3 Marco Teórico

En esta sección se presentarán los Antecedentes Teóricos necesarios para participar exitosamente en los proyectos en que me he involucrado. Esta sección asume que el lector cuenta con un conocimiento básico de la operación de redes de datos. El enfoque será el siguiente:

- Breve explicación de teoría de redes de datos
- La importancia del análisis del desempeño de una red de datos
- Metodología de administración de proyectos

3.1 TCP/IP

En el mundo de Internet y de las redes en general, la suite de protocolos TCP/IP es la que domina el mercado. Al percatarse de que TCP estaba realizando funciones de capa 3 y capa 4, se realizó una separación de TCP inicial en dos protocolos: TCP para la capa de transporte e IP en la capa de red, de ahí el nombre final: TCP/IP. El primer estándar formal usado en redes modernas fue la versión 4, de ahí que la versión de IP que primero se uso masivamente fue IPv4.

El crecimiento en la popularidad de la suite de protocolos TCP/IP se debe a muchos factores; como los siguientes:

- Sistema integrado de direccionamiento: TCP/IP lleva integrado un sistema para otorgar direccionamiento a los dispositivos de red en redes de cualquier tamaño[2]. Este sistema está diseñado para permitir a los dispositivos obtener direcciones de capa 3 independientemente de los detalles respecto a las tecnologías de capas 2.

- Diseño para enrutamiento: TCP/IP está diseñado para facilitar el enrutamiento en una red de tamaño arbitrario. Los routers permiten que el intercambio de información entre dispositivos se realice un paso a la vez (enrutamiento *hop-by-hop*). Existen varios tipos de protocolos de enrutamiento, de los cuales se hablará en secciones posteriores.

- Independencia de la tecnología subyacente: TCP/IP puede funcionar en prácticamente cualquier tecnología de capas inferiores, incluidas *Frame Relay*, ATM, PPP, HDLC, SONET. Esta flexibilidad permite que se puedan mezclar diferentes tecnologías de capas inferiores, e interconectarlas usando TCP/IP.

- Proceso de desarrollo y estándares abiertos: los estándares TCP/IP son estándares abiertos disponibles gratuitamente para el público. Son desarrollados siguiendo los procesos democráticos dictados por las RFC, en los cuales todas las partes interesadas son invitadas a participar.

[2] Esto era parcialmente cierto, aunque con el crecimiento que ha tenido Internet, las direcciones de IPv4 disponibles ya se han agotado. Para solucionar ese problema, se ha creado IPv6.

3.1.1 Operación Cliente/Servidor de TCP/IP

TCP/IP opera bajo la estructura cliente/servidor. Los servidores (generalmente con un gran poder de procesamiento) están dedicados a proporcionar servicios a un número mucho mayor de equipos (los clientes). Con este esquema, en lugar de que todos los dispositivos operen como pares (*peers*), tanto clientes como servidores operan como sistemas complementarios.

Generalmente el cliente inicia la comunicación al enviar una petición o solicitud de información a un servidor; éste responde al cliente, dándole la información que solicitó, o con una respuesta alternativa (por ejemplo, un mensaje de error).

Existen varias ventajas al usar el modelo cliente/servidor. Por ejemplo, las aplicaciones de software desarrolladas para el cliente o para el servidor son optimizadas para realizar sus tareas de la forma más eficiente posible.

3.1.2 Modelo TCP/IP

Hay correspondencia natural entre el las capas del modelo TCP/IP y el modelo OSI. Sin embargo esta correspondencia no es uno a uno. La siguiente figura compara ambos modelos. Dado que las funciones de cada capa del modelo TCP/IP son muy similares a las capas equivalentes del modelo OSI, no profundizaré en las características de cada capa.

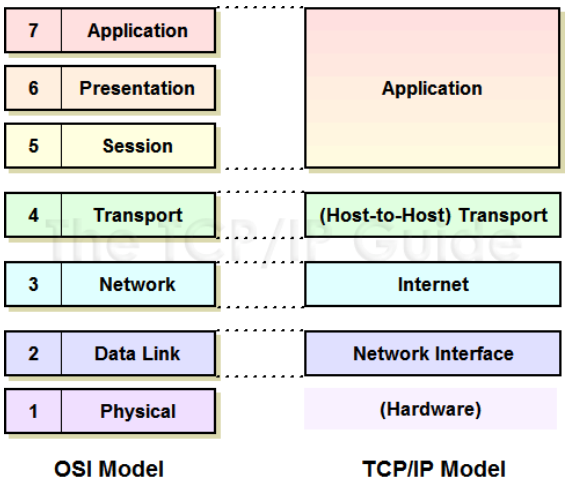

Figura 1. Modelo OSI vs. Modelo TCP/IP

3.1.3 Protocolos de la suite TCP/IP

La figura 2 muestra un diagrama de los protocolos más comunes de la suite TCP/IP, así como la capa en la que residen. Los protocolos más importantes de la suite TCP/IP son: IP (capa de Internet), TCP y UDP (ambos residentes en la capa de transporte). Estos protocolos medulares soportan el funcionamiento de muchos otros protocolos. Dada su importancia, se profundizará al respecto.

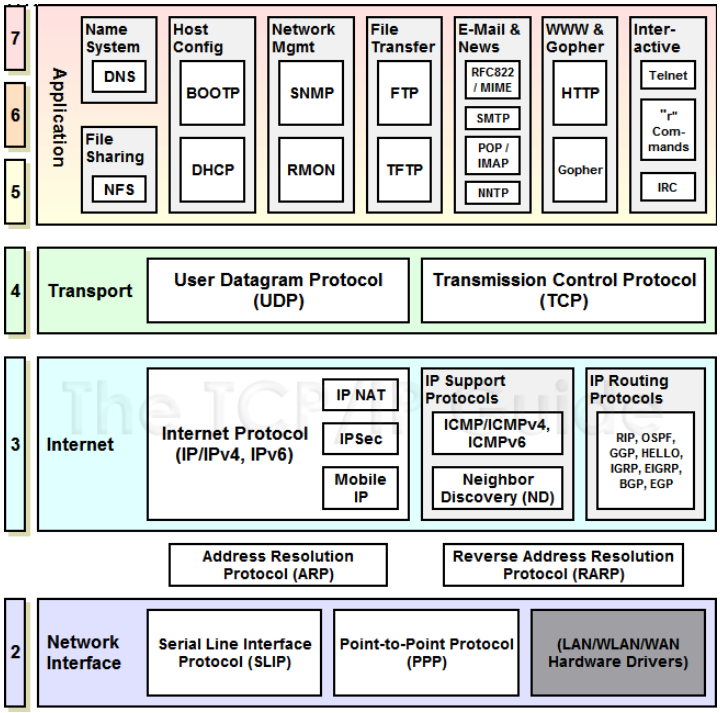

Figura 2. Algunos protocolos de la suite TCP/IP

3.2 Internet Protocol (IP)

Muchos otros protocolos de capas superiores (TCP, UDP, ICMP e IGMP, por citar algunos) transmiten su información mediante datagramas o paquetes[3] IP. IP es un protocolo no confiable (*best-effort*) y no orientado a conexión (*connectionless)*.

Best-effort se refiere a que no existen garantías de que un paquete IP alcance exitosamente su destino. Cuando algo falla en el proceso de transmisión, IP tiene un mecanismo simple de manejo de errores: descartar el paquete y enviar un mensaje a la fuente, indicándole del error.

Connectionless significa que cada paquete se manipula de forma independiente a todos los otros paquetes. Esto implica que los paquetes pueden ser entregados en un orden distinto al cual fueron enviados por la fuente.

A continuación se abordarán algunas de las características más importantes de IP: el formato del encabezado, el enrutamiento de paquetes, así como el direccionamiento IP.

[3] Los términos datagrama y paquete se refieren a los PDUs de capa 3. Por tanto, “paquete” y “datagrama” se usarán de forma indistinta, y tienen el mismo significado.

3.2.1 Formato del encabezado IP

La figura 6 muestra el formato del encabezado IP.

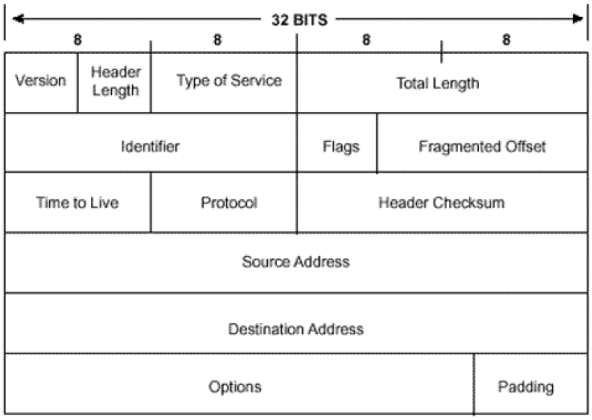

Figura 3. Formato del encabezado de un datagrama IP

Los campos más importantes del encabezado son:

- *Version*: identifica la versión de IP a la cuál pertenece el paquete. Tiene una longitud de 4 bits y regularmente su valor es 0100 binario.

- *Header Length*: tiene una longitud de cuatro bits, la cual indica la longitud total del encabezado IP, expresada en bytes. La longitud mínima del encabezado IP es de 20 *bytes*, mientras que la máxima es de 24 *bytes*. El valor normal de este campo (sin *options*) es 5.

- *Type of Service* (ToS): es un campo de ocho bits que se usa para proporcionar **Quality of Service (QoS)**. Usualmente se le divide en dos subcampos: IP *Precedence* y ToS. Este campo se conoce también como DiffServ desde la introducción del modelo *Differentiated Services Code Point* (DSCP) para la implementación de QoS. Los primeros 6 bits se conocen como los bits DSCP y los dos bits restantes se usan para control de congestión.

- *Total Length*: es un campo de 16 bits que indica la longitud **total del paquete IP** en *bytes*. El tamaño máximo de un paquete IP es 65535 *bytes*. (65KB).

- *Identifier*: tiene una longitud de 16 bits; se le usa en conjunto con los campos *Flags* y *Fragment Offset* para manejar la fragmentación de paquetes.

- *Flags*: es un campo de 3 bits; el primero de ellos no se usa. El segundo es el bit *Don't Fragment*. Si este bit está habilitado (1), entonces se impide la fragmentación de un paquete. El tercer bit es *More Fragments*; cuando hay fragmentación, se habilita este bit en todos los fragmentos excepto en el último, de manera que el dispositivo destino conoce cuándo se han enviado todos los fragmentos.

- *Time-to-Live* (TTL): es un campo de ocho bits que funciona como un "conteo de saltos". A medida que el paquete pasa de router en router, cada uno de ellos irá disminuyendo el valor del TTL en uno.

- *Protocol*: tiene una longitud de ocho bits e indica el número de protocolo de la capa de transporte (o *host*-to-*host*) para quien está destinada la información. Actualmente se tienen asignados

aproximadamente 100 números de protocolo. La siguiente tabla muestra los más comunes de ellos.

Números de protocolos más comunes	
Protocolo	**Protocolo de la capa Host-to-Host**
1	Internet Control Message Protocol (ICMP)
2	Internet Group Management Protocol (IGMP)
6	Transmission Control Protocol (TCP)
17	User Datagram Protocol (UDP)
88	EIGRP
89	Open Shortest Path First (OSPF)

Tabla 1. Números de protocolo más comunes

- *Source Address, Destination Address*: Son las direcciones IP de 32 bits, tanto del equipo que originó el paquete, como del equipo destino. El formato de la dirección IP se cubrirá con mayor detalle en la siguiente sección.

