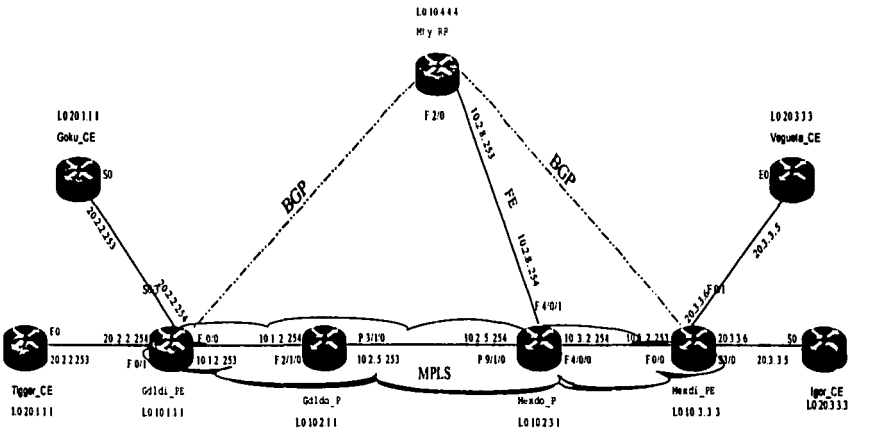


Figura 5.5. Paquete MPLS viajando de Tigger_CE a Igor_CE.

```
Record #4    (From Hub To Node) Captured on 06.26.03 at 19:58:52.675223400 Length =   70
Runtime Frame# 4
---------- ETHER Header ----------
ETHER: Destination: 00-03-31-B2-70-80
ETHER: Source: 00-09-7C-E4-27-00
ETHER: Protocol: MPLS_UCAST
ETHER: FCS: FA80A12F

---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x0001E0FE
MPLS:    0000 0000  0000 0001  1110 ....  .... .... Label = 30
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...0  .... .... Bottom Of Stack (S Bit) = FALSE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254
MPLS: Label Stack Entry[2] = 0x00017FFE
MPLS:    0000 0000  0000 0001  0111 ....  .... .... Label = 23
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254

---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:    0000 00.. DS Codepoint = Default PHB (0)
IP:    .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:    .0.. .... .... .... Don't Fragment Bit = FALSE
IP:    ..0. .... .... .... More Fragments Bit = FALSE
IP:    ...0 0000  0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.3.3.3
IP: Destination address = 20.1.1.1

---------- TCP Header ----------
TCP: Source port = 11050
TCP: Destination port = telnet (23)
TCP: Sequence number = 2929286641
TCP: Ack number = 0
TCP: Data offset = 24
TCP: Flags = 0x02
TCP:    ..0. .... URGENT Flag = FALSE
TCP:    ...0 .... ACK Flag = FALSE
TCP:    .... 0... PUSH Flag = FALSE
TCP:    .... .0.. RST Flag = FALSE
TCP:    .... ..1. SYN Flag = TRUE
TCP:    .... ...0 FIN Flag = FALSE
TCP: Window = 2144
TCP: Checksum = 2F8F
TCP: Urgent pointer = 00000000
TCP: Options = (mss 536)

Record #4    (From Hub To Node) Captured on 06.26.03 at 19:58:52.675223400 Length =   70
  00 03 31 b2 70 80 00 09   7c e4 27 00 88 47 00 01   ..1.p... |.'..G..
  e0 fe 00 01 71 fe 45 00   00 2c 00 00 00 00 fe 06   ....q.E. .,......
  90 c4 14 03 03 03 14 01   01 01 2b 2a 00 17 ae 99   ........ ..+*....
  5d f1 00 00 00 00 60 02   08 60 2f 8f 00 00 02 04   ]..... `..`/.....
  02 18 fa 80 a1 2f
```

Ejemplo 5.14. Captura entre el enrutador Mexdi_Pe y Mexdo_P, analizador # 1

```
Record #5      (P1) Captured on 06.26.03 at 20:00:55.515574635 Length =  58
HDLC:
  Address          = 015
  Frame Type       = 0x00 (Information)
  Control Information = 0x00
   000. .... N(R) = 0
   ...0 .... P/F Bit = 0 (Poll)
   .... 000. N(S) = 0
  FCS              = 0xe9-21 (Good)
Ethertype          = 0x8847 ( MPLS )

---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x0001E0FD
MPLS:    0000 0000  0000 0001  1110 ....  .... .... Label = 30
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...0  .... .... Bottom Of Stack (S Bit) = FALSE
MPLS:    .... ....  .... ....  .... ....  1111 1101 Time To Live = 253
MPLS: Label Stack Enury[2] = 0x000171FE
MPLS:    0000 0000  0000 0001  0111 ....  .... .... Label = 23
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254
---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:    0000 00.. DS Codepoint = Default PHB (0)
IP:    .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:    .0.. .... .... .... Don't Fragment Bit = FALSE
IP:    ..0. .... .... .... More Fragments Bit = FALSE
IP:    ...0 0000  0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.3.3.3
IP: Destination address = 20.1.1.1
---------- TCP Header ----------
TCP: Source port = 11050
TCP: Destination port = telnet (23)
TCP: Sequence number = 2929286641
TCP: Ack number = 0
TCP: Data offset = 24
TCP: Flags = 0x02
TCP:    ..0. .... URGENT Flag = FALSE
TCP:    ...0 .... ACK Flag = FALSE
TCP:    .... 0... PUSH Flag = FALSE
TCP:    .... .0.. RST Flag = FALSE
TCP:    .... ..1. SYN Flag = TRUE
TCP:    .... ...0 FIN Flag = FALSE
TCP: Window = 2144
TCP: Checksum = 2F8F
TCP: Urgent pointer = 00000000
TCP: Options = (mss 536)

Record #5      (P1) Captured on 06.26.03 at 20:00:55.515574635 Length =  58
  0f 00 88 47 00 01 e0 fd   00 01 71 fc 45 00 00 2c   ...G.... ..q.E..,
  00 00 00 00 fe 06 90 c4   14 03 03 03 14 01 01 01   ........ ........
  2b 2a 00 17 ae 99 5d f1   00 00 00 00 60 02 08 60   +*....]. ....`..`
  2f 8f 00 00 02 04 02 18   e9 21                     /........ .!
```

