ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

ESTUDIO DE LOS FACTORES TÉCNICOS Y OPERATIVOS QUE INTERVIENEN EN LA INFRAESTRUCTURA DE CALIDAD DE SERVICIO EN INTERNET

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO ESPECIALISTA EN ELECTRÓNICA Y TELECOMUNICACIONES

TOMO II

LOOR FONSECA DIEGO CHRISTIAN
PICHOASAMÍN MORALES LUIS HUMBERTO

DIRECTOR: Ing. María Soledad Jiménez

Quito, Enero 2001

Capítulo 5

ARQUITECTURA DE SERVICIOS INTEGRADOS

CAPÍTULO 5

ARQUITECTURA DE SERVICIOS INTEGRADOS

El surgimiento de nuevas aplicaciones multimedia así como su extensa difusión ha sido fomentado principalmente por tres tecnologías: a) el desarrollo de poderosas estaciones de trabajo equipadas con hardware dedicado a voz y video, b) el desarrollo de aplicaciones sofisticadas (por ejemplo tele-educación, tele-inmersión[1], etc.), y c) la propiedad de *multicast* de IP.

Sin embargo, la diferencia de retardo de encolamiento y las pérdidas por congestión presentes en el actual Internet son los mayores problemas que tienen que afrontar las aplicaciones en tiempo real, haciéndose necesaria la implementación de calidad de servicio extremo a extremo en tiempo real.

Ha surgido además la necesidad de administrar el ancho de banda del enlace controlando la porción de recurso entregado a cada usuario en condiciones de sobrecarga. Los Servicios Integrados (*IntServ*) son un modelo de servicio de Internet que incluye el actual servicio *best effort*, el servicio en tiempo real y la compartición controlada del enlace. La Arquitectura de Servicios Integrados no fue diseñada (y de hecho no puedo hacerlo) para reemplazar al básico servicio IP, en cambio añade nuevos componentes y mecanismos que permiten implementar Calidad de Servicio.

5.1 ELEMENTOS DE LA ARQUITECTURA *IntServ*

Esta Arquitectura tiene dos elementos: el Modelo de Servicios Integrados (sección 5.1.1) y una Implementación de Referencia (sección 5.1.2).

El Modelo *IntServ* define el comportamiento que debe mostrar la arquitectura, el cual no puede cambiar con el tiempo; la implementación de referencia muestra

[1] Teleinmersión.- se refiere a un sistema que permite a personas situadas en distintos lugares compartir el mismo entorno virtual.

una forma de implementación del modelo, por lo tanto está sujeta a modificaciones.

5.1.1 MODELO DE SERVICIOS INTEGRADOS

El Modelo de Servicios Integrados se basa en los requerimientos generales de calidad de servicio que demandan las aplicaciones y el entorno actual mas no en un dispositivo de red específico. Su afán no es el de cambiar la actual infraestructura de Internet sino complementarla con la adición de nuevos elementos y mecanismos.

El modelo consiste de un conjunto de compromisos en respuesta a un requerimiento de servicios. Los compromisos pueden ser categorizados para un flujo individual o un conjunto de flujos. En el caso de un flujo individual se busca la optimización de una aplicación básicamente refiriéndose al tiempo de envío de los paquetes, diferenciándose dos clases de servicio: servicio de carga controlada y servicio garantizado. En el segundo caso se busca la optimización de los recursos tanto de red como económicos en circunstancias de compartición del enlace.

Para el desarrollo del modelo se asume lo siguiente:

- Administración explícita de los recursos; la única forma de garantizar los servicios es a través de la reservación de recursos y para ello es necesario definir políticas y controles de admisión que permitan la autentificación de los flujos, los cuales pueden basarse en usuarios o paquetes. El mayor cambio que implica la reservación de recursos es el establecimiento en los *routers* de un estado específico para cada flujo.

- Internet como infraestructura común para la comunicación de tráfico en tiempo real y tráfico que no es en tiempo real, pues el desarrollo de una infraestructura paralela para tráfico en tiempo real haría más difícil la administración del Internet.

5.1.1.1 Requerimiento de QoS para un flujo

El requerimiento principal de un flujo es la prontitud de envío de sus paquetes, por consiguiente la única forma cuantitativa de entregar calidad de servicio es la fijación de un retardo mínimo y máximo que recibirá el flujo. Mediante la sensibilidad al retardo se puede distinguir dos tipos de aplicaciones: aplicaciones en tiempo real y aplicaciones elásticas.

Las **aplicaciones en tiempo real** incluyen la mayoría de aplicaciones *playback*, que son aquellas cuya forma de funcionamiento implica empaquetamiento, envío, desempaquetamiento y reproducción de una señal original. Para su operación se define el punto de *playback* correspondiente a un tiempo de compensación desde que se envía el paquete hasta que se reproduce; todos los paquetes que tienen un retardo menor al punto de *playback* son datos útiles, mientras los paquetes con un retardo mayor son desechados. El retardo de compensación se define en función del retardo de los primeros paquetes enviados o del compromiso de la red cuando ofrece algún nivel de calidad de servicio.

Las medidas del rendimiento de estas aplicaciones son la latencia y la fidelidad, relacionadas con el retardo y la pérdida de paquetes respectivamente. El retardo es el responsable directo de la latencia pero también de la fidelidad ya que una aplicación pierde fidelidad directamente a través de la pérdida de paquetes o en forma de distorsión si acepta los paquetes tardíos desplazando el punto de *playback*. La latencia es crítica en aplicaciones interactivas como la telefonía, mientras la fidelidad es importante en aplicaciones de difusión de audio y video.

Existen aplicaciones en tiempo real que no resisten distorsión ni pérdida de paquetes, y que exigen de la red el menor retardo de compensación, el cual será superior al retardo de cualquier paquete de la aplicación, para estas aplicaciones intolerantes existe el Servicio Garantizado (sección 5.2.2) . Otras aplicaciones en tiempo real no son tan exigentes y aceptan un retardo de compensación variable entre ciertos límites e incluso soportan alguna pérdida a cambio de mayor eficiencia de transmisión y menores costos, para estas aplicaciones tolerantes se definió el Servicio de Carga Controlada (sección 5.2.1).

Las **aplicaciones elásticas,** a diferencia de las aplicaciones en tiempo real, no son tan sensibles al retardo. Se caracterizan porque siempre esperan la llegada de todos sus datos y éstos se procesan inmediatamente después de su arribo. Entre las aplicaciones elásticas se encuentran aquellas que presentan un comportamiento de ráfaga interactiva como Telnet y NFS (*Network File System*), aplicaciones de transferencia interactiva en grandes proporciones como FTP y aplicaciones de transferencia asincrónica en grandes proporciones como *e-mail* y fax. Las aplicaciones elásticas no necesitan control de admisión, siendo el tradicional servicio *best effort* el más adecuado para cubrir sus necesidades.

A pesar que se ha intentado clasificar en cierta forma el tráfico, cualquier aplicación puede solicitar el nivel de calidad de servicio que desee basándose en sus requerimientos de fidelidad, latencia y costo.

5.1.1.2 Requerimientos de Compartición de Recursos

Un enlace de comunicaciones involucra varios flujos con diferentes requerimientos de recursos. El enlace puede estar compartido por varias entidades o empresas, cada una de las cuales demanda un ancho de banda fijo en condiciones de sobrecarga así como el uso del ancho de banda desocupado en condiciones de baja carga de tráfico. Por otro lado no todos los protocolos que existen en Internet tienen la misma importancia por lo que en condiciones de sobrecarga se puede dar prioridad a ciertos protocolos como IP, IPX, SNA o cualquier otro; incluso se puede hacer una diferenciación dentro de un mismo protocolo multiservicio como IP asignando una fracción de ancho de banda para aplicaciones en tiempo real y otra fracción para las aplicaciones elásticas.

El modelo de compartición de recursos más adecuado puede ser uno jerárquico, ubicándose en una posición superior la organización a la cual pertenece el tráfico, luego el protocolo que utiliza y finalmente el servicio, tal como se muestra en la figura 5.1.

La diferenciación del tráfico expuesta torna imprescindible el uso de un control de admisión que permita cristalizar los compromisos de compartición del enlace. El

control de admisión debe guardar recursos para el tráfico garantizado y predictivo (perteneciente al servicio de carga controlada) de cada entidad.

Idealmente la compartición de recursos del enlace debe ser instantánea pero debido a que ello es imposible, pues se requiere una perfecta sincronización en el manejo de los paquetes, el modelo debe aproximarse lo más que se pueda a esta característica ideal.

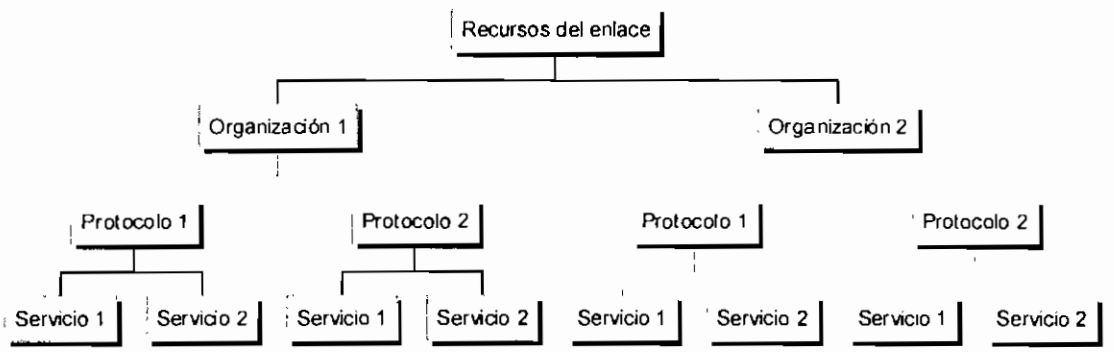

Figura 5.1 Modelo jerárquico de la compartición de recursos en un enlace

5.1.1.3 Descarte de Paquetes

No todos los paquetes pertenecientes a un mismo flujo tienen la misma importancia, en muchos flujos de audio o video algunos paquetes son más importantes que otros. Se puede entonces marcar a los paquetes menos importantes con probabilidad de descarte, que se hará efectiva en el *router* cuando se presente algún problema como por ejemplo el arribo tardío del paquete o el arribo fuera de orden.

Un paquete marcado con probabilidad de descarte no implica necesariamente su descarte, por el contrario, la mayoría de paquetes marcados son enviados a través del Internet. Otra forma de tratar los paquetes menos importantes es excluyéndoles de la descripción del tráfico usado en el control de admisión y por tanto el *router* no tendrá el compromiso de enviarlos.

5.1.1.4 Modelo de Reservación

El modelo de reservación es la manera en que una aplicación reserva un nivel de calidad de servicio. En la forma más simple se refiere a la respuesta afirmativa o negativa de la red a un requerimiento de una aplicación, pero en realidad el funcionamiento es más complejo debido a que existen diferentes niveles de servicio que pueden trabajar bien con una determinada aplicación.

En general una reservación empieza con la difusión (*broadcast*) de la información que define las características del tráfico ofertado por el emisor, ésta información es grabada y tal vez modificada en los *routers* que se hallan en el camino al receptor, el cual en respuesta envía por el mismo camino las expectativas de tráfico requeridas para una comunicación normal, finalmente se realiza la respectiva correlación en cada *router*. Una descripción más detallada acerca de la reservación de recursos se hace en el estudio del protocolo de reservación de recursos RSVP (sección 5.3).

5.1.2 IMPLEMENTACIÓN DE REFERENCIA

Para el análisis de la Arquitectura de Servicios Integrados es necesario definir el término flujo:

> Flujo.- "Es una corriente diferenciable de datagramas relacionados que resultan de la actividad de un simple usuario y que requiere el mismo QoS [42]. Por ejemplo, un flujo puede consistir de una conexión de capa transporte como la corriente de video entre un par de *hosts* dados. Es la más fina granularidad que puede distinguir los servicios integrados *IntServ*. Un flujo es *simplex*[1], por ejemplo una teleconferencia con *N* miembros requiere *N* flujos que se originan, uno en cada sitio".

Las funciones básicas que realiza el *router* son el direccionamiento de los paquetes y la administración de las colas de salida, usualmente realizado con la técnica de encolamiento FIFO y el descarte de paquetes *tail drop* como se explica

[1] Comunicación Simplex.- Es aquella en la cual los datos circulan en un solo sentido.

en el capítulo 3. Sin embargo los Servicios Integrados requieren que el *router* implemente calidad de servicio por cada flujo, necesitándose mecanismos adicionales para cumplir este objetivo. La función del *router* que permite la asignación de QoS se conoce como Control de Tráfico y tiene tres mecanismos principales:

- Organizador de Paquetes (*Packet Scheduler*)
- Clasificador
- Control de Admisión

5.1.2.1 Organizador de Paquetes

Este mecanismo administra el envío de paquetes y su operación la realiza en las colas de salida del *router* con la ayuda de varios algoritmos de encolamiento y de descarte de paquetes. Pertenece a la capa 2 o capa enlace del modelo de referencia OSI lo cual le hace dependiente del medio de transmisión.

Los algoritmos de encolamiento básicamente reordenan la cola de salida teniendo en cuenta por ejemplo la prioridad relativa de cada paquete, un paquete marcado con mayor prioridad será enviado antes que uno con prioridad menor. Esta forma de encolamiento tiene desventajas, una de ellas es el retardo profundo que puede experimentar un paquete con baja prioridad en presencia de tráfico continuo de alta prioridad. Estas desventajas han incentivado el surgimiento de otros algoritmos como el encolamiento justo ponderado WFQ que solucionan estos problemas a cambio de mayor complejidad.

Tradicionalmente en Internet el descarte de paquetes se lleva a cabo cuando una cola ha llegado a su capacidad máxima y sirve además como notificación de congestión para el accionar de protocolos de capa transporte. La filosofía que tienen los Servicios Integrados es diferente, el descarte de paquetes se efectúa antes que la cola se sature consiguiendo mayor eficiencia en el control de congestión. Por otro lado, el descarte de un paquete encolado disminuye el retardo de los paquetes que están detrás de él, se prefiere perder un paquete en beneficio de muchos, el punto clave aquí es determinar qué paquete descartar.

La correcta combinación de técnicas de encolamiento y descarte permite que no se pierdan las características de cada una. Un estudio más detallado de la organización de paquetes se encuentra en el capítulo 3.

Se considera dentro de la organización de paquetes un elemento vital para el funcionamiento del modelo *IntServ* denominado estimador. El **estimador** es un algoritmo que mide las propiedades de la corriente de tráfico saliente para desarrollar estadísticas que controlen la organización de paquetes y el control de admisión [42].

5.1.2.2 Clasificador

Para definir diferentes conductas de reordenamiento y descarte para los paquetes que ingresan al *router* se necesita una clasificación previa de éstos. La clasificación se hace de acuerdo a información contenida en la cabecera del paquete o a algún número de clasificación adicional.

El primer modelo de clasificación necesita conocer las direcciones IP de origen y destino, los puertos de origen y destino, y el protocolo utilizado; la combinación de estos datos permite identificar un flujo particular. Por ejemplo el tráfico de video puede ser reconocido por el puerto, ubicado en la cabecera UDP, o un flujo particular puede ser reconocido por las direcciones IP de origen y destino.

El segundo modelo de clasificación emula un circuito virtual con requerimientos específicos de calidad de servicio, en el cual los paquetes pertenecientes al circuito tendrían un identificador especial.

La clasificación que se realice depende también del tipo de *router* en donde se efectúa. Un *router* perteneciente al *backbone*[1] de la red puede agrupar un gran número de categorías de flujos en una sola clase lo que le permitiría aumentar el rendimiento de la transmisión al no utilizar demasiada información de control; en cambio un *router* periférico podría distinguir flujos individuales sin mayor pérdida de eficiencia.

---

[1] Backbone.- Porción de una red de comunicaciones por la cual circula la más alta densidad de tráfico de conectividad.

5.1.2.3 Control de Admisión

El control de admisión es la decisión de aceptar o rechazar un requerimiento de calidad de servicio en función de la disponibilidad de recursos, la cual indica si hay capacidad de hacer compromisos con un determinado flujo.

Para realizar esta tarea el *router* debe entender los algoritmos de control de admisión. Un método para realizar control de admisión es mediante el cómputo de las características más críticas de cada flujo aceptado, en el instante en que se hace un nuevo requerimiento, y luego tomar la decisión. Un método más actual propone el cálculo instantáneo de la utilización de recursos, el cual a un costo de información de control adicional (*overhead*) permite optimizar el uso del ancho de banda.

Aunque el control de admisión es parte del modelo global de servicio, el algoritmo particular en cada *router* es materia local, por lo tanto los fabricantes pueden competir en el desarrollo de mejores algoritmos que optimicen el enlace y la sobrecarga de información de control [42].

5.1.2.4 Aplicación de los mecanismos de Control de Tráfico

Un análisis teórico (encontrado en la referencia [43]) demuestra que si se implementa WFQ en los *routers* y es posible caracterizar la fuente de tráfico se asegura un retardo máximo en una red, posibilitando la entrega de servicio garantizado. WFQ también permite una compartición controlada del enlace donde la prioridad no es el retardo sino limitar la sobrecarga en el enlace; este uso de WFQ está disponible actualmente en los *routers* comerciales, y es utilizado para segregar tráfico basado en parámetros como el tipo de protocolo o la aplicación.

WFQ garantiza un retardo constante pero no necesariamente un retardo mínimo, en general si se mezcla el tráfico en una sola cola se obtiene retardos menores con tal que el tráfico tenga las mismas características, por ejemplo tráfico de video de múltiples codificadores y no tráfico de video con FTP. Este servicio se conoce como servicio predictivo en tiempo real cuyos objetivos son entregar un retardo tan pequeño como sea posible y suficientemente estable para que pueda ser estimado por el receptor. Este servicio se lo puede realizar en dos etapas, la

primera es segregar el tráfico en clases con requerimientos similares, y la segunda es aplicar los mecanismos de organización en cada clase para alcanzar sus objetivos de servicio.

5.1.2.5 *Router* capacitado con Servicios Integrados.

Un *router* IP necesita ciertas adecuaciones para proveer los servicios descritos en la Arquitectura de Servicios Integrados. La figura 5.2 muestra estos componentes distribuidos en las dos divisiones funcionales del *router* : el camino de transporte (plano de datos) en la parte inferior y el código de fondo (plano de control) en la parte superior.

El **camino de transporte** o envío está dividido en tres partes: controlador de entrada, agente de transporte de Internet y controlador de salida. El agente de transporte de Internet interpreta la cabecera del protocolo o arquitectura de red, por ejemplo la cabecera IP para TCP/IP, ejecuta el clasificador para cada paquete y luego lo pasa al controlador de salida.

El controlador de salida implementa el organizador de paquetes (*packet scheduler*), dentro de éste se distinguen dos secciones, la una que corresponde al organizador de paquetes propiamente, la cual es independiente del hardware, y la otra es la interfaz normal de entrada y salida, dependiente del hardware. El estimador está entre las dos secciones.

El **código de fondo** (*background*) es un conjunto de rutinas que crean estructuras de datos para controlar el camino de transporte. El código está almacenado en la memoria del *router* y es ejecutado por una unidad central de proceso de propósito general. Se distinguen tres componentes: agente de enrutamiento, agente de configuración de reservaciones y agente de administración.

El agente de enrutamiento implementa un protocolo de enrutamiento particular construyendo la respectiva base de datos. El agente de configuración de reservaciones implementa el protocolo de reservación de recursos, está capacitado para modificar el clasificador y organizador de paquetes si el control de admisión faculta la entrega de QoS a un nuevo flujo. El agente de

administración permite la administración de la red y también está facultado a modificar el organizador y clasificador de paquetes con el fin de controlar la compartición del enlace y configurar las políticas que rigen el control de admisión.

Para la implementación de servicios integrados se requiere cambios en las dos secciones del *router*, siendo la sección del camino de envío la más costosa y difícil, pues para obtener la eficiencia requerida se necesita cambios en el hardware.

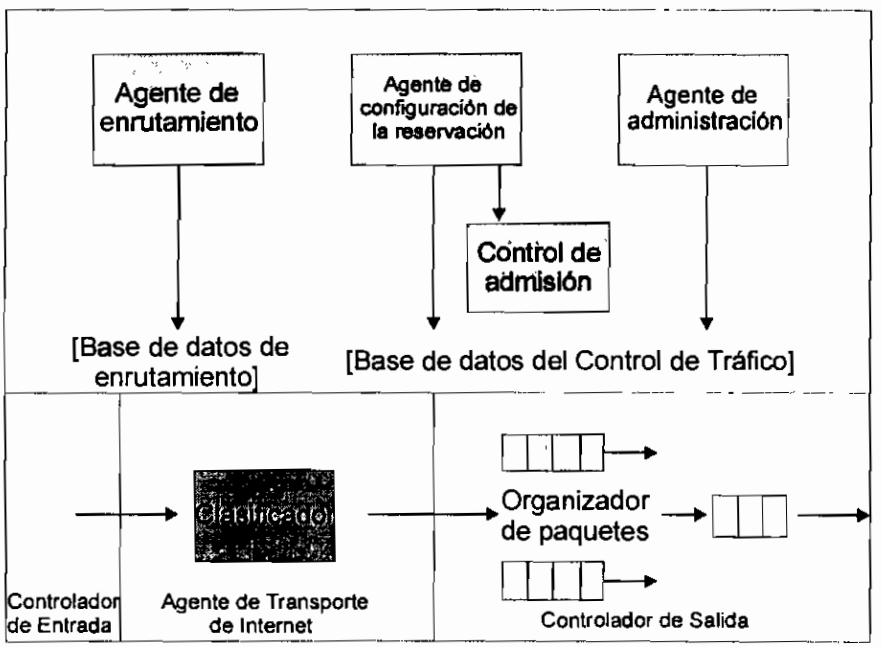

Figura 5.2 Modelo de implementación de referencia para los *routers*. [42]

La implementación para un *host* es similar al *router* con la adición de las aplicaciones, por tanto los datos del *host* se originan y terminan en la aplicación. Cuando una aplicación requiere QoS invoca localmente al agente de configuración de reservaciones, existiendo la posibilidad de no utilizar clasificador ya que la clase del paquete puede estar especificada en la respectiva estructura de control de entrada y salida.

5.2 SERVICIOS DE LA ARQUITECTURA IntServ

La Arquitectura de Servicios Integrados define dos clases de servicios:

- Servicio de Carga Controlada o Servicio Predictivo
- Servicio Garantizado

5.2.1 SERVICIO DE CARGA CONTROLADA

De las dos clases de servicios que ofrece la arquitectura *IntServ*, la de carga controlada necesita menores requerimientos de recursos. A continuación se detalla las características del Servicio de Carga Controlada.

5.2.1.1 Conducta extremo a extremo

Este servicio está dirigido a aplicaciones que funcionan bien en las actuales condiciones de Internet pero que su rendimiento se degrada en situaciones de sobrecarga. El servicio de carga controlada es similar al servicio *best effort* en condiciones de baja carga de la red, presentando las siguientes características:

- Un alto porcentaje de paquetes transmitidos arriban exitosamente a los nodos finales.
- El retardo que experimentan los paquetes es relativamente uniforme y su valor depende básicamente de la velocidad de la luz más el tiempo de procesamiento en los dispositivos de comunicación. La velocidad de la luz define, aproximadamente, el tiempo de propagación de la señal electromagnética a través del medio de transmisión; y, el tiempo de procesamiento en los dispositivos de red no contabiliza el retardo de encolamiento, pues éste sería mínimo.

Para obtener el servicio, la aplicación indica a la red las características del tráfico que generará (TSpec, *Traffic Specification*), en respuesta la red reserva recursos para cumplir el requerimiento; si la aplicación genera más tráfico que el indicado recibirá prácticamente un servicio *best effort* en condiciones de sobrecarga.

5.2.1.2 Requerimientos de los elementos de red

1. Cada elemento de red que acepta un requerimiento de servicio de carga controlada debe reservar los recursos suficientes que le permitan cumplir los compromisos de QoS. Los recursos más importantes incluyen el ancho de banda del enlace, el espacio en *buffers* y la capacidad de cómputo en la sección de direccionamiento de paquetes.

2. Un elemento de red puede usar herramientas estadísticas para decidir la aceptación de un nuevo flujo. Dependiendo de la conducta pasada que hayan mostrado los flujos puede disponer de recursos que le permitan asegurar a un nuevo flujo sin perjudicar a los existentes.

3. Un elemento de red puede hacer uso de los algoritmos de ordenamiento apropiados para cumplir los compromisos de servicio. El algoritmo implementado debe disponer un ancho de banda mayor que el especificado en el TSpec a fin de superar los momentos de ráfaga del flujo, de no hacerlo el retardo de encolamiento se incrementaría permanentemente. El algoritmo de ordenamiento puede implementar esta condición en una forma explícita, "pidiendo prestado" ancho de banda, o en forma implícita mediante técnicas de multiplexaje estadístico[1].

   Similarmente, la implementación debe reservar mayor espacio en *buffers* que el especificado en TSpec con el objetivo de reducir las pérdidas de paquetes. Los servicios de carga controlada no rediseñan el TSpec en cada nodo, distorsionándose éste a su paso por los puntos de encolamiento, distorsión que se produce especialmente en momentos de ráfagas. El algoritmo implementado puede usar técnicas de multiplexaje estadístico para cumplir esta condición.

4. Un dispositivo de red no puede fragmentar los paquetes de un flujo que recibe el servicio de carga controlada. Si los paquetes son más grandes que la máxima unidad de transporte (MTU) del enlace, éstos no son

[1] Multiplexaje Estadístico.- Realizar medidas estadísticas que permitan predecir el comportamiento de los flujos en base a las cuales se toman decisiones para asignar recursos.

considerados parte del TSpec y se procede de una forma explicada más adelante (sección 5.2.1.4).

5.2.1.3 Caracterización del tráfico (TSpec)

El elemento de red antes de aceptar el requerimiento de servicio de carga controlada para un flujo, necesita conocer la caracterización del tráfico generado, el cual determina los recursos a reservarse. La especificación del tráfico se efectúa con un conjunto de parámetros definidos en TSpec. Los parámetros son: velocidad promedio de transmisión "*bucket rate*" (*r*), espacio en *buffer* "*bucket size*" (*b*), velocidad de transmisión pico (*p*), mínimo tamaño de paquete (*m*) y máximo tamaño de paquete (*M*). Los parámetros *r* y *b* definen el *token bucket* detallado en el ANEXO 4.

Velocidad de Transmisión *(r)*: Corresponde a la velocidad promedio de transmisión, es medida en bytes de datagramas IP por segundo y su valor puede variar desde 1 byte/s hasta 40 terabytes/s.

Espacio en *buffers (b)*: Es el máximo espacio del *buffer* a ocuparse cuando se producen ráfagas de datos, está medido en bytes y su valor varía entre 1 byte y 250 gigabytes.

Velocidad de Transmisión Pico (*p*): Es la máxima velocidad admitida, la cual se presenta cuando existe ráfagas de datos. Es medida en bytes de datagramas IP por segundo y su valor puede variar desde 1 byte/s hasta 40 terabytes/s. Generalmente este parámetro es ignorado ya que el módulo de servicio del elemento de red puede asumir siempre que la velocidad pico de arribo al elemento es la velocidad de la línea en la interfaz de entrada, y el criterio de evaluación del servicio no requiere que el elemento de red considere el valor de *p*.

Mínimo tamaño de paquete (*m*): Es un número entero positivo medido en bytes que define el tamaño de paquete más pequeño a generarse. Si la aplicación produce paquetes más pequeños que este valor, el elemento de red considerará que su tamaño es *m*.

Máximo tamaño de paquete (*M*): Es un número entero positivo medido en bytes que define el máximo tamaño de paquete a generarse. El elemento de red rechazará el requerimiento de servicio si *M* es mayor que el MTU del enlace.

Para representar estos valores se utilizan tres números de punto flotante para *r*, *b* y *p*; y dos enteros de 32 bits para *m* y *M*. Los valores admitidos para *r*, *b* y *p* son intencionalmente grandes para futuras tecnologías, en la práctica solo los primeros dígitos son significativos y se permiten patrones de bits que representen números de punto flotante no negativos. Para *p* se permite un valor infinito representado con un exponente de todos unos (255) y un bit de signo y mantisa de todos ceros; si el valor de *p* no es especificado, éste se considera infinito.