3.2.2 Direccionamiento IP

Las direcciones IP tienen 32 bits de longitud. Teóricamente se cuenta con 2^{32} direcciones disponibles. En un inicio las direcciones fueran asignadas de manera poco eficiente, lo cual provocó un gran desperdicio en las direcciones. En fechas recientes, las direcciones IPv4 se han agotado.

Las direcciones IP constan de dos partes: *Network* ID o *Network Number*, y *Host* ID o *Host Number*.

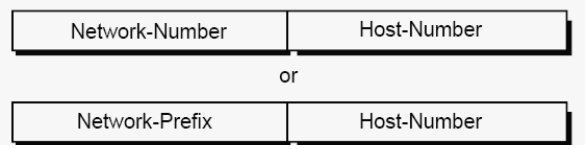

Figura 4. Formato del encabezado de un

La porción de *Network Number* identifica la red donde reside el *host*; el *Host* ID es la parte restante de los bits de la dirección IP; identifica a un dispositivo particular conectado a la red descrita por el *Network* ID. Todos los *hosts* en una red comparten el mismo *Network Number*, pero deben tener un *Host Number* único.

El formato más común para representar una dirección IP es empleando la notación decimal punteada. Los 32 bits de una dirección IP se separan en 8 octetos, cada uno de los cuales se representa con un número decimal del 0 al 255, utilizando puntos para separar cada octeto. La siguiente figura muestra cómo se efectúa este mapeo:

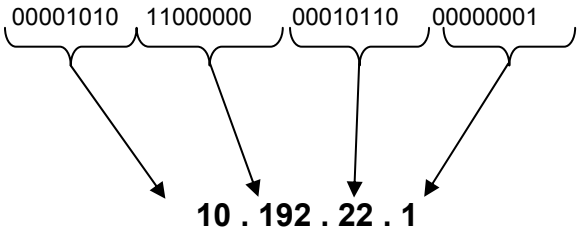

Figura 5. Ejemplo del mapeo de una dirección IP

3.2.3 Direccionamiento classful y clases de direcciones

En lugar de emplear un direccionamiento plano, se definió una jerarquía o clasificación de direcciones, con base en el número de *host*s a los cuales se les deberá dar direccionamiento.

El esquema de direccionamiento *classful* se llama así ya que el espacio de direcciones se separa en clases predefinidas. Cada una de estas clases establece la frontera entre el *Network Number* y el *Host Number* en un punto diferente dentro del espacio de 32 bits. Las diferencias entre estas clases son:

- Redes clase A: La porción del *Network* ID es de 8 bits. Las redes clase A tienen en el bit más significativo del primer octeto el valor "**0**".
- Redes clase B: La porción del *Network* ID es de 16 bits. Las redes clase B tienen en los dos bits más significativos del primer octeto el valor "10".
- Redes clase C: La porción del *Network* ID es de 24 bits. Las redes clase C tienen en los tres bits más significativos del primer octeto el valor "110".

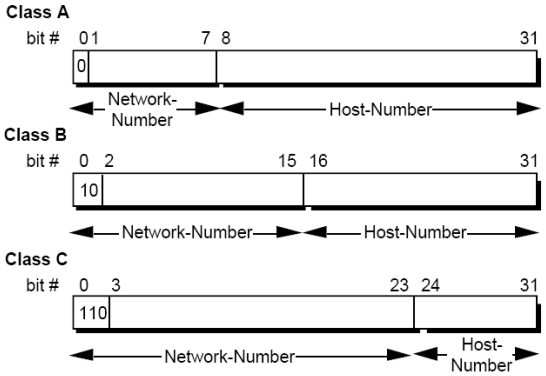

Figura 6. Formatos de las redes clase A, B y C

Existen además otras dos clases de direcciones: Clase D y Clase E. Las clases D se usan para tráfico *multicast*. La siguiente tabla resume las clases de direcciones y sus usos comunes

Clase	Bits del *Network* ID	Bits del *host* ID	Número máximo de redes y Número máximo de *hosts*	Rango de direcciones (notación decimal punteada)	Uso
Clase A	8	24	Red: 126 (2^7-2) *Hosts*: 16,777,214 (2^{24}-2)[4]	**0**.0.0.0 - **127**.255.255.255[5]	Direccionamiento Unicast para organizaciones muy grandes
Clase B	16	16	Red: 16,384 (2^{14}) *Hosts*: 65,534 (2^{16}-2)	**128**.0.0.0 - **191**.255.255.255	Direccionamiento Unicast para organizaciones de tamaño medio a grande
Clase C	24	8	Red: 2,097,152 (2^{21}) *Hosts*: 254 (2^8–2)	**192**.0.0.0 - **223**.255.255.255	Direccionamiento Unicast para organizaciones pequeñas con no más de 250 *hosts*
Clase D	n/a	n/a	n/a	**224**.0.0.0 - **239**.255.255.255	Multicast IP
Clase E	n/a	n/a	n/a	**240**.0.0.0 - **247**.255.255.255	Reservado para uso experimental

Tabla 2. Las diferentes clases de direcciones IP

[4] Se restan dos direcciones de *hosts*, debido a que las direcciones "***all-zero***" y "***all-ones***" no son válidas para un *host*. Se emplean para identificar la subred misma y definir la dirección de *broadcast*, respectivamente.

[5] Las redes 0.0.0.0 y 127.0.0.0 están reservadas.

La principal desventaja del direccionamiento *classful* es el enorme salto en el número de *host*s disponibles que se tienen entre una u otra clase de direcciones. Una clase C permite asignar direcciones para 254 *host*s, mientras que una clase B permite asignar direcciones para 65534. La diferencia entre una clase B y una clase A es aún más dramática (65,534 vs. 16,777,214). La disparidad entre el número de *host*s disponibles obliga a un enorme desperdicio de direcciones; esta es la razón fundamental de la escasez de direcciones IPv4 que provocaron su agotamiento.

Para dar solución a estos problemas se han desarrollado varios métodos, el más importante es VLSM (Variable Length Subnet Masks).

3.2.4 VLSM (Variable-Length Subnet Masks)

El objetivo fundamental de VLSM es usar más eficientemente el espacio de direccionamiento disponible. El *subnetting* ofrece la posibilidad de tener una tercera capa dentro de la jerarquía de direccionamiento IP, y posibilita la administración más eficiente del espacio de direcciones IP disponibles.

Los bits para obtener la dirección de *subnet* se toman de los bits del *Host* ID, lo cual reduce el número de direcciones disponibles para *host*s. Una dirección a la que se ha realizado *subnetting* luce exactamente igual que una dirección que no lo está, por ello se debe contar con un medio que nos permita saber cuántos bits del *Host* ID fueron tomados para definir la dirección de *subnet*. Esto se logra mediante el uso de máscaras de *subnet*.

3.3 TCP y UDP

La capa de transporte es responsable de ciertas funciones específicas. Debido a la diferente naturaleza de las aplicaciones de capas superiores y a sus también variados requerimientos, existen dos protocolos en la capa de transporte: TCP y UDP.

Una de las limitaciones más importantes de IP es que no ofrece garantía de que todos los paquetes de un *flujo* de datos seguirán la misma ruta, y por tanto tampoco hay garantía alguna de que los paquetes lleguen en orden y con el mismo intervalo de tiempo entre ellos.

Estas características se convierten en serios problemas para algunas aplicaciones que confían en que los datos que envían llegarán a su destino sin ninguna pérdida o error, además de que sean entregados en orden. Sin embargo, también existen aplicaciones que no requieren de confiabilidad y no se benefician del hecho de que los paquetes sean entregados en orden.

La solución fue la creación de dos protocolos de capa de transporte: uno de ellos cumpliría las demandas de las aplicaciones que requieran una entrega confiable de datos, sin errores, y en orden, en el entendido de que esto implica un aumento en el *overhead*; el otro protocolo tendría una operación mucho más simple, pero con la ventaja de ser fácil de usar y con poco *overhead*. Dichos protocolos son TCP y UDP.

- *Transmission Control Protocol* (TCP): es un protocolo orientado a conexión y confiable. Permite que un par de dispositivos establezcan una conexión virtual y después intercambien datos bidireccionalmente. La transmisión de datos se administra usando un sistema especial de tamaño de ventana variable (*windowing*), en el cual las transmisiones para las cuales no se recibió un acuse de recibo se detectan y retransmiten automáticamente. Además puede administrar la tasa de transmisión de datos entre dos dispositivos de manera dinámica. Algunos de los protocolos de capas superiores que emplean TCP son HTTP, FTP) y SMTP.

- *User Datagram Protocol* (UDP): es un protocolo sencillo, con muy pocas funcionalidades adicionales. Es no orientado a conexión, las transmisiones no son confiables y los datos se pueden perder en el camino. Su ventaja principal es la velocidad y el reducido *overhead* en que incurre. Su uso más común se encuentra en las aplicaciones interactivas y de tiempo real, en las cuales la retransmisión no tiene ninguna ventaja (VoIP y Video *flujo*ing son sólo algunos ejemplos).

3.4 Enrutamiento IP

Una de las funciones de la IP es el enrutamiento (la determinación de trayectorias lógicas entre un origen y un destino). Esta trayectoria puede atravesar enlaces capa 2 de diversos tipos: *Ethernet*, HDLC, PPP, *Frame Relay*, etc. La información para la determinación de las trayectorias para llegar a un destino se almacena en la tabla de enrutamiento.

3.4.1 La tabla de enrutamiento

Como mínimo, cada entrada en la tabla de enrutamiento debe contener dos elementos:

- Una dirección destino. Es la dirección de la red que el router puede alcanzar. El router puede tener más de una ruta hacia la misma dirección.

- Un apuntador hacia el destino. Este apuntador puede indicar ya sea si la dirección está directamente conectada o indicará la dirección de otro router conectado también a una red directamente conectada. Este otro router se conoce como *next-hop*.

El router buscará hallar una correspondencia entre la dirección destino con la dirección más específica que tenga y que coincida con dicha red.

Si no se puede encontrar una correspondencia entre la dirección destino del paquete y una entrada en la tabla de enrutamiento, el paquete se descarta y se envía una notificación al origen, a través de un mensaje ICMP *Destination Unreachable*.

A continuación se muestra una *subnet* sencilla, mostrando las tablas de enrutamiento para cada uno de los routers involucrados. Las direcciones destino que el router puede alcanzar se encuentran en la columna *Network*. Los apuntadores a los destinos se encuentran en la columna Next Hop.

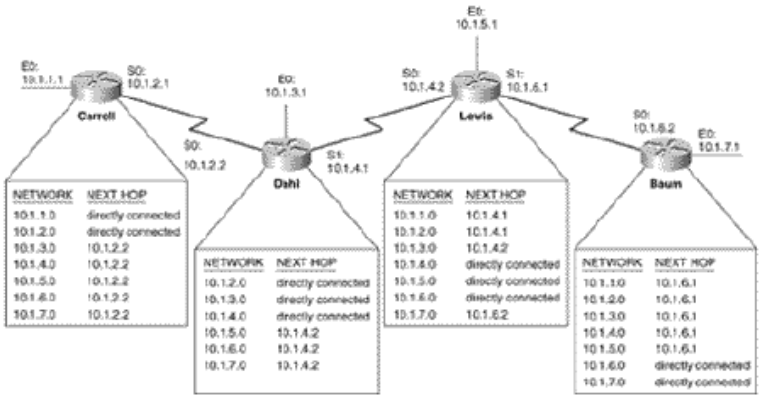

Figura 7. Tablas de enrutamiento

3.4.2 Enrutamiento Estático y Enrutamiento Dinámico

La tabla de enrutamiento obtiene su información mediante dos métodos: estáticamente o de forma dinámica.