Ejemplo 5.15.Captura entre el enrutador Mexdo_P y Gdldo_P, analizador # 2

```
Record #5      (From Node to Hub) Captured on 06.27.03 at 07:59:05.719098300 Length =   66
 Runtime Frame# 5

---------- ETHER Header ----------
ETHER: Destination: 00-0B-46-5B-89-80
ETHER: Source: 00-07-EC-98-78-48
ETHER: Protocol: MPLS_UCAST
ETHER: FCS: 17217BB2

---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x000171FC
MPLS:     0000 0000  0000 0001  0111 ....  .... .... Label = 23
MPLS:     .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:     .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:     .... ....  .... ....  .... ....  1111 1100 Time To Live = 252

---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:     0000 00.. DS Codepoint = Default PHB (0)
IP:     .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:     .0.. .... .... .... Don't Fragment Bit = FALSE
IP:     ..0. .... .... .... More Fragments Bit = FALSE
IP:     ...0 0000  0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.3.3.3
IP: Destination address = 20.1.1.1

---------- TCP Header ----------
TCP: Source port = 11050
TCP: Destination port = telnet (23)
TCP: Sequence number = 2929286641
TCP: Ack number = 0
TCP: Data offset = 24
TCP: Flags = 0x02
TCP:     ..0. .... URGENT Flag = FALSE
TCP:     ...0 .... ACK Flag = FALSE
TCP:     .... 0... PUSH Flag = FALSE
TCP:     .... .0.. RST Flag = FALSE
TCP:     .... ..1. SYN Flag = TRUE
TCP:     .... ...0 FIN Flag = FALSE
TCP: Window = 2144
TCP: Checksum = 2F8F
TCP: Urgent pointer = 00000000
TCP: Options = (mss 536)

 Record #5      (From Node to Hub) Captured on 06.27.03 at 07:59:05.719098300 Length =   66

   00 0b 46 5b 89 80 00 07   ec 98 78 48 88 47 00 01   ..F[.... ..xH.G..
   71 fc 45 00 00 2c 00 00   00 00 fe 06 90 c4 14 03   q.E..,.. ........
   03 03 14 01 01 01 2b 2a   00 17 ae 99 5d f1 00 00   ......+* ....]...
   00 00 60 02 08 60 2f 8f   00 00 02 04 02 18 17 21   ..`..`/. .......!
   7b b2                                               {.
```

Ejemplo 5.16. Captura entre el enrutador Gdldo_P y Gdldi_PE, analizador # 3

```
Record #6     (From Hub To Node) Captured on 06.27.03 at 07:59:05.723385900 Length =   70
Runtime Frame# 6
---------- ETHER Header ----------
ETHER: Destination: 00-07-EC-98-78-48
ETHER: Source: 00-0B-46-5B-89-80
ETHER: Protocol: MPLS_UCAST
ETHER: FCS: 62D2188B
---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x0001C0FE
MPLS:    0000 0000  0000 0001  1100 ....  .... .... Label = 28
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...0  .... .... Bottom Of Stack (S Bit) = FALSE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254
MPLS: Label Stack Entry[2] = 0x000181FE
MPLS:    0000 0000  0000 0001  1000 ....  .... .... Label = 24
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254
---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:    0000 00.. DS Codepoint = Default PHB (0)
IP:    .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:    .0.. .... .... .... Don't Fragment Bit = FALSE
IP:    ..0. .... .... .... More Fragments Bit = FALSE
IP:    ...0 0000  0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.1.1.1
IP: Destination address = 20.3.3.3

---------- TCP Header ----------
TCP: Source port = telnet (23)
TCP: Destination port = 11050
TCP: Sequence number = 1756416365
TCP: Ack number = 2929286642
TCP: Data offset = 24
TCP: Flags = 0x12
TCP:    ..0. .... URGENT Flag = FALSE
TCP:    ...1 .... ACK Flag = TRUE
TCP:    .... 0... PUSH Flag = FALSE
TCP:    .... .0.. RST Flag = FALSE
TCP:    .... ..1. SYN Flag = TRUE
TCP:    .... ...0 FIN Flag = FALSE
TCP: Window = 4128
TCP: Checksum = F58B
TCP: Urgent pointer = 00000000
TCP: Options = (mss 556)

Record #6     (From Hub To Node) Captured on 06.27.03 at 07:59:05.723385900 Length =   70

  00 07 ec 98 78 48 00 0b    46 5b 89 80 88 47 00 01    ....xH.. F[...G..
  c0 fe 00 01 81 fe 45 00    00 2e 00 00 00 00 fe 06    ......E. ........
  90 c4 14 01 01 01 14 03    03 03 00 17 2b 2a 68 b0    ........ ....+*h.
  c9 6d ac 99 5d f2 60 12    10 20 f5 8b 00 00 02 04    .m..].`. . ......
  02 2c 62 d2 18 8b                                     ..b...
```