Para el funcionamiento de un elemento de red que curse tráfico de carga controlada es necesario diferenciar varios tipos de TSpec, a continuación se analizan algunas reglas [44].

a. En el servicio de carga controlada un TSpec A es tan bueno o mejor que un TSpec B si y solo si:

   1. $r_A \geq r_B$
   2. $b_A \geq b_B$
   3. $p_A \geq p_B$
   4. $m_A \leq m_B$
   5. $M_A \geq M_B$

b. Si dos TSpec difieren y no se cumple la regla de arriba, el mínimo de los dos se determina comparando los valores respectivos y eligiendo:

   1. El mínimo valor de r
   2. El máximo valor de b
   3. El mínimo valor de p
   4. El mínimo valor de m
   5. El mínimo valor de M

c. Para determinar la reservación compartida de *n* flujos de carga controlada se calculan los siguientes parámetros:

1. $r_T = \sum_{i=1}^{n} r_i$

2. $b_T = \sum_{i=1}^{n} b_i$

3. $p_T = \sum_{i=1}^{n} p_i$

4. $m_T = \text{MIN}(m_1, m_2, m_3, \ldots\ldots, m_n)$

5. $M_T = \text{MAX}(M_1, M_2, M_3, \ldots\ldots, M_n)$

d. Cuando se hace reservaciones, a veces es necesario caracterizar el tráfico de varios flujos con un solo conjunto de parámetros TSpec, los cuales representarán el mínimo común de todos los flujos y serán calculados de la siguiente manera:

1. $r_c = \text{MAX}(r_i)$

2. $b_c = \text{MAX}(b_i)$

3. $p_c = \text{MAX}(p_i)$

4. $m_c = \text{MIN}(m_i)$

5. $M_c = \text{MAX}(M_i)$

5.2.1.4 Políticas de Funcionamiento

Los módulos de servicio de carga controlada proveen QoS al tráfico enmarcado en el TSpec. Se debe cumplir la condición de que la cantidad de datos enviados, para todo tiempo, no exceda *rT+b*, donde *r* y *b* son los parámetros especificados anteriormente y *T* es la longitud del periodo de tiempo; los paquetes que lleguen al dispositivo de red y violen esta regla no son considerados parte del TSpec. Adicionalmente los paquetes de menor tamaño que el especificado en *m* son considerados de tamaño *m*; y, los paquetes de mayor tamaño que el MTU del enlace tampoco son considerados parte del TSpec.

Si al dispositivo de red arriba tráfico que no es considerado parte del TSpec se debe cumplir las siguientes tres reglas:

- El elemento de red DEBE continuar proveyendo el QoS contratado a todos los flujos que no experimenten exceso de tráfico.
- El elemento de red DEBERÍA prevenir que el exceso de tráfico de carga controlada perjudique al tráfico *best effort*.
- El elemento de red DEBE enviar el exceso de tráfico de carga controlada si existen recursos disponibles.

El control de admisión puede sobredimensionar los recursos disponibles a fin de dar cabida a un mayor número de flujos, siempre y cuando garantice el servicio a cada uno de ellos. Aunque la suma total de los TSpec puede ser mayor que los recursos de red, la utilización real de los recursos es menor que la totalidad disponible; el elemento de red conoce la utilización real mediante el uso de técnicas estadísticas o midiendo los recursos utilizados al momento que existe la negociación para un nuevo flujo.

El elemento de red no puede asumir la presencia de tráfico que no cumple el TSpec como un acto inusual, pues bajo ciertas circunstancias un gran número de paquetes caen fuera del TSpec, lo cual es considerado normal. Además, el elemento de red tampoco puede asumir que los elementos precedentes realicen algún control, en cambio se debe realizar un control local tal vez con algoritmos de ordenamiento.

Como se indicó en el párrafo anterior, en ciertas circunstancias un alto número de paquetes caen fuera del TSpec especificado en el control de admisión, particularmente si un *router* actúa como punto de división en un árbol de distribución *multicast* soportando una reservación compartida. El servicio indica que al exceso de tráfico de carga controlada se lo maneje como *best effort*, sin embargo existe el inconveniente de que éste se apodere de la mayoría de recursos asignados al tráfico *best effort*.

Se utilizan dos formas para proteger al tráfico elástico del inelástico, la primera es manejar prioridades asignando una mayor prioridad al tráfico elástico, la cual es adecuada en presencia de aplicaciones en tiempo real con velocidad de transmisión no adaptativa; la segunda es mantener diferentes asignaciones de recursos, adecuada para aplicaciones en tiempo real con velocidad de transmisión adaptativa. La implementación de lo mencionado puede hacerse a través de algoritmos de control de tráfico como WFQ o RED. [44]

El servicio no especifica si degradar la calidad de todo el flujo o dar un trato de menor calidad sólo a los paquetes que no cumplen el TSpec. En el primer caso se incrementa el retardo y la pérdida de paquetes de todo el flujo pero se mantiene el orden de los paquetes, haciéndolo útil para protocolos como TCP cuyo rendimiento depende bastante del orden de llegada de los paquetes. En el segundo caso un subgrupo de paquetes tiene bajo retardo y pérdida de paquetes mientras el otro subgrupo tiene altos niveles de retardo y pérdida de paquetes, siendo adecuado para aplicaciones altamente sensitivas al tiempo como una conferencia interactiva. Sin embargo, previo a la manipulación del tráfico no contemplado en el TSpec, se tratará de rediseñar el mismo a fin de que cumpla con el TSpec, una manera de lograrlo es retardando un paquete lo máximo permisible.

5.2.1.5 Implementación del Servicio de Carga Controlada

Una forma de implementación del servicio de carga controlada es proveer un mecanismo de encolamiento con dos niveles de prioridad: un nivel de prioridad alta para la carga controlada y un nivel de baja prioridad para el servicio *best effort*. Para limitar el tráfico en la cola de alta prioridad se utiliza control de admisión, cuyo algoritmo puede basarse en los parámetros especificados en el TSpec de los flujos existentes, o en las características medidas de los flujos existentes y en el TSpec del nuevo requerimiento.

Otra forma de implementación aprovecha la capacidad de los actuales elementos de red de soportar clases de tráfico basándose en mecanismos como el encolamiento justo ponderado WFQ o el encolamiento basado en clases CBQ. *En*

este caso es suficiente mapear los flujos de datos en una clase de tráfico con capacidad disponible.

El algoritmo de control de admisión puede basarse en los TSpec especificados o en las características medidas del tráfico existente. El primer caso es apropiado cuando el número de flujos en la clase de prioridad alta es pequeño ó las características de tráfico de los flujos tienen alta variación, bajo estas situaciones la conducta medida de la corriente de tráfico no permite pronosticar efectivamente el tráfico futuro, incurriendo en resultados incorrectos. En el segundo caso, si la conducta medida permite pronosticar el tráfico futuro se puede admitir más tráfico en el servicio de carga controlada sin degradar el rendimiento. Para la implementación se puede elegir conmutar entre estos dos tipos de control de admisión dependiendo de las condiciones descritas.

El servicio de carga controlada puede ser requerido por cualquier aplicación que utilice el servicio *best effort* pero es más adecuado para aquellas aplicaciones que pueden caracterizar sus requerimientos de tráfico, por ejemplo aquellas basadas en el transporte continuo de datos multimedia como audio o video digitalizados. Debido a que los servicios integrados no proveen información explícita del retardo de transmisión, las aplicaciones sensibles al retardo necesitan mecanismos de control, los cuales pueden ser dados por el protocolo de transporte en tiempo real RTP (*Real Time Protocol*).

En general, cualquier aplicación sensible a la carga de la red puede solicitar el servicio de carga controlada cuando el servicio *best effort* se ha degradado, este tipo de funcionamiento asegura la entrega de calidad de servicio estable durante toda la sesión, además ofrece gran flexibilidad y ahorro si se tiene en cuenta que cualquier nivel de servicio superior al *best effort* tiene un costo adicional.

5.2.1.6 Evaluación del Servicio de Carga Controlada

Dado que los servicios de carga controlada pretenden entregar un servicio similar al *best effort* en condiciones de baja carga, se deduce el siguiente modelo de evaluación:

1. Comparar el servicio dado al tráfico *best effort* y al de carga controlada en condiciones de baja carga.

- Medir la pérdida de paquetes y el retardo de un flujo de prueba utilizando servicio *best effort* en ausencia de carga en el elemento de red.

- Comparar las medidas anteriores con las hechas al mismo flujo utilizando servicio de carga controlada en ausencia de carga en el elemento de red.

Mientras más cercanas estén las medidas, el servicio obtendrá un mayor puntaje de evaluación.

2. Comparar el servicio dado al tráfico *best effort* y al de carga controlada mientras se incrementa la carga.

- Medir las características de retardo y pérdida de paquetes de un flujo servido con carga controlada mientras se incrementa la carga de tráfico en el elemento de red.

Características que se mantienen esencialmente constantes mientras aumenta la carga merecen un mayor puntaje de evaluación.

Este modelo aunque da una idea global no realiza una evaluación completa del rendimiento del servicio. Tres variables adicionales afectan la evaluación: la primera es la característica de ráfaga utilizada por el tráfico empleado en las pruebas, la segunda es el grado de cambio de los parámetros especificados en el TSpec, y la tercera es la relación entre el tráfico de carga controlada y cualquier otro tipo de tráfico. La tercera variable puede evaluarse con el siguiente procedimiento:

- Sobrecargar el elemento de red únicamente con tráfico *best effort*.

- Hacer un requerimiento de servicio de carga controlada especificando altos valores para *r* y *b* (incluidos en el TSpec). Si el requerimiento es aceptado comprobar el desempeño utilizando el modelo descrito.

- Repetir estos experimentos para determinar el rango de *r* y *b* bajo el cual se obtiene un correcto funcionamiento. Con los valores obtenidos de *r* y *b* se define una región de manipulación exitosa del TSpec, mientras mayor sea esta región, mayor será el puntaje de evaluación otorgado al elemento de red.

5.2.2 SERVICIO GARANTIZADO

El Servicio Garantizado supone un grado mayor de Calidad de Servicio que el Servicio de Carga Controlada. Este servicio se detalla a continuación.

5.2.2.1 Conducta extremo a extremo

Este servicio es dirigido a aquellas aplicaciones que demandan un límite fijo de retardo para funcionar correctamente, un ejemplo son las aplicaciones de *playback* de audio o video, aunque en general cualquier aplicación en tiempo real puede solicitar el servicio.

Este servicio garantiza un máximo retardo total y un ancho de banda fijo, asegurando que ningún paquete se pierda debido al desbordamiento de la cola. Las garantías que ofrece el servicio se pueden perder sólo si hay fallas en la red o cambios en la ruta.

El servicio garantizado calcula el retardo máximo basándose en la combinación de varios parámetros obtenidos en cada *router* a lo largo de toda la ruta, los cuales se deducen a partir de la información otorgada por la fuente y contenida en el TSpec (expectativa de tráfico).

Permite un máximo nivel de control del retardo y para ello necesita que todos los *routers* involucrados sean capaces de ofrecer información del retardo producido en sí mismos y en el enlace aledaño. Esta condición no quiere decir que todos los *routers* en Internet sean capaces de ofrecer este servicio para la implementación del mismo en la red, pues un ISP podría implementarlo en su intranet para garantizar el servicio en sus diferentes puntos de presencia (POPs, *Points Of Presence*).

El retardo total está formado por el retardo de transmisión, inherente al medio, y el retardo de encolamiento en los elementos de red, éste último está bajo el control de la aplicación y depende de los parámetros *b* y *R* definidos posteriormente (sección 5.2.2.3).

Existen varios mecanismos de configuración y de identificación de flujos, por ejemplo se puede garantizar una reservación por medio de un protocolo de reservación de recursos como RSVP o también manualmente, sin embargo los servicios garantizados son definidos independientes de tales mecanismos. Además, como es lógico, debe existir un control de admisión.

Este servicio garantiza un retardo máximo y no un retardo promedio ni una variación del mismo o *jitter*, usualmente el retardo garantizado es mucho mayor que el retardo normal y ocasionalmente se produce un elevado nivel de *jitter*, situación que las aplicaciones deben considerar sobretodo al dimensionar el tamaño de los *buffers*. [45]

5.2.2.2 Requerimientos de los elementos de red

1. El nivel de servicio de cada flujo se caracteriza en cada elemento de red por los parámetros *R*, que define la velocidad del servicio, y *B*, que define el espacio en los *buffers*. El elemento de red debe ser capaz de asegurar el cumplimiento de los valores *R* y *B* mientras dure el flujo.

2. Si un flujo recibe una velocidad de servicio *R*, el servicio garantizado se comportará lo más próximo posible al servicio que recibiría el flujo si existiera un enlace dedicado entre el origen y el destino, independientemente de todos los demás flujos.

3. Si *r* y *b* representan respectivamente la velocidad y el espacio en *buffers* del tráfico generado, y *R* es la velocidad de servicio, el retardo máximo ideal producido en el elemento será igual a *b/R*; sin embargo, el elemento de red debe asegurar un retardo menor o igual al especificado en la ecuación 5.1, en la cual *C* y *D* son dos constantes que definen el máximo nivel de error (refiérase a la sección 5.2.2.4).

$$retardo_{elemento} \leq \frac{b}{R} + \frac{C}{R} + D \qquad (5.1)$$

4. Los enlaces no deben fragmentar los paquetes correspondientes a un flujo con servicio garantizado. Los datagramas más grandes que el MTU del enlace no se consideran parte del flujo garantizado y se procede de acuerdo a lo indicado en la sección de las políticas de funcionamiento.

5.2.2.3 Caracterización del tráfico (TSpec) y de los recursos (RSpec)

Un flujo que requiere servicio garantizado debe especificar las características del tráfico que se generará, contenido en el TSpec, y en respuesta recibirá de la red los recursos asignados contenidos en el Rspec (*Resource Spectation*). Si se desea cambiar la reservación se deberá invocar nuevamente al control de admisión a menos que la nueva relación entre TSpec y RSpec sea menor que la relación existente.

Los parámetros que conforman el **TSpec** son: velocidad de transmisión promedio "*bucket rate* " (*r*), espacio en *buffer* "*bucket size*" (*b*), velocidad de transmisión pico (p), mínimo tamaño de paquete (*m*) y máximo tamaño de paquete (*M*). Estos parámetros son los mismos que aquellos definidos en el TSpec utilizado por el servicio de carga controlada y por lo tanto cumplen lo especificado en el punto 5.2.1.3.

Los parámetros que conforman el **RSpec** son: velocidad de servicio (*R*) e indicativo de inactividad (*S*).

Velocidad de Servicio (*R*): Medida en bytes de datagramas IP por segundo, su valor puede variar desde 1 byte/s hasta 40 terabytes/s. Especifica la velocidad que un *router* asignará a un flujo con servicio de carga controlada y, por lo tanto, debe ser mayor que el parámetro *r*.

Indicativo de Inactividad (*S*): Medido en microsegundos, especifica la diferencia entre el retardo deseado y el retardo obtenido usando un nivel de reservación *R*.

Es utilizado por el elemento de red para reducir el nivel de reservación de un flujo. Si su valor no es especificado se configura *S* igual a cero.

Los servicios garantizados utilizan un número de punto flotante de precisión simple para representar el valor de *R* y un entero de 32 bits para la representación de *S*, parámetro que puede adoptar valores desde 0 a $(2^{32}-1)$ microsegundos. El exponente de los números de punto flotante se representa con ocho bits, siendo el primero el correspondiente al signo, en la tabla 5.1 se detallan los valores permitidos para el servicio.

EXPONENTE	REPRESENTACIÓN	TIPO DE RANGO
[-126,0]	[0,127]	NO VÁLIDO
[+1,+34]	[128,161]	VÁLIDO
[+35,+127]	[162,254]	NO VÁLIDO
[+128]	[255]	VÁLIDO

Tabla 5.1 Exponente en los números de punto flotante. [45]

Para el funcionamiento de un elemento de red que curse tráfico garantizado es necesario diferenciar varios tipos de RSpec, a continuación se analizan algunas reglas. [45]

a. En el servicio garantizado un RSpec A es tan bueno o mejor que un RSpec B si y solo si:

1. $R_A \geq R_B$
2. $S_A \leq S_B$

b. Cuando se hace reservaciones a veces es necesario caracterizar los recursos asignados a varios flujos con un solo conjunto de parámetros RSpec, los cuales representarán el mínimo común para todos los flujos y serán calculados de la siguiente manera:

1. $R_c = \mathrm{MAX}(R_i)$
2. $S_c = \mathrm{MIN}(S_i)$

5.2.2.4 Información a exportar

Cada módulo del servicio garantizado debe exportar dos parámetros característicos, *C* y *D*, los cuales representan el error entre la implementación real y el modelo ideal.

Término de error *C*: Es el término de error dependiente de la velocidad de transmisión, representa el retardo que un datagrama puede experimentar debido a la velocidad del flujo.

Término de error *D*: Es el término de error independiente de la tasa de transmisión, representa el retardo máximo que introduce el elemento de red. Generalmente se determina o ajusta al instante de arranque (*boot*) o al momento de la configuración.

Los nodos finales necesitan calcular el retardo extremo a extremo y para ello utilizan los términos de error C_{tot} y D_{tot}, los cuales corresponden a la suma de todos los términos *C* y *D* en cada *router*. Además, los términos de error parciales C_{sum} y D_{sum}, que corresponden a la suma de C y D en todos los *routers* previos, incluido el actual, son utilizados por el *router* para dimensionar el espacio adecuado en los *buffers* a fin de evitar pérdidas.

El término de error C se mide en bytes y su valor oscila entre 1 y 2^{28} (algo más de 250 megabytes) en un elemento individual, mientras el valor total, sobre todos los elementos, no puede ser mayor que 2^{32}-1. El término de error D se mide en microsegundos y su valor oscila entre 1 y 2^{28} (algo más de dos minutos) para un elemento individual, además, el retardo total no puede ser mayor que 2^{32}-1. [45]

5.2.2.5 Políticas de Funcionamiento

Existen dos políticas de funcionamiento de los servicios garantizados: simple y rediseño (*reshaping*). La primera simplemente compara el tráfico generado con el especificado en el TSpec, de acuerdo a ello se acepta o rechaza una reservación; la segunda intenta rediseñar el tráfico de manera que le permita cumplir con los parámetros del TSpec, esto se consigue almacenando en *buffers* el tráfico transmitido en exceso.

En la política de funcionamiento simple, la negociación de recursos se la hace en el extremo de la red. El rediseño se lo hace en cambio en todos los puntos de división ó fusión de tráfico en los cuales cada ramal presenta diferentes parámetros TSpec y se necesita obtener un mínimo común de todos ellos. La identificación de estos puntos es responsabilidad del protocolo de reservación o del mecanismo de configuración del servicio en general.

Los servicios garantizados entregan QoS a un flujo si la cantidad de datos enviados no supera M + MIN(pT, $rT+b-M$), si no se cumple esta condición, los datos enviados no son considerados parte del flujo con servicio garantizado.

En el extremo de la red, si el tráfico no cumple el TSpec, éste debería ser tratado como tráfico *best effort*, pero si el objetivo no es suministrar QoS sino compartir el enlace sería mejor descartar el paquete.

Dentro de la red, las políticas de funcionamiento no producen los resultados deseados porque los efectos del encolamiento ocasionalmente causarán que el tráfico que ingresa a la red cumpliendo el TSpec experimente cambios que lo hagan incumplir en algunos elementos de red. Bajo estas circunstancias se utiliza el rediseño del tráfico.

El rediseño se logra almacenando en *buffers* el exceso de tráfico hasta que éste pueda ser enviado de acuerdo a los parámetros r, b y p especificados en el TSpec. El espacio en *buffers* requerido para rediseñar el tráfico es $b+C_{sum}+D_{sum}.r$ que corresponde a las ráfagas de tráfico comprometidas más los errores introducidos en los elementos de red previos.

Los servicios garantizados no permiten la fragmentación de un paquete cuyo flujo ha recibido el servicio, pero si el paquete es más grande que el MTU del enlace no es considerado parte del flujo y se le entrega un servicio *best effort*, pudiendo ser fragmentado o descartado si hay congestión.

5.2.2.6 Implementación del Servicio Garantizado

Para la entrega de servicio garantizado las subredes y los *routers* deben ser capaces de soportar el servicio, pero típicamente, las subredes no son capaces

de negociar servicios basadas en protocolos IP, por lo tanto, los *routers* deben actuar como *proxies* de las subredes aledañas.

En ciertos casos el servicio de *proxy* será fácil, por ejemplo en una línea dedicada administrada por el algoritmo de organización de tráfico WFQ, en la cual el *proxy* simplemente necesita asegurarse que la suma de las tasas de transmisión de todos los flujos no excedan el ancho de banda del enlace y además debe anunciar los valores de retardo *C* y *D*. En otros casos el servicio de proxy será complejo, por ejemplo en una red ATM se debe establecer primero un circuito virtual y luego calcular los valores de *C* y *D* para ese circuito virtual.

Los parámetros correspondientes a las tasas de transmisión contenidos en el TSpec miden el tráfico IP y no contabilizan las cabeceras de la capa enlace, por lo tanto, los elementos de las subredes deben ajustar la velocidad considerando las cabeceras añadidas en el nivel de enlace. Esta consideración es aplicable también al caso de túneles en el cual se añaden cabeceras que hacen posible la encriptación.

En una red de paquetes, la máxima velocidad de la cabecera se calcula de acuerdo a la siguiente ecuación, en la cual *r* es la tasa de transmisión, *b* es la ráfaga de datos permitida y m es el tamaño de paquete mínimo (cabecera).

$$r_{overhead} = \frac{r.b}{m} \qquad (5.2)$$

Para redes que tienen fragmentación interna como ATM el cálculo es más complejo pues se debe tomar en cuenta la fragmentación y los bytes de relleno. Un estimativo de la tasa de transmisión adicional para ATM AAL5 más segmentación y reensamblaje ATM se da en la ecuación 5.3, la cual representa la velocidad dividida en celdas de 48 bytes multiplicada por la cabecera ATM de 5 bytes, más la máxima velocidad del datagrama (r/m) multiplicada por el costo de 8 bytes de cabecera AAL5 más el máximo espacio que puede ser gastado en una celda, el cual es 52 bytes en una celda que contiene 1 byte de datos. [45]

$$r_{overhead} = \left(\frac{r}{48} * 5\right) + \left(\frac{r}{m} * (8 + 52)\right) \qquad (5.3)$$

Para asegurar que los datagramas no se pierdan se debe configurar en el *router* un espacio en *buffers B* igual al parámetro *b* (en el TSpec) más un término de error. Si la red se comportara idealmente bastaría con configurar *B* = *b*, pero como se espera que el tráfico adquiera mayores características de ráfaga a medida que pasa por la red, se debe hacer que *B* cumpla la ecuación 5.4.

$$B = b + C_{sum} + D_{sum}.R \qquad (5.4)$$

Si se contabiliza la tasa de transmisión pico (*p*) se obtendrían tres casos:

1. Si $\frac{b-M}{p-r} < \frac{C_{sum}}{R} + D'_{sum}$ entonces $B = b + C_{sum}\frac{r}{R} + D'_{sum}.r$

2. Si $\frac{b-M}{p-r} \geq \frac{C_{sum}}{R} + D'_{sum}$ y $p > R$

   entonces $B = M + \frac{(b-M)(p-R)}{p-r} + C_{sum} + D'_{sum}R$

3. cualquier otro caso $B = M + C_{sum}\frac{P}{R} + D'_{sum}.p$

Para estas tres ecuaciones se considera *p* la velocidad a la cual se pone la ráfaga de datos *b* en la cola y además que $D'_{sum} = D_{sum} + S_{out}$[1].

La utilización del parámetro *p* no es crítica pudiéndose omitir si la aplicación lo desea pues ello introducirá errores muy pequeños. Para el cálculo del espacio en *buffers* generalmente se utiliza la ecuación 5.4. Así mismo, el cálculo del retardo total se lo obtiene a través de la ecuación 5.1, reemplazando los valores *C* y *D* por C_{tot} y D_{tot} respectivamente.

El parámetro *D* en cada elemento de red debería representar la máxima variación del retardo (independiente de la tasa de transmisión) en la transferencia de un

[1] S_{out}.- Es el valor que tiene el término de inactividad (sección 5.2.2.3) a la salida del router. Este término es susceptible a cambios en el router, si se desea modificar el nivel de reservación.

datagrama. Por ejemplo si el algoritmo de encolamiento es WFQ, *D* debe ser igual al MTU divido para el ancho de banda del enlace, ya que si un paquete arriba a la cola de salida justo cuando está siendo transmitido otro paquete del tamaño del MTU deberá esperar que se termine la transmisión del paquete actual antes de ser transmitido.

El término *D* es muy difícil de calcular en transmisión *multicast* en la cual la característica del retardo depende del camino tomado; para solucionar este problema se calcula un valor representativo de variación del retardo para la subred.

El parámetro *C* corresponde a los datos acumulados debido a la desviación entre la implementación de un algoritmo específico y el servicio ideal bit a bit. Por ejemplo, para WFQ el valor de *C* corresponde al máximo tamaño del paquete (*M*).

El parámetro de inactividad *S* se utiliza para reducir el nivel de la reservación, su valor es almacenado en el elemento de red y éste debe permanecer constante durante la sesión pues ello garantiza consistencia en el proceso de reservación. Para ilustrar el uso de *S*, considérese el caso en el cual el retardo extremo a extremo requerido D_{req} es mayor que el retardo obtenido, *S* está definido de la siguiente manera:

$$S = D_{req} - \left(\frac{b}{r} + \frac{C_{tot}}{r} + D_{tot} \right) \qquad (5.5)$$

S puede ser usado por los elementos de red para ajustar sus reservaciones y que les permitan admitir flujos adicionales. Un elemento de red intermedio que diferencia entre garantías de retardo y velocidad puede usar esta información para disminuir la cantidad de recursos asignados a un flujo. Por ejemplo un elemento de red que use WFQ puede disminuir su reservación local de R_{in} a R_{out} usando algo de *S*.

Un claro ejemplo del uso de este servicio está en las aplicaciones intolerantes al retardo y a la pérdida de paquetes, las cuales usan los valores C_{tot} y D_{tot} para calcular el límite del retardo de un servicio que requiere una tasa de transmisión

R. Asumiendo que $R < p$ se puede configurar el punto de *playback* de la aplicación de acuerdo a la siguiente ecuación:

$$retardo_{playback} = \left(\frac{b-M}{R}\right)\left(\frac{p-R}{p-r}\right) + \frac{M+C_{tot}}{R} + D_{tot} \qquad (5.6)$$

5.2.2.7 Evaluación del Servicio Garantizado

Los algoritmos de control de admisión y de organización deben asegurar que nunca se violen los límites del retardo ni se pierdan paquetes, si el tráfico generado se encuentra enmarcado en los parámetros especificados en el TSpec. Los elementos de red deben asegurar que los flujos con comportamiento inadecuado no afecten el desempeño de los demás flujos.

5.3 PROTOCOLO DE RESERVACIÓN DE RECURSOS RSVP

5.3.1 INTRODUCCIÓN

El Protocolo de Reservación de Recursos RSVP es usado para establecer y mantener reservaciones de recursos que permitan suministrar calidad de servicio a un flujo de datos. Los requerimientos son evaluados en cada *router* en el camino del flujo, siendo RSVP el medio de transporte de éstos.

Las principales características de este protocolo son:

- RSVP permite reservaciones para aplicaciones *unicast* y *multicast*.
- RSVP es *simplex*, es decir hace reservaciones a flujos de datos unidireccionales.
- RSVP es orientado al receptor, es decir el receptor de un flujo de datos inicia y mantiene la reservación de recursos para el flujo.
- RSVP mantiene un estado "suave" (refiérase a la sección 5.3.3.1.4) en los *routers* y *hosts*, soportando los cambios dinámicos de membresías y adaptándose dinámicamente a los cambios de rutas.

- RSVP no es un protocolo de enrutamiento pero depende de los actuales y futuros protocolos de enrutamiento.
- RSVP transporta y mantiene los parámetros de control de tráfico y de políticas.
- RSVP provee varios estilos de reservación.
- RSVP provee operación transparente a través de todos los *routers* que no lo soportan.
- RSVP soporta las dos versiones del Protocolo de Internet, IPv4 e IPv6.

La siguiente figura muestra la estructura funcional de un *host* y un *router* que soportan el protocolo RSVP.