El enrutamiento estático es preferido en escenarios donde se requiere de un control preciso de las trayectorias de tráfico. El precio a pagar por esta precisión es la complejidad administrativa: se necesita cambiar manualmente las rutas en todos los routers involucrados cuando existan cambios en la topología.

La otra forma de obtener información de enrutamiento es mediante el uso **protocolos de enrutamiento dinámico**, los cuales no sólo son aplicaciones de obtención de rutas, sino que trabajan en tiempo real realizando un rastreo de la conectividad en la red para proporcionar información de enrutamiento tan actual y válida como sea posible.

Los protocolos de enrutamiento dinámico son más convenientes para los operadores de red en el sentido de la facilidad de administración. El precio a pagar por esta conveniencia es la complejidad en la configuración y la resolución de problemas. Los protocolos de enrutamiento dinámico pueden también requerir un intenso uso de recursos de procesamiento y memoria en los routers. Por ello, el trabajar con protocolos de enrutamiento dinámico requiere de conocimientos y un nivel de dominio avanzados para lidiar con aspectos como el diseño de la red, configuraciones del router, ajuste fino y resolución de problemas.

A continuación sigue una breve explicación acerca de los protocolos de enrutamiento dinámico, sus clasificaciones y características generales.

3.4.3 Protocolos de enrutamiento dinámico

Un protocolo de enrutamiento es el lenguaje que utiliza un router para comunicarse con otros routers para compartir información acerca de las redes que conoce y su estado. Los protocolos de enrutamiento dinámico también determinan la siguiente mejor trayectoria si la mejor trayectoria que se está utilizando actualmente queda inutilizable. La característica más importante de los protocolos de enrutamiento dinámico es la capacidad de adaptar y compensar los cambios en la topología de la red de forma automática.

Existen 8 protocolos de enrutamiento comunes; para que un router se comunique con otro. Evidentemente deben utilizar el mismo protocolo de enrutamiento. Sin embargo, existe un método de hacer un router “multilingüe”: la redistribución entre protocolos de enrutamiento.

Todos los protocolos de enrutamiento se construyen con base en un algoritmo. Como mínimo, un algoritmo para un protocolo de enrutamiento dinámico debe especificar lo siguiente:

- Un procedimiento para transmitir y recibir información acerca de las redes alcanzables
- Un procedimiento para determinar las mejores rutas con base en la información que se obtuvo de otros routers, así como un procedimiento para almacenar esta información en la tabla de enrutamiento.
- Un procedimiento para reaccionar, compensar y anunciar los cambios en la topología de la red

3.4.3.1 Métricas

Cuando un router conoce múltiples rutas hacia un destino dado, debe contar con un mecanismo para determinar qué ruta es la mejor. La métrica es un mecanismo utilizado por todos los protocolos para organizar las rutas, de mayor a menor preferencia. Diferentes protocolos usan diferentes variables como métricas. La tabla 4 muestra las diferentes variables usadas como métricas, así como el significado de cada una de ellas

Variable	Significado

Conteo de saltos (*Hop Count*)	Conteo de saltos de router. Se prefiere la ruta con el menor número de saltos
Ancho de Banda (BW)	Se prefiere la ruta con el mayor ancho de banda
Retraso (*Delay*)	Es una medida del tiempo que le toma a un paquete recorrer una ruta.
Carga (*Load*)	Refleja carga de tráfico en los enlaces a lo largo de una trayectoria. Se prefiere la ruta con la menor carga.
Confiabilidad (*Reliability*)	Esta variable mide la probabilidad de que un enlace falle de alguna forma, y puede ser fija o variable. Se prefiere la ruta con la mayor confiabilidad.
Costo (*Cost*)	Se define con base en las características de un enlace, o también se puede definir arbitrariamente. Se prefiere aquella ruta con el menor costo.

Tabla 4. Variables comunes empleadas como métrica

En algunos casos se puede utilizar más de una variable para obtener la métrica, es decir, se pueden tener métricas compuestas. Ejemplos de protocolos que usan métricas compuestas son EIGRP y BGP (aunque BGP es un protocolo especial, que usa otro tipo de métricas; se hablará más detalladamente de él en las secciones subsecuentes).

3.4.3.2 Convergencia

El proceso de llevar las tablas de enrutamiento de todos los routers de una red a un punto en que coinciden en la información que contienen se conoce como **convergencia**. El tiempo que toma el distribuir la información de enrutamiento a todos los routers de la red, así como que los routers calculen sus mejores trayectorias se conoce como tiempo de convergencia. Casi en todos los casos se prefiere un tiempo de convergencia más rápido; la única excepción es BGP, el cual fue diseñado teniendo en mente sino la escalabilidad y la confiabilidad.

3.4.3.3 Balanceo de carga

El balanceo de carga consiste en distribuir el tráfico entre múltiples trayectorias hacia el mismo destino, para usar el ancho de banda disponible de forma eficiente. El balanceo puede ser de costo igual o costo desigual, además de que se puede realizar un balanceo por paquete o por destino.

3.4.3.4 Distancia Administrativa

Si un router está empleando más de un protocolo de enrutamiento, y aprende una ruta hacia el mismo destino a través de varios protocolos, ¿qué ruta debe seleccionar? Como ya se ha visto, cada protocolo emplea una métrica diferente; el tratar de comparar las métricas de diferentes protocolos es imposible.

La respuesta a este problema es la Distancia Administrativa. Esta variable es una medida acerca de qué tan preferible es una ruta. Cada protocolo tiene asignada una Distancia Administrativa; entre más baja sea esta Distancia, más confiable será el protocolo. Cuando varios protocolos tienen información acerca de una ruta al mismo destino, el factor decisivo será la Distancia Administrativa.

La tabla 5 muestra las Distancias Administrativas para los protocolos de enrutamiento más comunes, según el fabricante Cisco.

Origen	Distancia Administrativa
Interface directamente conectada	0
Ruta estática	1

Ruta sumarizada EIGRP	5
BGP Externo	20
EIGRP Interno	90
IGRP	100
OSPF	110
IS-IS	115
RIP-1/RIP-2	120
EGP	140
EIGRP externo	170
BGP interno	200
Desconocido	255

Tabla 5. Distancias Administrativas para protocolos de enrutamiento

3.4.4 Clasificación de los protocolos de enrutamiento

Existen varias formas de clasificar a los protocolos de enrutamiento. A continuación se muestran las más comunes:

3.4.5 Classful vs. Classless

Los protocolos *classful* no anuncian en sus actualizaciones de enrutamiento las máscaras de red asociadas a cada ruta. Por otro lado, los protocolos *classless* tienen capacidad enviar siempre las máscaras asociadas a cada ruta en sus actualizaciones.

La tabla 6 muestra una clasificación de los protocolos de enrutamiento con base en su naturaleza *classful* o classless.

Classful	*Classless*
RIP v.1	RIP-2
IGRP	EIGRP
	OSPF
	Integrated IS-IS
	BGP

Tabla 6. Protocolos de enrutamiento *classful* y classless

3.4.6 Protocolos Intradominio (Interior Gateway) vs. Protocolos Interdominio (Exterior Gateway)

Un dominio o Sistema Autónomo (*Autonomous System*, AS) de enrutamiento es un conjunto de routers que se encuentran bajo una administración común. Los protocolos de enrutamiento que se emplean al interior de un dominio se conocen como Protocolos de Enrutamiento Interior (*Interior Gateway Protocols*, IGPs). Los protocolos de enrutamiento que se emplean para conectar dominios se conocen como Protocolos de Enrutamiento Exterior (*Exterior Gateway Protocols*, EGPs)[6]. Actualmente existe sólo un protocolo de Enrutamiento Exterior: BGP. Con la excepción de BGP, el resto de los protocolos de enrutamiento en uso son IGPs.

Interior Gateway			*Exterior Gateway*
Distance Vector	*Advanced Distance Vector*	*Link-State*	*Path Vector*
RIP-1	EIGRP	OSPF	BGP

[6] Este nombre tiene su origen en el hecho de que el protocolo que dio origen a BGP se conocía como EGP, *Exterior Gateway Protocol*. Dicho protocolo se encuentra en desuso.

RIP-2		Integrated IS-IS	
IGRP			

Tabla 7. Protocolos de Enrutamiento Interior vs Protocolos de Enrutamiento Exterior

3.4.7 Protocolos de enrutamiento Distance Vector

La mayoría de los protocolos de enrutamiento caen en una de dos clases: *distance vector* o *link state* (exceptuando BGP). Los protocolos *distance vector* se basan en el algoritmo de Bellman-Ford. El nombre *distance vector* se deriva del hecho de que las rutas son anunciadas como vectores del tipo **[distancia, dirección]**. Por ejemplo, el vector [5,X] se interpreta como "el destino A está a una distancia de 5 saltos, en la dirección del *next-hop* X".

Cada router anuncia sus mejores rutas a sus vecinos, desde su propia perspectiva. A los enlaces que conectan a los routers se les asigna un costo o métrica. La métrica asociada con una trayectoria específica hacia un destino dado desde cualquier router es la suma de las métricas de cada uno de los enlaces a lo largo de dicha trayectoria.

3.4.7.1.1 Actualizaciones Periódicas, Triggered y Broadcast

En las primeras implementaciones de los protocolos de enrutamiento, al final de cierto periodo de tiempo (*update timer*) los routers transmitían a todos sus vecinos como *broadcast* (IP 255.255.255.255) todas las rutas en su tabla de enrutamiento. Un problema con este mecanismo es que si las actualizaciones se envían muy frecuentemente, puede ocurrir congestión en enlaces de baja velocidad; por otra parte, si se envían con un intervalo de tiempo muy grande entre ellos, el tiempo de convergencia puede ser inaceptablemente alto.

Si las actualizaciones se enviaran sólo periódicamente, los cambios en la red no se comunicarán lo suficientemente rápido. Las actualizaciones de tipo *triggered* son un mecanismo que permite que un router envíe de forma inmediata los cambios que detecte, en lugar a esperar a que transcurra el siguiente *update timer*.

3.4.7.1.2 Convergencia en Protocolos Distance Vector

Los protocolos *distance vector* son mucho más simples que los protocolos *link state*. Sin embargo, esta simplicidad tiene un precio. Debido a que los routers que emplean estos protocolos confían ciegamente en sus vecinos, estos protocolos son muy propicios a *loop*s de enrutamiento. Un *loop* de enrutamiento ocurre cuando dos nodos apuntan el uno al otro como *next-hop* para el mismo destino. La consecuencia más obvia de estos *loop*s es que prolongan el tiempo que le toma a un router el determinar que una trayectoria ya no es válida o seleccionar una trayectoria alterna, aumentando notablemente el tiempo de convergencia. Por tanto, es deseable que las trayectorias inútiles sean removidas de la red tan pronto como sea posible. Para lograrlo se emplean varios métodos que permiten prevenir o limitar el efecto de los *loop*s de enrutamiento y mejoran la convergencia. Dichos mecanismos son:

- Conteo a infinito

- *Holddown timers*
- *Split Horizon* y *Poison Reverse*
- *Triggered Updates*

Los detalles de dichos mecanismos quedan fuera del alcance de este documento.