Ejemplo 5.17. Captura entre el enrutador GdldiPE y GdldoP . analizador # 3

```
Record #6      (P2) Captured on 06.26.03 at 20:00:55.520115035 Length =   58
HDLC:
 Address          = 015
 Frame Type       = 0x00 (Information)
 Control Information = 0x00
  000. .... N(R) = 0
  ...0 .... P/F Bit = 0 (Poll)
  .... 000. N(S) = 0
 FCS               = 0x78-94 (Good)

Ethertype          = 0x8847 ( MPLS )
---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x0001D0FD
MPLS:    0000 0000  0000 0001  1101 ....  .... .... Label = 29
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...0  .... .... Bottom Of Stack (S Bit) = FALSE
MPLS:    .... ....  .... ....  .... ....  1111 1101 Time To Live = 253
MPLS: Label Stack Entry[2] = 0x000181FE
MPLS:    0000 0000  0000 0001  1000 ....  .... .... Label = 24
MPLS:    .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:    .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:    .... ....  .... ....  .... ....  1111 1110 Time To Live = 254
---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:    0000 00.. DS Codepoint = Default PHB (0)
IP:    .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:    .0.. .... .... .... Don't Fragment Bit = FALSE
IP:    ..0. .... .... .... More Fragments Bit = FALSE
IP:    ...0 0000 0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.1.1.1
IP: Destination address = 20.3.3.3
---------- TCP Header ----------
TCP: Source port = telnet (23)
TCP: Destination port = 11050
TCP: Sequence number = 1756416365
TCP: Ack number = 2929286642
TCP: Data offset = 24
TCP: Flags = 0x12
TCP:    ..0. .... URGENT Flag = FALSE
TCP:    ...1 .... ACK Flag = TRUE
TCP:    .... 0... PUSH Flag = FALSE
TCP:    .... .0.. RST Flag = FALSE
TCP:    .... ..1. SYN Flag = TRUE
TCP:    .... ...0 FIN Flag = FALSE
TCP: Window = 4128
TCP: Checksum = F58B
TCP: Urgent pointer = 00000000
TCP: Options = (mss 556)

Record #6      (P2) Captured on 06.26.03 at 20:00:55.520115035 Length =   58
  0f 00 88 47 00 01 d0 fd   00 01 81 fe 45 00 00 2c   ...G.... ....E...
  00 00 00 00 fe 06 90 c4   14 01 01 01 14 03 03 03   ........ ........
  00 17 2b 2a 68 b0 c9 6d   ae 99 5d f2 60 12 10 20   ..+*h..m ..].`..
  f5 8b 00 00 02 04 02 2c   78 94                    ....... x.
```


Ejemplo 5.18.Captura entre el enrutador Gdldo_P y Mexdo_P, analizador # 2

```
Record #5      (From Node to Hub) Captured on 06.26.03 at 19:58:52.679938700 Length =    66
Runtime Frame# 5

---------- ETHER Header ----------
ETHER: Destination: 00-09-7C-E4-27-00
ETHER: Source: 00-03-31-B2-70-80
ETHER: Protocol: MPLS_UCAST
ETHER: FCS: 924357B0

---------- MPLS Header ----------
MPLS: Label Stack Entry[1] = 0x000181FC
MPLS:     0000 0000  0000 0001  1000 ....  .... .... Label = 24
MPLS:     .... ....  .... ....  .... 000.  .... .... Experimental Use = 0
MPLS:     .... ....  .... ....  .... ...1  .... .... Bottom Of Stack (S Bit) = TRUE
MPLS:     .... ....  .... ....  .... ....  1111 1100 Time To Live = 252

---------- IP Header ----------
IP: Version = 4
IP: Header length = 20
IP: Differentiated Services (DS) Field = 0x00
IP:     0000 00.. DS Codepoint = Default PHB (0)
IP:     .... ..00 Unused
IP: Packet length = 44
IP: Id = 0
IP: Fragmentation Info = 0x0000
IP:     .0.. .... .... .... Don't Fragment Bit = FALSE
IP:     ..0. .... .... .... More Fragments Bit = FALSE
IP:     ...0 0000  0000 0000 Fragment offset = 0
IP: Time to live = 254
IP: Protocol = TCP (6)
IP: Header checksum = 90C4
IP: Source address = 20.1.1.1
IP: Destination address = 20.3.3.3

---------- TCP Header ----------
TCP: Source port = telnet (23)
TCP: Destination port = 11050
TCP: Sequence number = 1756416365
TCP: Ack number = 2929286642
TCP: Data offset = 24
TCP: Flags = 0x12
TCP:     ..0. .... URGENT Flag = FALSE
TCP:     ...1 .... ACK Flag = TRUE
TCP:     .... 0... PUSH Flag = FALSE
TCP:     .... .0.. RST Flag = FALSE
TCP:     .... ..1. SYN Flag = TRUE
TCP:     .... ...0 FIN Flag = FALSE
TCP: Window = 4128
TCP: Checksum = F58B
TCP: Urgent pointer = 00000000
TCP: Options = (mss 556)

Record #5      (From Node to Hub) Captured on 06.26.03 at 19:58:52.679938700 Length =    66

  00 09 7c e4 27 00 00 03    31 b2 70 80 88 47 00 01    ..|.'... 1.p..G..
  81 fc 45 00 00 2c 00 00    00 00 fe 06 90 c4 14 01    ..E.,... ........
  01 01 14 03 03 03 00 17    2b 2a 68 b0 c9 6d ae 99    ........ +*h..m..
  5d f2 60 12 10 20 f5 8b    00 00 02 04 02 2c 92 43    ].`.. .. .....C
  57 b0                                                 W.
```