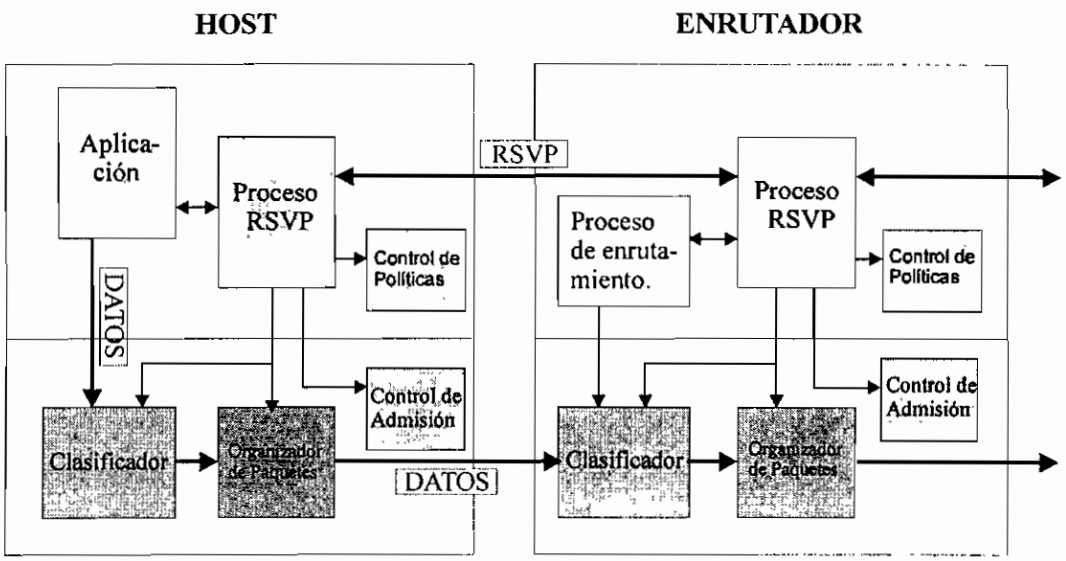

Figura 5.3 Protocolo RSVP en los *hosts* y *routers*. [46]

RSVP define como sesión al flujo de datos que tiene un destino particular y un protocolo de capa transporte. Una sesión está definida por tres campos: (*DestAddress*, *Protocol Id*, *DstPort*) en donde, *DestAddress* es la dirección de destino IP (*unicast* o *multicast*); *Protocol Id* es la identificación del protocolo IP; y, *DstPort* representa un puerto de destino generalizado que podría ser el puerto de destino UDP/TCP. Este tercer parámetro es opcional en ciertas ocasiones, pero

es necesario cuando se debe distinguir entre varios flujos *unicast* direccionados al mismo destino, además debe ser consistente en todos los nodos a lo largo del camino.

El flujo de datos dentro de una sesión RSVP puede ser *multicast* o *unicast*. Si es *multicast* se envía una copia de cada paquete de la fuente s_i a cada destino r_j; si la sesión es *unicast* habrá un solo *host* destino R, pero pueden existir varias fuentes.

5.3.2 MODELO DE RESERVACIÓN

El proceso RSVP empieza cuando una fuente genera un mensaje *Path*, el cual es enviado a través de la red para documentar la ruta de la reservación de extremo a extremo (obsérvese la figura 5.4). El paquete contiene información que permite identificar el flujo que demanda el servicio y los parámetros que definen el servicio esperado. En un punto dado el paquete contiene la dirección IP del salto previo e información de la capacidad y retardo introducido en el nodo.

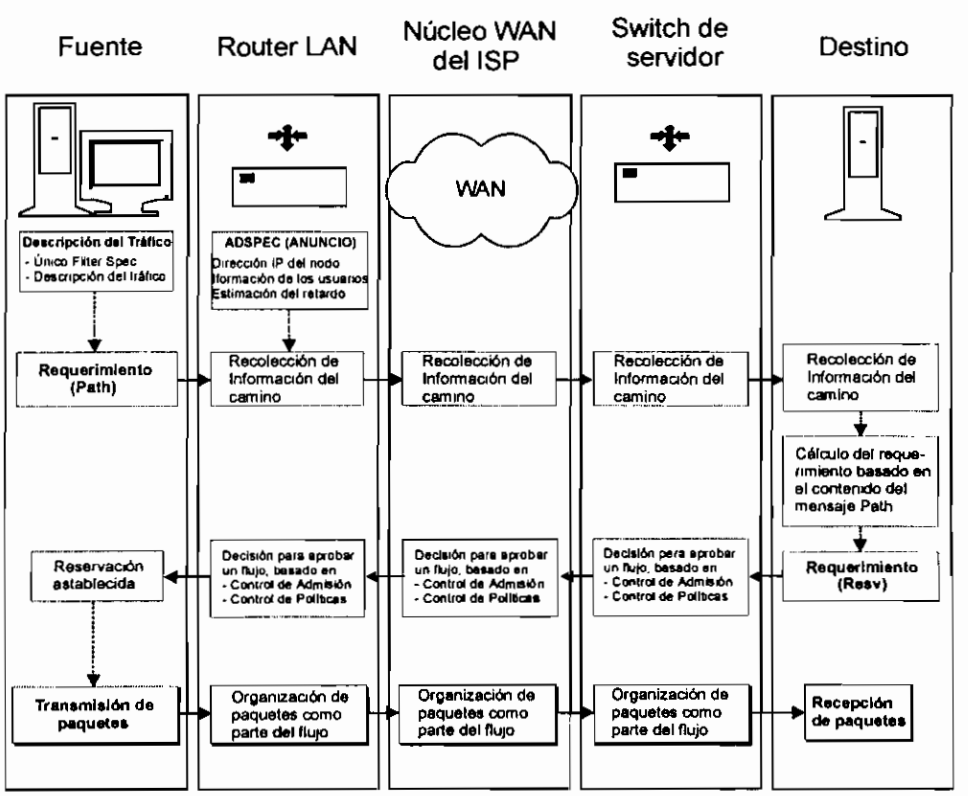

Figura 5.4 Modelo de una reservación RSVP. [6]

Luego que este mensaje llega al destino, el receptor tiene una idea clara de la ruta, de los servicios y capacidades relativas que la red puede ofrecer, las cuales comparándolas con la descripción del tráfico de la fuente, genera un mensaje de reservación *Resv*. Este mensaje contiene el mismo clasificador que permite identificar al flujo y además información que describe el tipo de reservación, la cual puede ser garantizada o de carga controlada.

En cada nodo se realizan las pruebas de control de admisión y control de políticas, la primera para verificar si hay recursos disponibles y la segunda verifica si el transmisor tiene permiso para hacer la reservación; si ambas pruebas son exitosas entonces se realiza la reservación.

Se debe observar que quien realmente empieza la reservación es el receptor, pues aunque aparentemente lo más obvio sería que la inicie el transmisor, ya que éste conoce las características del tráfico que enviará, en cambio el receptor sabe lo que desea (o lo que puede) recibir. Si se dejara la tarea de iniciación de la reservación al transmisor, ello provocaría problemas de escalamiento en árboles *multicast* grandes, dinámicos, y con receptores heterogéneos.

Estos problemas de escalamiento se resuelven dejando que el receptor sea el que inicie la reservación, de este modo se manipula fácilmente una reservación para receptores heterogéneos ya que cada receptor simplemente solicita la reservación para sí mismo, y si se presentan reservaciones diferentes, simplemente éstas se combinan dentro de la red. La iniciación del receptor también es consistente con el *multicast* del protocolo IP, en el cual un grupo *multicast* se crea implícitamente por los receptores que se unen a él.

En el modelo de reservación, el receptor envía el mensaje *Resv* y cada nodo en el camino acepta o rechaza la reservación, esta forma de reservación puede resultar difícil cuando se desea hallar un determinado servicio extremo a extremo. Para evitar ello se puede utilizar el objeto *ADSPEC* que recoge información en cada nodo y que permite hacer una predicción de la calidad de servicio extremo a extremo. Esta información puede ser usada por el receptor para construir o ajustar dinámicamente la reservación.

La anterior es una explicación bastante simple del modelo de reservación que utiliza el protocolo RSVP, una explicación más detallada se encuentra en el punto 5.3.3 que corresponde a los mecanismos del protocolo.

5.3.2.1 Estilos de Reservación

Existen varias formas de hacer reservaciones, una alternativa es realizar una reservación para cada transmisor (reservación distinguible), o efectuarla para un conjunto de transmisores (reservación compartida); otra opción involucra la selección de los transmisores, la cual puede basarse en una lista (selección explícita), o puede ser abierta a cualquier transmisor (selección *wildcard* o comodín). Una selección explícita necesita que el *filter spec*[1] coincida exactamente con un transmisor, mientras que en una selección comodín no se necesita *filter spec*.

De acuerdo a estas opciones se tiene los siguientes estilos de reservación:

Selección del transmisor	Reservación	
	Distinguible	Compartida
Explícita	**Estilo Filtro Fijo (FF)**	**Estilo Compartición Explícita (SE)**
Comodín	**No definido**	**Estilo Filtro Comodín (WF)**

Tabla 5.2 Estilos de Reservación. [46]

5.3.2.1.1 Estilo Filtro Comodín (WF, Wildcard Filter)

Este estilo crea una sola reservación compartida por los flujos de todos los transmisores (de una misma sesión), esta reservación corresponde a la mayor cantidad de recursos requeridos por todos los receptores y es independiente del número de transmisores que estén usándola. En este estilo, la reservación se propaga a través de todos los transmisores y se extiende automáticamente a nuevos transmisores de acuerdo a como éstos vayan apareciendo.

[1] El ***filter spec***, junto con la especificación de la sesión, define el conjunto de paquetes de datos (flujo) a recibir el QoS definido por el *flowspec*. El formato exacto depende del protocolo que se utilice, Ipv4 o Ipv6.

Simbólicamente se representa por WF(* {Q}) donde el asterisco representa la forma de selección del transmisor (comodín) y Q representa el *flowspec*[1].

5.3.2.1.2 Estilo Filtro Fijo (Fixed Filter)

Este estilo crea una reservación distinta para los paquetes de datos de un emisor en particular y no la comparte con los paquetes de otros emisores. La selección de los transmisores es explícita y se la puede hacer por medio de una lista.

Simbólicamente se representa por FF(S {Q}) donde S es el emisor seleccionado y Q corresponde al *flowspec*. RSVP permite múltiples reservaciones FF elementales solicitadas al mismo tiempo, para lo cual utiliza una lista de descriptores del flujo FF($S_1\{Q_1\}$, $S_2\{Q_2\}$, ...$S_N\{Q_N\}$); en este caso la reservación total del enlace es la suma de Q_1, Q_2, ...Q_N.

5.3.2.1.3 Estilo Compartición Explícita (SE, Shared-Explicit)

Este estilo implica una reservación compartida y una forma explícita de selección del transmisor.

Simbólicamente se representa por SE((S_1, S_2, ..., S_N) {Q}) donde Q es el *flowspec* y (S_1, S_2, ..., S_N) es la lista de emisores.

Los estilos SE y WF que permiten compartir la reservación son adecuados para aplicaciones *multicast* en las que es bastante improbable el funcionamiento simultáneo de todas las fuentes, un ejemplo son las aplicaciones de audio. En cambio el estilo FF puede ser utilizado por señales de video.

Las reglas que rigen el protocolo RSVP no permiten la combinación de estos estilos, por lo tanto WF, SE y FF son mutuamente incompatibles. Sin embargo se puede simular WF con SE si se considera un alto número de transmisores, aunque se introduce un *overhead* significativo.

A continuación se muestra un ejemplo de cada estilo de reservación, para ello considérese la figura 5.5 en la cual (a) y (b) son las interfaces de entrada; (c) y (d)

[1] El ***flowspec*** especifica el QoS deseado e incluye generalmente dos conjuntos de parámetros numéricos: TSpec y RSpec.

son las interfaces de salida; S1, S2 y S3 los transmisores; y, R1, R2 y R3 los receptores, además R2 y R3 están en una LAN de difusión. Se asume que los paquetes provenientes de cada transmisor S_i son enrutados a ambas interfaces de salida.

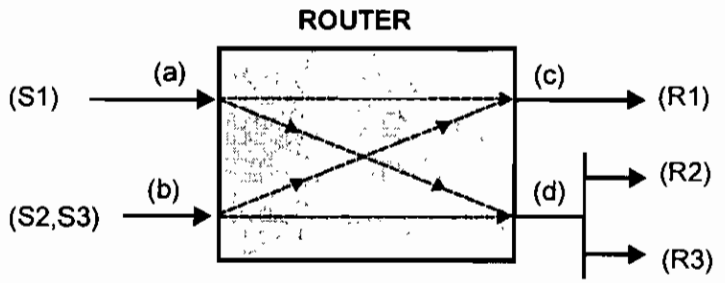

Figura 5.5 Configuración del *router* [46]

Por simplicidad, los ejemplos consideran *flowspecs* unidimensionales y múltiplos enteros de una cantidad de recurso base B. En las figuras 5.6, 5.7 y 5.8 la columna "recibe" muestra los requerimientos de reservación sobre cada interfaz de salida, la columna "reserva" muestra la reservación resultante en cada interfaz de salida y la columna "envía" muestra los requerimientos de reservación que son enviados hacia los saltos previos.

La figura 5.6 ilustra el estilo de reservación filtro comodín (WF). Obsérvese que R2 y R3 hacen requerimientos de 4B y 2B respectivamente, siendo 4B la combinación de las dos y ésta cantidad de recurso se reserva en la interfaz (d). En la interfaz (c) la reservación es directa y es igual a 5B. Debido a la configuración del *router*, por las interfaces (a) y (b) se envía el mismo requerimiento de recursos, el cual corresponde a la combinación de las reservaciones en (c) y (d) y su valor es igual a 5B.

La figura 5.7 muestra el estilo de reservación filtro fijo (FF), el mismo que selecciona explícitamente las fuentes de tráfico y además no permite la compartición de la reservación. Nótese la discriminación del emisor en cada interfaz.

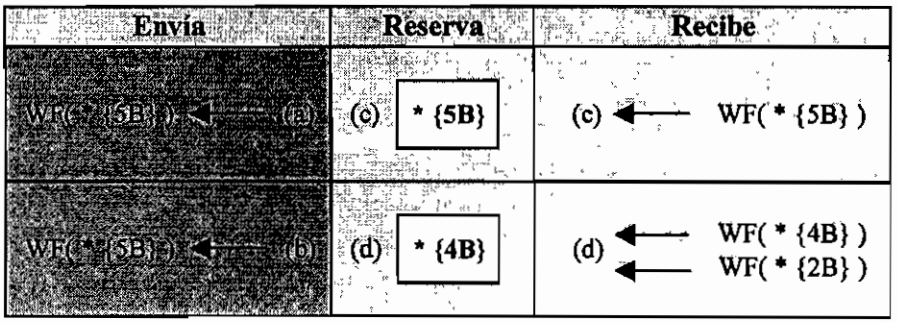

Figura 5.6 Ejemplo del estilo de reservación filtro comodín (WF)

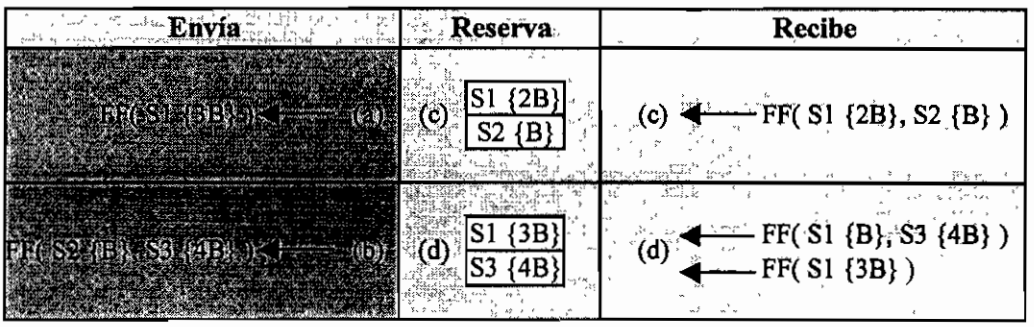

Figura 5.7 Ejemplo del estilo de reservación filtro fijo (FF)

El estilo de reservación compartición explícita (SE) se indica en la figura 5.8, en la misma se aprecia la distinción del emisor y la característica de compartición de la reservación en cada interfaz.

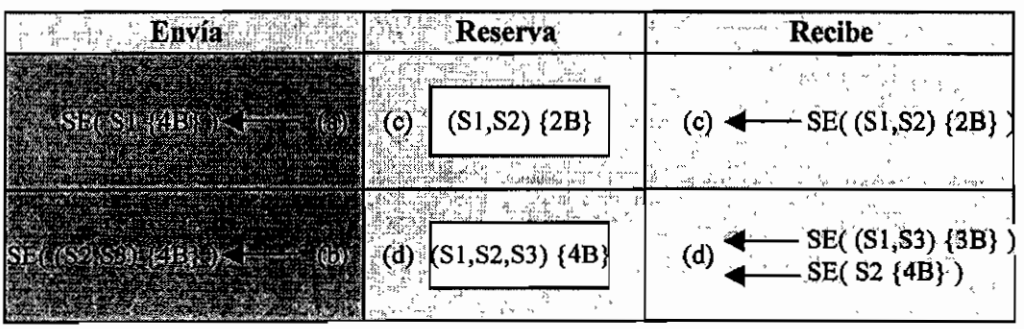

Figura 5.8 Ejemplo del estilo de reservación compartición explícita (SE)

5.3.3 MECANISMOS DEL PROTOCOLO RSVP

5.3.3.1 Mensajes *Path* y *Resv*

Existen dos tipos de mensajes fundamentales en el protocolo RSVP: *Path* y *Resv*.

5.3.3.1.1 Mensaje Path

Cada *host* RSVP fuente transmite periódicamente un mensaje *Path* a través de la ruta *unicast* o *multicast* dada por el respectivo protocolo de enrutamiento, esta ruta es la misma que utilizarán los paquetes de datos. La transmisión de este mensaje es no confiable, es decir no existe asentimiento alguno si el mensaje alcanza al receptor.

Este mensaje es el encargado de almacenar en los nodos el Estado del Camino (*Path State*) y contiene por lo menos la dirección IP del nodo previo, la cual será utilizada por los mensajes *Resv*, originados en el receptor, para llegar a la fuente. En entornos *multicast* este mensaje puede contener varias direcciones IP si ha salido por distintas interfaces.

Opcionalmente, este mensaje puede transportar información de la fuente, tal como el formato de los datos y las características de tráfico de los flujos de datos originados en la fuente, los cuales son usados por los mecanismos de control de tráfico para prevenir una sobre reservación y por lo tanto provocar fallas innecesarias en el control de admisión.

5.3.3.1.2 Mensaje Resv

Cada *host* destino envía hacia las fuentes un requerimiento de reservación RSVP a través de un mensaje *Resv*, el cual sigue la ruta inversa de los paquetes de datos y es el encargado de establecer y mantener el Estado de Reservación (*Resv State*) en el nodo. Para llegar a la fuente, el mensaje *Resv* lee la información contenida en el estado del camino (*Path* State) en cada nodo de la ruta.

Un requerimiento de reservación elemental consiste de un objeto *flowspec* y un objeto *filter spec*.

El ***flowspec*** especifica el nivel de calidad de servicio deseado y es usado para configurar los parámetros del organizador de paquetes u otros mecanismos de capa enlace del elemento de red. El *flowspec* incluye dos conjuntos de parámetros numéricos: RSpec que define los recursos y TSpec que describe los parámetros del tráfico a generarse.

El objeto ***filter spec***, junto a la especificación de la sesión, define el flujo o conjunto de paquetes que recibirán calidad de servicio y es usado para configurar los parámetros en el clasificador de paquetes. El formato específico del *filter spec* depende de la versión del Protocolo de Internet usada, IPv4 o IPv6, pero su forma más simple incluye la dirección IP de la fuente y el número del puerto UDP/TCP (ver ANEXO 5). Cada *router* debe ser capaz de examinar el campo correspondiente al puerto UDP/TCP pero no siempre es fácil pues la fragmentación de paquetes IP y la encriptación ocultan esta información, además si se usa IPv6 se tienen cabeceras variables incrementando las dificultades de la clasificación para QoS.

5.3.3.1.3 Reservación

Un requerimiento de reservación efectúa dos acciones generales en cada nodo intermedio: 1) efectúa una reservación en el enlace y 2) envía el requerimiento hacia los demás nodos.

Para realizar la reservación, el proceso RSVP pasa el requerimiento a través del control de admisión y control de políticas; si uno de los dos falla, la reservación es rechazada y el proceso RSVP regresa al receptor un mensaje de error; si ambos procesos son exitosos, el nodo configura el clasificador de paquetes para seleccionar los paquetes de datos definidos en el *filter spec*, e interactúa con la capa de enlace apropiada para obtener el QoS deseado y definido en el *flowspec*. Las reglas específicas para satisfacer un requerimiento de QoS bajo RSVP dependen sobre todo de la tecnología de capa enlace que se use en cada interfaz.

El nodo efectúa la segunda acción propagando el requerimiento de reservación hacia las fuentes apropiadas. El conjunto de *host*s fuente para los cuales se

propaga un requerimiento de reservación dado se conoce como "campo" del requerimiento. El requerimiento de reservación que envía un nodo puede diferir con el que recibe pues los mecanismos de control de tráfico pueden cambiar el *flowspec* en cada salto ó en ambientes *multicast*, en los cuales se debe combinar las reservaciones de varios receptores hacia un mismo transmisor o conjunto de transmisores.

El mensaje *Resv* envía al nodo previo un *flowspec* que es el más grande de todos los *flowspecs* requeridos en cada interfaz. A veces es imposible diferenciar un *flowspec* superior, por ejemplo entre dos *flowspec* en el que uno demanda ancho de banda y otro demanda bajo retardo, en este caso se calcula un *flowspec* efectivo que puede ser una combinación de los dos y que puede ser un *flowspec* al menos tan grande como cada uno. Las reglas para comparar *flowspecs* están fuera del alcance del protocolo RSVP pues están definidas en el servcio específico (servicio garantizado o servicio de carga controlada) y por lo tanto para su implementación se debe llamar a las correspondientes rutinas de combinación dependientes del servicio.

Una característica importante del protocolo es su capacidad de soportar cambios en la ruta. Cuando existe un cambio en la ruta, el protocolo de enrutamiento es el encargado de notificar lo ocurrido al respectivo proceso RSVP, el cual a su vez genera mensajes de refresco *Path* y recibe los correspondientes mensajes *Resv*.

5.3.3.1.4 Estados de la Reservación

El protocolo RSVP utiliza un estado "suave" para administrar el estado de la reservación. El estado de la reservación es creado y refrescado periódicamente por los mensajes *Path* y *Resv*; y, es borrado del nodo si no han arribado mensajes de refresco luego de cumplido un determinado periodo de tiempo (*cleanup timeout*). Un estado de reservación también se puede cancelar explícitamente con la ayuda del mensaje Cancelar (*teardown*) descrito en la siguiente sección.

Los mensajes de refresco sirven para actualizar el estado de la reservación y se envían a un intervalo de tiempo definido por el parámetro tiempo de refresco (*refresh timeout*). El formato de los mensajes (*Path* y *Resv*) iniciales y de refresco

es el mismo y solo los diferenciará el nodo, de acuerdo a su existencia previa en el mismo. Para mantener una reservación dinámica basta con cambiar los mensajes *Path* y *Resv*, y propagarlos inmediatamente hasta que alcancen un punto en el cual la combinación no produzca cambio en el estado de la reservación. Esto minimiza el *overhead* debido a los cambios producidos y es esencial para obtener escalabilidad en grandes grupos *multicast.*

Debido a que los mensajes RSVP se envían como datagramas IP no confiables, éstos pueden perderse, pero la situación de pérdida ocasional de paquetes se puede manejar: si el tiempo *cleanup timeout* es *k* veces el tiempo de refresco, entonces el protocolo RSVP puede tolerar *k-1* pérdidas sucesivas de los mensajes de refresco. A pesar de la flexibilidad a alguna pérdida de paquetes, los mecanismos de control de tráfico deben reservar recursos que permitan proteger a los mensajes RSVP de las pérdidas por congestión y además que los envíen sin retardo cuando han ocurrido cambios del estado de reservación.

5.3.3.2 Mensaje Cancelar (*Teardown*)

Este mensaje remueve el estado de la reservación inmediatamente y, aunque no es estrictamente necesario, su uso es recomendado.

Existen dos tipos de mensajes cancelar: *PathTear* y *ResvTear.* El mensaje *PathTear* viaja en el mismo sentido que los mensajes *Path* y borra los estados de *Path* a lo largo de la ruta. De la misma forma un mensaje *ResvTear* viaja por el camino que recorren los mensajes *Resv* borrando las reservaciones en cada nodo que atraviesan.

Para que termine la reservación en un nodo debe coincidir exactamente la información de identificación contenida en el nodo (*Path State* o *Resv State*) con la información transportada en el mensaje correspondiente (*PathTear* o *ResvTear*), si esto no ocurre, los mensajes serán descartados y no podrán seguir propagándose.

Un mensaje cancelar puede ser generado por los nodos extremos (emisor o receptor) o por un *router* intermedio que ha cumplido un determinado tiempo sin recibir mensajes de refresco.

Igualmente a los anteriores, este mensaje es enviado en una forma no confiable, sin embargo si se llegara a perder no causaría ninguna falla en el protocolo. Asumiendo que la probabilidad de pérdida del paquete es pequeña, el tiempo máximo para cancelar una reservación rara vez sería mayor que un intervalo de tiempo de refresco.

Es posible cancelar toda la reservación o solo un subconjunto del estado establecido, dependiendo de las condiciones de combinación de flujos presentes en cada nodo. Para el estado de reservación, el término más pequeño es un *filter spec* individual; y, para el estado de *Path*, el término más pequeño es un emisor simple.

5.3.3.3 Mensajes de errores y Estado de Bloqueo

Existen dos mensajes de errores: *ResvErr* y *PathErr*, generados si se produce un error en el procesamiento de los mensajes *Resv* y *Path* respectivamente.

Los mensajes *PathErr* viajan en dirección contraria a los mensajes *Path*, siendo enviados por el nodo donde se produjo el error hacia la fuente que los generó, utilizando para ello el estado *Path*; su objetivo es informar a la fuente que existió un error, y que ello no implique modificar el estado de cada nodo en la ruta.

Los mensajes *ResvErr* viajan en dirección contraria a los mensajes *Resv*, para ello utilizan el estado de la reservación presente en cada nodo sin alterarlo, sin embargo los errores en la reservación son más complejos de manejar que los anteriores pues son más probables que ocurran y si se producen pueden implicar varias reservaciones combinadas debiéndose enviar mensajes *ResvErr* a todos los receptores responsables. La combinación de requerimientos heterogéneos puede ocasionar que un requerimiento niegue el servicio a otro, normalmente existen dos acciones que producen este tipo de problemas:

- El primer problema se presenta cuando ya existe una reservación Q_0 y otro receptor hace una reservación mayor $Q_1>Q_0$, la combinación resultante puede ser rechazada por el control de admisión de algún nodo. La solución a este problema es simple: si el control de admisión rechaza un nuevo requerimiento, se debe dejar la reservación existente.

- El segundo problema se presenta cuando un receptor efectúa persistentemente una reservación Q_1 aunque ésta no pase el control de admisión en algún nodo, ello impediría que un flujo menor $Q_0<Q_1$ reciba el servicio debido a que se está combinado con Q_1. Para resolver este problema, los mensajes *ResvErr* establecen un estado adicional conocido como estado de bloqueo (en este caso para Q_1) que cambia las reglas de combinación posibilitando la reservación de un flujo menor.

Si un requerimiento no pasa el control de admisión en algún nodo, se crea un estado de bloqueo en ese nodo pero se mantiene la reservación en los demás. Aunque podría pensarse que estas reservaciones están consumiendo recursos innecesariamente y deberían terminarse, existen motivos para mantenerlas. Un receptor persiste en una reservación fallida si busca disponibilidad de recursos para una reservación extremo a extremo o si desea obtener QoS en la mayor parte de la ruta como le sea posible.

El estado de bloqueo provee protección contra fallas transitorias, por ejemplo si se cambia a una ruta congestionada debido a una falla repentina, se debe volver lo más pronto posible a la ruta original. El estado de bloqueo no cambia la reservación en los demás nodos, actúa solamente en el nodo local. Para indicar estado de bloqueo, el nodo activa el bit "*in place*" presente en el objeto ERROR_SPEC (ver el ANEXO 5).

El estado de bloqueo está definido por dos parámetros: el *flowspec* de bloqueo (Q_b) y un tiempo de bloqueo (t_b). El *flowspec* de bloqueo es igual al *flowspec* que no pasó el control de admisión en un determinado nodo, y está contenido en el mensaje *ResvErr*. El tiempo de bloqueo corresponde a un múltiplo configurable del tiempo de refresco del estado de reservación, e indica el tiempo de expiración

del estado de bloqueo. La figura 5.9 muestra el estado de bloqueo producido en la interfaz (a) del *router* debido a la falla de la reservación en algún nodo previo; el estado se mantiene hasta que expire el tiempo t_b luego del cual se intenta nuevamente la reservación en la interfaz (a), si ésta no sucede, la recepción de un mensaje *ResvErr* retornará al nodo al estado mostrado.