3.4.8 Protocolos de enrutamiento Link State.

Los protocolos *link state* se basan en el algoritmo de Dijkstra para encontrar la ruta más corta desde un origen hasta un destino. Los protocolos *link state* reciben información de primera mano de todos sus vecinos. Cada router origina información acerca de sí mismo, sus enlaces directamente conectados, así como el estado de dichos enlaces (de ahí el nombre). Esta información pasa de router en router; cada router, después de recibir la información, hace una copia de dicha información, y la envía a sus vecinos, pero jamás la modifica. El objetivo es que cada router en un área[7] tenga la misma información, y que cada router calcule de forma individual e independiente sus mejores rutas.

Como ejemplos de protocolos de enrutamiento *link state* se tienen, principalmente:

- *Open Shortest Path First* (OSPF)
- *Integrated Intermediate System to Intermediate System* (Integrated IS-IS)

Una característica fundamental de los protocolos Link State es que estos no envían updates de enrutamiento, sino que el intercambio de información se realiza con paquetes diferentes: *Link State Advertisements* o *Link State Updates*.

Los protocolos *link state* requieren el uso de áreas para facilitar el escalamiento. Las áreas se comunican entre sí a través del área *backbone*; esta área constituye el segundo nivel de jerarquía y siempre debe estar presente. Véase la figura 18.

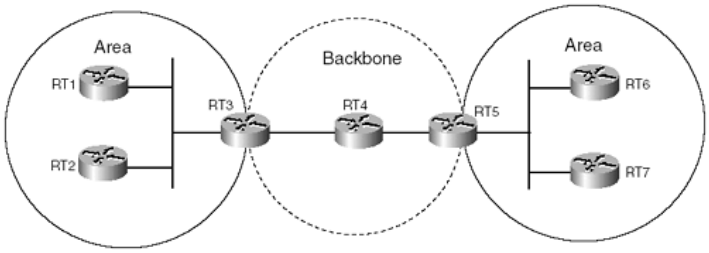

Figura 9. Áreas y jerarquías en un dominio de enrutamiento *link state*

Aunque los protocolos *link state* se consideran más complejos que los *distance vector*, la funcionalidad básica no es compleja:

1. Cada router establece una relación – una adyacencia – con cada uno de sus vecinos.
2. Cada router envía paquetes conocidos como LSAs o LSPs (*Link state Advertisements* o *Link state Packets*) a cada vecino. Se genera un LSA para cada uno de los enlaces del router, identificando el enlace, el estado del mismo, el costo de cada interface que conecta con el enlace, y todos los vecinos que pudieran estar conectados al enlace. Cada vecino, al recibir el LSA, reenvía el mismo a sus vecinos sin modificarlo.

[7] Los protocolos *link state* soportan jerarquías de enrutamiento (también llamadas áreas), las cuales no son más que un subconjunto de los routers que componen una red.

3. Cada router almacena en una base de datos una copia de todos los LSAs que ha recibido. El objetivo es que las bases de datos de todos los routers ubicados dentro de una misma área deben ser idénticas.
4. La base de datos topológica completada (también llamada *Link State Database* LSDB), describe una gráfica de la red. Empleando el algoritmo de Dijkstra, cada router calcula la ruta más corta a cada prefijo de la red e introduce esta información en la tabla de enrutamiento.

Como en el caso de los protocolos *distance vector*, los protocolos *link state* comparten ciertas características comunes que se abordarán a continuación:

3.4.8.1.1 *Link State Flooding*

Después de que las adyacencias se establecen, los routers pueden comenzar a enviar sus LSAs a todos los vecinos. En cambio, cada LSA recibido es copiado a una base de datos y después reenviado a todos los vecinos, excepto por supuesto a aquel vecino que envió originalmente el LSA. Este proceso es el origen de una de las ventajas de los protocolos *link state* sobre los *distance vector*: los LSAs se reenvían de inmediato, mientras que los protocolos *distance vector* deben correr su algoritmo y actualizar su tabla de enrutamiento antes de que los *updates* (incluso los llamados *triggered*) se puedan reenviar. Como resultado, los protocolos *link state* convergen mucho más rápido que los *distance vector*. Existen muchas formas de hacer el *flooding* eficiente y confiable, tales como el uso de direcciones *unicast* y *multicast*, *checksum*s y *acknowledgements*.

3.4.8.1.2 Link state Database.

Otra tarea crítica de los protocolos *link state* es la construcción de la LSDB, también conocida como tabla topológica. En esta estructura de datos se almacenan los LSAs. La información más importante contenida en la tabla es el Router ID del router que envió el LSA, las redes conectadas y el costo asociado con estas redes o vecinos.

La topología de cada área se obtiene corriendo el algoritmo *link state* sobre la LSD. Los detalles acerca de la operación del algoritmo *link state* son extensos y se dejarán fuera de este documento.

3.5 Análisis del Desempeño de una Red

3.5.1 Determinación de una Línea Base

El primer paso para determinar qué tan eficiente es el desempeño de una red involucra el efectuar y comparar varias mediciones de indicadores de desempeño, (tales como ***delay***, ***jitter***, ***throughput***, etc., conceptos que serán definidos más adelante) realizadas durante diferentes periodos de tiempo. De ello pueden surgir preguntas tales como: ¿El desempeño de la red ha mejorado o se ha degradado? ¿Cuál es el efecto de la implementación de una nueva característica o servicio de red en el desempeño? La única manera de conocer la respuesta a estas preguntas es tener un marco de referencia válido contra el cual comparar las mediciones actuales. Este punto de comparación se conoce como línea base, la cual es el nivel de desempeño que es aceptable cuando el sistema está manejando una carga de tráfico típica.

Una línea base ayuda a lograr los siguientes objetivos:

- Identificar los usuarios que emplean el mayor número de recursos
- Mapear diaria, semanal o mensualmente los patrones de utilización de red
- Identificar patrones de tráfico relativos a protocolos específicos
- Justificar el costo de actualizar componentes de red

Para establecer la línea base, se debe medir el desempeño de la red durante su uso típico. Las mediciones no deben realizarse en la hora más saturada del día, y tampoco deben realizarse en las horas de menor utilización. Ambos enfoques arrojarían resultados tergiversados y poco útiles. Un buen enfoque para efectuar las mediciones es tomar lecturas separadas a intervalos espaciados y después obtener el promedio.

La estabilidad, confiabilidad y desempeño de la red es algo obligado. Para un Network Consulting Engineer como yo, es nuestra misión. Como analista de red, debo usar nuestras habilidades para abstraer la información necesaria y después analizarla. En este proceso, se deben extraer métricas precisas para exponer problemas que afectan la operación y el desempeño de la red. Después se deben analizar detalladamente los problemas encontrados para desarrollar una sinopsis técnica concisa, soportada por una recomendación clara y definitiva.

La definición de una línea base debe reflejar de la forma más precisa el estado actual de una red. Por tanto, se debe asegurar que las mediciones estadísticas que caracterizan nuestra red reflejen los hallazgos y recomendaciones. La siguiente figura representa la definición de una línea base de red:

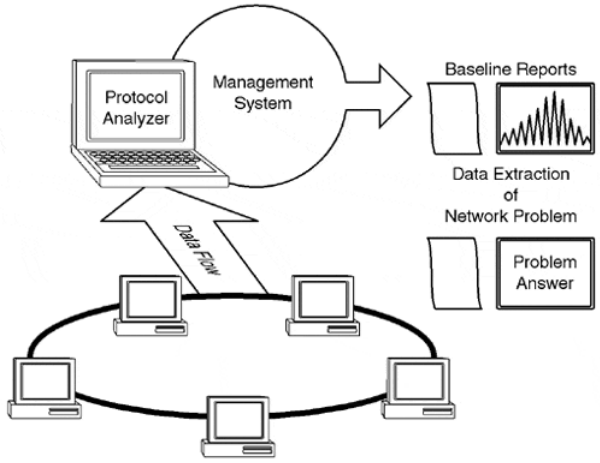

Figura 10. Definición de una línea base de red

Cada día en una red, aparecen muchos escenarios que requieren decisiones bajo la marcha. Cuando se enfrenta un problema de red, el impulso inmediato de un ingeniero de red es reaccionar. Incluso en tiempos cuando la red está estable, el ingeniero de red debe ponderar maneras para mejorar el desempeño global y las características operacionales de la red. Aquí es cuando el ingeniero considera el añadir nuevos productos o características para hacer a la red más efectiva.

Varias razones nos obligan a optimizar una red mediante la definición de una línea base. Una razón importante es la resolución reactiva de problemas. Muchas veces los ingenieros de red encaran situaciones problemáticas que requieren una reacción inmediata. Una línea base de red, por ejemplo, permite aislar ciertos errores debidos a la topología o al medio físico de transmisión, que pueden estar causando problemas intermitentes que afectan la estabilidad de la red. Mediante el análisis empleando herramientas de monitoreo, se pueden localizar los elementos de red que causan estos errores.

Otra razón para realizar un análisis proactivo de la red es asegurar que nuestra red siempre está operando de forma estable y confiable. Considerando el número de problemas reactivos de red con los cuales un ingeniero de red podría encontrarse, esto puede parecer una tarea imposible. El camino para lograr la meta de usar un muestreo de red proactivo y consistente diariamente siempre está impedido por las restricciones de tiempo y recursos que resultan de encarar y solucionar los problemas de red cotidianos.

El monitoreo de las modificaciones de configuración y las implementaciones de nuevos productos es una razón más para emplear una línea base. Como ingeniero de red, debo implementar rápidamente muchos cambios en las redes de nuestros clientes. Algunos de esos cambios o implementaciones son planeadas con anticipación y se realizan de forma cíclica, mientras otros cambios son requeridos de forma dinámica e inmediata, para reaccionar a cambios no previstos.

Finalmente, una de las metas más importantes de la definición de una línea base de red es predecir los efectos de la implementación de nuevas aplicaciones en una red. Hoy, la implementación de aplicaciones es una actividad clave que impulsa el diseño y los ciclos de vida de las Tecnologías de la Información. Los ingenieros de red conocen con certeza que toda la infraestructura de red existe en principio para soportar el transporte de aplicaciones críticas de negocio. Las aplicaciones son las entidades que impulsan la necesidad de tener cimientos fuertes dentro de las infraestructuras LAN y WAN.

Así, se puede emplear una línea base de red y técnicas de optimización para reducir la ocurrencia de problemas que requieren un enfoque reactivo y movernos así hacia el terreno proactivo. Esta es una de las metas fundamentales de un Network Consulting Engineer: nuestra labor no es resolver problemas de naturaleza proactiva, sino por el contrario, el evitar la ocurrencia de los mismos realizando un proceso constante de optimización de las redes de nuestros clientes. Ejemplos fundamentales de las herramientas que empleo para lograr dicha optimización son la Auditoría de Red y el Análisis de Mejores Prácticas de Configuración.

3.5.2 Desempeño de la Red

Al analizar los requerimientos técnicos para un diseño de red, se deben aislar los criterios del cliente para aceptar el desempeño de la red, incluyendo throughput, eficiencia, delay y tiempo de respuesta.

El objetivo de esta sección es ofrecer una manera sencilla de comprender el desempeño de la red, incluyendo conclusiones prácticas que son muy útiles cuando no hay tiempo para un análisis matemático.

El analizar la red del cliente ayuda a determinar qué cambios deben realizarse para cumplir con las metas de desempeño. Dichas metas también están estrechamente ligadas a metas de escalabilidad.