Ejemplo 5.19. Captura entre el enrutador Mexdo_P y Mexdi_PE, analizador # 1

Al final revisando los paquetes de datos se puede comprobar que efectivamente portan dos etiquetas, una que permanece constante en toda la trayectoria, etiqueta VPN y otra que conmuta al pasar por cada nodo, etiqueta MPLS y que estas etiquetas corresponde exactamente a las que se encuentran en las tablas que cada enrutador generó en su momento.

En los paquetes que se capturaron en el analizador de protocolos podemos observar que efectivamente el primer paquete que viaja desde la dirección 20.3.3.3 hasta la dirección 20.1.1.1 lleva una solicitud **syn**. Este paquete al pasar por tres puntos clave nos permite analizar la conmutación real de etiquetas MPLS, que tal como lo hicimos anteriormente con sólo las tablas y como la etiqueta VPN en este caso la # 23 se mantiene intacta en todo el trayecto hasta llegar al último PE. También se puede corroborar el pop tag que Gdldo_P le aplica al paquete ya que el desplegado de este paquete antes de entrar al enrutador Gdldo_P lleva dos etiquetas la 30 de MPLS y la 23 de VPN, mientras que el desplegado del paquete dirigiéndose a Gdldi_PE viaja solamente con la etiqueta 23 de VPN, debiéndose por supuesto a un pop tag solicitado anteriormente. Al llegar el paquete a Gdldi_PE, este elimina la última etiqueta y envía un paquete IP puro hacia el destino Tigger_CE.

Tigger_CE al recibir la solicitud responde a Igor_CE (20.3.3.3) con un **syn + ack**, originándose así el segundo paquete que viaja a través de la red, el comportamiento es el mismo al pasar por cada nodo la etiqueta MPLS ve cambiando, como lo refleja los desplegados de cada analizador y la etiqueta VPN en este caso la etiqueta 24 que corresponde al destino 20.3.3.3 se mantiene constante hasta alcanzar el último PE. En estos desplegados también se puede comprobar el **pop tag** que Mexdo_P le hace al paquete antes de enviarlo hacia Mexdi_PE, en el desplegado del analizador # 2 (ejemplo 5.18) se ve el paquete entrando a Mexdo_P con la etiqueta 29 de MPLS y la 24 de VPN, al salir de Mexdo_P éste ya ha eliminado la etiqueta 29 y se dirige a Mexdi_PE con sólo la etiqueta 24, que identifica la VPN, ver desplegado del analizador # 1 (ejemplo 5.19).

CONCLUSIONES

Las VPN's cumplen su objetivo de entregar conectividad implementada sobre una infraestructura compartida con las mismas políticas que se disfrutan en una red privada. A través de esta tesis se implemento el servicio VPN sobre una infraestructura IP, en donde gracias a que se construyen tablas de transporte y enrutamiento para cada VPN, los negocios de los clientes que están corriendo sobre un servicio VPN disfrutan de la misma seguridad, calidad de servicio, confiabilidad y manejabilidad que ellos tienen en su propia red privada, ya que aún cuando el tráfico viaje sobre una infraestructura compartida, éste nunca podrá invadir una VPN a la que no pertenezca, además, gracias a esta separación de tablas de enrutamiento los proveedores de servicio pueden manejar el trafico de diferentes clientes que tengan los mismos prefijos de direcciones IP en su red, es decir, manejan direcciones IP duplicadas sin ningún problema.

La implementación VPN-MPLS ofrece a los empresarios una solución no solo para establecer accesos remotos sino también para remplazar un legado de redes costosas y de difícil administración. Esto es de alguna forma la construcción de un nuevo mundo en redes. En donde las VPNs son la piedra angular de los servicios del nuevo mundo.

Cuando son implementadas propiamente, las VPN's pueden perfeccionar la operación de las redes mientras reducen los costos de capital. Este cambio fundamental en estrategia abre las oportunidades para un continuo crecimiento, incrementa la rentabilidad y la eficiencia para ambos, proveedor de servicio y clientes. En el viejo mundo el proveedor de servicio destaca el transporte de nivel bajo, como son las líneas alquiladas y Frame Relay. En el nuevo mundo el proveedor de servicio hace equipo con los negocios de los clientes para enfrentar los requerimientos de sus redes a través de VPN's.

Compañías que formalmente manejan sus propios requerimientos en comunicaciones se están asociando con los proveedores de servicio que pueden ayudarle a desarrollar, crecer y manejar sus redes en una escala global. Para la mayoría de ellos, el punto de partida es el conectar grupos de trabajo ampliamente dispersos en una manera eficiente y con un costo efectivo. De allí, el proveedor de servicio usa la tecnología del Core (núcleo de la red) como una fundación para ofrecer los servicios incrementados como son la telefonía viajando en paquetes, videoconferencia, e-comercio y aplicaciones de host.

El pago es sustancial; los proveedores de servicio se vuelven expertos confiables en las necesidades de comunicación de sus clientes, disfrutando de sus ingresos incrementados mientras se distinguen en un altamente competitivo y lucrativo mercado de trabajo. Las VPN ayudan al proveedor de servicio a construir la lealtad del cliente mientras entregan servicios de red que son parte fundamental para las operaciones de los negocios de sus clientes.