Envía	Qb	Reserva	Recibe
(a) ◄—— WF(* {B})	{3B}	(c) * {3B}	(c) ◄—— WF(* {3B})
(b) ◄—— WF(* {3B})	(none)	(d) * {B}	(d) ◄—— WF(* {B})

Figura 5.9 Estado de bloqueo. [46]

5.3.3.4 Mensaje de Confirmación (*ResvConf*)

Un receptor que origina un requerimiento de QoS puede también pedir a la red una confirmación, para ello el receptor incluye en el mensaje *Resv* un objeto de requerimiento de confirmación el cual contiene la dirección IP del solicitante. En cada punto de combinación de flujos solo el *flowspec* mayor y los objetos de requerimiento de confirmación continúan su camino hacia la fuente; si el requerimiento de reservación es menor o igual que el existente en el nodo entonces se genera un mensaje *Resv*Conf que es devuelto al solicitante de la reservación.

Si se hace un nuevo requerimiento de reservación con un *flowspec* mayor que uno existente en una sesión dada, normalmente se produce un mensaje *ResvErr* o *ResvConf*, negando o aceptando la reservación respectivamente. En este caso, la generación de un mensaje *ResvConf* significa una confirmación extremo a extremo.

Recibir un mensaje de confirmación no representa que el servicio está completamente garantizado, supóngase que se hacen requerimientos desde los

receptores R1 y R2, y que el *flowspec* de R1 es mayor que el de R2, si el requerimiento de R2 arriba al nodo después que el de R1, el nodo envía de regreso un mensaje *Resv*Conf aún si el requerimiento de R1 no ha completado la reservación extremo a extremo. Si el requerimiento de R1 no completa la reservación, el *host* R2 recibirá un mensaje *ResvErr* luego del mensaje *ResvConf.*

5.3.3.5 Parámetros de Tiempo

Existen dos parámetros de tiempo principales: el tiempo de refresco (R) y el tiempo de vida del estado (L). Los valores que tomen estos parámetros están especificados en el objeto *TIME_VALUES* y varían en cada salto de la ruta. Un *router* configura un valor de refresco aleatorio comprendido entre $0.5R$ y $1.5R$, y un valor de tiempo de vida mayor a $(k+0.5)(1.5R)$ el cual asegura un tiempo de vida no tan corto, siendo k un entero configurable cuyo valor predeterminado es 3. Un valor recomendado para R es 30 segundos.

Elegir un valor para R significa un compromiso entre *overhead* y dinamismo; un valor pequeño de R implica rápida adaptación a los cambios a un costo de incrementar el *overhead* y viceversa. El protocolo RSVP permite elegir valores para R y k (y por tanto para L y T_b) en cada interfaz del elemento de red.

5.3.3.6 Control de Políticas

El Control de Políticas es el encargado de verificar si un usuario o una clase de usuarios tienen permiso administrativo para efectuar una reservación. Los "datos de políticas" se encuentran en el objeto *POLICY_DATA* del protocolo RSVP y pueden incluir credenciales de identificación de usuarios, números de cuentas, etc. Por seguridad, los datos de políticas pueden usar certificados de usuarios encriptados.

Al igual que los *flowspecs*, los datos de política son independientes del protocolo RSVP, por lo tanto la combinación de estos datos y las reglas que las rigen corresponden a los mecanismos de control de políticas.

Se requiere control de políticas en tres lugares: a) en el extremo de la red, b) en los puntos de combinación de flujos y c) en los puntos de división de tráfico; esta

información es conocida por el protocolo RSVP y éste la comunica al control de tráfico para que tome las acciones necesarias.

Acarrear los datos de políticas en los mensajes *Resv* presenta un potencial problema de escalamiento especialmente si pertenecen a un alto número de receptores de un grupo *multicast*, para evitar el exceso de estos datos se debe procurar combinarlos lo más cerca al receptor.

5.3.3.7 Seguridad

Requerimientos de reservación dañados pueden provocar que la reservación se haga a usuarios no autorizados o tal vez que se niegue el servicio causado por el bloqueo de los recursos de red. El protocolo RSVP se protege de estas reservaciones defectuosas con un mecanismo de autentificación que usa una función encriptada y que es soportado por todos los objetos que pueden aparecer en el mensaje RSVP.

5.3.3.8 Nubes No-RSVP

Para proveer el tipo de servicio requerido se necesita que todos los *routers* en el camino "entiendan" el protocolo RSVP, lo cual es imposible obtener a lo largo de todo el Internet. Sin embargo si existe una nube con suficiente capacidad entre dos *routers* RSVP todavía se puede suministrar servicio en tiempo real.

El protocolo RSVP es diseñado para operar correctamente a través de nubes que no lo soportan. Los *routers* RSVP y aquellos que no lo son direccionan los mensajes *Path* hacia el destino usando la tabla de enrutamiento, por lo tanto estos mensajes no son afectados por los *routers* que no soportan RSVP. Cuando un mensaje *Path* atraviesa una nube que no soporta RSVP, éste lleva hacia el siguiente nodo RSVP la dirección IP del nodo RSVP previo.

5.3.3.9 Compatibilidad Futura

El protocolo RSVP está diseñado para proveer compatibilidad futura, presentando elasticidad y adaptación a la introducción de nuevas clases en un objeto e incluso de nuevos objetos. Cada objeto tiene el campo *Class-Num*, cuyos dos bits más significativos determinan el trato que se le dará al objeto, de acuerdo a éstos se

puede rechazar el objeto generando un mensaje de error, enviar el objeto sin generar error ó simplemente ignorar el objeto.

5.3.4 ESPECIFICACIÓN FUNCIONAL

El protocolo RSVP, para su funcionamiento, utiliza un conjunto de siete mensajes, los cuales están formados por una cabecera común y un campo de longitud variable que contiene un conjunto preestablecido de objetos. A continuación se detallan lineamientos generales sobre el formato de estos mensajes, sin embargo el formato detallado se encuentra en el ANEXO 5.

Los mensajes definidos en el RFC 2205 para el protocolo de Reservación de Recursos RSVP son los siguientes:

1. Mensaje *Path*
2. Mensaje *Resv*
3. Mensaje *PathTear*
4. Mensaje *ResvTear*
5. Mensaje *PathErr*
6. Mensaje *ResvErr*
7. Mensaje *ResvConf*

La identificación de cada mensaje se encuentra en el campo Tipo de Mensaje (*Msg Type*) de la cabecera común. La cabecera común, como su nombre lo indica, está presente en los siete mensajes RSVP, tiene una longitud de 8 bytes y su formato se observa en la figura 5.10.

Los campos en la cabecera común son:

Versión (*Vers*) [4 bits]: Número de la versión del protocolo, su valor es 1.

Banderas (*Flags*) [4 bits]: Aún no se especifica bits de banderas.

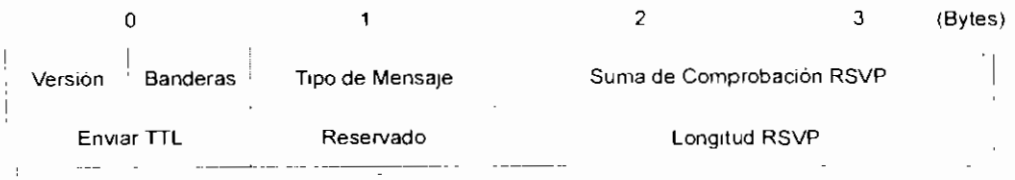

Figura 5.10 Formato de la cabecera RSVP [46]

Tipo de Mensaje (*Msg Type*) [8 bits]: Identifica al mensaje, de la siguiente manera:

Valor	Mensaje
1	Path
2	Resv
3	PathErr
4	ResvErr
5	PathTear
6	ResvTear
7	ResvConf

Tabla 5.3 Valor del campo Tipo de Mensaje para los mensajes RSVP. [46]

Suma de comprobación (RSVP *Checksum*) [16 bits]: Se calcula como el uno complemento de la suma uno complemento del mensaje, con el campo *checksum* reemplazado por ceros. Si todos los bits de este campo son ceros entonces se deduce que el *checksum* no fue transmitido.

Enviar_TTL (*Send_TTL*) [8 bits]: Es el valor IP TTL[1] (*Time To Live*) del mensaje.

Longitud del mensaje (RSVP *Length*) [16 bits]: Es la longitud total del mensaje RSVP en bytes, incluye la cabecera común y el campo de longitud variable correspondiente a los objetos.

Todos los mensajes RSVP contienen varios objetos, cada uno de los cuales (objetos) transporta información específica, útil para la realización de los procesos

[1] Time To Live (TTL).- Es un contador presente en el paquete IP que comunica a un router de red si el paquete ha permanecido demasiado tiempo en la red (sin encontrar el destino) y debe ser descartado.

RSVP. Cada objeto consiste de una o más palabras de 32 bits, una de las cuales es la cabecera. El formato se encuentra en la figura 5.11.

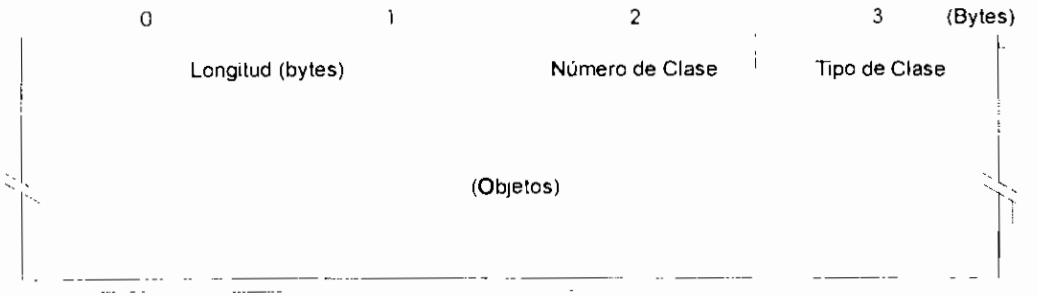

Figura 5.11 Formato de los Objetos RSVP. [46]

La cabecera del objeto tiene los siguientes campos:

Longitud (*Length*) [16 bits]: Contiene la longitud total del objeto, en bytes; ésta puede ser 4 o múltiplos de 4.

Número de Clase (*Class-Num*) [8 bits]: Contiene el número correspondiente a la clase del objeto, el cual permite su identificación.

Tipo de Clase (*C-Type*) [8 bits]: Cada objeto depende de la versión del protocolo IP que se utilice, de acuerdo a ello se define el tipo de clase del objeto, cuyo valor es 1 y 2 para IPv4 e IPv6 respectivamente.

Los objetos definidos en el RFC 2205 para el protocolo RSVP son:

- Objeto NULO (*NULL*): Este objeto puede transmitirse en cualquier momento y su contenido es ignorado por el receptor.
- Objeto SESIÓN (*SESSION*): Contiene la información necesaria para definir una sesión específica para los demás objetos que siguen, es necesario en todos los mensajes RSVP.
- Objeto SALTO (*RSVP_HOP*): Este objeto transporta la dirección IP del último nodo RSVP.

- Objeto VALORES DE TIEMPO (*TIME_VALUES*): Contiene el periodo de refresco R en milisegundos. Es usado por la fuente del mensaje y su presencia es necesaria en todos los mensajes *Resv* y *Path*.

- Objeto ERROR (*ERROR_SPEC*): Sirve para especificar el error en los mensajes *ResvErr* y *PathErr*, ó la confirmación en el mensaje *ResvConf*.

- Objeto CAMPO (*SCOPE*): Contiene el campo de la reservación, el cual es una lista de direcciones IP de los *hosts* transmisores hacia los cuales va dirigida la información. Este objeto puede aparecer en los mensajes *Resv*, *ResvErr* o *ResvTear*.

- Objeto ESTILO (*STYLE*): Este campo permite definir el estilo de la reservación. Los valores binarios 10001, 01010 y 10010 identifican los estilos de reservación WF, FF y SE respectivamente.

- Objeto ESPECTATIVA DE FLUJO (*FLOWSPEC*): Este objeto permite definir las características de la calidad de servicio deseada, se utiliza con los mensajes *Resv*. Está formado por el TSpec y RSpec especificados anteriormente en el numeral 5.2.2.3.

- Objeto EXPECTATIVA DE FILTRO (*FILTER_SPEC*): Define el conjunto de paquetes que pertenecen a la sesión que recibirá el QoS. Este objeto es transportado por los mensajes *Resv*.

- Objeto TRANSMISOR (*SENDER_TEMPLATE*): Contiene información que permite identificar a los *host*s transmisores.

- Objeto TRANSMISOR_*TSpec* (*SENDER_TSPEC*): Define las características de tráfico que generará una fuente. Se utiliza en los mensajes *Path*.

- Objeto ANUNCIO (*ADSPEC*): Transporta los datos de anuncio en los mensajes *Path*. Está formado por una cabecera general y por los fragmentos correspondientes a cada servicio.

- Objeto DATOS DE POLÍTICA (*POLICY_DATA*): Este objeto fue creado para transportar datos que permitan conocer si un usuario está administrativamente permitido de hacer un requerimiento de calidad de servicio, sin embargo aún no se especifican parámetros para este objeto.

- Objeto CONFIRMACIÓN DE RESERVACIÓN (*RESV_CONFIRM*): Transporta la dirección IP del receptor que pide confirmación de su requerimiento, por lo tanto puede aparecer en los mensajes *Resv* o *ResvConf*.

- Objeto INTEGRIDAD (INTEGRITY): Transporta datos criptográficos para auntentificar el nodo que origina el mensaje y verificar el contenido del mensaje RSVP.

Para el funcionamiento del protocolo RSVP se requieren operaciones o llamadas producidas en sus interfaces (lógicas). El protocolo RSVP tiene interfaces para enrutamiento y control de tráfico en un *router*, e, interfaces para la aplicación y control de tráfico (si existe) en un *host*. Una explicación más detallada se encuentra en el ANEXO 5.

5.4 ARQUITECTURA *IntServ* SOBRE TECNOLOGÍAS ESPECÍFICAS DE CAPA ENLACE

La arquitectura de servicios integrados introduce una extensión a la actual arquitectura IP permitiendo la entrega de calidad de servicio a las aplicaciones que lo requieran. Para la implementación completa de estos servicios se deben especificar sus interpretaciones en las diferentes tecnologías de capa enlace. La presente sección estudia a los servicios integrados y al protocolo RSVP sobre tecnologías específicas de capa 2.

5.4.1 ARQUITECTURA DE SERVICIOS INTEGRADOS SOBRE ATM

Es posible transportar tráfico IP con una determinada calidad de servicio sobre la arquitectura ATM debido a que su diseño es adecuado para ello. ATM cuenta con un conjunto de parámetros y categorías de servicio capaces de interpretar los

servicios presentes en la Arquitectura de Servicios Integrados, e incluso soporta el protocolo de señalización RSVP.

5.4.1.1 Parámetros ATM

Parámetro	Significado	Relevante a
Peso Administrativo (AW)	Impuesto por el Administrador para indicar preferencia por un determinado circuito	Todos
Velocidad Disponible de Celda (ACR)	Medida de la capacidad normal disponible	CBR, rtVBR nrtVBR
Variación del Retardo de Celda (CDV)	Medida del cambio en el retardo que ofrece un circuito	CBR, rtVBR
Tolerancia de la variación del retardo de celda (CDVT)	Indica el jitter que una fuente puede tolerar	CBR, rtVBR nrtVBR
Relación de Pérdida de Celdas (CLR)	Relación entre las celdas perdidas y las celdas transmitidas	CBR, rtVBR nrtVBR
Margen de Velocidad de Celdas (CRM)	Capacidad extra entre la velocidad disponible y la velocidad del tráfico de la fuente	rtVBR (opcional) nrtVBR (opc.)
Máximo Tamaño de Ráfagas (MBS)	Mayor número de celdas continuas que transmitirá una fuente en el circuito.	Todos
Máxima Velocidad de Celdas	Máxima velocidad de envío de las conexiones en una categoría de servicio.	ABR, UBR CBR, rtVBR (opc. nrtVBR (opc.)
Mínima Velocidad de Celda (MCR)	Mínima Velocidad de envío de celdas, la cual es todavía útil para la aplicación.	
Máximo Retardo de Transferencia de celdas (MCTD)	Medida del peor retardo extremo a extremo en una nube ATM.	CBR, rtVBR
Velocidad Pico de Celdas (PCR)	Máxima velocidad a la cual se enviarán celdas en la red	CBR
Velocidad Sustentable de Celdas (SCR)	Típica tasa de transmisión del transmisor	
Factor de Varianza	Indicador de la varianza de la velocidad de la celda en un enlace	rtVBR (opcional) nrtVBR (opc.)

Tabla 5.4 Parámetros ATM. [6]

ATM tiene un gran número de parámetros que permiten describir el tráfico generado por una fuente y los recursos de la red. Estos parámetros son usados en los procesos de admisión de ATM para establecer nuevos circuitos, en una forma análoga al control de admisión dentro de los servicios integrados. La tabla

5.4 detalla los principales parámetros de ATM que intervienen en la negociación de circuitos virtuales.

5.4.1.2 Categorías de Servicio de ATM

Las categorías de servicio especifican propiedades generales de un circuito virtual, tales como el retardo, la tasa de transmisión (variable o constante) y la aplicabilidad de parámetros de QoS.

Las especificaciones de Administración de Tráfico y de Interfaz entre Usuario y Red TM/UNI (*Traffic Management / User Network Interface*) v4.0, definen cinco categorías de servicio: CBR, rtVBR, nrtVBR, ABR y UBR. [47]

5.4.1.2.1 Velocidad Constante de Bit (CBR, Constant Bit Rate)

Este servicio es diseñado para transportar tráfico de aplicaciones que necesitan un ancho de banda constante y una respuesta de tiempo predecible; es decir se emula un enlace dedicado. Una conexión CBR puede ser descrita por el parámetro de velocidad de celda pico PCR a la cual normalmente transmitirá la aplicación. Un circuito CBR acarrea tráfico inelástico sensitivo al retardo.

5.4.1.2.2 Velocidad Variable de Bit para Tiempo Real (rtVBR, real time Variable Bit Rate)

Este servicio es adecuado para aplicaciones que generan ráfagas de datos y que necesitan sincronización de tiempo entre la fuente y el destino. Un circuito rtVBR puede ser definido por los parámetros de velocidad de celda sustentable (SCR, *Sustainable Cell Rate*), velocidad de celda pico (PCR, *Peak Cell Rate*) y máximo tamaño de ráfaga (MBS, *Maximum Burst Size*). Un circuito rtVBR transmite los datos de la fuente a una velocidad dada por SCR y permite ráfagas de datos hasta alcanzar el MBS a una velocidad PCR. Un circuito rtVBR puede transmitir tráfico elástico conversacional sensitivo al retardo.

5.4.1.2.3 Velocidad Variable de Bit inadecuado para Tiempo Real (nrtVBR, no real time Variable Bit Rate)

Este servicio es similar al anterior pero asume que no se necesita sincronización de tiempo. Es adecuado para transacciones en las cuales el servicio garantizado de los datos es preferible que el envío rápido.

5.4.1.2.4 Velocidad Disponible de Bit (ABR, Available Bit Rate)

Este servicio es adecuado para protocolos como TCP con capacidad para adaptarse a los cambios de ancho de banda de la red. Un circuito ABR intenta transportar los datos a la velocidad de celda pico proporcionando un servicio mejor que el ofrecido por el *best effort*.

5.4.1.2.5 Velocidad No Especificada de Bit (UBR, Unspecified Bit Rate)

Este servicio es realmente un servicio *best effort* en el cual no se tienen características garantizadas de la red.

5.4.1.3 Interpretación de los Servicios Integrados

Como se mencionó anteriormente, ATM cuenta con un conjunto de parámetros y categorías de servicio que le permiten interpretar la arquitectura de Servicios Integrados. Básicamente consiste en traducir los parámetros de cada servicio del nivel IP (Servicio Integrado) en su correspondiente de la Arquitectura ATM. Esta traducción se la hace en dispositivos de red denominados "dispositivos frontera" ya que se encuentran entre la red IP y la red ATM, obsérvese la figura 5.12.

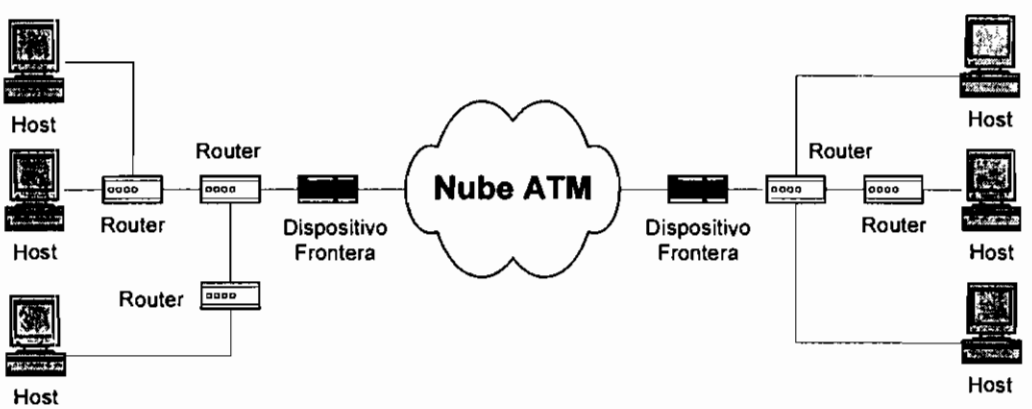

Figura 5.12 Arquitectura de red IP/ATM que permite QoS. [47]

La figura 5.12 muestra una determinada topología de red en la cual los dispositivos frontera permiten la coexistencia de las redes de acceso y la red ATM. Los dispositivos frontera pueden ser considerados como *routers* IP o como interfaces ATM ya que son capaces de crear y administrar circuitos virtuales en la interfaz ATM usuario-red (UNI).

La función principal de estos dispositivos es traducir los parámetros de calidad de servicio de nivel IP en los correspondientes parámetros de ATM, conocida como función de conectividad (IWF, *InterWorking Function*); sin embargo se llevan a cabo otras funciones de IP y ATM como se muestra en la figura 5.13.

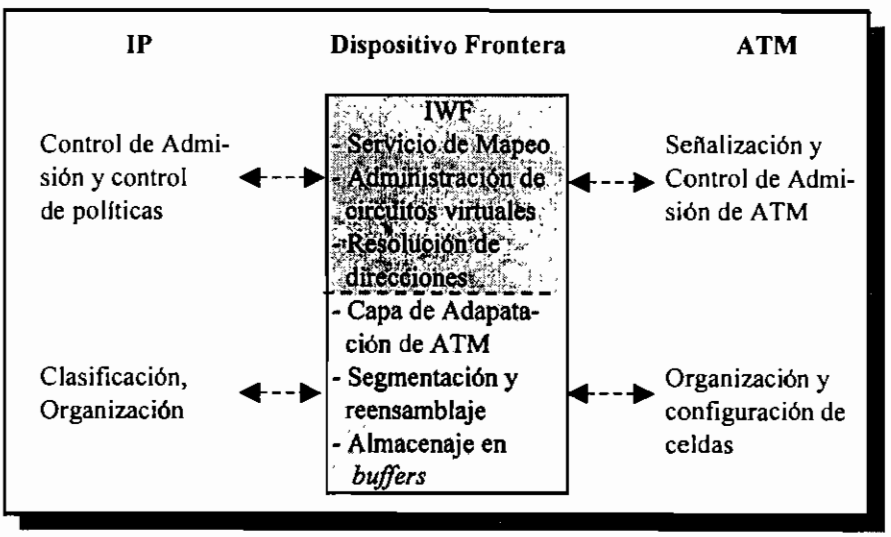

Figura 5.13 Funciones del Dispositivo Frontera. [47]

A continuación se analizan las traducciones de parámetros y categorías de servicio específicas a los servicios garantizado, de carga controlada y *best effort* de la Arquitectura de Servicios Integrados.

5.4.1.3.1 Servicio Garantizado y ATM

Las categorías de servicio de ATM que permiten la entrega de Servicio Garantizado son:

- CBR (velocidad constante de bit), y

- rtVBR (velocidad variable de bit en tiempo real).

Los servicios garantizados utilizan ciertos parámetros que caracterizan el tráfico que recibirá el servicio y los recursos de red empleados. Estos parámetros son la velocidad de transmisión pico (*p*), la velocidad de transmisión normal (*r*) y espacio en *buffers* (*b*) de la especificación de tráfico TSpec; y, la velocidad de transmisión asignada (*R*) de la especificación de recursos RSpec.

Cuando se utiliza los servicios garantizados sobre CBR sería suficiente configurar PCR = *R* si el dispositivo frontera tiene un *buffer* con capacidad para cumplir el contrato de retardo, es decir si su tamaño es al menos $b+C_{sum}+R.D_{sum}$ (ecuación 5.4).

Cuando se utiliza los servicios garantizados sobre rtVBR se debe hacer PCR = *p*, SCR = *R* y MBS = *b*. Esta equivalencia es la más lógica y adecuada pero podrían presentarse variaciones, específicamente, se puede hacer $R \leq PCR < p$ en un intento por aminorar los costos del circuito, si éste depende en gran medida del parámetro PCR.

ATM da la posibilidad de no usar *buffers* para absorber ráfagas de datos, si no existen o son insuficientes, debiéndose elegir un valor mayor para PCR, siendo éste el mayor valor entre R y p; sin embargo siempre debe existir un *buffer* que absorba el *jitter* ($C_{sum}+R.D_{sum}$) suministrado en el elemento frontera o en la red ATM mediante el parámetro MBS (máximo tamaño de ráfagas). Esta característica es válida para las dos categorías de servicio descritas.

5.4.1.3.2 Servicio de Carga Controlada y ATM

Existen tres categorías de servicio adecuadas para la entrega de servicio de carga controlada:

- CBR (velocidad constante de bit),
- nrtVBR (velocidad variable de bit inadecuado para tiempo real), y
- ABR (velocidad disponible de bit)

Los servicios de carga controlada utilizan ciertos parámetros que caracterizan el tráfico que recibirá el servicio. Estos parámetros son la velocidad de transmisión pico (*p*), la velocidad de transmisión normal (*r*) y el espacio en *buffers* (*b*) de la especificación de tráfico TSpec.

Cuando se utiliza el servicio de carga controlada sobre nrtVBR se debe hacer PCR = *p*, SCR = *r* y MBS = *b*. En este caso, al igual que en el servicio de carga controlada sobre rtVBR, también se puede manipular el tamaño de los *buffers* para obtener valores adecuados para PCR y/o MBS. Además en este caso también es necesaria la presencia de *buffers* para absorber el *jitter*, pero el tamaño del mismo depende de la implementación ya que no existen los parámetros *C* y *D* para el servicio de carga controlada.

Cuando se utiliza el servicio de carga controlada sobre ABR se debe configurar la mínima velocidad de celda MCR = *r*. El servicio ABR asegura el transporte de celdas al menos a una velocidad especificada por MCR, se puede superar esta velocidad (si *r* aumenta) pero ello implicará pérdida de celdas. Dentro de esta categoría de servicio no existe señalización para *b*, por lo tanto el dispositivo frontera debe contar con la capacidad suficiente en *buffers* que le permita absorber ráfagas de datos y el *jitter* de la red.

Cuando se utiliza CBR se configura PCR = *r*. Similar al caso anterior, el elemento frontera debe proveer capacidad de almacenaje en *buffers* para absorber ráfagas de datos y *jitter*.

5.4.1.3.3 Servicio Best effort y ATM

Debido a que el servicio *best effort* es el servicio predeterminado de Internet y no garantiza calidad de servicio puede utilizar cualquier categoría de servicio de ATM, sin embargo la categoría de servicio natural es UBR (velocidad no especificada de Bit).

UBR no tiene descriptores de tráfico, en este caso se suele asignar recursos a un servicio global (no por circuito virtual) usando las características de admisión de los *switches* ATM propias de la implementación.

Cuando se utiliza CBR se asigna un ancho de banda para un conjunto de flujos que se desea agregar. CBR y nrtVBR pueden ser útiles si se desea una asignación explícita de ancho de banda que tiene como finalidad implementar costos diferenciales.

La clase de servicio ABR funciona bien con tráfico *best effort* ya que su diseño fue concebido para transportar el protocolo TCP/IP, el cual se adapta a las condiciones de la red. Sin embargo, utilizar rtVBR con el servicio *best effort* implicaría una complejidad innecesaria y por lo tanto no es recomendado su uso [47].

Una característica importante que presenta ATM, es la capacidad de marcar las celdas que no cumplen los descriptores de tráfico especificados, para ello se utiliza un bit de la cabecera ATM denominado bit de prioridad de pérdida de celdas CPL (*Cell Priority Loss*). Para etiquetar las celdas con prioridad de marcado, ATM utiliza mecanismos similares al control de admisión de los Servicios Integrados.