3.5.2.1 Definiciones de Desempeño de Red

La siguiente lista proporciona algunas definiciones de metas de desempeño de red que se pueden usar al analizar requerimientos precisos:

- **Capacidad** (ancho de banda): la capacidad de transporte de datos de un circuito o una red, usualmente medida en bits por segundo.
- **Utilización**: el porcentaje de la capacidad total disponible que está en uso
- **Utilización óptima**: Máxima utilización promedio antes de que la red sea considerada saturada
- **Throughput**: la cantidad de datos libres de errores transferidos exitosamente entre nodos por unidad de tiempo, usualmente segundos.
- **Carga Ofrecida:** suma de todos los datos que todos los nodos de red tienen que enviar en un momento particular
- **Precisión**: la cantidad de tráfico útil que se transmite correctamente, en relación con el tráfico total. Es la razón entre el tráfico sin errores entre el tráfico total.
- **Eficiencia**: es un análisis de qué tanto esfuerzo se requiere para producir una cierta cantidad de throughput de datos
- **Delay** (latencia): El tiempo que pasa desde que un frame está listo para ser transmitido en un nodo y la entrega de dicha trama en otro punto de la red
- **Jitter**: variación del delay. La cantidad de tiempo promedio que varía el delay.

- **Tiempo de respuesta:** la cantidad de tiempo entre el una petición de un servicio de red y la respuesta a dicha solicitud.

3.5.3 Optimización de Red

Las organizaciones de IT están batallando con dos retos opuestos: por un lado, proveer altos niveles de desempeño de aplicaciones[8] para una fuerza de trabajo cada vez más grande y más distribuida, y por otro consolidar una infraestructura costosa para una mejor administración, mejorar la protección de datos y contener los costos.

La optimización de red es el proceso de usar los datos de la línea base de la red para obtener métricas para mejorar la capacidad de la red para alcanzar niveles más altos de desempeño. La optimización es un paso crítico en el diseño de red para todas las empresas y Proveedores de Servicios de Internet hoy en día. Para lograr las metas de negocio, estas organizaciones esperan que sus redes empleen el ancho de banda de forma eficiente, que tengan control del delay y del jitter, y que otorguen un servicio preferencial para aplicaciones esenciales. Fabricantes de equipo de redes como Cisco y organismos de estandarización como el IEEE e IETF ofrecen numerosas opciones para cumplir con las expectativas mencionadas.

Para iniciar con el proceso de optimización de una red, se debe tener un conocimiento sólido de la topología lógica y física de la red en cuestión.

Existen varias áreas de optimización, entre las más importantes se cuentan las siguientes:

- Optimización de la utilización de ancho de banda
- Optimización del delay
- Quality of Service (QoS)
- Técnicas de switching
- Aceleración de aplicaciones en la WAN (WAAS)

A continuación se profundizará en las áreas mencionadas.

3.5.3.1 Optimización de la utilización de ancho de banda: IP Multicast y Wide Area Application Services

Una de las razones principales por las que se requieren técnicas de optimización de red es el uso creciente de aplicaciones multimedia, tales como streaming de video y audio, video en tiempo real, aprendizaje a distancia, herramientas de colaboración en línea, entre otras. Dichas aplicaciones demandan gran BW y son utilizadas por múltiples usuarios simultáneamente, lo cual plantea el reto de que no usen mucho BW o podrían causar problemas en el desempeño de otras aplicaciones y de ellas mismas.

IP Multicast

Una solución para este problema es emplear IP Multicast. Esta solución optimiza la transmisión de todo tipo de tráfico que va de un origen a múltiples destinos. Las aplicaciones multimedia antiguas que no

[8] En esta sección, una aplicación es un conjunto de software que opera en capas 4 a 7 del modelo OSI

usan tecnologías multicast envían un stream de datos a cada usuario. Tales aplicaciones emplean un método unicast (punto a punto) para manejar el tráfico multimedia, lo cual desperdicia el ancho de banda.

Con IP Multicast, un único stream de datos es enviado sólo a aquellas estaciones que lo solicitan, optimizando así el uso de ancho de banda y reduciendo los problemas de desempeño en dispositivos finales y equipos de red.

Empresas, universidades, ISPs y otras organizaciones pueden emplear IP Multicast para muchas aplicaciones que diseminan información, como entrenamiento virtual, reuniones y colaboración en línea, transmisión de noticias, video y audio tanto en tiempo real como en streaming, entre otras. Además, muchos protocolos de red como OSPF, IS-IS, EIGRP, STP, etc., usan Multicast como parte fundamental de su operación regular.

IP Multicast transmite paquetes IP a un grupo de hosts que se identifica por una dirección IP clase D. El rango de éstas va de 224.0.0.0 a 239.255.255.255. Un grupo de multicast se identifica también por una dirección MAC multicast. Usando una dirección MAC multicast se optimiza el desempeño de la red al permitir que las Network Interface Cards (NICs) que no son parte de un grupo ignoren los stream de datos que no van dirigidos a ellas.

Parte fundamental de la operación de IP Multicast es el protocolo Internet Group Management Protocol (IGMP). IGMP permite que un host se una a un grupo multicast e informe a los routers de su necesidad de recibir un stream de datos en particular. Los hosts usan IGMP para reportar su membresía a grupos multicast a routers vecinos.

Cuando un usuario (o un proceso de sistema) inicia una aplicación que requiere que un host se una a un grupo de multicast, el host transmite un mensaje de membresía para informar a los routers en su segmento de red que el tráfico multicast del grupo en cuestión debe enviarse a dicho segmento.

Además de permitir que los hosts se unan a grupos, IGMP especifíca que un router multicast envíe un query a través de cada interface a intervalos regulares, para verificar si todavía hay algún host que pertenezca al grupo multicast. Si lo hay, dicho host (o hosts) responde(n), enviando un mensaje de membresía para cada grupo del cuál es todavía miembro.

Para reducir la utilización de ancho de banda, los hosts establecen un contador aleatorio antes de responder a los queries. Si el host ve otro host respondiendo para un grupo al cual el primer host pertenece, éste cancela su respuesta. El router no necesita saber cuántos o qué hosts en específico pertenecen a un grupo. Simplemente necesita reconocer que un grupo tiene al menos un miembro en un segmento, de modo que envía tráfico a dicho segmento usando las direcciones multicast IP y MAC.

Wide Area Application Services

Por lo que respecta a la red WAN posee una serie de ineficiencias y limitaciones que impactan el desempeño de las aplicaciones, las cuales se están volviendo cada vez más robustas y complejas. Esto las hace más sensibles a las condiciones de red. Aunque el BW no es un factor limitante en una red LAN, este no es el caso en una red WAN.

Las limitaciones de BW e ineficiencias de throughput se pueden definir como sigue:

Ineficiencias de BW: La falta de BW disponible, en conjunto con las ineficiencias de capas de aplicación crean una barrera para el correcto desempeño de las aplicaciones, en particular cuando una aplicación es ineficiente en la forma en que intercambia la información entre nodos.

Algunos de los factores que provocan ineficiencias de BW en la red WAN son:

- Archivos anexos de email redundantes siendo descargados desde la WAN de servidores de email en múltiples ocasiones por múltiples usuarios durante cierto periodo de tiempo.
- Un email con un archivo anexo que es enviado por un usuario en una oficina remota hacia uno o más usuarios en el mismo edificio cuando el servidor de email está en una parte distante de la red.
- Múltiples copias del mismo archivo almacenado en servidores distantes, a los cuales múltiples usuarios tienen acceso sobre la WAN durante cierto periodo de tiempo desde la misma ubicación.
- Un usuario en una oficina remota que tiene acceso a un archivo en un servidor, y después envía una copia de dicho archivo a diferentes personas dentro de la organización.

Limitaciones de Throughput: Las limitaciones de throughput pueden dificultar o entorpecer significativamente el desempeño. Una limitación de throughput se refiere a la incapacidad de una aplicación para aprovechar las recursos de red que están disponibles. Esto es comúnmente un resultado directo de ineficiencias de latencia y BW. A medida que la latencia de aplicación aumenta en una aplicación *send-and-wait*, la cantidad de tiempo que transcurre esperando una respuesta del peer se traduce directamente en tiempo donde la aplicación es incapaz de operar. Aunque muchas aplicaciones permiten que ciertas operaciones se realicen en paralelo – es decir, que no son bloqueadas por el proceso *send-and-wait* -, muchas operaciones críticas para la integridad de los datos, seguridad y coherencia deben ser manejadas de forma serial. En tales casos, antes de que se puedan manipular mensajes subsecuentes, estas operaciones críticas deben ser completadas de forma satisfactoria.

Similarmente, las ineficiencias de BW se relacionan directamente con limitaciones de throughput asociadas con una aplicación dada. A medida que el volumen de datos enviado aumenta, la probabilidad de encontrar congestión también aumenta – no solo en capas inferiores del modelo OSI, sino también en capas superiores. Con la congestión viene la pérdida de paquetes causada por el agotamiento de buffers por falta de memoria para almacenar los datos, lo que lleva a retransmisión, situación que puede aumentar el problema de congestión.

La solución WAAS ofrece una solución para las barreras de desempeño en la WAN mediante el empleo de optimizaciones independientes de la aplicación (también conocidas como optimización WAN) en conjunto con optimizaciones específicas para aplicaciones (también conocidas como aceleración de aplicaciones). La optimización WAN se refiere al empleo de técnicas en la capa de transporte que se aplican en cualquier protocolo de aplicación que usa la red o el protocolo de transporte.

Entre los mecanismos de WAAS que ayudan a optimizar la utilización de BW en la red WAN están:

> **Data Redundancy Elimination (DRE):** es un mecanismo de des-duplicación de datos bidireccional que usa discos duros y memoria RAM para minimizar la cantidad de datos redundantes encontrados en la WAN. DRE aprovecha una historia de compresión por cada peer vagamente sincornizada. Cuando se identifican datos redundantes, el dispositivo WAAS envía una firma referenciando esos datos al peer en lugar de enviar los datos originales, proveyendo así altos niveles de compresión. Los datos no redundantes son añadidos al historial de compresión en ambos peers y es enviado a través de la WAN al peer con firmas recién generadas.

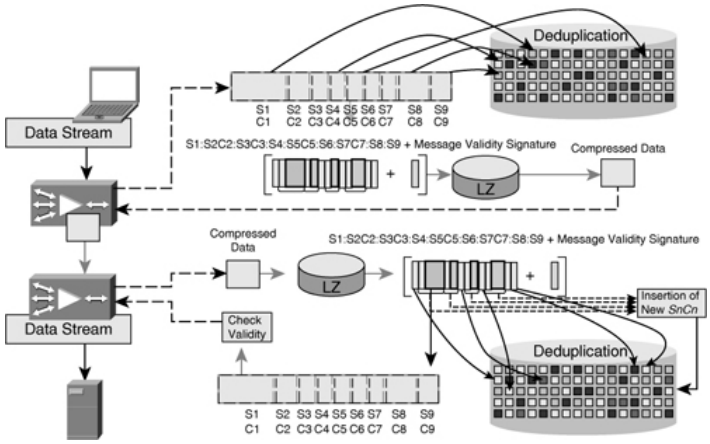

Figura 11. Data Redundancy Elimination

Persistent LZ Compression (PLZ): PLZ emplea compresión Lempel-Ziv con memoria extendida por conexión para proveer altos niveles de compresión, lo que ayuda a minimizar el BW consumido cuando los datos se transfieren sobre la red WAN. PLZ es útil para datos que han sido identificados como no redundantes por DRE y puede comprimir firmas que son enviadas por DRE en lugar de datos redundantes.