Resumiendo los beneficios que las VPN's aportan diremos que; para los proveedores de servicio las VPN's son la clave para mantenerse competitivos en los siguientes años y a los clientes les garantiza que sus aplicaciones atravesaran la red de manera segura y confiable, mejorarán la conectividad y tendrán la oportunidad de reducir los costos.

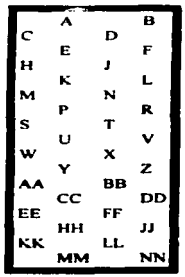
APÉNDICE A
Especificaciones de las diferentes Interfaces utilizadas en las redes LAN y WAN.

Dentro de las redes existe varios estándares que definen los atributos físicos y lógicos de las interfaces como son el número de Pines en el conector, las señales eléctricas en cada Pin.

Entre las más usuales encontramos la norma RS-232C, la cual define una interfaz de capa física de baja velocidad, menor a 20 kbps en una distancia menor a 15 m. En donde las especificaciones eléctricas están definidas en la norma V.28, las funcionales por la norma V24, esta interfaz maneja un conector de 25 pines, DB-25, con los siguientes señales principales, asociadas a cada pin:

Función	Código	# de Pin
Frame ground	101/FGND	1
Signal ground	102/SGND	7
Transmit Data	103/TD	2
Receive Data	104/RD	3
Request to Send	105/RTS	4
Clear to Send	106/CTS	5
Data Set Ready	107/DRS	6
Data Terminal Ready	108/DTR	20
Data Carrier Detect	109/DCD	8

Otra recomendación es la V.35, la cual maneja una transmisión de datos desde 48 kbps hasta 4 Mbps, esta interfaz requiere un conector cuadrado llamado Winchester de 34 pines, distribuidos de la siguiente manera:



Winchester de 34 pines

Función	Codigo	# de Pin
Frame ground	101/FGND	A
Signal ground	102/SGND	B
Transmit Data	103/TD	P S
Receive Data	104/RD	R T
Request to Send	105/RTS	C
Clear to Send	106/CTS	D
Data Set Ready	107/DRS	E
Data Terminal Ready	108/DTR	H
Data Carrier Detect	109/DCD	F
Terminal Tx clock	113/TTC	U W
Tx clock	114/TC	Y AA
Rx clock	115/RC	V X

La siguiente interfaz es la RS-449. esta es una versión más rápida de RS-232, velocidad de hasta 2 Mbps, con capacidad de utilizar cables más largos, está definida para dos conectores; un DB-37 y un DB-9. Las características eléctricas están definidas por las normas siguientes:

EIA-422: Transmisión Balanceada de alta velocidad sobre dos hilos para cada señal. Características eléctricas compatibles con la norma V.11.
EIA-423: Transmisión no Balanceada de RS-449 para compatibilidad con RS-232. Características eléctricas compatibles con la norma V.10.

Las velocidades que cada norma maneja son las siguientes:

Norma	Distancia		
	10 m	100m	1000m
422	10 Mbps	1 Mbps	100 kbps
423	100 kbps	10 kbps	1 kbps

Las especificaciones de IEEE 802.3. es decir, 10baseT, 100baseT y 1000baseT, utilizan un conector estándar de 8 hilos, conocido como RJ-45. Esta interfaz Ethernet utiliza dos pares de alambres para las señales de transmisión y dos para la recepción. Los pines que se utilizan son los siguientes:

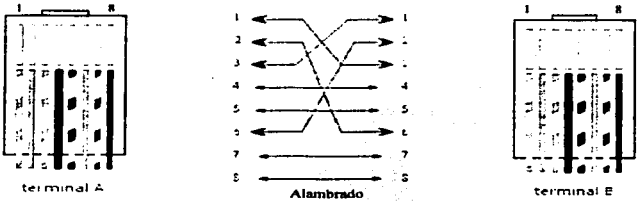
Función	**Código**	**# de PIN**
Transmisión	TX	1
Transmisión	TX	2
Recepción	RX	3
Recepción	RX	6

La conexión de una interfaces Ethernet con cable UTP puede hacerse de dos formas. la primera es una conexión 1 a 1, con el cual podemos conectar por ejemplo una computadora a un switch o a un Hub (conexión DTE a DCE); la segunda es una conexión conocida como nula o cruzada. con la cual se puede conectar una computadora a otra computadora. un enrutador a otro enrutador o dos switches (DCE a DCE).

Los siguientes diagrama muestran como se debe alambrar para conectar los pines en cada una de estas conexiones.