El trato que se dará a las celdas marcadas con prioridad de descarte depende de la implementación, ya que ATM no lo define explícitamente; sin embargo es deseable que al tráfico en exceso perteneciente a los servicios garantizado y de carga controlada se les otorgue un tratamiento *best effort*, mientras que el exceso de tráfico *best effort* sea descartado.

Es preferible que la tarea de marcación de celdas con prioridad de descarte se la realice en el dispositivo frontera, pues ello aumentaría la eficiencia. El dispositivo frontera conoce el paquete IP y si éste no cumpliera las condiciones se procedería a marcar con prioridad de descarte todas las celdas pertenecientes al paquete; si se dejara esta tarea solamente a ATM podría descartarse una celda del paquete y transmitirse el resto, lo cual sería un gasto innecesario de recursos pues cuando lleguen las demás celdas al receptor serían descartadas.

5.4.2 SERVICIOS INTEGRADOS SOBRE ENLACES DE BAJA TASA DE TRANSMISIÓN

En general, los servicios integrados no pueden utilizarse eficientemente en un enlace de baja tasa de transmisión. Cuando se refiere a enlaces de baja tasa de transmisión se consideran principalmente los enlaces de módem (14.4, 28.8, 33.6 y 56 kbps) y los enlaces ISDN de 56 y 64 kbps [48].

El problema radica en dos aspectos:

- El elevado retardo que experimenta un paquete pequeño de una aplicación en tiempo real mientras espera la culminación de la transmisión de un paquete grande. Por ejemplo la transmisión de un paquete de 1500 bytes en un enlace de 28.8 kbps se demora 400 ms [48], lo cual haría imposible la comunicación en una aplicación interactiva.

- El elevado *overhead* introducido en un paquete debido a las cabeceras de cada protocolo lo cual hace ineficiente la transmisión. Por ejemplo el overhead del *stack* de protocolos HDLC/PPP–IP–UDP-RTP es 44 bytes (4-20-8-12 bytes, respectivamente) mientras la carga útil puede ser 19.75 bytes si se utiliza G.723.1[1]. [48]

En un enlace de esta naturaleza lo primordial es obtener bajo retardo, aunque también es deseable la compresión de la cabecera; una buena implementación debería contener estos dos aspectos. A continuación se presentan las características y las soluciones a estos dos problemas descritos.

5.4.2.1 Encapsulamiento en tiempo real

Para evitar el retardo de un paquete pequeño de una aplicación de tiempo real, mientras espera la transmisión de otro paquete grande, se debe interrumpir permanente o temporalmente la transmisión de dicho paquete y dar paso a la transmisión de los paquetes de mayor prioridad.

[1] G.723.1.- Norma de la UIT que define un método de transmisión de voz sobre IP.

Para lograr esta tarea de encapsulamiento del tráfico en tiempo real y permitir una rápida conmutación se debe primero identificar el tráfico en tiempo real, para lo cual se pueden utilizar técnicas heurísticas[1] o no relacionales que distinguen este tráfico debido al pequeño tamaño de los paquetes y a su característica periódica. Otra forma de distinguir este tráfico es mediante el uso de la precedencia en el protocolo IP (capítulo 4, sección 4.4); sin embargo, se necesitan parámetros extras si se utiliza el protocolo RSVP (sección 5.3).

5.4.2.1.1 Fragmentación IP

En lugar de pausar la transmisión de un paquete grande simplemente se podría aprovechar la capacidad de fragmentación del actual protocolo IP (IPv4) y trabajar con paquetes pequeños.

Una implementación del protocolo PPP en tiempo real sería elegir el MTU adecuado a las características de retardo que se desea obtener. Esta solución tiene el inconveniente del elevado *overhead* introducido, pues se necesita una cabecera IP de mínimo 20 bytes, y por ello se recomienda su uso solo si no existen otros protocolos de fragmentación. [48]

5.4.2.1.2 Mecanismos de Capa de Enlace

El protocolo HDLC (*High-level Data Link Control*) y una variación de él, utilizados en líneas sincrónicas y asincrónicas respectivamente, utilizan un mecanismo para la delimitación de tramas el cual debería adecuárselo para proveer fragmentación mediante la utilización de un indicativo que relacione las partes de una trama.

Existen actualmente dos especificaciones de encapsulación de tráfico en tiempo real: la extensión multiclase del protocolo PPP multienlace y el protocolo PPP orientado a tramas tipo HDLC.

---

[1] Técnica Heurística.- Es una técnica que aplica para la resolución de los problemas "reglas de buena lógica" que presentan visos de ser correctas aunque no se garantiza su éxito, modelizando el problema de una forma adecuada.

- *Protocolo PPP multienlace*

El protocolo PPP multienlace soluciona el problema de retardo presente en la transmisión de un paquete grande mediante la fragmentación de éste. Utiliza el formato de la figura 5.14 para transmitir los paquetes de menor prioridad, los cuales son fragmentados en el transmisor y reconstruidos en el receptor con la ayuda de una secuencia gradual de números. Los paquetes de mayor prioridad se transmiten fuera de este formato.

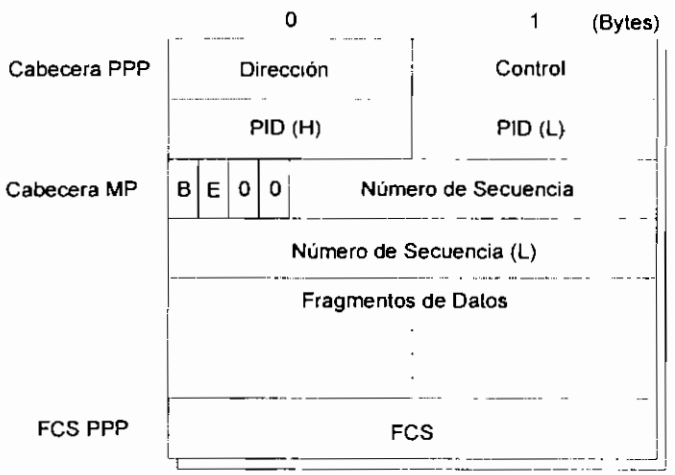

Figura 5.14 Formato de un fragmento utilizando PPP multienlace. [49]

La limitación de esta técnica es que únicamente permite un nivel de suspensión, pausando la transmisión de un paquete grande para dar lugar a la transmisión de un paquete pequeño no fragmentado; además, la cabecera que se añade es relativamente grande.

Para superar este problema se proponen dos extensiones al protocolo PPP multienlace: extensión corta y extensión larga; las cuales posibilitan 4 y 16 niveles de suspensión mediante el uso de 12 y 24 bits de cabecera respectivamente. El formato de las dos extensiones se presenta en las figuras 5.15 y 5.16.

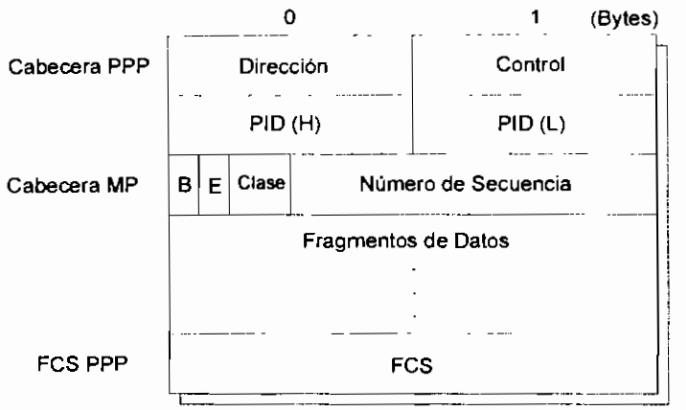

Figura 5.15 Extensión corta al protocolo PPP multienlace. [49]

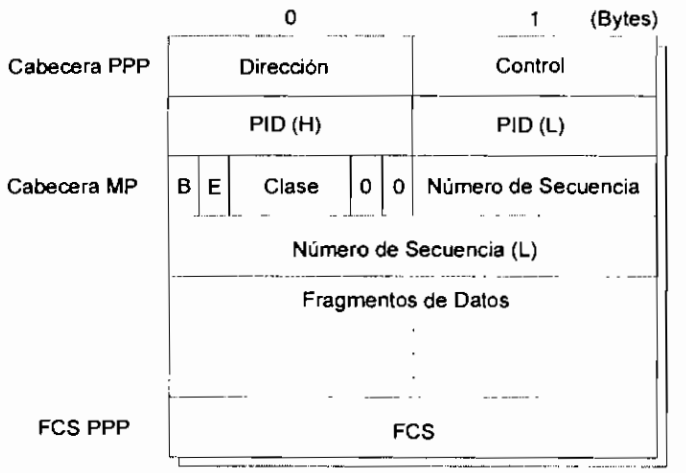

Figura 5.16 Extensión larga al protocolo PPP multienlace. [49]

Protocolo PPP orientado a tramas tipo HDLC

Esta extensión del protocolo PPP permite suspender la transmisión de un paquete grande mientras se transmite un paquete de mayor prioridad. Utilizar el formato del protocolo PPP multienlace para señalizar los fragmentos de tramas significa la inserción de una cabecera del formato por cada pausa en la transmisión. Para evitar el *overhead* así producido se define un nuevo formato compacto de fragmentos y cabecera, los cuales se estudian a continuación.

El formato compacto de un fragmento se ilustra en la figura 5.17. El bit *R* permite identificar fragmentos pertenecientes a una misma trama; si *R* = 1 significa que este fragmento reinicia la transmisión de un paquete que ya tiene otros fragmentos enviados.

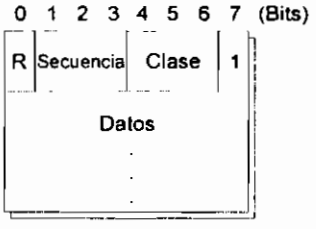

Figura 5.17 Formato compacto de un fragmento del protocolo PPP orientado a tramas tipo HDLC. [50]

El número de secuencia permite identificar los fragmentos sucesivos de un paquete perteneciente a una clase especificada en el campo correspondiente. Se puede pensar que se permite ocho clases de tráfico (del 0 al 7), sin embargo es prohibitivo tener todos unos (R=1, *sequence*=7, *class*=7) en la cabecera del fragmento por su coincidencia con la cabecera estándar del protocolo PPP en HDLC. Para evitar esta ambigüedad se hace que la clase 7 no permita pausa en la transmisión, tomando R siempre el valor de cero; como el campo de secuencia no tiene sentido en este caso, se lo utiliza para definir 8 clases adicionales (de la 8 a la 15) las cuales tampoco permiten la pausa en la transmisión de paquetes.

Los tres bits del número de secuencia son insuficientes cuando se tiene múltiples enlaces, y para este caso se define el formato de fragmento compacto extendido, mostrado en la figura 5.18.

En este caso se definen 7 clases (0 a 6) con capacidad de hacer pausa en la transmisión. Debido a que se tiene la misma restricción para la clase 7 que en el caso anterior, se utilizan los números de secuencia para definir 1024 clases (7 a 1030) sin capacidad de pausar la transmisión.

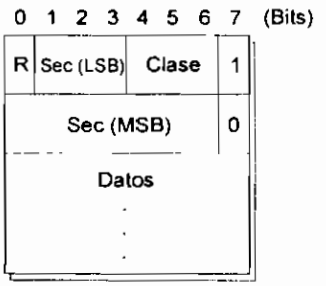

Figura 5.18 Formato Compacto Extendido de un fragmento del protocolo PPP orientado a tramas tipo HDLC. [50]

Para utilizar estos dos formatos en tramas tipo HDLC se necesita un byte adicional al final de cada fragmento. Este byte se denomina FSE (F*ragment Suspend Escape*) y sirve para delimitar los fragmentos, además contiene un bit *E* el cual toma el valor de 1 para el último fragmento de la trama y 0 para los demás fragmentos.

El byte FSE es transparente para el protocolo HDLC y su procesamiento se lo debe hacer en capas superiores. El valor hexadecimal escogido para este byte es DE debido a su poca probabilidad de ocurrencia en los datos válidos; sin embargo pueden existir datos válidos que contengan el valor DE, necesitándose por lo tanto algún mecanismo de transparencia.

El mecanismo de transparencia consiste en agregar un byte 1xxx1111 luego de la ocurrencia de un byte FSE entre los datos; se elige este byte debido a que su valor es prohibitivo en la cabecera PPP multienlace (indicado previamente) que sucedería a un byte FSE en condiciones normales. El valor que tomen los tres bits representados por xxx permite diferentes opciones de ocurrencias del byte FSE.

5.4.2.2 Compresión de Cabeceras

La compresión de las cabeceras introducidas en el *stack* de protocolos que se utilice permite incrementar la eficiencia de la transmisión. Esta compresión se la puede realizar a todo nivel y en diferentes elementos de red; por ejemplo algunas aplicaciones de telefonía emplean técnicas de compresión que permiten eliminar 11 de los 12 bytes de cabecera RTP, al nivel de capa red el protocolo IP puede reducir la cabecera IPv4/UDP de 28 a 6 bytes. [48]

Capítulo 6

ARQUITECTURAS HÍBRIDAS

CAPÍTULO 6

ARQUITECTURAS HÍBRIDAS

Los capítulos 5 y 6 estudian la entrega de calidad de servicio a través de las Arquitecturas de Servicios Diferenciados e Integrados respectivamente. Este estudio se lo realiza en forma independiente para cada arquitectura; sin embargo, se puede aprovechar los beneficios de cada una de ellas con el objetivo de formar una arquitectura robusta y escalable que permita la entrega de calidad de servicio extremo a extremo.

El presente capítulo estudia la arquitectura híbrida *IntServ/DiffServ* que trata la operación de los Servicios Integrados sobre una red que soporta Servicios Diferenciados. Además se estudiará una forma alternativa de proveer calidad de servicio a través del MultiProtocolo de Conmutación de Etiquetas (MPLS, *MultiProtocol Label Switching*) y su interacción con los Servicios Integrados y Diferenciados.

6.1 SERVICIOS INTEGRADOS SOBRE UNA RED DiffServ

Esta arquitectura utiliza la capacidad de reservación de recursos de la Arquitectura de Servicios Integrados (*IntServ*) y la escalabilidad de la Arquitectura de Servicios Diferenciados (*DiffServ*) para ofrecer un servicio extremo a extremo robusto, escalable y garantizado.

La poca capacidad de escalabilidad en redes grandes debido a la gran cantidad de *overhead* que se necesitaría para la señalización de cada flujo, además que únicamente pocos *host*s pueden generar la señalización RSVP y que mecanismos de control de políticas recién están disponibles, han hecho difícil la implementación extremo a extremo de la Arquitectura de Servicios Integrados.

Surge entonces la necesidad de utilizar una arquitectura híbrida, en la cual la incorporación de la Arquitectura de Servicios Diferenciados soluciona el problema

de escalabilidad en redes grandes mediante la separación del tráfico en pocas clases, para lo cual utiliza el DSCP (*DiffServ Code Point*) del paquete y sobre la base de éste se da un tratamiento diferenciado especificado por el PHB (*Per Hop Behavior*).

Esta arquitectura híbrida utiliza la Arquitectura de Servicios Integrados ideada para la provisión de calidad de servicio extremo a extremo y considera a las regiones *DiffServ* como un elemento del camino extremo a extremo, es decir se puede considerar a la red *DiffServ* como enlaces virtuales dentro de la red *IntServ*. Para garantizar los recursos de la red *DiffServ* se debe emplear control de admisión en la periferia de la misma.

6.1.1 BENEFICIOS

Las principales ventajas que ofrece la presente arquitectura son:

- Escalabilidad de la región *DiffServ*, pues realiza el control de tráfico para un conjunto de flujos denominados agregados, en lugar del control de tráfico para flujos independientes como en *IntServ*.

- Control de Admisión basado en recursos, el cual permite asegurar los recursos suficientes de la región *DiffServ* para los requerimientos de los flujos *IntServ*; ello se obtiene introduciendo en la región *DiffServ* un agente con capacidad de interpretar los requerimientos de calidad de servicio de cada flujo *IntServ*. Si no se realiza este control de admisión se puede degradar el servicio que la red *DiffServ* ofrece a la red *IntServ*. Para entender este enunciado considérese el siguiente ejemplo: si en una red *DiffServ* se asigna 50 kbps para un DSCP determinado, todos los flujos marcados con este DSCP compartirán el enlace, así si se tiene 10 conversaciones telefónicas de 10 kbps cada una, todas comparten los 50 kbps produciendo la degradación de todas las conversaciones; bajo las mismas condiciones, en una red *IntServ* (y en la red híbrida propuesta) solo 5 conversaciones utilizarán los 50 kbps asignados a un determinado servicio, mientras el resto de conversaciones serán negadas o tal vez se procesarán con un nivel de servicio menor [51].

- Control de Admisión basado en políticas, el cual lo efectúa el mismo agente anterior. Este control de admisión se realiza basándose en usuarios y/o aplicaciones en forma similar al efectuado en una red *IntServ*, sin embargo su implementación en una red *DiffServ* pura sería imposible debido a que en ésta no se puede diferenciar flujos individuales.

- Asistencia para la clasificación e identificación del tráfico ofrecida por la señalización existente en la región *IntServ* (protocolo RSVP), mediante la cual se marcan los flujos individuales de tráfico con el adecuado DSCP para su interpretación y tratamiento en la región *DiffServ*. La marcación del tráfico puede realizarse en los *hosts* extremos o en los *routers* intermedios; la marcación en los *hosts* tiene la ventaja de que éste tiene el conocimiento del tipo de tráfico que genera y en función de ello le asigna el DSCP más adecuado, mas aún, si el tráfico generado pertenece al protocolo IPSec[1], la marcación solo puede ser realizada aquí, pues la información para identificar al flujo se encuentra encriptada en el resto de la ruta; la marcación en los *routers* puede efectuarse en forma dinámica con la ayuda de algún protocolo o en forma estática (manual), esta última presenta poca versatilidad a los cambios en la red y solo funciona con patrones de tráfico previamente definidos.

6.1.2 MODELO DE LA ARQUITECTURA *IntServ/DiffServ*

La Arquitectura *IntServ/DiffServ* considera a la Arquitectura de Servicio Integrados como el medio para proporcionar calidad de servicio extremo a extremo, dentro de la cual existen regiones pertenecientes a la Arquitectura de Servicios Diferenciados que tienen nodos capaces de interpretar el protocolo de señalización RSVP.

De esta forma, las redes de acceso, en las cuales la disponibilidad de recursos es escasa, utilizarán la Arquitectura de Servicios Integrados; y, los núcleos de las redes o *backbones*, en donde los datos de control (*overhead*) por cada flujo

[1] IPSec (IP Security): Protocolo de seguridad utilizado en las redes privadas virtuales que transportar los datagramas IP encriptados.

presentan problemas de escalabilidad, utilizarán la Arquitectura de Servicios Diferenciados. El modelo referido se grafica en la figura 6.1.

El transmisor (Tx) y el receptor (Rx) utilizan el protocolo de señalización RSVP para comunicar los requerimientos de calidad de servicio a la red. Además, utilizan control de tráfico local (dentro del *host*) o externo (en el primer *router* capaz de realizarlo) para marcar el tráfico con el DSCP apropiado.

La implementación del modelo puede realizarse de dos maneras dependiendo de la región *DiffServ:* si la región no contiene nodos capaces de interpretar la señalización *IntServ* se efectuará un aprovisionamiento de recursos estático, si la región contiene nodos con capacidad de interpretar la señalización *IntServ*, el aprovisionamiento de recursos será dinámico.

Cuando se efectúa aprovisionamiento estático, los *routers* frontera (ER, *Edge Routers*) actúan como agente de control de admisión para la región *DiffServ* y los *routers* del borde (BR, *Border Routers*) son *routers DiffServ* puros. Cuando se realiza aprovisionamiento dinámico, el agente de control de admisión de la región *DiffServ* se encuentra en los *routers* del borde BR, mientras los *routers* frontera ER son *routers IntServ* puros y realizan solamente control de admisión local.

6.1.2.1 Aprovisionamiento estático

El aprovisionamiento de recursos de la región *DiffServ* se realiza en forma estática o manual cuando la mencionada región no tiene nodos capaces de interpretar la señalización RSVP. Para ello se negocia la especificación de nivel de servicio SLS (*Servive Level Specification*) la cual detalla parámetros como ancho de banda, velocidad pico, hora del día, entre otros.

En este modelo el *router* frontera ER tiene dos funciones que le permiten interactuar con cada región: se comporta como un *router IntServ* (RSVP) con capacidad de señalización por flujos y además es el interfaz con la región *DiffServ*, valiéndose de varias interfaces virtuales para cada nivel de servicio SLS. Este *router* contiene una tabla que indica la capacidad asignada a cada SLS.

A continuación se indica la secuencia de eventos que ocurren para proveer calidad de servicio extremo a extremo mediante este modelo:

1. El sistema operativo del *host* transmisor (Tx) genera un mensaje *Path* que se propaga y procesa en todos los *routers* RSVP, incluidos ER1 y ER2. Este mensaje se procesa de igual forma que en una red *IntServ* estableciendo el correspondiente estado de reservación en cada *router IntServ*. En la región *DiffServ* el mensaje *Path* atraviesa en forma transparente.

2. El sistema operativo del receptor (Rx) genera en respuesta un mensaje *Resv*, el cual recibe idéntico tratamiento que en la Arquitectura de Servicios Integrados en cada nodo *IntServ*, incluidos ER1 y ER2. Igual que el mensaje *Path*, el mensaje *Resv* atraviesa la región *DiffServ* en forma transparente. Cuando el mensaje *Resv* arriba al *router* frontera ER1, dispara el control de admisión (y el control de políticas, si existe) chequeando la disponibilidad de recursos de la red *DiffServ* asignados al DSCP especificado para el flujo; si el flujo es admitido se debe actualizar la disponibilidad de recursos.

3. Cuando el mensaje *Resv* arriba al transmisor, éste entiende que su requerimiento fue aceptado y empieza a enviar paquetes de datos, y tal vez procede a marcarlos con el DSCP contenido en el mensaje *Resv*. La correspondencia entre un determinado servicio integrado y el DSCP de la red *DiffServ* será explicada más adelante (sección 6.1.2.3).

6.1.2.2 Aprovisionamiento Dinámico

Esta forma de aprovisionamiento exige la existencia de *routers* capaces de interpretar la señalización RSVP dentro de la red *DiffServ*. Los *routers* de borde BR pertenecen a esta clase, sin embargo pueden existir otros *routers* dentro de la región *DiffServ* que interpreten la señalización RSVP. Los *routers* frontera ER no desempeñan otro papel que no sea el de un *router* RSVP/*IntServ* puro, y su acción se limita a la región *IntServ*.

Debe entenderse que los *routers* BR no pueden diferenciar el tráfico por flujos individuales como lo hace un *router IntServ* puro, pues a pesar que su plano de control "entiende" RSVP, y por tanto obtiene las ventajas de la señalización, en cambio, su plano de datos trabaja con agregados o conjuntos de flujos.

A diferencia del modelo anterior en el cual existe solo una interfaz entre las regiones, el *router* ER1, éste modelo tiene varios puntos de comunicación, uno en cada *router* capaz de interpretar RSVP dentro de la región *DiffServ*. Aprovechando esta capacidad de señalización dentro de la región *DiffServ* se puede aumentar la eficiencia del uso de los recursos disponibles en la misma ya que se conoce la capacidad de algún camino en particular.

Una ventaja importante que tiene este modelo es la capacidad para realizar un ajuste dinámico de los recursos de la red *DiffServ* cuando varíen las reservaciones en la red *IntServ*. Este ajuste puede realizarse a partir de una configuración inicial y no necesariamente de una manera instantánea, además su consecución es más fácil si la región *DiffServ* pertenece a un mismo dominio administrativo.

Se debe tener claro el compromiso que implica la utilización de un determinado número de *routers* "RSVP" en la región *DiffServ*, para esto se considera los dos casos extremos:

- Si solo se utilizan los *routers* de borde BR se obtiene una red ineficiente en la administración de recursos, la cual para su funcionamiento debe estar sobredimensionada, o en su defecto sujetarse a patrones preestablecidos de datos.

- Si se emplean muchos *routers* "RSVP" en la región *DiffServ* se puede incrementar la eficiencia en el uso de recursos, pero posiblemente la gran cantidad de datos de control que se necesita impediría la obtención de una red escalable.

Este modelo utiliza la señalización RSVP porque es la señalización predeterminada en la Arquitectura de Servicios Integrados, sin embargo se puede

utilizar otros protocolos siempre y cuando tengan el conocimiento suficiente de la disponibilidad de recursos y de la topología de la red que le permitan efectuar un correcto control de admisión.

PROTOCOLO RSVP SOBRE LOS SERVICIOS DIFERENCIADOS

La arquitectura híbrida expuesta basa su funcionamiento en la escalabilidad sobre redes grandes ofrecida por los Servicios Diferenciados. Esta escalabilidad se da porque la señalización del protocolo RSVP se utiliza con un conjunto de flujos de tráfico, y por lo tanto no se añade datos de control representativos en la ruta establecida.

El protocolo RSVP se utiliza solo para el establecimiento de las sesiones *DiffServ* y para la realización del control de admisión, y su procesamiento es mucho más simple comparado con el procesamiento estándar RSVP ya que las sesiones y los estados son mucho más simples.

A continuación se presentan los mecanismos que usa el protocolo RSVP para establecer una sesión *DiffServ* (obsérvese la figura 6.1):

El *router* de borde BR1 inicia la sesión *DiffServ S(a,i)* determinada por el identificador de sesión *"i"* asignado en el *router* BR1, y su dirección IP *"a"*. La información de la sesión es transportada inicialmente hacia el receptor en el mensaje *Path*.

Cada nodo a lo largo de la ruta crea una sesión *DiffServ* luego de la recepción del mensaje *Path*, y lo reenvía hacia el destino.

Cuando el mensaje *Path* ha alcanzado el nodo receptor, éste envía en respuesta un mensaje *Resv* que contiene el objeto *DS_FLOWSPEC* con la especificación del flujo; la cual incluye la identificación del tratamiento por salto (*PHB_ID*) y los parámetros que lo definen (*PHB_PARAM*). El mensaje *Resv* puede contener opcionalmente el objeto *RESV_CONFIRM* si se desea confirmación de la reservación.

ARQUITECTURA *IntServ-DiffServ*

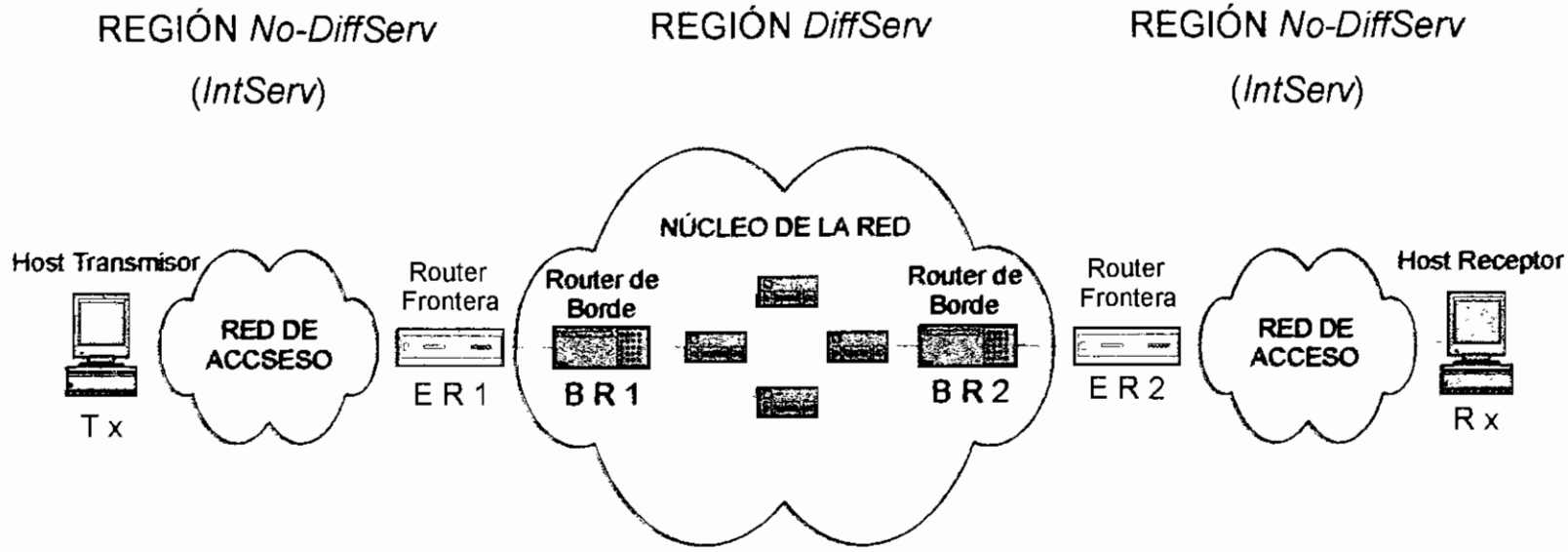

Figura 6.1 Arquitectura Híbrida *IntServ DiffServ*. [51]

A diferencia del protocolo RSVP estándar, los nodos intermedios *DiffServ* puros no realizan procesamiento de los mensajes *Resv* y se limitan a reenviarlos hacia el nodo BR1.