Transport Flow Optimization (TFO): TFO es una serie de optimizaciones TCP que ayuda a mitigar las barreras de desempeño asociadas a TCP. TFO incluye el anuncio y negociación de grandes ventanas de transmisión iniciales, acknowledgements selectivos, escalamiento de ventana, buffers grandes y un algoritmo avanzado de prevención de congestión que ayuda a "llenar el tubo" mientras se conserva un balance entre conexiones optimizadas y no optimizadas.

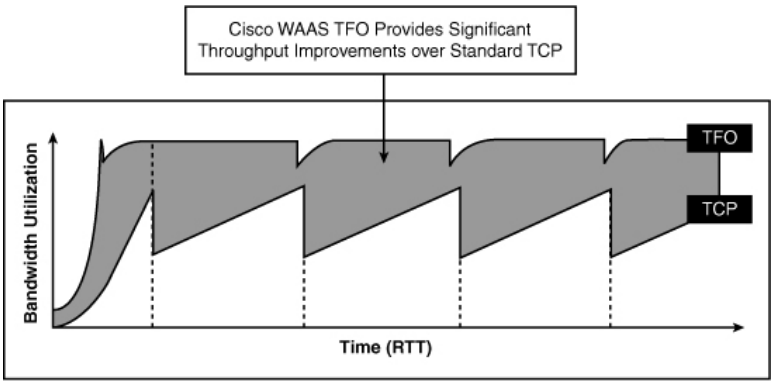

Figura 12. Transport Flow Optimization

Web Cache Communication Protocol

WCCP es una tecnología de content-routing que permite integrar engines de cache en la infraestructura de red.

WCCP permite el uso de otras plataformas que soportan WCCP (como los WAE, Wide Area Application Engines) para localizar patrones de tráfico web en la red, permitiendo que las solicitudes de contenido reciban respuesta de forma local. La localización de tráfico reduce los costos de transmisión y el tiempo de descarga.

WCCP forma parte de WAAS. WCCP se encarga de la redirección de las solicitudes de contenido hacia los WAEs, mientras que estos, empleando tecnologías ya explicadas como DRE, TFO y PLZ se encargan de realizar la optimización a través de la red WAN, como se muestra en la siguiente figura:

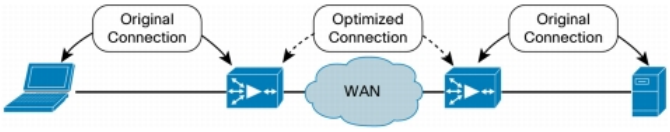

Figura 13. Optimización de conexiones en la red WAN (WAAS)

WCCP permite que las plataformas que emplean el sistema operativo IOS redirijan de forma transparente las solicitudes de contenido. El principal beneficio de la redirección transparente es que los usuarios no necesitan configurar sus browsers para usar un proxy web. En su lugar, pueden usar la URL destino para solicitar contenido, y sus solicitudes son redirigidas de forma automática a un engine de cache. La palabra “transparente” en este caso significa que el usuario final no se entera de que un archivo solicitado (tal como una página web) vino del WAE en lugar del servidor especificado originalmente.

Cuando un WAE recibe una solicitud, intenta darle servicio desde su propio cache local. Si la información solicitada no está presenta, el WAE envía su propia solicitud al servidor objetivo original para obtener la información requerida. Cuando el WAE recibe la información solicitada, la reenvía al cliente que la solicitó y guarda una copia local de la información para solicitudes futuras, maximizando el desempeño de las descargas y reduciendo sustancialmente los costos de transmisión.

WCCP permite que un conjunto de WAEs, llamados cache engine cluster, provean contenido a uno o varios routers. Los administradores de red pueden fácilmente escalar sus WAEs para manejar grandes volúmenes de tráfico mediante estas capacidades de clustering. Cada miembro del cluster trabaja en paralelo con los demás, lo que resulta en una escalabilidad lineal. Además, se obtiene una importante mejora en la escalabilidad, redundancia y disponibilidad.

Actualmente existen dos versiones de WCCP. La versión 1 tiene como característica principal que sólo un router da servicio a un cluster de engines. En este escenario, el router es el que realiza toda la redirección de paquetes IP.

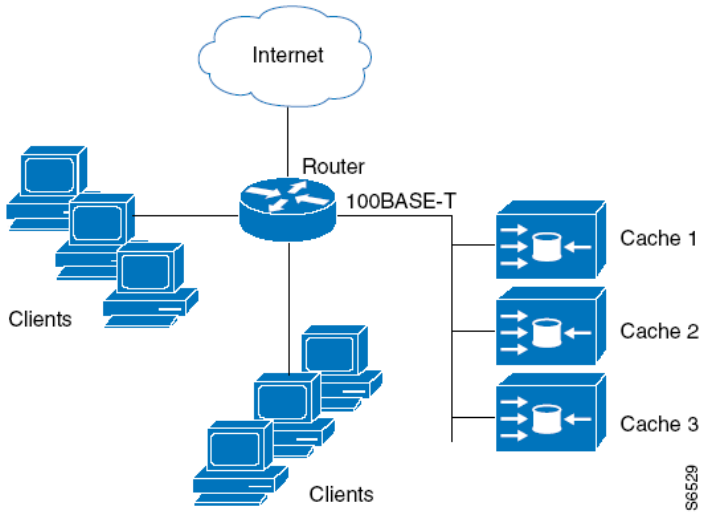

Figura 13. WCCP Versión 1

Con WCCP versión 2, varios routers pueden dar servicio a un cluster de engines. WCCPv2 requiere que cada engine conozca todos los routers en el grupo de servicio. Para especificar la dirección de todos los routers en un grupo de servicio, se puede escoger como método unicast o multicast.

El uso de múltiples routers en un grupo de servicio permite tener redundancia, agregación de interfaces y distribución de la carga de redirección.

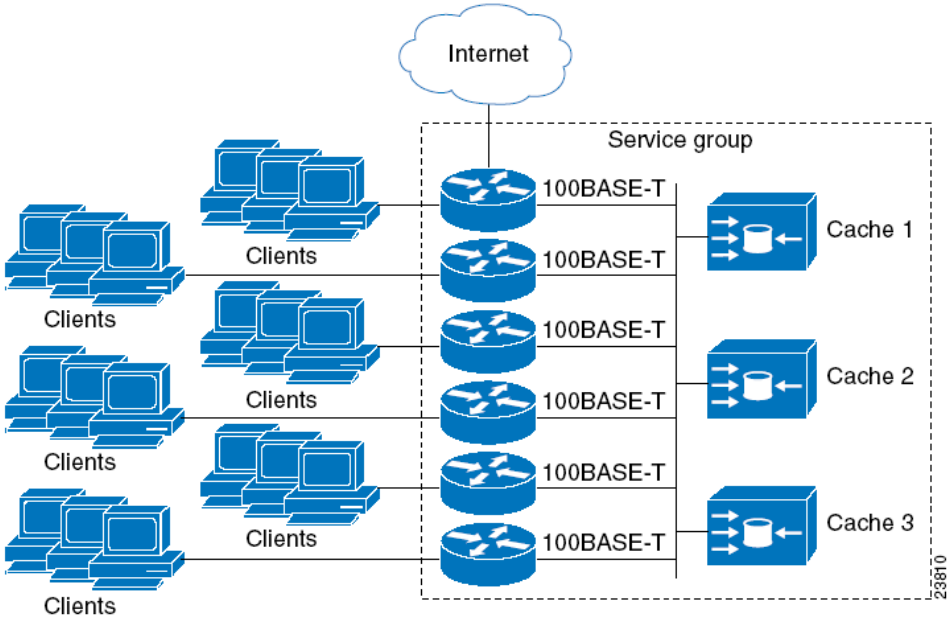

Figura 14. WCCP Versión 2

La siguiente secuencia detalla cómo funciona la configuración de WCCPv2:

1. Cada WAE se configura con una lista de routers
2. Cada WAE anuncia su presencia y una lista de todos los routers con los cuales ha establecido comunicación. Los routers responden con una lista de los WAE que ven en el grupo.
3. Una vez que la vista es consistente a lo largo de todos los cache engines en el cluster, un WAE se designa como el líder y establece la política que los routers del grupo necesitan implementar para la redirección de paquetes.

WCCPv2 permite la redirección de tráfico diferente a HTTP (TCP puerto 80), incluyendo una gran variedad de tráfico UDP y TCP. WCCPv1 soporta únicamente la redirección de tráfico HTTP. WCCPv2 soporta la redirección de paquetes destinados a otros puertos, incluyendo aquellos usados para manejo de cache proxy-web, caching de FTP, web caching para puertos diferentes al 80, y aplicaciones de telefonía, audio y video en tiempo real.

3.5.3.2 Optimización del delay

En enlaces WAN de baja velocidad, el tiempo para transmitir un paquete grande es significativo. Dicho tiempo se conoce como delay de serialización. El delay de serialización se vuelve un problema cuando un enlace WAN es usado por aplicaciones que envían paquetes largos, tales como transferencia de archivos, y aplicaciones que son sensibles al delay, como la voz, video y Telnet. Como soluciones a este problema se tienen varias alternativas, entre ellas Link Fragmentation and Interleaving y compresión tanto de paquetes como de encabezados.

Link-Layer Fragmentation and Interleaving (LFI)

LFI reduce el delay en enlaces WAN fragmentando los paquetes grandes e intercalando los fragmentos con paquetes de aplicaciones sensibles al delay. Tecnologías WAN como PPP, Frame Relay y ATM se ven beneficiadas con LFI en enlaces de baja velocidad, como un T1 o E1.

LFI para PPP está definido por el estándar Multilink Point-to-Point Protocol. Su operación es relativamente simple. Los paquetes grandes son encapsulados y fragmentados en un tamaño lo suficientemente pequeño para satisfacer los requerimientos de delay del tráfico sensible al delay. Los paquetes pequeños sensibles al delay no son encapsulados en multilink sino entrelazados entre fragmentos de los paquetes grandes.

MPPP permite que los paquetes sean enviados al mismo tiempo sobre múltiples enlaces punto-a-punto hacia la misma dirección remota. Los múltiples enlaces se levantan en respuesta a un umbral de carga de tráfico que se predefine. La carga puede ser calculada en tráfico de entrada, tráfico de salida o en ambos, como se requiera según el tráfico entre sitios específicos. MPPP proporciona ancho de banda en demanda y reduce el delay en enlaces WAN.

3.5.3.3 Quality of Service

El conjunto de herramientas para garantizar un servicio preferencial o deferencial a ciertas clases de tráfico se conoce como QoS.

Evolución de QoS

Las redes IP de mediados de los 90's del siglo pasado eran invariablemente redes best-effort, y el Internet, como un todo, permanece así hasta el momento. Sin embargo, las redes privadas de empresas y de ISPs se han transformado radicalmente, pasando de modelos best-effort a modelos mucho más complejos de servicios diferenciados, lo que significa que la red otorga a diferentes aplicaciones diferentes niveles de servicio.