Conexión 1-1

Conexión cruzada



APÉNDICE B

Tablas TFIB y tablas de transporte y enrutamiento VPN, de cada uno de los enrutadores utilizados dentro de la implementación de la red VPN-MPLS del capítulo V.

```
mexdi_pe#sh ip bgp vpnv4 vrf winnie tags

   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)
   20.1.1.0/24      10.1.1.1        notag/23
   20.2.2.0/24      10.1.1.1        notag/24
   20.3.3.0/24      0.0.0.0         24/notag
   20.3.3.4/30      0.0.0.0         25/aggregate(winnie)
```

Tabla de transporte y enrutamiento de la VPN winnie dentro del enrutador Mexdi_PE

```
mexdi_pe#sh ip bgp vpnv4 vrf dragonball tag
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:2 (dragonball)
   20.1.1.0/24      10.1.1.1        notag/26
   20.2.2.0/24      10.1.1.1        notag/25
   20.3.3.0/24      20.3.3.5        26/notag
   20.3.3.4/30      0.0.0.0         27/aggregate(dragonball)
```

Tabla de transporte y enrutamiento de la VPN dragonball dentro del enrutador Mexdi_PE

```
mexdi_pe#sh tag-switching forwarding-table

Local  Outgoing    Prefix           Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched   interface
16     Pop tag     10.2.8.0/24      0          Fa0/0       10.3.2.254
18     28          10.2.1.0/24      0          Fa0/0       10.3.2.254
19     27          10.1.2.0/24      0          Fa0/0       10.3.2.254
20     31          10.4.4.0/24      0          Fa0/0       10.3.2.254
21     30          10.1.1.0/24      0          Fa0/0       10.3.2.254
22     Pop tag     10.2.3.0/24      0          Fa0/0       10.3.2.254
23     Pop tag     10.2.5.0/24      0          Fa0/0       10.3.2.254
24     Untagged    20.3.3.0/24[V]   4060       Se3/0       point2point
25     Aggregate   20.3.3.4/30[V]   0
26     Untagged    20.3.3.0/24[V]   570        Fa0/1       20.3.3.5
27     Aggregate   20.3.3.4/30[V]   0
```

Tabla de TFIB dentro del enrutador Mexdi_PE.

```
mexdo_p#sh tag forwarding-table
Local  Outgoing     Prefix          Bytes tag   Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched    interface
27     Pop tag      10.1.2.0/24     0           PO9/1/0     point2point
28     Pop tag      10.2.1.0/24     1517        PO9/1/0     point2point
29     Pop tag      10.3.3.0/24     4858        Fa4/0/0     10.3.2.253
30     30           10.1.1.0/24     224         PO9/1/0     point2point
31     Untagged     10.4.4.0/24     30255       Fa4/0/1     10.2.8.253
```

Tabla TFIB dentro del enrutador Mexdo_P.

```
gdldo_p#sh tag for
Local  Outgoing     Prefix          Bytes tag   Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched    interface
26     Pop tag      10.2.8.0/24     0           PO3/1/0     point2point
28     29           10.3.3.0/24     224         PO3/1/0     point2point
29     31           10.4.4.0/24     2207        PO3/1/0     point2point
30     Pop tag      10.1.1.0/24     14367       Fa2/1/0     10.1.2.253
31     Pop tag      10.3.2.0/24     0           PO3/1/0     point2point
32     Pop tag      10.2.3.0/24     0           PO3/1/0     point2point
```

Tabla TFIB dentro del enrutador gdldo_P.

```
gdldi_pe#sh tag for
Local  Outgoing     Prefix          Bytes tag   Outgoing    Next Hop
tag    tag or VC    or Tunnel Id    switched    interface
16     26           10.2.8.0/24     0           Fa0/0       10.1.2.254
17     Pop tag      10.2.1.0/24     0           Fa0/0       10.1.2.254
18     29           10.4.4.0/24     0           Fa0/0       10.1.2.254
19     28           10.3.3.0/24     0           Fa0/0       10.1.2.254
20     31           10.3.2.0/24     0           Fa0/0       10.1.2.254
21     32           10.2.3.0/24     0           Fa0/0       10.1.2.254
22     Pop tag      10.2.5.0/24     0           Fa0/0       10.1.2.254
23     Untagged     20.1.1.0/24[V]  114         Fa0/1       20.2.2.253
24     Aggregate    20.2.2.0/24[V]  1238
25     Aggregate    20.2.2.0/24[V]  416
26     Untagged     20.1.1.0/24[V]  1904        Se0/3       point2point
```

Tabla de TFIB dentro del enrutador gdldi_PE.

```
gdldi_pe#sh ip bgp vpnv4 vrf winnie tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1 (winnie)
   20.1.1.0/24      20.2.2.253        23/notag
   20.2.2.0/24      0.0.0.0           24/aggregate(winnie)
   20.3.3.0/24      10.3.3.3          notag/24
   20.3.3.4/30      10.3.3.3          notag/25
```

Tabla de transporte y enrutamiento de la VPN winnie dentro del enrutador gdldi_PE

```
gdldi_pe#sh ip bgp vpnv4 vrf dragonball tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:2 (dragonball)
   20.1.1.0/24      0.0.0.0         26/notag
   20.2.2.0/24      0.0.0.0         25/aggregate(dragonball)
   20.3.3.0/24      10.3.3.3        notag/26
   20.3.3.4/30      10.3.3.3        notag/27
```

Tabla de transporte y enrutamiento de la VPN dragonball dentro del enrutador gdldi_PE

```
mty_rr#sh ip bgp vpnv4 rd 20.0.0.0:1 tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:1
   20.1.1.0/24      10.1.1.1        notag/23
   20.2.2.0/24      10.1.1.1        notag/24
   20.3.3.0/24      10.3.3.3        notag/24
   20.3.3.4/30      10.3.3.3        notag/25
```

Tabla de transporte y enrutamiento de la VPN winnie dentro del enrutador mty_RR.

```
mty_rr#sh ip bgp vpnv4 rd 20.0.0.0:2 tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 20.0.0.0:2
   20.1.1.0/24      10.1.1.1        notag/26
   20.2.2.0/24      10.1.1.1        notag/25
   20.3.3.0/24      10.3.3.3        notag/26
   20.3.3.4/30      10.3.3.3        notag/27
```

Tabla de transporte y enrutamiento de la VPN dragonball dentro del enrutador mty_RR.