Cuando el mensaje *Resv* ha llegado al *router* BR1, éste activa el control de admisión y si existe disponibilidad de recursos de la red *DiffServ*, empieza a enviar los paquetes de datos con el tratamiento por salto (PHB) especificado para el correspondiente DSCP si los paquetes cumplen la especificación de tráfico. Si no existe disponibilidad de recursos, BR1 envía un mensaje *ResvErr* terminando el intento de reservación.

Al igual que en el protocolo RSVP estándar, se puede terminar una sesión *DiffServ* con el envío de mensajes *PathTear* o cuando haya transcurrido un tiempo predeterminado sin que se haya recibido mensajes de refresco.

6.1.2.3 Correlación entre las regiones *IntServ* y *DiffServ*

De igual forma como se interpretan los Servicios Integrados en tecnologías de capa enlace, como se estudió en el capítulo 5, debe existir una adecuada interpretación de los servicios de *IntServ* en los correspondientes servicios de *DiffServ*. Este modelo no define una relación directa entre las dos arquitecturas, sin embargo delinea ciertas condiciones a cumplirse, las cuales son:

- Seleccionar los tratamientos por salto (PHBs) adecuados para el tráfico generado.
- Exportar los parámetros *IntServ* presentes en la región *DiffServ*, por ejemplo la actualización de los objetos *ADSPEC*.
- Realizar control de admisión para los requerimientos *IntServ* tomando en cuenta la disponibilidad de recursos de la región *DiffServ*.

6.1.2.4 Objeto RSVP DCLASS (*Differentiated CLASS*)

Cuando un mensaje *Path* atraviesa un nodo de la región *DiffServ* (usualmente ER1 o BR1) que cuenta con un agente de admisión, se instala en éste el estado apropiado y se reenvía el mensaje hacia el destino. Cuando en el mismo nodo se recibe el mensaje *Resv*, el elemento toma la decisión de admitir o rechazar el requerimiento dependiendo de la disponibilidad de recursos en la región *DiffServ*.

Si el flujo es aceptado se determina el tratamiento por salto PHB adecuado y se procede a marcar el DSCP correspondiente; la marcación se hace dentro del mismo nodo, en un nodo previo o en el *host* transmisor, y se la hace previo acuerdo o por medio de un protocolo de negociación.

Si la marcación se hace en el *host* o en un nodo previo al nodo que contiene el agente de control de admisión, éste debe comunicar el DSCP al nodo en el cual se realizará la marcación; esta información se transporta en el mensaje *Resv* en el objeto DCLASS.

El formato del objeto DCLASS se ilustra en la figura 6.2, este objeto está formado por una cabecera y una o más palabras de 32 bits. El formato de la cabecera es el mismo que el de cualquier objeto, en ésta se indica la longitud del objeto (en bytes) y además el número de clase del objeto, 255. El cuerpo del mensaje puede contener una o más palabras de 32 bits, en las cuales únicamente se utilizan 6 bits para el DSCP. Como se observa se pueden utilizar varios DSCP para permitir la diferenciación de subflujos.

El objeto DCLASS es bastante crítico pues de su contenido depende el trato que se le dé al tráfico; un ataque a la red que modifique este objeto, cambiando el DSCP por uno de menor calidad, podría causar la interrupción de una aplicación. El mencionado ataque se produciría en algún punto entre el nodo que realiza la marcación del tráfico y el nodo que tiene el agente de control de admisión, y como se dijo, sería completamente nocivo si se intenta bajar el nivel de calidad de servicio, sin embargo, si se intenta subir la calidad del servicio, su efecto sería contrarrestado por el mismo control de admisión de la red.

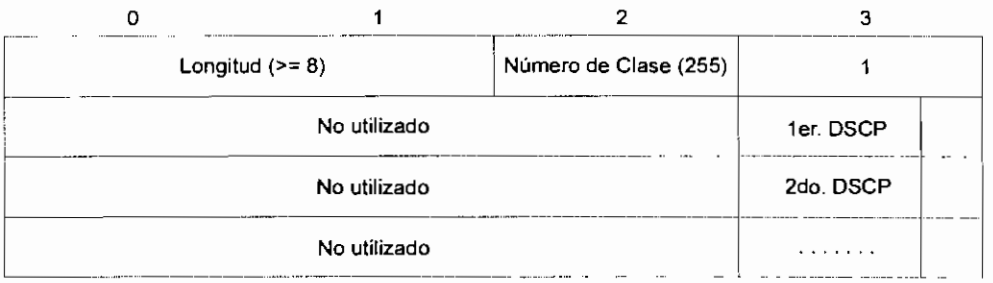

Figura 6.2 Objeto DCLASS utilizado por RSVP. [53]

6.1.3 CONSIDERACIONES ADICIONALES

Para un correcto funcionamiento del modelo se debe proteger al tráfico *IntServ* (altos requerimientos de calidad) del trafico *No-IntServ* (medianos requerimientos de calidad) y *best effort* (mínimos requerimientos de calidad). Para ello se debe aislar las tres clases de tráfico mediante una adecuada configuración de los mecanismos de clasificación y políticas, al ingreso de la red, y de los mecanismos de aprovisionamiento, dentro de la red. Una forma de implementación es haciendo que no coincidan los DSCP de cada tipo de tráfico.

El funcionamiento del modelo es limitado en entornos *multicast*, pues no existe forma de diferenciar, dentro de la región *DiffServ*, requerimientos heterogéneos procedentes de varios destinos hacia una misma fuente. El establecimiento de sesiones *multicast* se logra con la introducción de nodos con señalización RSVP dentro de la región *DiffServ* y además se debe hacer que éstos sean los puntos de división del árbol *multicast*.

Para garantizar la seguridad de los mensajes RSVP se puede implementar la misma técnica de encriptación utilizada en el protocolo RSVP estándar. Por otro lado, la marcación DSCP de los paquetes efectuada en la fuente haría suponer que la fuente intenta acaparar los recursos del enlace, pero esto no sucede si existe el correcto control de admisión en la entrada a la región *DiffServ*.

6.2 MPLS y *DiffServ*

La entrega de paquetes bajo el modelo *DiffServ*, estudiado en el Capitulo 4 define una variedad de mecanismos para poder clasificar el tráfico en un reducido número de clases de servicio, con diferentes prioridades. Según los requisitos de los usuarios, *DiffServ* permite diferenciar servicios tradicionales tales como el WWW, el correo electrónico o la transferencia de ficheros (para los que el retardo no es crítico), de otras aplicaciones mucho más dependientes del retardo y de la variación del mismo, como son las de vídeo y voz interactiva. Para ello se emplea el campo ToS (*Type of Service*), redefinido en *DiffServ* como el octeto DS.

MPLS (*MultiProtocol Label Switching*) se adapta perfectamente al modelo *DiffServ*, ya que las etiquetas MPLS tienen el campo EXP para poder propagar la clase de servicio CoS en el correspondiente LSP (*Label Switch Path*). De este modo, una red MPLS puede transportar distintas clases de tráfico. Para una estudio mas profundo sobre la historia, ventajas y descripción funcional de MPLS refiérase al Anexo 6.

De acuerdo con la información contenida en los bits del campo EXP, el tráfico que fluye a través de un determinado LSP se puede asignar a diferentes colas de salida en los diferentes saltos LSR (*Label Switch Router*). Entre cada par de LSR exteriores se pueden provisionar múltiples LSPs, cada uno de ellos con distintas prestaciones y con diferentes garantías de ancho de banda. Por ejemplo, un LSP puede ser para tráfico de máxima prioridad, otro para una prioridad media y un tercero para tráfico *best effort*, tres niveles de servicio que lógicamente tendrán distintos precios.

Como puede notarse, debido a que *DiffServ* y MPLS trabajan en forma similar con respecto a la Calidad de Servicio que cada uno de ellos habilita (mediante la clasificación de paquetes), el envío de tráfico *DiffServ* sobre LSPs es relativamente simple.

Para soportar un modelo *DiffServ* por saltos, un operador de red MPLS necesita asignar un conjunto de recursos de envío para cada clase *DiffServ* en cada *router* MPLS, y asignar etiquetas.

MPLS permite el mapeo de los valores del byte DS del paquete IP a diferentes trayectos virtuales (LSPs) que podrían ser enrutados selectivamente en la red del proveedor de servicio, la ventaja adicional de su uso es que crea una topología de caminos a través de la red. Estos caminos permiten enrutar el tráfico IP únicamente en el nodo de entrada al dominio MPLS, mientras que en los nodos intermedios los datos son enviados mediante las etiquetas de la cabecera MPLS anexada al paquete, en lugar de un análisis de enrutamiento ejecutado en la cabecera IP. Por lo tanto los paquetes pueden viajar a través de la red más rápidamente, debido a que los dispositivos intermedios no necesitan llevar a cabo las complejas tareas de enrutamiento tradicional.

Si la etiqueta es codificada en el interior de la cabecera de alguna tecnología de capa 2 tal como ATM o Frame Relay, luego esos *switches* pueden ser parte de una red de envío de paquetes usando etiquetas MPLS. Así, si la red consiste de nodos enrutadores en los extremos y nodos MPLS ATM en el centro, los LSPs son establecidos entre cada par de *routers* de los extremos a través de switches ATM, lo cual crea un camino dedicado entre *routers* frontera del dominio.

6.2.1 MPLS Y CONMUTACIÓN DE HARDWARE ATM

Debido a que ATM es una tecnología que provee envío de datos a altas velocidades usando conmutación de celdas, el desarrollo de *backbones* de redes de alta velocidad últimamente ha estado basado sobre ATM.

Además el Modo de Transferencia Asincrónica fue diseñado para soportar múltiples clases de servicio con diferentes requerimientos de retardo y variación del mismo. Estas características y contribuciones proveen una excelente plataforma para el despliegue de una red MPLS, motivo por el cual se analiza la implementación de Servicios Diferenciados sobre una red ATM habilitada con MPLS.

Los LSPs dentro de un ambiente MPLS, son similares en cierta forma a una conexión ATM aún cuando existen ciertas diferencias derivadas principalmente del hecho que MPLS puede ser utilizado de dos maneras distintas [54]:

1. MPLS topológicamente administrado, y

2. MPLS en base a caminos explícitos.

6.2.1.1 MPLS TOPOLÓGICAMENTE ADMINISTRADO

En una red que utilice enrutamiento tradicional, los nodos contiguos intercambian información a través del protocolo correspondiente. Los datos son enrutados a través de los nodos intermedios y enviados a su nodo de destino de acuerdo a la tabla presente en cada nodo. La aplicación MPLS usa la información de la topología de red adquirida mediante el protocolo de enrutamiento a fin de establecer segmentos a través de los LSPs hacia la red de destino.

La ruta formada por los segmentos LSPs es controlada mediante protocolos de enrutamiento normal en el interior de la red, de tal modo que cuando cambian las condiciones de enrutamiento del próximo salto los caminos son redefinidos.

Como se puede pensar, al estar estos segmentos LSPs compartiendo los mismos enlaces con otros caminos directos vecinos, debe haber una compartición justa de recursos de red tanto para tráfico perteneciente a caminos directos como para tráfico perteneciente a caminos formados por segmentos.

En ATM, cada conexión es caracterizada a través de toda la red. Cada una de las conexiones posee un conjunto de parámetros que controlan las características de conexión tales como: velocidad máxima de celda, máximos tamaños de ráfagas y retardo. La programación de datos, sobre un enlace que contenga múltiples conexiones de diferentes clases, se la realiza siguiendo un esquema de prioridad para cada una de las clases de servicio. Así, el tráfico de alta prioridad es organizado en el nodo de ingreso para asegurar que los enlaces de la red tengan una suficiente capacidad para las conexiones.

En el caso de MPLS, no es factible determinar los parámetros para cada camino individual, debido a que éstos varían tanto sobre períodos cortos como largos dependiendo de la duración del tráfico circulante. Por lo tanto no es fácil aplicar funciones de formación de tráfico para cada conexión a fin de controlar la programación de datos. En lugar de ello, el tráfico puede ser caracterizado para

cada enlace entre un par de nodos vecinos, y esta caracterización es luego aplicada a un conjunto de conexiones MPLS compartiendo un mismo enlace. La programación de servicio luego puede ser aplicada ya no individualmente a cada conexión, sino mas bien al conjunto de conexiones sobre el enlace que comparten la misma clase de servicio, mediante alguna forma de compartición justa de la cola.

La utilización de MPLS topológicamente administrado habilita la conmutación por hardware (por ejemplo, *switches* ATM) para ejecutar el envío de datos a través de la red. Siendo esto aplicable en redes en las cuales la capacidad de conmutación excede grandemente la capacidad de enrutamiento.

En la actualidad únicamente el Protocolo de Distribución de Etiquetas (LDP, *Label Distribution Protocol*) ha sido definido para establecer LSPs topológicamente administrados dentro de la red. Éste emplea un formato general de mensaje para permitir que parámetros adicionales sean introducidos dentro de MPLS. Uno de los parámetros utilizados es el PFC (*Per-hop behavior Forwarding Class*), el cual es utilizado para especificar un grupo de PHBs que gobernará el envío de paquetes dentro de cada LSR.

El concepto de PFC es muy similar al definido por PHB en la Arquitectura *DiffServ*, para cada *router* que habilite diferenciación del servicio. Al establecer múltiples LSPs entre los nodos el dominio soporta múltiples *DiffServ* PHBs, y por lo tanto provee un dominio completamente *DiffServ*.

El esquema de señalización de MPLS permite dos opciones de utilizar los LSPs para soportar múltiples clases de servicio. La primera opción es tener un LSP separado, previamente establecido para cada clase de servicio que se haya definido. Mientras que la otra opción, es usar un LSP para un grupo de clases de servicio tales como el Grupo de Envío Asegurado AF_{1X}, en cuyo caso, la probabilidad de descarte para cada una de las subclases (AF_{11}, AF_{12} y AF_{13}) es indicada mediante el bit CLP dentro de la cabecera ATM.

6.2.1.2 MPLS para modo explícito

En la actualidad se han estandarizados dos protocolos con el fin de establecer caminos explícitos LSPs a través de la red, éstos son el CR-LDP y el RSVP.

El CR-LDP (*Constraint Based Routing LDP*) constituye una extensión del protocolo LDP, el cual provee la información necesaria para controlar el enrutamiento LSP y especifica algunas otras características requeridas para el establecimiento de la conexión, tales como velocidades de transmisión. Éste provee parámetros adicionales sobre el protocolo básico LDP, CR-LDP define [54]:

- Una velocidad de datos comprometida (CDR, *Committed Data Rate)*
- Una velocidad de datos límite (PDR, *Peak Data Rate)*
- Una tolerancia límite para ráfagas de datos *(*CBT, *Committed Burst Tolerance)*

El protocolo RSVP para MPLS ha sido adaptado del tradicional RSVP para implementar los objetivos de escalabilidad que no se alcanzaban y que no permitían que éste sea totalmente útil en los núcleos de las redes. De igual manera se adaptó RSVP a fin que provea la distribución de etiquetas y habilite capacidades de ingeniería de tráfico (refiérase a la sección 6.3).

Así, los protocolos RSVP y CR-LDP proveen la capacidad para controlar la ruta del LSP. Esto es, el LSP puede ser forzado a tomar un camino específico a través de la red, lo que converge hacia a un modo explícito de utilización de MPLS.

6.2.2 ATM EN LA DIFERENCIACIÓN DE SERVICIO

Si en el nodo ATM, se habilita una combinación de tratamiento de tráfico MPLS/ATM , es importante e imprescindible que el operador de red establezca y entienda la relación entre categorías de servicio ATM y clases de servicio MPLS dentro del nodo.

Un esquema modelo de mapeo entre categoría de servicio ATM y clase de servicio MPLS (que define algún DSCP) puede establecerse como indica la tabla 6.1.

Categoria de Servicio ATM	Diffser PHB
CRB	EF
rt - VBR	AF prioridad 1
nrt - VBR	AF prioridad 2
	AF prioridad 3
ABR, GFR, UBR, W-UBR	AF prioridad 4

Tabla 6.1 Equivalencia de Prioridades de Servicio entre ATM y *DiffServ*. [54]

El soporte de *DiffServ* se lo provee mediante el establecimiento de LSPs, con un especificado PFC, el cual controla las características del camino de datos.

EL PHB EF y las dos primeras clases de PHBs AF tienen características similares de servicio que las definidas para las conexiones de ATM CRB y rt-VBR, respectivamente. Aún mas tienen similares restricciones, tales como su aplicación en una topología punto a punto exclusivamente y el cumplimiento de perfiles de tráfico estrictos.

Las otras clases AF típicamente podrían tener mayores asignaciones de encolamientos, debido a su mayor probabilidad de descarte, lo que otorga al tráfico servido por estas clases un servicio con mayor retardo y niveles de calidad inferiores.

Finalmente, el servicio definido para el PHB *Best Effort*, existe para proveer una clase de servicio como el actualmente entregado en Internet.

Las clases *DiffServ* controlan el tratamiento de los paquetes dentro del nodo creando múltiples clases de servicio. Por lo tanto, para definir un SLA entre un proveedor y el cliente se utilizan las capacidades de un nodo *DiffServ* y las

conexiones MPLS/ATM, que permitan el dimensionamiento necesario de la red, a fin de cumplir con los compromisos adquiridos mediante el SLA.

6.3 RSVP EXTENDIDO PARA MPLS

Los problemas acusados a la escalabilidad y la gran demanda de *overhead* requerido por RSVP para soportar potencialmente millones de flujos *host a host*, han constituido el punto inicial de arranque para el desarrollo de nuevas tecnologías que tiendan a minimizar los impactos antes mencionados en las redes de transmisión de datos.

Como solución actual y de gran eficiencia surge la combinación de RSVP / MPLS. A fin de alcanzar el trabajo conjunto de estas dos tecnologías se ha desarrollado una extensión del protocolo RSVP estándar (estudiado en el capítulo 5) que da lugar a la definición del Protocolo RSVP Extendido (RSVP Ext).

MPLS selecciona la utilización del RSVP Extendido como protocolo de señalización para soportar la creación de LSPs que se caracterizan por ser automáticamente enrutados sin importar posibles estados de falla o congestión de la red. El uso y definición de RSVP Ext involucra ciertas ventajas sobre el protocolo RSVP estándar, por ejemplo:

- Soporta el establecimiento y mantenimiento de rutas LSPs explícitas.
- RSVP Ext posee la habilidad de aplicar un mismo tratamiento a una colección de flujos que compartan un camino común e iguales reservaciones compartidas de red, en lugar de aplicar un tratamiento por cada flujo *host a host*, tal como lo indica la figura 6.3. La agregación de numerosos flujos *host* a *host* dentro de un mismo LSP, reduce significativamente la cantidad de *overhead*.
- Los requerimientos de escalabilidad, latencia y *overhead* se consiguen mediante un conjunto de extensiones que reducen el número de mensajes de refresco y los requerimientos de procesamiento de mensajes.

- RSVP Ext puede reservar recursos de red en los *routers* LSRs a través del LSP, también permite a un LSP cursar tráfico *best effort* sin la necesidad de realizar una específica reservación de recursos.

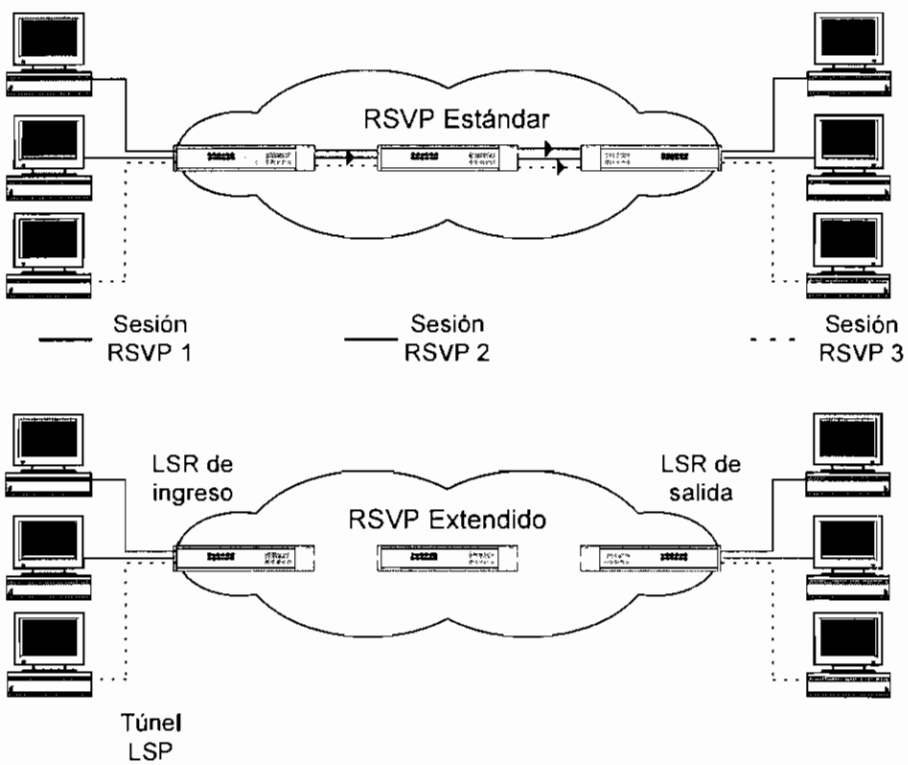

Figura 6.3 RSVP Estándar y RSVP Extendido. [55]

6.3.1 TÚNELES LSP

Un LSP establecido mediante RSVP, es mas conocido como un túnel LSP debido a que el tráfico que fluye a través de él no es visible para cada uno de los *routers* internos.

Para soportar el establecimiento de un túnel LSP, RSVP Ext define un nuevo Objeto *SESSION* llamado *LSP_TUNNEL_IPv4*. Este tipo de objeto informa a cada nodo LSR que el tráfico perteneciente a un túnel LSP específico debe ser identificado únicamente en el *router* de entrada a aquel túnel. Cabe recordar que en las especificaciones de RSVP solo se definen dos tipos de sesión: IPv4/UDP e IPv6/UDP, para mayor información refiérase al ANEXO 5.

6.3.1.1 Establecimiento de un túnel LSP

Para el establecimiento de un túnel LSP, el nodo LSR de ingreso envía un mensaje RSVP *Path* hacia el nodo LSR destino. El nodo LSR destino responde a la recepción del mensaje RSVP *Path* transmitiendo un mensaje RSVP *Resv* hacia el LSR de ingreso. Una vez que el nodo de ingreso recibe el RSVP *Resv*, el LSP es establecido y puede utilizarse el mismo para enviar tráfico entre el LSR ingreso y el LSR destino.

6.3.1.2 Mensaje RSVP *Path*

Cuando un nodo LSR de ingreso desea establecer un túnel LSP entre si y un nodo final, éste transmite un mensaje RSVP *Path* con un tipo de sesión LSP_TUNNEL_IPv4. El mensaje RSVP *Path* puede incluir ciertos objetos, algunos de ellos de carácter opcional. Los objetos soportados por el mensaje RSVP *Path* son los siguientes:

- Objeto *LABEL_REQUEST*
- Objeto *EXPLICIT_ROUTE*
- Objeto *RECORD_ROUTE*
- Objeto *SESSION_ATRIBUTE*
- Objeto *FLOWSPEC CoS*
- Nuevos *C-types*

6.3.1.2.1 Objeto LABEL_REQUEST

El objeto *LABEL_REQUEST* tiene como finalidad indicar que un LSP solicita que se le asigne una etiqueta, a fin de poder ser identificado. El objeto *LABEL_REQUEST* contiene también el campo Protocolo ID, que identifica el protocolo de capa 3 que atravesará el túnel.

Existen tres posibles tipos de objetos *LABEL_REQUEST*:

- LSP solicita una etiqueta, pero no especifica el rango en el cual ésta debe caer. Este tipo de objeto es el más común.

- El LSP solicita una etiqueta con una rango que especifique los valores máximos y mínimos de identificadores de camino virtual (VPI, *Virtual Path Identifier*) y de circuito virtual (VCI, *Virtual Circuit Identifier*). Este tipo de solicitud se utiliza cuando la etiqueta MPLS es transportada en la cabecera ATM (capa 2).

- El LSP solicita una etiqueta con una rango que especifica los valores DLCI (*Data Link Connection Identifier*) máximos y mínimos para una red Frame Relay. Este tipo de solicitud se utiliza cuando la etiqueta MPLS es transportada en la cabecera Frame Relay.

Cuando un mensaje *Path* arriba a un nodo LSR, éste almacena el objeto *LABEL_REQUEST* para ese LSP.

6.3.1.2.2 Objeto EXPLICIT_ROUTE

Mediante el objeto *EXPLICIT_ROUTE*, el LSR de ingreso puede especificar una ruta explícita[1] predeterminada para el LSP, independientemente de los resultados del enrutamiento convencional.

Una ruta explícita es codificada como una serie de subobjetos[2] contenidos en este objeto. Cada uno de los subobjetos puede identificar un grupo de *routers* a través de la ruta explícita o puede especificar una operación a ser ejecutada a través del camino.

[1] Ruta Explícita.- Especificación de un grupo de nodos a ser atravesados y un conjunto de operaciones a ser ejecutadas a través del camino.
[2] Subobjeto.- Es información insertada por varios nodos dentro de un mismo objeto.

6.3.1.2.3 Objeto RECORD_ROUTE

El objeto *RECORD_ROUTE* permite recolectar información acerca de la ruta que atraviesa el túnel LSP. Al igual que en el objeto anterior, su contenido es una serie de subobjetos.

Cuando un LSR de ingreso intenta establecer un túnel LSP, éste envía un mensaje que contiene un objeto *LABEL_REQUEST*. El mensaje *Path* puede también contener el objeto *RECORD_ROUTE*. Así el Objeto grabación de ruta inicial contiene la dirección IP del LSR de ingreso, a continuación el siguiente nodo al recibir un mensaje *Path* que contiene un objeto grabación de ruta almacena una copia de ese valor y adiciona su propia dirección IP al objeto mencionado (es decir se establece un subobjeto). Finalmente cuando el LSR de salida recibe el mensaje *Path* junto a éste objeto, adiciona un objeto grabación de ruta a su mensaje RSVP. Luego del intercambio de mensajes *Path* y *Resv*, cada uno de los *routers* a través del camino establecen la ruta completa para el LSP, desde el nodo de ingreso hasta el nodo final.

6.3.1.2.4 Objeto SESSION_ATTRIBUTE

Este objeto suele ser adicionado al mensaje *Path* a fin de controlar la prioridad y la preferencia, entre otras características del LSP.

Este objeto está constituido por un campo de 8 bits que se divide en dos subcampos de 3 bits cada uno y los 2 bits restantes no son utilizados, tal como se indica mediante la figura 6.4.

Figura 6.4 Formato del Objeto *SESSION_ATRIBUTE*. [55]

El primero de los subcampos es llamado de "Configuración de Prioridad" y define la prioridad que posee ese túnel LSP a la asignación de recursos con respecto a

otro túnel LSP. El rango de valores permitidos es de 0 a 7, siendo representada la prioridad más alta con el valor 0.

El otro subcampo es conocido como de "Mantenimiento de Prioridad", éste define la prioridad de ese túnel LSP con respecto a la posesión de recursos que desean ser consumidos por nuevos LSPs. El rango de valores permitidos es de 0 a 7, siendo representada la prioridad más alta con el valor 0.

6.3.1.2.5 Objeto FLOWSPEC

Este objeto es básicamente el mismo que se define en el protocolo RSVP estándar, para mayor información refiérase al ANEXO 5.

6.3.1.2.6 Nuevos Tipos de Objetos

De igual forma se han definido nuevos tipos de objetos en mensajes previamente ya definidos. RSVP Ext adiciona nuevos tipos de objetos para una clase de objeto dada (*SESSION, SENDER_TEMPLATE,* Y *FILTER_SPEC*). El protocolo RSVP estándar define solo dos tipos de objeto: los objetos compatibles con el protocolo IPv4 y aquellos que son compatibles con IPv6.

- *Objeto SESSION (C-Type = LSP_TUNEL_IPv4)*

Este objeto sesión es adicionado al mensaje *Path* para ayudar en la identificación y diagnóstico de la sesión. El nuevo tipo de objeto contiene la dirección del nodo de salida del túnel y un identificador único de 16 bits que permanece constante mientras dura el túnel LSP, aún si por alguna razón el túnel ha sido reenrutado.