La siguiente figura muestra la evolución de QoS desde inicios de la década de los 1990s:

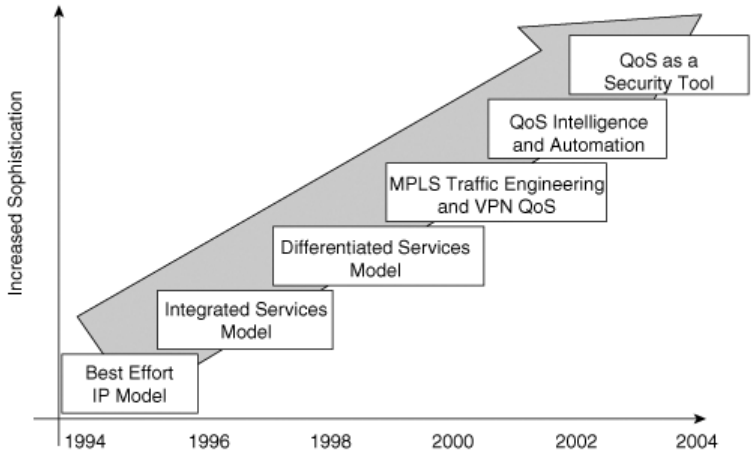

Figura 13. Evolución de QoS

El primer modelo de QoS fue Integrated Services (IntServ). Estos RFCs se centraban en RSVP. Este protocolo señaliza los requerimientos de BW y latencia para cada sesión a cada nodo a lo largo de una trayectoria que los paquetes tomarían desde su origen hacia su destino. Inicialmente, RSVP requería que cada nodo realizara reservaciones, lo que era muy impráctico en Internet.

Para superar esta limitante, se publicó otro conjunto de estándares – el modelo DiffServ –el cual emergió como un segundo intento para estandarizar QoS. El modelo DiffServ describe varios comportamientos que serán adoptados en cada nodo que se ajuste a DiffServ. Los nodos pueden usar cualquier característica disponible, para cumplir con los requerimientos de servicio del tráfico. El marcado de paquetes con IP Precedence o su sucesor DSCP fueron definidos en conjunto con Per-Hop-Behaviors para tipos clave de tráfico.

A medida que los modelos IntServ y DiffServ han evolucionado, la popularidad de un modelo contra el otro ha ido cambiando, y su coexistencia se ha vuelto una batalla creciente, con partidarios en ambos lados. La conclusión al día de hoy es que ningún método ofrece una solución integral y que elementos de

ambos modelos deben ser combinados para proveer el método más general aplicable al más amplio rango de tráfico y tipos de aplicaciones.

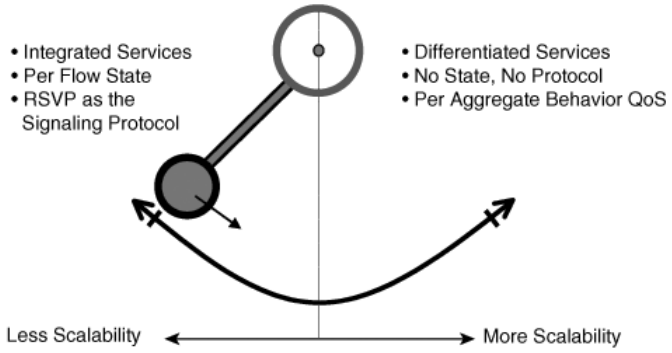

Figura 14. IntServ vs. DiffServ

Los modelos IntServ y DiffServ se definen como sigue:

IntServ se basa en un concepto de flujos en conjunto con un protocolo de señalización (RSVP) a lo largo de la trayectoria del paquete. El protocolo de señalización garantiza que estén disponibles los recursos necesarios para el flujo (en cada salto) antes de admitirlo en la red. El modelo IntServ sufrió de problemas de escalabilidad debido a los muchos flujos que necesitaban ser manejados en el backbone de las redes.

DiffServ usa marcados de paquetes para clasificar y tratar cada paquete de manera independiente. Aunque esta solución escala muy bien (lo cual es la razón por la cual tiene una implementación mayor que IntServ) no ofrece garantías específicas de ancho de banda a paquetes que pertenecen al mismo flujo y por tanto, falla en proveer control de admisión para nuevos flujos.

Sin ventaja clara para ningún modelo, los mecanismos de QoS continúan usando una mezcla de ambos modelos para ofrecer la variedad de servicios requeridos en las redes.

La tendencia más reciente en QoS es la simplificación y automatización, con la meta de provisionar simple y eficientemente QoS “inteligente” en redes IP. Toda la gama de características que ofrecen las tecnologías de QoS son muy útiles en manos de administradores de red expertos, pero pueden convertirse en configuraciones muy complejas a la vez. Por ello, la tendencia a simplificar su implementación.

Servicios de Queuing (Encolamiento)

Las técnicas de switching de alta velocidad discutidas anteriormente sólo son útiles hasta el punto en que una red se congestiona. En ese momento, se hacen necesarias métodos de queuing rápidos e inteligentes. Los métodos de queuing permiten a un dispositivo de red manejar un sobreflujo de tráfico. Los mecanismos de encolamiento más usados son los siguientes:

- First-in, first-out (FIFO) queuing
- Priority queuing

- Weighted fair queuing (WFQ)
- Class-based WFQ (CBWFQ)
- Low-latency queuing (LLQ)

First-In, First-Out Queuing: FIFO proporciona funcionalidades básicas de almacenamiento y reenvío de paquetes. FIFO tiene la ventaja de que no requiere configuración, al ser el método de queuing por default en algunas instancias. Por otra parte, tiene la desventaja de que no considera la prioridad de los paquetes. Todos son tratados por igual, el orden en que llegaron determina el orden en que serán procesados y servidos.

En la práctica FIFO no proporciona ninguna funcionalidad de QoS y ninguna protección contra una aplicación que use los recursos de una forma que afecte negativamente el desempeño de otras aplicaciones.

Priority Queuing: Priority Queuing asegura que el tráfico importante sea procesado primero. Fue diseñado para dar prioridad a aplicaciones críticas. La priorización de paquetes se puede realizar con base en varios criterios, como protocolo, interface de entrada, tamaño de paquete y dirección origen o destino.

Priority Queuing es particularmente útil en casos donde los enlaces WAN sufren de congestión ocasional. Si los enlaces WAN están congestionados constantemente, el cliente debe investigar posibles ineficiencias de protocolo y aplicación, considerar el uso de compresión o incluso aumentar el ancho de banda. Si los enlaces WAN nunca están congestionados, Priority Queuing es innecesario. Debido a que esta funcionalidad demanda procesamiento extra y puede causar problemas de desempeño para el tráfico de baja prioridad, no debe utilizarse a menos que sea necesario.

Priority Queuing tiene 4 colas: high, medium, normal y low. La cola de prioridad alta se vacía siempre antes que el resto de las colas, como se muestra en la siguiente figura.

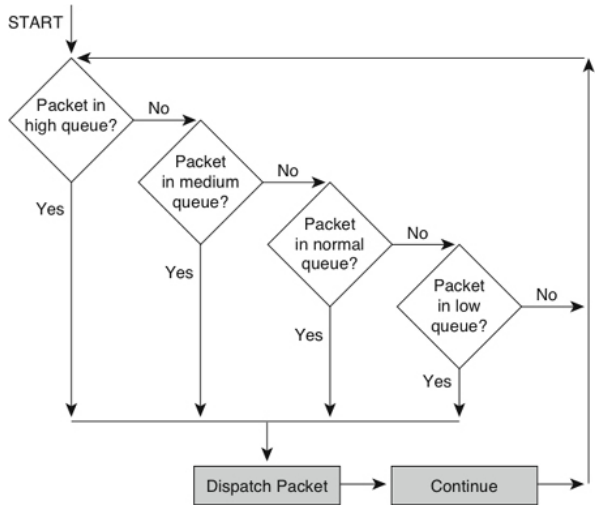

Figura 15. Priority Queuing

Weighted Fair Queuing (WFQ): WFQ es un conjunto de algoritmos sofisticados diseñados para reducir la variación en el delay y proveer un *throughput* y tiempo de respuesta predecible para flujos de tráfico. Una meta de WFQ es ofrecer un servicio uniforme para usuarios tanto con baja

como con alta demanda de tráfico. WFQ asegura que el tiempo de respuesta para aplicaciones de volumen bajo es consistente con el de aquellas aplicaciones de volumen alto. Las aplicaciones que envían paquetes pequeños no son sometidas a falta de BW debido a aplicaciones que envían paquetes grandes. El BW se divide de forma equitativa en forma automática.

WFQ es un algoritmo de queuing basado en flujos que reconoce un flujo para una aplicación interactiva y envía el tráfico de esa aplicación al frente de la queue para reducir el tiempo de respuesta. A los streams de tráfico de bajo volumen para aplicaciones interactivas (lo que representa un gran porcentaje de las aplicaciones en la mayoría de las redes), se les permite transmitir toda su carga ofrecida a tiempo. Los streams de tráfico de alto volumen comparten la capacidad sobrante.

WFQ se adapta automáticamente a condiciones cambiantes de tráfico y requiere poca configuración. Es el método de queuing default en la mayoría de las interfaces seriales con velocidades ≤ 2.048 Mbps.

WFQ puede asignar ancho de banda con base en el valor de IP Precedence, usando este último como factor de ponderación. El algoritmo asigna más BW a las conversaciones con mayor precedencia, y se asegura de que dichas conversaciones sean atendidas más rápidamente cuando ocurre congestión. WFQ asigna un peso a cada flujo, lo que determina el orden de la transmisión para paquetes encolados.

WFQ también funciona con RSVP, el cual usa WFQ para asignar espacio en buffer, organizar el despacho de paquetes, y garantizar el BW basándose en las reservaciones de recursos.

Class-Based Weighted Fair Queuing (CBWFQ): CBWFQ combina los mejores elementos de Priority Queuing y Weighted Fair Queuing. CBWFQ resulta en una configuración más compleja que los otros métodos de queuing, pero dicha complejidad añade una flexibilidad que no se encuentra en otros métodos. CBWFQ permite definir clases de tráfico con base en criterios de *match* como protocolos, ACLs, interfaces de entrada, direcciones IP origen, entre otras. Los paquetes que cumplen el criterio definido para una clase constituyen el tráfico para dicha clase. Una cola FIFO se reserva para cada clase, y el tráfico que pertenece a una clase se dirige a la queue de dicha clase.

Después de que una clase ha sido definida, se pueden asignar características a ella tales como BW y el número máximo de paquetes que pueden ser encolados para dicha clase, llamado *queue limit*. El BW asignado a una clase es el BW garantizado entregado a una clase durante congestión. El queue limit para la clase es el máximo número de paquetes que se pueden acumular en la queue para la clase. Para optimizar cómo se tiran los paquetes si se excede el *queue limit*, se puede habilitar WRED, que será cubierto más adelante.

La clasificación de flujos para CBWFQ se basa en WFQ. Esto es, los paquetes con la misma dirección IP origen, la misma IP destino, puerto TCP o UDP origen y puerto TCP o UDP destino son clasificados dentro del mismo flujo. WFQ asigna una cantidad igual de BW para cada flujo. WFQ basado en flujo también se conoce como fair queuing debido a que todos los flujos son ponderados por igual.

Cuando se habilita CBWFQ, los paquetes que llegan a una interface de salida se clasifican de acuerdo al criterio de match que se definió. El peso para un paquete que pertenece a una clase en específico se deriva del BW que se asignó a la clase cuando se configuró. En este sentido, el peso para cada clase es configurable por el usuario. Después de que el peso para un paquete se

asigna, el paquete se coloca en la queue apropiada. CBWFQ usa los pesos asignados para los paquetes en queue para asegurar que la queue de la clase se sirva de forma justa.

Low Latency Queuing (LLQ): LLQ combina Priority Queuing con CBWFQ. LLQ agrega a CBWFQ una cola de Strict Priority. Strict priority queuing permite que los datos sensibles al delay, como la voz, sean enviados antes que se envíen los paquetes en otras queues.