APÉNDICE C

Recomendaciones RFC (Request For Comments), para la arquitectura MPLS y VPN's.

RFC 3031 MPLS Architecture

RFC 3032 MPLS Label Stack Encoding

RFC 3036 LDP Specification

RFC 3037 LDP Applicability

RFC 3197 Carrying Label Information in BGP-4

RFC 3063 MPLS Loop Prevention Mechanism

RFC 3443 y **RFC 3032** Time to Live (TTL) Processing en MPLS Networks

RFC 2547 BGP/MPLS VPN's

RFC 2917 A core MPLS IP VPN Architecture

Glosario

AS	Sistema Autónomo, es una colección de redes que están bajo una administración común compartiendo una estrategia de administración común.
BGP	Border Gateway Protocol. Es un protocolo de enrutamiento interdominio designado para el Internet global. El exterior BGP (eBGP), comunica entre diferentes sistemas autónomos, el interior BGP (iBGP), comunica entre enrutadores con el mismo sistema autónomo, está definido en la RFC 1163.
CE	Customer Edge. Enrutador al borde del cliente, es un enrutador que es parte de la red del cliente y conecta al enrutador al borde del proveedor (PE), un CE puede unir cualquier conjunto de Redes Privadas Virtuales (VPN's). Cada CE conecta un sitio del cliente con un PE, obteniendo el servicio VPN para ese sitio. Los CE no están enterados de la VPN asociada
CEF	Cisco Express Forwarding. Es un adelanto en la tecnología de conmutación de capa 3 IP. CEF optimiza el desempeño y escalabilidad de la red, es útil para redes que manejan una gran cantidad tráfico, las tablas de transporte y enrutamiento (VRFs) usan la tecnología CEF, además, las VPN's basadas en MPLS deben tener habilitado el CEF.
Customer	Cliente. Solicita servicio VPN de un proveedor (provider). Cada cliente puede tener sus propios sitios de clientes.
Customer Networt	Red del cliente, es una red que está bajo el control de un cliente final. Las VPN conectan las redes de los clientes individuales conectando los sitios aislados.
Enrutar	Determinar que camino deben de tomar los datos a través de la red, basados en las condiciones de la red, prioridad de servicio y otros factores, como el número de saltos requeridos para alcanzar el destino.

Etiqueta (Tag/Label)	Es un pequeño identificador de valor arreglado que les dice a los nodos conmutadores como enviar los paquetes de datos.
FEC	Forwarding Equivalence Classes (Clase Equivalente de Transporte). Se refiere un grupo de paquetes IP que son enviados de la misma manera, sobre el mismo camino, con el mismo trato de trasporte, es decir, es una forma de agrupar los paquetes IP de una misma clase, el FEC al cual es asignado el paquete es codificado con un identificador, conocido como etiqueta.
FIB	Forwarding Information Base (Base de Información de Transporte), es la tabla que CEF utiliza para lograr que la conmutación de paquetes seas más rápida.
Host	Como host se entiende cualquier computadora personal, mini computadora, enrutador, servidor o cualquier equipo de computo con CPU.
IGP	Interior Gateway Protocol, es un protocolo usado par intercambiar información de enrutamiento con un sistema autónomo. Algunos ejemplos son el iBGP, IGRP, OSPF, EIGRP y RIP.
Ingeniería de Trafico	La ingeniería de tráfico (traffic engineering) es la técnica usada para redirigir trafico a través de la red sobre una trayectoria diferente a la que habría sido escogida si se hubiera usado un método de enrutamiento estándar.
IPv4	Protocolo de Internet (IP) versión 4, es una versión de IP que soporta direcciones de 32 bits.
LFIB	Label Forwarding Information Base. Es la estructura de datos usada para conmutar los paquetes etiquetados. Utilizado en las especificaciones IETF.
LIB	Label Information Base. Es una base de datos usada por los LSR para almacenar etiquetas aprendidas de otros LSR's así como las etiquetas asignadas por el LSR local. Utilizado en las especificaciones IETF.

Loopback address	Dirección loopback. Es una interfaz lógica sobre un enrutador cisco que está siempre arriba (up) y que no está conectada a nada. Una loopback es configurable dentro del enrutador y como no pertenece a ninguna interfaz del enrutador no es vulnerable a sufrir una caída, es decir siempre se mantiene activa a menos que el administrador la desactive manualmente.
LSP	Label Switch Path. Trayectoria conmutada por etiqueta es una secuencia de saltos por los que un paquete debe viajar a través de un mecanismo de conmutación por etiquetas. Un LSP puede ser establecido dinámicamente basándose en los mecanismos de enrutamiento normal. Utilizado en las especificaciones IETF.
LSR	Label Switching Router. Enrutador de capa 3 que envía paquetes basándose en el valor de una etiqueta encapsulada en el paquete.
MPLS	Multi Protocol Label Switching. Es un estándar emergente basado en la tecnología de conmutación de etiquetas.
MPLS-VPN	Red Privada Virtual basada en MPLS. La solución VPN-MPLS es un conjunto de PE's que están conectados en común al backbone (columna vertebral) de la red para proporcionar conectividad IP privada entre dos o más sitios del cliente, para un cliente en particular. Cada VPN maneja un conjunto de políticas.
Next Hop	El next hop es la dirección del próximo enrutador sobre la trayectoria hacia el destino. En el caso particular de rutas VPN que fueron anunciadas por BGP, esta es la direccion del enrutador PE anunciante, es decir, el que anunció la ruta destino, con su respectiva etiqueta.
NLRI	Network Layer Reachability information. BGP envia mensajes de actualización conteniendo NLRI para describir una ruta y como poder llegar ahí, una actualización BGP porta una o más prefijos NLRI y los atributos de la ruta, los cuales incluyen la dirección del next hop BGP, los valores de las comunidades extendidas de BGP y algún otro valor.