- *Objeto SENDER_TEMPLATE (C-Type = LSP_TUNEL_IPv4)*

Este objeto contiene la dirección IPv4 del nodo en el cual se originan los datos.

- *Objeto FILTER_SPEC (C-Type = LSP_TUNEL_IPv4)*

El objeto *FILTER_SPEC*, junto con el objeto *SESSION* definen el conjunto de paquetes de datos (flujos) que reciben el servicio definido por el objeto *FLOWSPEC*. Este nuevo objeto contiene la dirección IPv4 del origen de datos.

6.3.1.3 Mensaje RSVP *Resv*

Un mensaje *Resv* es trasmitido por el nodo de salida hacia el LSR de ingreso al túnel, en respuesta a la recepción de un mensaje *Path*. El mensaje RSVP establece en cada nodo LSR el estado de la reservación: distribución de etiquetas, reservación de recursos solicitados y especifica el estilo de reservación (filtro fijo o compartición explícita).

Un mensaje *Resv* puede contener un número diferente de objetos RSVP, entre los cuales están:

- Objeto *LABEL:* ejecuta la respuesta a un proceso de demanda de etiqueta.
- Objeto *RECORD_ROUTE:* retorna el camino LSP al nodo que envió el mensaje *Path*.
- Objeto *SESSION*: únicamente identifica el LSP que está siendo establecido.
- Objeto *STYLE*: especifica el estilo de reservación a llevarse a cabo. RSVP EXP soporta estilos de reservación filtro fijo y compartición explícita.

Los objetos enunciados han sido analizados en la sección 6.3.1.2, a excepción del objeto *LABEL*, detallado a continuación.

OBJETO LABEL

El objeto *LABEL* es transportado en el mensaje *Resv*, y puede contener tanto una sola etiqueta como una pila de ellas. En aquellas implementaciones MPLS que no soporten una pila de etiquetas, únicamente se examina la etiqueta superior. Para estilos de reservación FF y SE, se provee una etiqueta por cada emisor.

Como se observa en la figura 6.5, el procesamiento realizado por un nodo LSR al objeto *LABEL* del mensaje *Resv* es el siguiente:

- Cuando el nodo LSR recibe un mensaje *Resv* correspondiente a un mensaje *Path* previo, éste confirma que el mensaje *Resv* fue transmitido

por el próximo salto en el LSP. El nodo LSR luego de realizada esa tarea, asigna la etiqueta entrante a una interfaz de salida y usa esa asignación para enviar el tráfico asociado con el LSP hacia su próximo nodo vecino en el mismo sentido que el tráfico de datos. Así por ejemplo, en la figura 6.5, el nodo LSR envía todo el flujo de tráfico con etiqueta de valor igual a 20 a través del túnel LSP mediante la interfaz 2.

- Cuando el nodo LSR recibe el mensaje *Resv*, éste configura una etiqueta asignada localmente (10 en el ejemplo) a la interfaz de entrada del LSP (Interfaz 1). La interfaz entrante es la única sobre la cual el LSR ha recibido el mensaje *Path* correspondiente al LSP.

- El nodo LSR construye un nuevo objeto *LABEL* con un valor igual a 10, reemplazando la etiqueta superior (en la pila) en el mensaje *Resv* recibido, y lo reenvía a su nodo inmediatamente anterior.

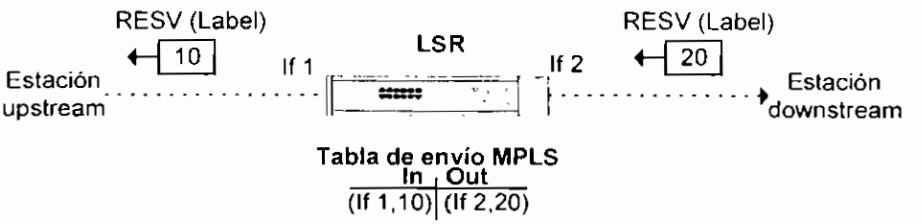

Figura 6.5 Procesamiento de un objeto LABEL por el LSR. [55]

6.4 EJEMPLO DE ESTABLECIMIENTO DE UN TÚNEL LSP MEDIANTE RSVP EXT

A continuación se da un ejemplo de cómo RSVP Ext establece un túnel LSP, para lo cual se ilustra en la figura 6.6 la topología a utilizarse.

En ella se asume que MPLS y RSVP Ext han sido habilitados en todos y cada uno de los nodos LSR 1, LSR 2, LSR 3 y LSR 4. Además de antemano se conoce que el LSP debería seguir la ruta explícita LSR 1 - LSR 2 - LSR 3 - LSR 4.

Mensaje RSVP Path

LSR 1 inicia el establecimiento del LSP, siguiendo procedimientos de señalización RSVP estándar. LSR 1 transmite un mensaje *Path* al LSR 4, el cual viaja a través de los nodos intermedios LSR 2 y LSR 3, hasta alcanzar el nodo final.

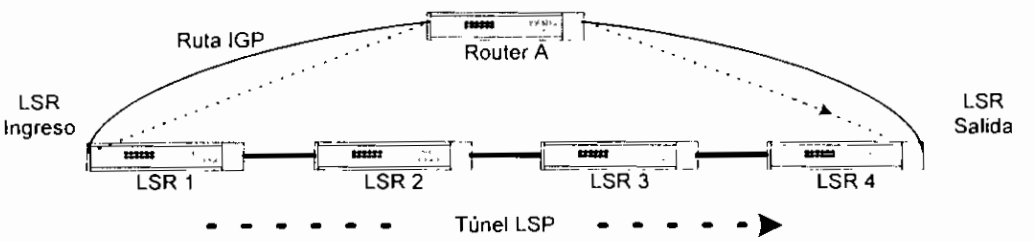

Figura 6.6 Topología de la Red para el ejemplo. [55]

Procesamiento en LSR 1:

1. Para establecer el LSP, el nodo LSR 1 crea un mensaje *Path* en el cual se especifican, los objetos *EXPLICIT_ROUTE* y *LABEL_REQUEST*, anteriormente definidos. Así como también pueden estar anexados los objetos *RECORD_ROUTE*, *SESSION*, *SESSION_ATRIBUTE*, *SENDER_TEMPLATE* Y *SENDER_TSPEC*.

2. En caso de que el LSP sea para cursar tráfico *best effort*, y no se requiera ninguna asignación de recursos, se solicita el Servicio de Carga Controlada con un tamaño de ráfaga y velocidad de transmisión igual a cero.

3. El mensaje *Path* es transmitido hacia el LSR 4 a través del camino especificado por el objeto *EXPLICIT_ROUTE*.

Procesamiento en LSR 2:

1. Una vez que el mensaje *Path* arriba al nodo LSR 2, éste graba los objetos *LABEL_REQUEST* y *EXPLICIT_ROUTE* en su Bloque de Almacenamiento de Estado del Camino. Donde también se almacenan la dirección IP del

salto previo, la sesión, el emisor y el TSpec; esta información es usada para enrutar el correspondiente mensaje RSVP de vuelta al LSR 1.

2. El LSR 2 envia el mensaje *Path* hacia el LSR 4 a través del camino especificado en el *EXPLICIT_ROUTE*.

3. En el caso que el LSR 2 no pueda asignar una etiqueta para el LSP, éste responde enviando un mensaje *PathErr* hacia el LSR 1.

Procesamiento en LSR 3:

1. Una vez que el mensaje *Path* arriba al LSR 3, graba los objetos *EXPLICIT_ROUTE* y *EXPLICIT_ROUTE* en su Bloque de Almacenamiento de Estado del Camino. Donde también se almacenan la dirección IP del salto previo, la sesión, el emisor y el TSpec. Esta información es usada para enrutar el correspondiente mensaje RSVP de vuelta al LSR 2.

2. El mensaje *Path* es enviado hacia el LSR 4 a través del camino especificado en el *EXPLICIT_ROUTE*.

3. En el caso que el LSR 3 no pueda asignar una etiqueta para el LSP, éste responde enviando un mensaje *PathErr* hacia el LSR 1.

Procesamiento en LSR 4:

1. Una vez que el mensaje *Path* arriba al nodo LSR 4, éste advierte que es el último nodo del túnel al analizar el valor *LABEL_REQUEST* y se prepara para enviar el mensaje *Resv* hacia LSR 1.

Mensaje RSVP Resv

De la misma manera que se generó el mensaje *Path* siguiendo una serie de procedimientos de señalización RSVP estándar, LSR 4 genera un mensaje *Resv* para la sesión, distribuyendo etiquetas y estableciendo en estado de envío para el túnel LSP. La dirección IP destino del mensaje *Resv* es obtenida del Bloque de Almacenamiento de Estado del Camino de aquel LSR.

Procesamiento en LSR 4:

1. LSR 4 asigna una etiqueta con un valor de 0, y la coloca en el objeto *LABEL* del mensaje *Resv*. El valor 0 tiene un significado especial para el LSR 4: cuando LSR 4 recibe un paquete MPLS con un valor de etiqueta igual a 0, éste conoce que es el nodo de salida para el LSP. En el flujo normal de datos, el nodo LSR 4 simplemente retira la etiqueta y envía los paquetes de datos basándose en la dirección IP de destino, contenida en la cabecera IP.

2. Si el LSR 1 inserta un TSpec en el mensaje *Path*, el nodo LSR 4 utiliza esta información para construir un apropiado receptor TSpec y RSpec.

3. El mensaje *Resv* es transmitido de vuelta hacia el LSR 1 a través del nodo LSR 3. El mensaje *Resv* no sigue un camino establecido por objetos *EXPLICIT_ROUTE*, en lugar de ello el mensaje *RESV* sigue el camino inverso seguido por el mensaje *Path* y almacenado en el Bloque de Almacenamiento de Estado del Camino.

Procesamiento en LSR 3:

1. El nodo LSR 3 recibe el mensaje *Resv* que contiene la etiqueta asignada por el LSR 4.

2. Luego de ello almacena la etiqueta (0) como parte del estado de reservación para el LSP, utilizando luego esta etiqueta para enviar tráfico de salida a través del LSP hacia LSR 4.

3. Finalmente asigna un nuevo valor de etiqueta (20) y lo coloca en el objeto *LABEL* (reemplazando el valor anterior), y lo envia hacia el LSR 2. Ésta es la etiqueta que LSR 3 usa para identificar tráfico entrante sobre el LSP desde LSR 2.

Procesamiento en LSR 2:

1. LSR 2 recibe el mensaje *Resv* que contiene la etiqueta asignada por el LSR 3.

2. Luego de ello almacena la etiqueta (20) como parte del estado de reservación para el LSP

3. Finalmente, asigna un nuevo valor de etiqueta (10) y lo coloca en el objeto *LABEL*, y lo envía hacia el LSR 1; el nodo LSR 2 utilizará esta etiqueta para identificar tráfico entrante sobre el LSP desde LSR 1.

Procesamiento en LSR 1:

1. LSR 1 recibe el mensaje *Resv* que contiene la etiqueta asignada por el LSR 2 y usa esa etiqueta para todo el tráfico de salida que es asignado al LSP.

2. Como un resultado de estas operaciones, el LSP es establecido desde el LSR 1 hasta el LSR 4 siguiendo el camino indicado explícitamente por el objeto *EXPLICIT_ROUTE*.

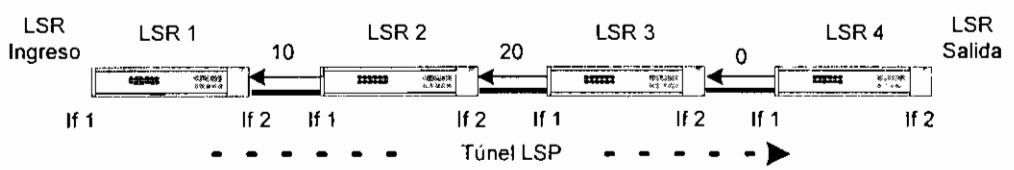

Figura 6.7 Asignación de Etiquetas en el ejemplo. [55]

Capítulo 7

SISTEMAS DE POLÍTICAS DE CALIDAD DE SERVICIO

CAPÍTULO 7

SISTEMAS DE POLÍTICAS DE CALIDAD DE SERVICIO

Una red con calidad de servicio debe permitir al administrador de la misma configurarla en forma eficiente de modo que la asignación de recursos se realice en concordancia con las necesidades específicas de la empresa y que además se tenga facilidad de ejecutar las demás tareas de administración cotidianas como por ejemplo la obtención de los respaldos de la información.

En la consecución de los objetivos señalados, desempeñan un rol fundamental el sistema de políticasy sus componentes: los servicios de Directorio y de Políticas, propuestos en este capítulo.

Este capítulo hace un estudio de varios protocolos de políticas y de servicios de directorio, analizándose en mayor detalle los protocolos COPS (*Common Open Policy Service*) y LDAP (*Lightweight Directory Access Protocol*), los cuales son la base del modelo. Se estudia también el modelo del sistema de políticas utilizado en una red de calidad de servicio.

7.1 SISTEMAS DE POLÍTICAS TRADICIONALES

El sistema de políticas propuesto en el numeral 7.2 y 7.5, comparado con las formas tradicionales de implementación de políticas presenta varias ventajas, entre las principales se cuentan la facilidad de administración y el dinamismo del sistema. Formas de implementación de políticas tradicionales son el aprovisionamiento manual y el aprovisionamiento por medio de "olfateadores" (*sniffers*).

El **aprovisionamiento manual** lo realiza en forma explícita el administrador de la red mediante la creación de listas de acceso en los routers que especifican la forma de tratar los paquetes. La implementación de las listas es una tarea

bastante compleja y aunque no es dinámica es una forma básica de contrarrestar la congestión.

El **aprovisionamiento por medio de *sniffers*** u olfateadores se lo realiza monitoreando el tráfico en un segmento de red susceptible a congestión y en base a la información recolectada, el administrador reconfigura el dispositivo de red si no está proveyendo al tráfico el trato adecuado. Esta forma de aprovisionamiento garantiza un nivel de servicio en un segmento de la red pero su implementación extremo a extremo sería bastante difícil y costosa pues involucra la utilización de un *sniffer* en cada segmento del camino.

7.2 SISTEMA DE POLÍTICAS GENÉRICO

Cuando se hace un requerimiento de reservación de recursos, el dispositivo de red utiliza información interna para determinar si hay disponibilidad de recursos, pero también utiliza información externa para determinar si el usuario o la aplicación solicitante está permitida a realizar la reservación. Un sistema de políticas provee los mecanismos necesarios para comunicar esta información externa.

Un sistema de políticas administra un conjunto de reglas de alto nivel que definen el comportamiento de la red ante aplicaciones o usuarios determinados cuando éstos requieren calidad de servicio. Para ello, se necesitan de entidades que hagan cumplir las reglas y de entidades que decidan cuándo y cómo aplicarlas, haciendo una analogía se necesita de "policías" y "jueces" respectivamente.

Un Sistema de Políticas se compone de estas tres entidades: reglas, "policías" y "jueces"; y, aunque su principal función es suministrar reglas para una red con calidad de servicio, está diseñado para trabajar en entornos de red sin calidad de servicio e incluso sobre aplicaciones individuales.

Las entidades encargadas de la toma de decisiones ("jueces") son los Servidores de Políticas, los cuales recuperan las políticas ("reglas") de los Servidores de Directorio y comunican las decisiones a los Clientes de Políticas ("policías") para

que las hagan cumplir. La interacción entre estos componentes se puede apreciar en la figura 7.1.

Los servidores de Políticas y de Directorios, pueden ser servidores comunes centralizados con características de hardware dependientes del protocolo, o software, que se utilice. Un cliente de Políticas puede ser un router que implemente calidad de servicio en la red.

Para la comunicación entre los Servidores de Directorios y Servidores de Políticas se utilizan Protocolos de Acceso a Directorios como LDAP[1] (*Lightweight Directory Access Protocol*) y SQL[2] (*Structured Query Language*), mientras que la comunicación entre el Servidor de Políticas y su Cliente está a cargo de un Protocolo de Políticas como por ejemplo COPS[3] (*Common Open Policy Service*).

El término "política" puede tener varios significados o interpretaciones, pero para que ello no suceda se presenta la siguiente definición:

Política: "Una Política es una regla o un conjunto de reglas que describen la o las acciones a tomarse cuando se presentan ciertas condiciones específicas." [56]

El sistema de políticas descrito tiene ciertas características, enunciadas a continuación:

- Una política puede componerse de varias políticas anidadas dentro de ella, es decir las políticas presentan una estructura jerárquica que permite, primero tener políticas complejas que representen eficientemente lo que se desea implantar, y segundo, otorgan facilidad de administración.

- Las políticas deben ser verificables y no deben presentar ambigüedades, debe existir una sola forma de proceder baja una determinada condición.

---

[1] LDAP.- Protocolo Ligero de Acceso a Directorios, refiérase a la sección 7.4.2.2
[2] SQL.- Lenguaje de programación que se utiliza para recuperar y actualizar la información contenida en una base de datos relacional. Desarrollado por IBM en los años 70, se ha convertido en estándar ISO y ANSI.
[3] COPS.- Protocolo Común de Servicio de Políticas, refiérase a la sección 7.6.

- Las políticas deben partir de criterios globales que atraviesen los dominios administrativos permitiendo a los sistemas de políticas interactuar entre sí.

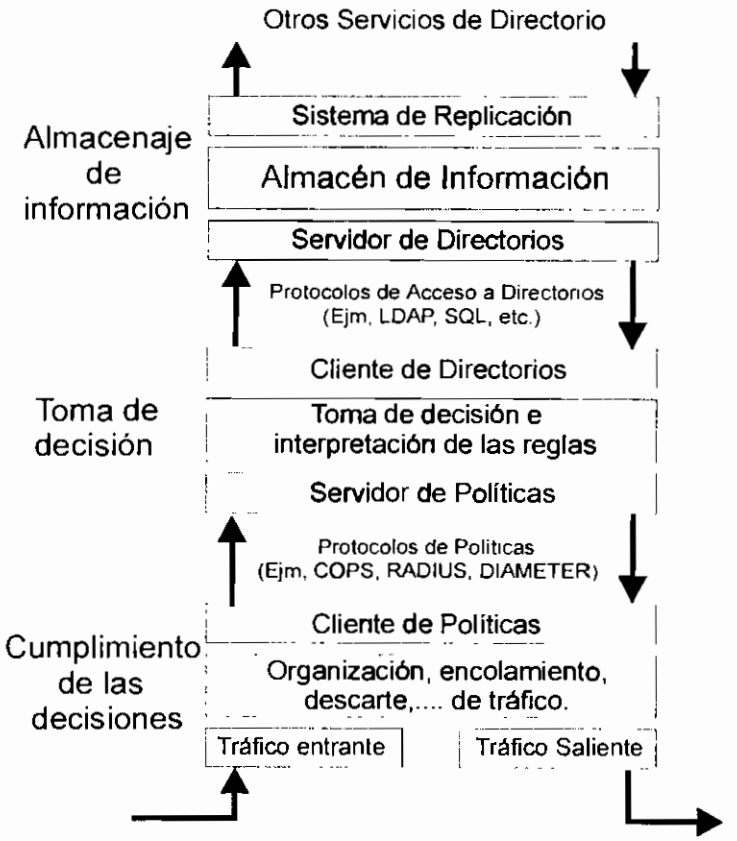

Figura 7.1 Sistema de Políticas Genérico. [6]

7.3 PROTOCOLOS DE POLÍTICAS

Existen varias formas o protocolos para transmitir a los dispositivos de red información relacionada con políticas, las cuales han evolucionado de un tradicional modelo cliente-servidor estático (*Radius* y *Diameter*) a uno más ágil y dinámico (COPS). La diferencia radica en que con el modelo tradicional no es posible transmitir y hacer cumplir una determinada configuración cada vez que las condiciones del sistema cambien.

Para ilustrar esta diferencia considérese como ejemplo la red de acceso de un Proveedor de Servicios de Internet ISP para sus clientes "*dial up*". Un usuario de este tipo para acceder a Internet establece una conexión temporal con el ISP a

través de una línea telefónica convencional. Cuando se desea establecer una conexión, el usuario realiza una llamada al Servidor de Acceso a la Red NAS (*Network Access Server*) del ISP, el cual luego de contestarle verifica los datos del usuario, estableciéndose la conexión si éstos son válidos.

En el modelo tradicional el usuario permanece conectado independientemente de las modificaciones que sufra su cuenta después de la conexión; este modelo se ilustra en la figura 7.2.

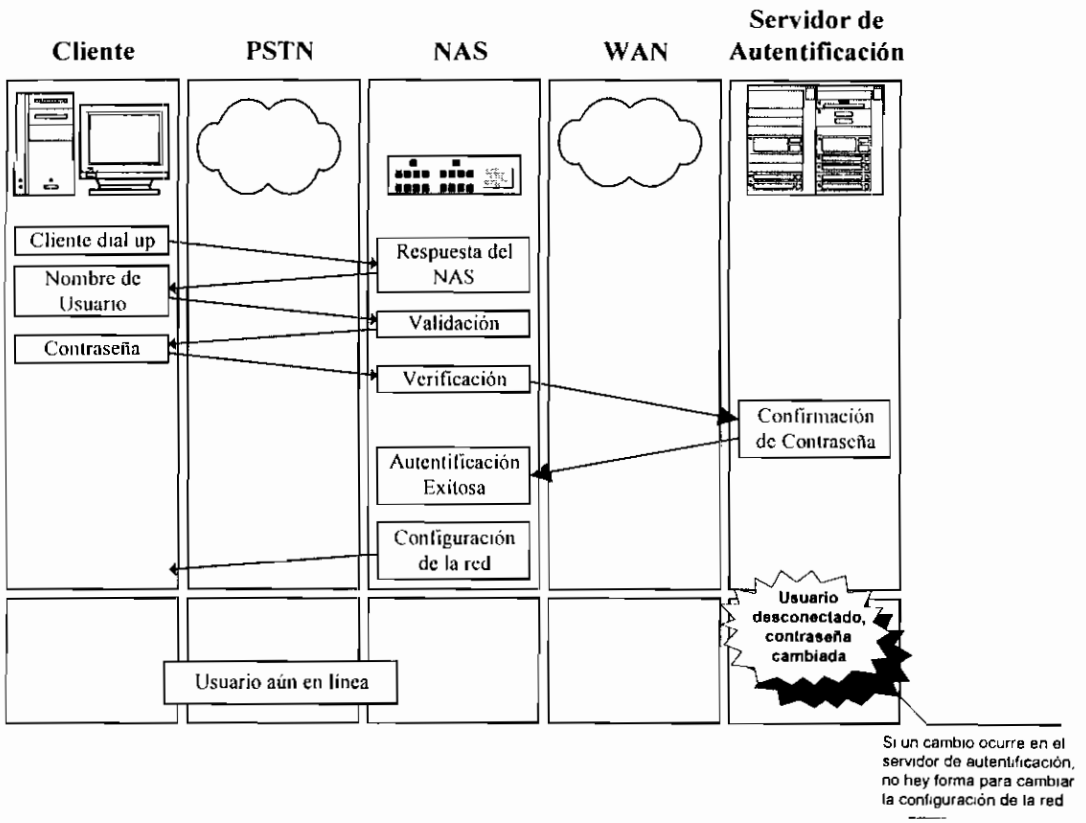

Figura 7.2 Modelo de Autentificación Tradicional. [6]

A diferencia del anterior, en un modelo más dinámico es posible efectivizar los cambios a medida que éstos ocurren; es decir, si se deshabilita la cuenta de un usuario mientras está conectado, éste debe ser inmediatamente desconectado del servicio; como se puede observar en la figura 7.3.

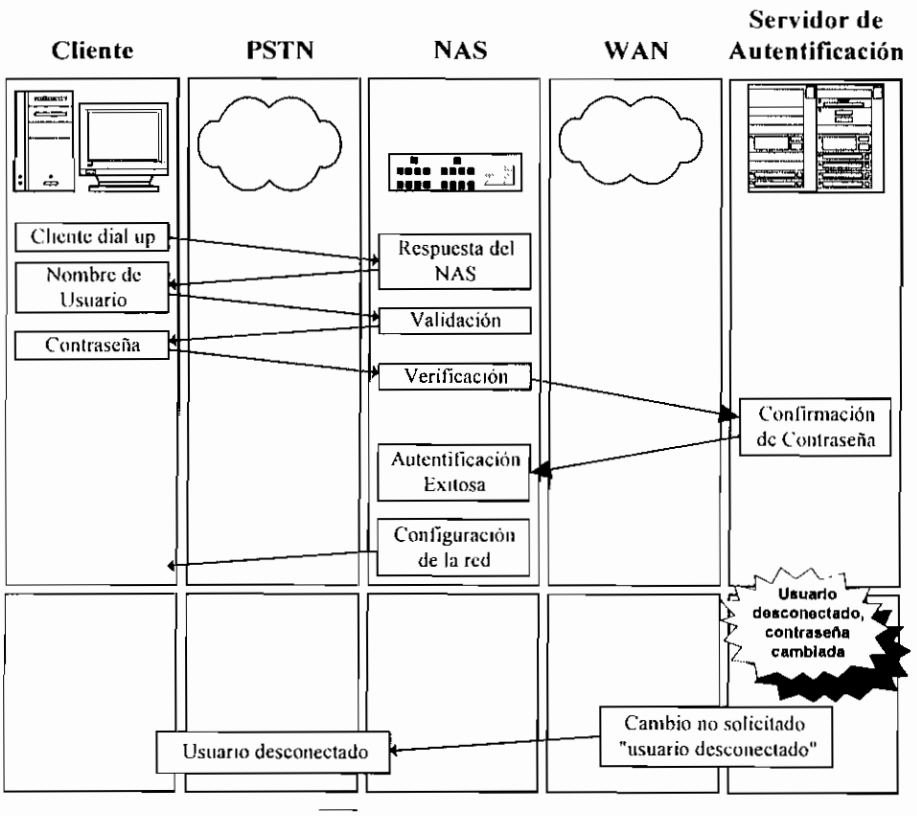

Figura 7.3 Modelo de Autentificación Dinámico. [6]

Este último modelo es bastante útil si se desea asignar dinámicamente los escasos recursos de un enlace entre varios clientes que lo comparten y que tienen diferentes derechos sobre él como en una red con calidad de servicio.

7.3.1 PROTOCOLO *RADIUS*

El protocolo de Servicio de Usuario de Acceso Remoto *Dial Up* (*RADIUS*, *Remote Access Dial Up User Services*) es ampliamente utilizado en las redes de acceso a los proveedores de servicio de Internet. A pesar de su extensa popularidad pertenece al grupo de protocolos tradicionales con los cuales es imposible realizar cambios sin que el cliente lo haya pedido.

Este protocolo es utilizado para transportar pares atributo-valor[1] entre los clientes y los servidores con el objetivo de efectuar autentificación. Si la autentificación es exitosa, el protocolo es capaz de transportar desde datos sencillos como la dirección IP asignada al cliente hasta datos más complejos como en el caso del establecimiento de una conexión de red privada virtual.

Debido al poco ancho de banda existente en los enlaces *Dial Up*, el cual típicamente es 28.8 kbps, es bastante difícil e ineficiente diferenciar entre las aplicaciones de un mismo usuario, pero en cambio se puede diferenciar entre los usuarios que acceden al ISP.

El protocolo *RADIUS* permite priorizar el tráfico de un usuario si existe la configuración correcta entre el servidor del protocolo y el router de acceso a Internet. La forma de dar prioridad a un cliente es evitando el descarte de su tráfico y permitiendo el descarte de otros, acción efectuada en el router mencionado. Es posible además marcar el tráfico con el adecuado DSCP si la red del proveedor lo soporta.

7.3.2 PROTOCOLO *DIAMETER*

Este protocolo es una continuación del anterior, en el cual se supera las principales falencias de *RADIUS*: dinamismo y extensibilidad. El protocolo *DIAMETER* es dinámico pues permite efectivizar los cambios sin requerimiento del cliente, además supera la poca extensibilidad de *RADIUS*, en el cual su funcionamiento depende del fabricante.

Este protocolo tiene total compatibilidad con su predecesor *RADIUS*, además al igual que éste, no especifica una implementación en particular, pero sugiere que los pares atributo-valor sean capaces de definir parámetros correspondientes al control de admisión con el protocolo RSVP, a la administración de un acceso *dial up* y a la administración de una sesión de voz sobre IP.