Sin LLQ, CBWFQ proporciona WFQ basado en clases definidas sin tener disponible una queue de Strict Priority para el tráfico de tiempo real. Cuando se usa CBWFQ, el peso para cada paquete que pertenece a una clase específica se obtiene del BW que se asignó a la clase cuando ésta se configuró. Por tanto, el BW asignado a los paquetes de una clase determina el orden en el cual los paquetes se envían. Todos los paquetes son servidos de forma justa con base en el peso, y ninguna clase recibe un trato de prioridad estricta. Este esquema representa problemas para el tráfico de voz que es intolerante al delay y al jitter.

LLQ permite el uso de una cola de Strict Priority dentro de CBWFQ, lo que permite enviar el tráfico que pertenece a una clase a la queue de Strict Priority. Se recomienda colocar únicamente el tráfico de voz en la queue de Strict Priority. El tráfico de voz requiere que el delay no varíe. El tráfico de tiempo real, como el video, puede introducir variaciones en el delay, alterando así la estabilidad en el delay que se requiere para la transmisión de tráfico de voz. LLQ es una herramienta exitosa para los ingenieros de red que planean transmitir voz en enlaces de bajo ancho de banda que también transportan otros tipos de tráfico.

Mecanismos de Manejo de la Congestión

Los mecanismos de queuing discutidos hasta este punto son técnicas de manejo de la congestión, es decir, nos permiten lidiar con ella una vez que se presenta, pero no prevenir su ocurrencia. Para ello se requieren nuevas herramientas que se discutirán a continuación.

Random Early Detection: Una de dichas herramientas es Random Early Detection (RED), la cual funciona monitoreando la carga de tráfico en puntos en la red y descarta paquetes de forma aleatoria si la congestión comienza a aumentar. El resultado es que los nodos origen detectan el tráfico que ha sido descartado y bajan su velocidad de transmisión. RED fue diseñado para trabajar con aplicaciones que operan sobre TCP.

Cuando ocurre una pérdida de paquetes, una sesión TCP disminuye su tasa de transmisión. Después se va incrementando de forma gradual hasta que se vuelve a encontrar congestión.

La experiencia muestra que si los routers no aplican alguna clase de aleatoriedad para descartar paquetes, múltiples sesiones TCP tienden a alentar su tasa de transmisión simultáneamente. (Las sesiones se sincronizan). Múltiples aplicaciones disminuyen y aumentan su tasa de transmisión simultáneamente, lo que significa que el BW no sea usado de forma efectiva. Cuando múltiples aplicaciones disminuyen su tasa de transmisión, se desperdicia BW. Cuando aumentan su tasa, tienden a incrementar la utilización de ancho de banda hasta que la congestión ocurre de nuevo.

La ventaja de RED es que los paquetes se descartan de forma aleatoria, por tanto reduciendo el potencial de que múltiples sesiones se sincronicen. RED es una buena solución para un router ubicado en un sitio central en una topología hub-and-spoke.

Weighted Random Early Detection (WRED): WRED combina las capacidades del algoritmo estándar de RED con IP Precedence. Esta combinación permite un tratamiento preferencial para paquetes de mayor prioridad. Descarta de forma selectiva el tráfico de baja prioridad cuando una interface comienza a congestionarse, en lugar de usar simplemente un método aleatorio. WRED también puede ajustar su comportamiento basándose en reservaciones RSVP.

WRED basado en flujo clasifica el tráfico de entrada en flujos con base en parámetros tales como IPs origen y destino y puertos. WRED basado en flujo usa esta información de clasificación y estado para asegurar que cada flujo no consuma más de los recursos que se le han permitido. WRED basado en flujo determina qué flujos monopolizan los recursos y los penaliza de forma más dura.

Traffic Shaping: Otra herramienta disponible es el Traffic Shaping, el cual permite manejar y controlar el tráfico de red para evitar cuellos de botella y cumplir así los requerimientos de QoS. Evita la congestión reduciendo el tráfico de salida de un flujo a un bit rate configurado, mientras mantiene en queue los bursts de tráfico para dicho flujo. En topologías con enlaces de BW variable, el tráfico puede ser shaped para evitar agobiar un router o enlace downstream.

Traffic shaping se configura por interface. El tráfico sobre el que se hará shaping se selecciona usando ACLs o class-maps. Traffic Shaping funciona con una gran variedad de tecnologías de capa 2, como Frame-Relay, ATM, y Ethernet.

3.5.3.4 Técnicas de Switching

Además de ejecutar protocolos de enrutamiento para obtener una topología de enrutamiento, la principal tarea de un router es reenviar paquetes de sus interfaces de entrada a sus interfaces de salida, proceso conocido como switching. El proceso de switching involucra recibir un paquete, determinar cómo enviarlo basado en la topología de enrutamiento y los requerimientos de QoS, y enviarlo a la interface o interfaces de salida. La velocidad a la cual un router puede realizar esta tarea es un factor mayor al determinar el desempeño de la red. En el caso particular de los routers Cisco, éstos soportan varios métodos de switching, con velocidades y comportamientos variables. A continuación se describirán algunos de estos métodos.

Por lo general, se debe usar el método de switching más rápido disponible. Usar un modo de switching rápido es especialmente imporante en el backbone de la red. Debe tenerse bajo observación el uso de memoria, debido a que los métodos de switching más rápidos usan más memoria, de modo que después de habilitar alguno de ellos el ingeniero de red se debe asegurar de monitorear la cantidad de memoria libre en el router durante un breve tiempo posterior.

Métodos tradicionales de Switching en Capa 3

Process Switching: es el método más lento de todos. Con este método, cuando una interface recibe un paquete de entrada, lo transfiere a la memoria input/output del router. El procesador de interface también genera una señal de *receive interrupt* del procesador central. El procesador

central determina el tipo de paquete y lo coloca en la cola de entrada apropiada. Por ejemplo, si es un paquete IP, el procesador central coloca el paquete en la cola ip_input. La siguiente vez que el scheduler de IOS corre, notará un paquete en la cola de entrada, y programará el proceso apropiado para ser ejecutado. Por ejemplo, para un paquete IP, programa el proceso ip_input para ser ejecutado.

El proceso de entrada, después de que el scheduler le indicó ser ejecutado, busca en la tabla de enrutamiento para determinar la interface de salida que debe ser usada para la dirección IP destino del paquete. El proceso reescribe el paquete con el encabezado capa 2 apropiado y copia el paquete a la interface. También coloca una entrada en el cache de fast-switching, conteniendo la interface de salida y la información para reescribir el encabezado capa 2. de modo que los paquetes subsecuentes dirigidos al mismo desino pueden utilizar dicha información para una búsqueda y referencia rápidas.

Nótese que la decisión de forwarding la toma un proceso programado por el scheduler de IOS para ser ejecutado. Este proceso corre a la par de otros en el router, como por ejemplo los procesos de los protocolos de enrutamiento. Los procesos que corren normalmente en el router no son interrumpidos para hacer process-switching de un paquete. Process switching debe esperar a que el resto de los procesos terminen de ser ejecutados, de ahí su lentitud.

Fast Switching: Permite un throughput más alto al hacer el switching de un paquete usando una entrada en el cache de fast switching que fue creada cuando el primer paquete hacia el destino fue procesado. Con Fast Switching, un paquete es manipulado de inmediato. El procesador de interface genera un mensaje de receive interrupt. Durante esta interrupción, el procesador central determina el tipo de paquete y comienza entonces a switchearlo. El proceso que corre en el procesador en ese momento es interrumpido para switchear el paquete. Los paquetes son switcheados bajo demanda, en lugar de serlo sólo cuando el proceso de forwarding puede ser programado para ser ejecutado. Con base en la información del cache de fast switching, el encabezado de capa 2 es reescrito y el paquete es enviado a la interface de salida a través de la cual se alcanza el next-hop del destino.

Nota: Si se está empleando balanceo de carga a través de múltiples interfaces con fast switching habilitado, la carga puede no estar balanceada de forma equitativa si una gran parte del tráfico se dirige a un único destino. Los paquetes dirigidos hacia un destino siempre saldrán por la misma interface incluso si el protocolo de enrutamiento soporta múltiples interfaces. En otras palabras, fast-switching realiza un balanceo por destino. Si esto representa un problema dada la naturaleza de las aplicaciones en la red, se debe usar CEF (Cisco Express Forwarding), que soporta tanto balanceo por destino y por paquete. El balanceo por paquete de CEF es en realidad un balanceo basado en pares origen/destino en lugar de basarse únicamente en direcciones destino, lo cual proporciona un mejor balanceo que fast-switching, sobre todo en el backbone.

Cisco Express Forwarding: Es el método más avanzado y el recomendado en el caso de plataformas Cisco. Es un método optimizado para el core de una red, que proporciona alta disponibilidad, predecibilidad, escalabilidad y resiliencia.

CEF es una técnica patentada por Cisco para realizar el switching de paquetes de forma muy rápida en redes con un backbone grande y en el Internet. En lugar de emplear las técnicas de cache usadas por los métodos explicados anteriormente, CEF emplea dos tablas: la Forwarding Information Base (FIB) y la tabla de adyacencias. En la FIB se almacena la información de las interfaces de salida y los encabezados de capa 2 para todos los destinos en la tabla de enrutamiento. La información de la FIB se calcula por adelantado, para todos los destinos.

CEF evolucionó para dar cabida a aplicaciones basadas en web y otras de naturaleza interactiva que están caracterizadas por sesiones de corta duración hacia múltiples destinos. CEF se volvió necesario cuando fue claro que un sistema basado en cache no estaba optimizado para este tipo de aplicaciones.

Consideremos una aplicación basada en web, por ejemplo. Cuando un usuario llega a un sitio, TCP abre una sesión con una nueva dirección destino. Es poco probable que el nuevo destino esté en el cache del router, a menos que se trate de un destino que el usuario u otros usuarios hayan visitado recientemente. Esto significa que el primer paquete es process-switched, lo cual es lento. Además, el CPU del router se ve impactado significativamente si hay muchos primeros paquetes a nuevos destinos. CEF mejora la velocidad de switching, y evita el overhead asociado con un cache que cambia continuamente, a través del uso de la FIB, la cual replica la información de todos los destinos de la tabla de enrutamiento.

CEF soporta balanceo tanto por paquete como por destino. Éste último es el default, lo que significa que los paquetes para un par origen/destino dado tendrán como interface de salida la misma interface. El tráfico dirigido a diferentes pares tiende a tomar caminos diferentes. A menos que el tráfico consista de flujos de un número reducido de pares origen/destino, el tráfico tenderá a ser distribuido equitativamente a lo largo de múltiples trayectorias con el balanceo por destino de CEF.

CEF también soporta balanceo por paquete. Este balanceo permite a un router enviar paquetes sucesivos sobre trayectorias sin importar las direcciones origen y destino. El balanceo por paquete usa un método round-robin para seleccionar la interface de salida para un paquete. El balanceo por paquete proporciona un mejor balance de tráfico sobre múltiples enlaces. Se puede usar balanceo por paquete para ayudar a asegurar que la trayectoria para un único par origen/destino no se vea sobrecargada.

Aunque la utilización de las trayectorias con el balanceo por paquete se ve optimizada, los paquetes para un par origen/destino pueden tomar diferentes caminos, lo cual puede provocar que los paquetes lleguen fuera de orden. Por esta razón, el balanceo por paquete es inapropiado para ciertos tipos de tráfico, como VoIP, que dependen de que los paquetes lleguen al destino en secuencia.