PE	Provide Edge Route. Es un enrutador al borde del proveedor que conecta a los enrutadores del cliente, CE. Todos los procesos VPN ocurren en el enrutador PE, cada PE pertenece a exactamente una región de la red del proveedor y conecta uno o más sitios del cliente. Cada enrutador PE puede manejar varias VRF's.
QoS	Quality of service. El mecanismo para proporcionar al administrador de red la habilidad de controlar la mezcla de ancho de banda, el retardo y la perdida de paquetes.
RD	Route Distinguisher. Es un valor de hasta 64 bits que se le agrega a los prefijos IPv4 para crear un prefijo VPN único. Cada VRF tienen un RD diferente.
Red del proveedor de servicio	Es la columna vertebral de una red bajo el control de un proveedor de servicio que proporciona servicio de transporte entre sitios de clientes.
Región	Un grupo de PE dentro de un mismo sistema autónomo
RT	Route Target. Es un valor de 64 bits con el cual el enrutador diferencia las actualizaciones de rutas que insertará dentro de cada VRF
Ruta estática	Ruta que está configurada explícitamente y se introduce en la tabla de enrutamiento. Una ruta estática tiene preferencia sobre las rutas que fueron escogidas por protocolos de enrutamiento dinámico.
TDP	Tag Distribution Protocol. Este Protocolo es usado para distribuir imposiciones de etiquetas a otros LSR's.
TFIB	Tag Forwarding Information Base, termino originado por CISCO y que es equivalente a LFIB, el cual es utilizado por las especificaciones IETF.

TIB Tag Information Base, termino originado por CISCO y que es equivalente a LIB, termino que es utilizado por las especificaciones IETF.

TSP Tag Switching Path, termino originado por CISCO y que es equivalente a LSP, termino que es utilizado por las especificaciones IETF.

Trama Una trama es una estructura lógica y organizada, en la cual los datos pueden ser transportados.

Sitio Es un elemento dentro de la VPN. Una colección de uno o más enrutadores Customer Edge (CE).

Sumarización Es la acción de resumir varios prefijos de red en un solo prefijo. Ejemplo: los prefijos 12.30.40.1, 12.30.40.2 y 12.30.40.3 se sumarizan en un solo prefijo que es 12.30.40.0 / 24, en donde el 24 indica la mascara de subred de 24 bits.

VPN Una Red Privada Virtual es una estructura que proporciona redes privadas IP, sobre una estructura pública. En una solución VPN-MPLS una VPN es un conjunto de sitios de clientes que son configurados para comunicarse a través de un servicio VPN. Una VPN es una red en la cual dos sitos pueden comunicarse sobre la red del proveedor de una manera privada, esto significa que ningún sitio fuera de la VPN puede interceptar sus paquetes o inyectar nuevos paquetes a la red privada. La red del proveedor es configurada de tal forma que solo los paquetes de la VPN pueden ser transmitidos a través de esa VPN, lo que quiere decir que ningún dato entra o sale de la VPN a menos que esto sea especificado a través de la configuración. Existe una conexión física que no se comparte, desde la red al borde del proveedor (PE) hacia la red al borde del cliente (CE).

Vpnv4 Es usado como palabra clave en algunos comandos dentro de los enrutadores para indicar prefijos VPN-IPv4. estos prefijos son direcciones de clientes VPN, cada uno de los cuales fue hecho único al agregársele un distinguidor de 64 bits conocido como Route Distinguisher (RD).

VRF Instancia de Transporte y Enrutamiento VPN. La VRF es un elemento clave en la tecnología VPN-MPLS, las cuales existen solo en los enrutadores PE. Las VRF's son pobladas por rutas VPN y consienten múltiples tablas de enrutamiento en un PE, una VRF es requerida por VPN sobre cada PE en la VPN.

BIBLIOGRAFÍA

Douglas E. Comer
Internetworking with TCP/IP
Segunda ed
Ed. Prentice-Hall New Jersey, 1991
547 pp.

Timothy Parker
Aprendiendo TCP/IP
Segunda ed
Ed. Prentice-Hall Mexico, 1996
480 pp.

Jim Guichard
Ivan Pepelnjak
MPLS and VPN Architectures
Primera ed.
Ed. CISCO Indianapolis, 2001
424 pp.

Cisco Internetwork Desing
Version 3.0
Ed. CISCO Systems, San Jose CA, 1997
600 pp

Documentos de la página de Cisco **http://www.cisco.com**

Using the Border Gateway Protocol for Interdomain Routing

Multiprotocol Label Switching (MPLS)

MPLS Architecture Overview

Multiprotocol Label Switching (MPLS) on Cisco routers

Virtual Private Network Architectures

How Virtual Private Networks Work

Sesiones tecnológicas Networkers 2003

INTRODUCCIÓN A MPLS

Duración 2 horas

IMPLEMENTANDO REDES VPN-MPLS

Duración 2 horas

SOLUCIÓN DE PROBLEMAS DE LAS REDES VPN-MPLS

Duración 2 horas