---

[1] Par atributo-valor.- Formato de la información intercambiada entre dos entidades cuando realizan una negociación. Por ejemplo, en la verificación del nombre de usuario para el establecimiento de una sesión de Internet, se utiliza el par: *UserName-Ihmp*

7.3.3 PROTOCOLO COPS

Cuando los elementos de red (clientes de un sistema de políticas) son completamente pasivos, el sistema de políticas debe censar constantemente al dispositivo para enterarse de su estado y realizar cambios si fuere necesario. Esta conducta involucraría la utilización de excesivos datos de control (*overhead*) que congestionarían la red.

"La evolución de las redes es bastante similar a la de la educación; así, en la primera mitad del siglo XX, la educación se enfocaba a un conocimiento general, sin embargo, debido a la gran cantidad de información, la tendencia actual no es tratar de saber todo sino saber como obtener el conocimiento. En el Internet ha sucedido algo semejante, es imposible trabajar con dispositivos que cuenten con toda la información que les permita trabajar en un entorno de usuarios debido al *overhead* que se necesitaría, en lugar de ello los dispositivos deben enviar requerimientos de configuración y en función de ésta, manipular el tráfico." [6]

Se puede pensar que se solucionaría el problema si se suministra a un router la información pertinente a su localidad, pero esto no es cierto, dadas las actuales condiciones, en las cuales un trabajador puede estar en varias ciudades en cortos periodos de tiempo. La solución es "enseñar" al router la forma de obtener la información.

A pesar de las ventajas que ofrece el protocolo *DIAMETER*, lo analizado últimamente, hace que el protocolo COPS sea el más adecuado para transportar información de políticas, pues su surgimiento fue motivado por esa filosofía. Este protocolo será estudiado con profundidad en la sección 7.6.

7.4 INFRAESTRUCTURAS DE DIRECTORIOS

7.4.1 DIRECTORIOS

Un directorio es una "base de datos que es optimizada para leer información de ella en un entorno de red" [6]. La información referida tiene un formato de pares **atributo-valor**, por ejemplo la información concerniente a la dirección de correo

electrónico de un usuario cualquiera puede tener el siguiente formato atributo-valor: ***e-mail*: abc@xyz.com**.

Un directorio se diferencia de una base de datos tradicional fundamentalmente en dos aspectos:

1. Un directorio contiene datos que se buscan a menudo pero rara vez se modifican; una base de datos contiene datos que se modifican constantemente.

2. Una base de datos almacena la información en tablas; un directorio lo hace con una estructura jerárquica orientada a objetos.

La función principal de un directorio de red con QoS es la administración de la misma, para ello debe contener información de usuarios y dispositivos de red que permita la asignación de recursos de red a los usuarios y aplicaciones permitidas. A pesar de la variedad de criterios y formas de administración, ellas deben basarse sobre un mismo depósito de información para que exista consistencia.

Entre las principales características de un directorio se pueden considerar:

- Un servicio de directorio es bastante consistente, ante una misma entrada debe existir una sola salida. Esta característica debe mantenerse si el directorio es centralizado o es distribuido en varios servidores de la red.

- El directorio de red debe estar disponible rápidamente, razón por la cual es optimizado para lograr respuestas rápidas, pero ello se logra a cambio de cierto estatismo en el cambio de la información.

Los directorios independizan los objetos lógicos del hardware del dispositivo de red. Cuando se configura un dispositivo en alguna forma tradicional se hace una traducción del sistema del dispositivo a través de interfaces SNMP, Web o de líneas de comandos; cuando la configuración es por medio de un servicio de directorios, el proceso es más rápido pues se lo realiza directamente sobre los objetos lógicos del dispositivo, sin traducción previa.

- El servicio de directorios debe soportar entornos de red heterogéneos. Para ello debe tener una infraestructura común, capaz de permitir compatibilidad entre los diversos fabricantes, además debe ser extensible, es decir debe tener la capacidad de aceptar cambios que le permitan adaptarse a condiciones o información desconocida al momento de su desarrollo.

- Los directorios por sí mismos no disponen de mecanismos que permitan una operación confiable, por lo tanto seguridad, control de acceso, autentificación y respaldo de la información se obtienen a través de aplicaciones externas.

La información dentro de un directorio está organizada en forma de un árbol jerárquico, en el cual cada nodo tiene una identificación única. La organización es similar a la utilizada en un sistema de archivos de un computador.

Esta forma de organización permite agrupar la información, habitualmente, diferenciando la ubicación geográfica de los diferentes tipos de objetos. Cada tipo de objeto contiene un conjunto de pares atributo-valor específicos a él y tal vez bastante diferentes a otro tipo de objeto. Por ejemplo, si el tipo de objeto es un usuario, los atributos pueden ser su nombre, dirección electrónica, puesto en la empresa, etc.; pero, si el tipo de objeto es una impresora, los atributos tal vez serían su marca y el número de serie.

Un ejemplo de la organización de un directorio se encuentra en la figura 7.4.

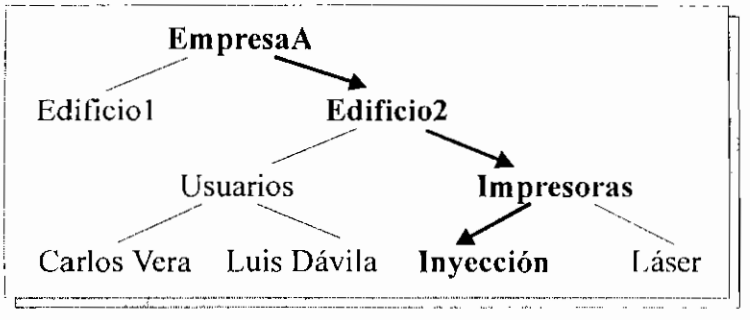

Figura 7.4 Estructura de un directorio jerárquico. [6]

En el ejemplo anterior, para referirse a la impresora nombrada Inyección, se da el nombre *EmpresaA.Edificio2.Impresoras.Inyección.*

Para optimizar la eficiencia y seguridad de una empresa, ésta debe poseer un solo servicio de directorios para cualquier proceso a realizarse. Actualmente, por ejemplo, si un empleado sale de la empresa, su salida tal vez conste en el rol de pagos pero muy probablemente no conste en el servidor de la red, estas inconsistencias tornan a la administración difícil e ineficiente.

7.4.2 PROTOCOLOS DE ACCESO A DIRECTORIOS

La estructura y forma de organización de la información dentro de un directorio depende básicamente de los protocolos de acceso. Los protocolos de acceso a directorios definen entre otras cosas, lo siguiente:

- Reglas que indican al cliente la forma de escribir y leer en un servicio de directorio.
- Reglas para identificar usuarios válidos (autentificación) y las acciones que se debe tomar (autorización).
- Reglas para realizar operaciones ejecutadas en el directorio, por ejemplo la realización de búsquedas.
- El sistema de réplicas, por medio del cual se distribuye la información entre los servidores de directorios, y los métodos de sincronización que el sistema utiliza para mantener consistencia.
- Una estructura común para el contenido del directorio, definiendo por ejemplo el formato de la información (datos binarios, de texto o de fechas).

7.4.2.1 Protocolo X.500

X.500 es un servicio de directorio estándar que describe los pasos para una comunicación entre un cliente y un servidor de directorios (acceso al directorio), definiendo además la estructura de la información (directorio). Fue diseñado

especialmente para hacer que gran cantidad de información esté disponible a un elevado número de servidores, manteniendo siempre consistencia.

Las principales características de este protocolo son:

- Define una estructura jerárquica.
- Define diferentes clases de objetos.
- Es un sistema complejo de varios agentes distribuidos, cada uno con labores específicas, las cuales las cumplen con sus respectivos agentes pares de otro servicio.

Por las características que presenta, X.500 es un protocolo demasiado pesado para su implementación en la mayoría de las redes, pues exige altas características del hardware. Ello ha hecho que no exista software disponible y mas bien X.500 es tomado como un modelo de referencia, similar al modelo de redes OSI, en base al cual se desarrollan otros protocolos como LDAP.

7.4.2.2 Protocolo LDAP (*Lightweight Directory Access Protocol*)

El Protocolo Liviano de Acceso a Directorios LDAP fue desarrollado en la Universidad de Michigan como una alternativa al Protocolo X.500, dada la complejidad y elevadas exigencias de recursos de este último; el directorio definido por LDAP puede ser instalado en PCs comunes, a diferencia de X.500 que exige *mainframes* o minicomputadores. Existen tres versiones del Protocolo LDAP expuestas en los RFCs 1487 (LDAPv1), 1777 (LDAPv2) y 2830 (LDAPv3).

Este protocolo define la forma de comunicación entre un servidor y un cliente de directorios, además define la estructura de la información dentro del directorio. Las características principales de este protocolo son:

- LDAP utiliza para la comunicación el Protocolo Internet (IP), y realiza control de congestión con la ayuda del Protocolo de Control de Transporte (TCP).

- LDAP define un modelo jerárquico de organización de la información, cuyas entradas contienen pares atributo-valor estándares.

- Para disminuir la complejidad de X.500, LDAP ofrece menor soporte de seguridad de la información.

La última versión del protocolo LDAPv3 tiene características adicionales:

- Si un servidor LDAP no puede satisfacer el requerimiento de un cliente, debido a falta de información o porque ésta no es LDAP, puede direccionar el requerimiento hacia otro servidor de directorios.

- Implementa mayor seguridad con la utilización de SSL[1] (*Secure Sockets Layer*) empleado en los navegadores de Internet cuando se realizan transacciones seguras.

- Presenta capacidad de adaptación a operaciones y objetos que aparezcan posteriormente.

El protocolo LDAP define cuatro términos que permiten acceder a la información en una estructura jerárquica, estos términos son: entrada, clase del objeto, atributo y nombre distinguido DN (*Distinguished Name*).

La **Entrada** hace referencia a un nodo del árbol jerárquico, la **Clase de Objeto** especifica el tipo de objeto que describe la entrada, el **Atributo** define las características que tiene una entidad en particular y depende del tipo de objeto al que se refiera, y, por último, el **Nombre Distinguido** (DN) presenta la ruta, los atributos y sus valores de una entrada.

Para entender cada uno de estos términos considérese el siguiente ejemplo basado en la figura 7.4:

Entrada: Impresora de Inyección localizada en el Edificio 2 de la Empresa A.

---

[1] *Secure Sockets Layer* SSL: Sistema de encriptación con claves privadas y públicas desarrollado por Netscape para administrar la seguridad en la transmisión de mensajes a través de la red.

Clase de Objeto: Impresora.

Atributos: Empresa (e), Edificio (b), Impresora (i).

DN: "i = Inyección, b = Edificio 2, e = Empresa A".

Un cliente LDAP para comunicarse con el servidor LDAP establece una sesión TCP, en el puerto predeterminado 389 generalmente [57]. Luego del establecimiento de la sesión se utilizan operaciones estándar para el acceso y actualización de la información en el directorio.

LDAP define las siguientes operaciones que ofrece un servidor a un cliente:

Vincular: Es el establecimiento de la sesión TCP entre un cliente y un servidor LDAP.

Desvincular: Es la finalización de la sesión TCP.

Buscar: Permite la realización de búsquedas de entradas. Esta operación presenta la facilidad de utilizar operadores lógicos, filtros, y permite además definir el campo de la búsqueda. Por ejemplo, con la opción de filtrado se pueden buscar los nombres de todas las personas que empiecen con la letra "e", y con la definición del campo se puede limitar la búsqueda a una rama específica del árbol. Otra opción importante es la habilidad para limitar los resultados a obtenerse de esta operación, en forma similar a un buscador de páginas web.

Modificar: Modifica los datos de alguna entrada.

Añadir: Añade entradas para una entidad nueva.

Borrar: Elimina una entrada existente.

Comparar: Permite comparar el valor de los atributos de un objeto con el valor almacenado en el directorio.

Abandonar Requerimiento: Abandona un requerimiento que está pendiente.

En respuesta a un requerimiento del cliente LDAP, siempre existirán resultados, si la operación fue exitosa, o en su defecto mensajes de error, si la operación fue defectuosa, generados por el servidor LDAP.

Finalmente, se puede decir, que el protocolo "LDAP es el potencial reemplazante de las listas específicas a la aplicación y que además es la fuente común de datos a la cual se dirigiría todo tipo de aplicaciones cuando lo requieran" [58]. Por ejemplo para la autentificación de usuarios bastaría definir los datos de éstos una sola vez en un directorio LDAP, y cada servidor realizaría la autentificación en este lugar independiente del sistema operativo de red (NT, Unix, Linux,..., etc.).

7.5 MODELO DEL SISTEMA DE POLÍTICAS

El modelo que se presenta a continuación fue propuesto por el IETF (*Internet Engineering Task Force*), inicialmente fue concebido para proveer un sistema de políticas a una red que soporte la Arquitectura de Servicios Integrados y RSVP, sin embargo el modelo es extensible a la Arquitectura de Servicios Diferenciados.

Los componentes básicos del modelo son:

- Punto de Ejecución de Políticas (PEP, *Policy Enforcement Point*): Es la entidad encargada de ejecutar o hacer cumplir las políticas, es decir son los "policías" del sistema, si se hace una analogía.

- Punto de Decisión de Políticas (PDP, *Policy Decision Point*): Es la entidad encargada de tomar decisiones basándose en un conjunto de reglas, si se hace una analogía, corresponderían a los "jueces" del sistema.

Estos componentes y su interacción se ilustran en la figura 7.5.

Las funciones principales del modelo son (ver figura 7.5):

- Tomar Decisiones: El PDP debe recibir los requerimientos del PEP, interpretarlos, detectar conflictos y enviar de respuesta las decisiones tomadas. Las decisiones que tome el PDP se basan en reglas relevantes especificadas para determinadas condiciones de la red. La comunicación

entre el PDP y el PEP se efectúa mediante el Protocolo COPS (*Common Open Policy Service*).

El PDP recupera la información concerniente a las reglas del sistema desde un servidor de autentificación o de un depósito de políticas por medio del Protocolo Ligero de Acceso a Directorios (LDAP). La forma de recuperación de las políticas debe ser eficiente, es decir, se debe recuperar lo estrictamente necesario, ni más ni menos de lo que se necesita, y esto debe ser realizado con el menor número de intentos posibles, pues corresponde a información de control de tráfico de tiempo real.

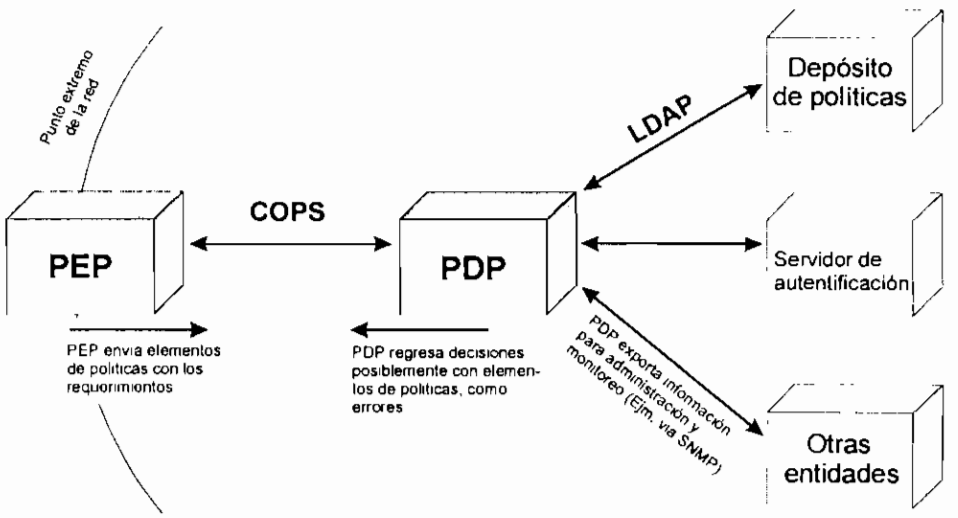

Figura 7.5 Modelo del Sistema de Políticas. [56]

- Ejecutar Decisiones: El PEP debe aplicar las acciones decididas por el PDP.

- Hacer Mediciones: Las mediciones son necesarias para verificar el buen estado de la red, es decir si se están cumpliendo las políticas y si los usuarios no están abusando de los recursos de la red. Para realizar esta función, el PEP es el encargado de efectuar las mediciones, mientras que el PDP se encarga de determinar si éstas (mediciones) indican el buen estado de la red.

El PDP tiene la capacidad de interactuar con otras entidades por medio del Protocolo Simple de Administración de Red (SNMP) con el objetivo de intercambiar información de monitoreo y administración de red.

7.6 PROTOCOLO COMÚN DE SERVICIO DE POLÍTICAS COPS

7.6.1 INTRODUCCIÓN

El Protocolo Común de Servicio de Políticas COPS (*Common Open Policy Service*) permite intercambiar información entre los servidores de políticas (PDPs) y sus clientes (PEPs). El cliente de políticas (PEP) puede ser un router que realiza control de admisión basándose en políticas, mientras el servidor de políticas (PDP) corresponde en sí a un servidor y se requiere la existencia de por lo menos uno dentro de cada dominio administrativo.

Las características del protocolo son:

- Utiliza el sencillo modelo cliente – servidor, en el cual el cliente o PEP envía al servidor o PDP requerimientos de inicio, actualización o finalización del estado del protocolo ante los cuales el PDP retorna decisiones aceptando o negando el cambio de estado.

- Utiliza el Protocolo TCP para una comunicación confiable entre el cliente (PEP) y el servidor (PDP).

- Soporta diversa información específica del cliente de políticas, sin requerir modificación al protocolo en sí, pues fue diseñado para la administración y configuración general de un sistema de políticas, lo cual lo hace extensible a varias implementaciones.

- Permite realizar autentificación segura, chequea integridad del mensaje y ofrece protección de repetición de un mensaje; además permite el uso del protocolo IPSec.

- Se establece un estado de requerimiento – decisión entre el cliente y el servidor, respectivamente, manteniéndose hasta que sea borrado explícitamente.

- Permite que el servidor (PDP) coloque en el cliente (PEP) información de configuración, la cual será retirada cuando no sea necesaria.

7.6.2 MODELO Y FUNCIONAMIENTO

El modelo básico de funcionamiento del protocolo COPS se grafica en la figura 7.6, en la cual se ratifica que la comunicación entre PEP y PDP es la tarea relevante del protocolo.

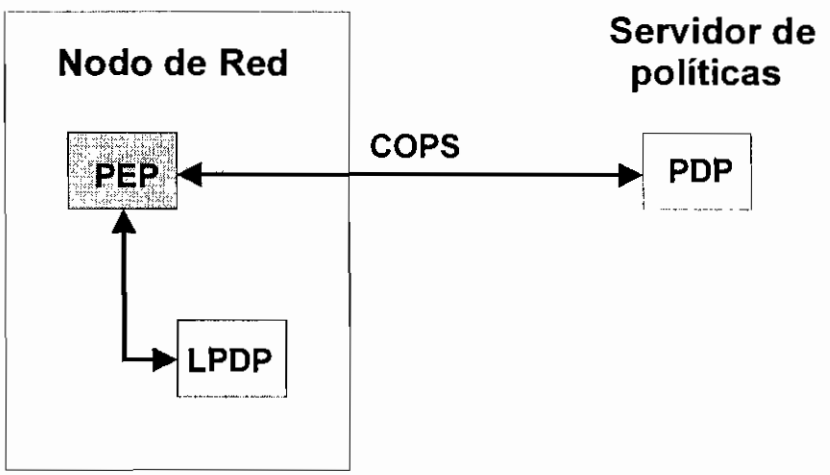

Figura 7.6 Modelo Básico del Protocolo COPS. [59]

El PEP puede soportar y, por tanto, tiene conocimiento de varios tipos de aplicaciones de clientes de políticas, convirtiéndolo en el indicado para inicializar la comunicación. Además tiene la capacidad de mantener varias conexiones abiertas hacia varios PDPs con el objetivo de cubrir todos los tipos de aplicaciones de clientes.

Cuando el PEP recibe un evento que necesita decisión, inicia una conexión TCP hacia el PDP utilizando por lo general el puerto 3288 [59]. Esta conexión instala un estado de requerimiento – aceptación en el PEP y PDP identificado por el puerto

TCP y el cliente de políticas específico, el cual será eliminado por el PEP cuando no sea necesario.

Para iniciar la conexión, el PEP envía a cada PDP un mensaje de requerimiento de servicio por cada tipo de cliente de políticas soportado, adicionalmente puede contener la dirección IP del último PDP utilizado, si se empleó previamente uno.

En respuesta el PDP envía un mensaje de aceptación por cada tipo de cliente que soporte, en el cual se especifica el valor del *Timer KA* que representa el intervalo de tiempo elegido entre dos mensajes de refresco sucesivos. Las siglas KA se derivan del inglés *Keep Alive* que significa "mantener vivo". Si el PDP no soporta el tipo de cliente especificado termina el requerimiento actual y lo direcciona hacia un servidor PDP de respaldo.

El PEP tiene la opción de solicitar del PDP información de configuración que luego de instalada controlará una interfaz, un módulo o una funcionalidad del PEP. El PDP actualiza la configuración mediante el envío de mensajes de refresco con la configuración especificada; si desea eliminar la configuración envía mensajes de terminación utilizando el objeto *FLAGS*, descrito mas adelante.

Como se determina en la figura 7.6, el PEP puede utilizar los servicios de un PDP local (LPDP, *Local Policies Decisión Point*) con el fin de aumentar la velocidad de la negociación; el LPDP además permite escalabilidad del sistema y la continuidad del servicio de políticas si el enlace entre el PEP y el PDP remoto está caído.

A pesar de la existencia del LPDP, la autoridad máxima sigue siendo el PDP remoto, del cual se obtiene la autorización de uso y la información de configuración para el LDAP. De esta forma, el PDP siempre tiene conocimiento del estado del PEP asociado pudiendo enviar mensajes de actualización del estado; el PEP tiene la obligación de notificar cualquier cambio en su estado, incluso después de utilizar el LPDP. El PEP reporta al PDP del estado de la configuración mediante el objeto *REPORT*, el cual además sirve para transportar información del cliente del servicio con el fin de realizar la contabilidad del consumo o el monitoreo del servicio.

Para indicar si una conexión entre el PEP y el PDP está funcionando se utiliza el mensaje "mantener vivo" (KA, *Keep Alive*). El PEP genera un mensaje KA a un intervalo de tiempo aleatorio comprendido entre ¼ y ¾ del tiempo especificado en el *Timer* KA, en respuesta el PDP genera un mensaje similar hacia el PEP; si ninguno de los dos recibe el mensaje KA generado por el otro durante el tiempo especificado por el *Timer* KA, se considera que la conexión se ha perdido.

Cuando se pierde la conexión debido a la inactividad explicada en el párrafo anterior, el PEP genera un mensaje de cierre para indicar error en la conexión y prosigue a buscar un PDP secundario o intenta reconectarse al PDP actual.

Una conexión TCP entre el PDP y el PEP finaliza cuando el primero sale de funcionamiento, para ello, el PDP envía mensajes de cierre a todos sus clientes PEPs conectados y tiene la opción de especificar la dirección IP de un PDP de respaldo. Como se mencionó anteriormente, una conexión también se cierra cuando el PDP genera un mensaje de cierre en respuesta a un requerimiento de un cliente de políticas no soportado.

Debido a que el protocolo COPS maneja información de administración y seguridad, debe tener tolerancia a las fallas en la conexión. La conexión se monitorea constantemente mediante el envío de mensajes "mantener vivo", si se detecta una desconexión, el PEP intentará inmediatamente reconectarse al mismo PDP o a uno de respaldo.

En caso de desconexión, el PEP puede utilizar el LPDP con el objetivo de mantener una completa funcionalidad del protocolo después que se ha producido la falla y antes que se logre la reconexión. El protocolo permite recobrar el PDP cuando se producen fallas transitorias mediante el uso del objeto *LastPDPAdd* que contiene la dirección IP del último PDP utilizado. Después de superada la falla se debe sincronizar nuevamente y de igual forma notificar todos los cambios al PDP.

Todas las implementaciones del protocolo COPS deben ser capaces de proveer autentificación segura, integridad del mensaje y evitar la repetición de los mensajes, utilizándose con este fin el objeto de integridad *INTEGRITY*.

Corresponde al PEP iniciar la negociación de seguridad en el primer mensaje de requerimiento que transmite al PDP.

La característica de protocolo seguro requiere la utilización de claves conocidas por el PEP y el PDP, las cuales combinando con el contenido del mensaje se calcula el objeto *INTEGRITY* por medio del cual se determina la validez de un mensaje y de esta forma se logra integridad del mensaje y autentificación segura. Se recomienda que las claves sean cambiadas periódicamente y, si es posible, tener una para cada cliente PEP con el objetivo de proteger la red de un cliente defectuoso.

Para evitar la repetición de un mensaje, los objetos Integridad tienen un número de secuencia que se incrementa a medida que se transmiten los mensajes. El número de secuencia para el PDP lo determina el PEP, y viceversa, al inicio de la negociación de la seguridad.

Si en algún mensaje se repite el número de secuencia del objeto Integridad o si éste llega con alteraciones, automáticamente se termina la conexión actual.

7.6.3 MENSAJES Y OBJETOS

Similar al protocolo RSVP, los mensajes del protocolo COPS están compuestos de varios objetos que determinan la característica del mensaje. A continuación se hará un estudio de los mensajes y objetos que utiliza el protocolo COPS. [59]

7.6.4 MENSAJES

7.6.4.1 Requerimiento (REQ, *Request*)

Permite al PEP iniciar el estado de requerimiento o realizar modificaciones a un estado previamente establecido. Este mensaje asigna un objeto "*handle*" usado para identificar los requerimientos subsecuentes.

7.6.4.2 Decisión (DEC, *Decision*)

Este mensaje lo genera el PDP en respuesta a un mensaje REQ y sirve para emitir una decisión o actualización del estado de requerimiento. No

necesariamente debe ocurrir un mensaje REQ para que se genere un DEC ya que el PDP tiene la capacidad de emitir decisiones aún sin haber sido solicitadas por el PEP. Este mensaje puede contener objetos de error si ocurre alguno.

7.6.4.3 Reporte del Estado (RPT, *Report State*)

Posibilita que el PEP reporte al PDP el éxito, la falla o la modificación producida en un estado de requerimiento. Es utilizado para proveer información de contabilidad a intervalos regulares.

7.6.4.4 Eliminación del Estado de Requerimiento (DRQ, *Delete Request State*)

Este mensaje lo genera el PEP cuando desea indicar que el PDP remoto, y su estado correspondiente, ya no está disponible o ya no es útil.

7.6.4.5 Requerimiento de Estado de Sincronismo (SSQ, *Synchronize State Request*)

Este mensaje indica que el PDP remoto desea reenviar su estado al cliente. El cliente realiza la sincronización del estado emitiendo consultas de requerimiento del estado existente en el PEP para el tipo de cliente especificado.

7.6.4.6 Abrir Cliente (OPN, *Client-Open*)

Este mensaje es usado por el PEP para especificar el tipo de clientes que soporta, el último PDP al cual el PEP estuvo conectado para un determinado tipo de cliente y/o las características de negociación específicas del cliente.

7.6.4.7 Aceptar Cliente (CAT, *Client-Accept*)

Este mensaje es utilizado por el PDP para responder afirmativamente a un mensaje Abrir Cliente, contiene el valor asignado al *Timer* KA que indica el intervalo de tiempo entre mensajes KA.

7.6.4.8 Cerrar Cliente (CC, *Client-Close*)

Este objeto es utilizado por el PEP o PDP para indicar que no soporta el tipo de cliente especificado. Contiene además la razón por la cual se cerró el requerimiento y tiene la opción de indicar la dirección IP de un PDP alterno.

7.6.4.9 Mantener Vivo (KA, *Keep Alive*)

Este mensaje indica que la conexión está activa y debe ser enviado por PEP con la frecuencia indicada por el *Timer* KA.

7.6.4.10 Estado de Sincronismo Completo (SSC, Synchronize State Complete)

Este mensaje lo envía el PEP hacia el PDP, en respuesta a un mensaje de Sincronismo del Estado de Requerimiento, luego que ha finalizado la sincronización.

7.6.5 OBJETOS

7.6.5.1 Indicador (*Handle*)

Este objeto encapsula un valor que identifica a un estado instalado. Se incluye en la mayoría de operaciones del protocolo COPS, su administración le corresponde al PEP, el cual lo transmite dentro de los mensajes de Requerimiento, Reporte y Eliminación de estado.

7.6.5.2 Contexto (*Context*)

Especifica el tipo de evento que genera la consulta, el cual puede ser control de admisión, asignación de recursos o solicitud de información de configuración. Se utiliza en los mensajes de requerimiento que genera el PEP hacia el PDP.

7.6.5.3 Interfaz de Entrada (*IN-Int*)

Es utilizado para identificar la interfaz de entrada de un requerimiento en particular y la dirección IP del nodo que lo originó. La interfaz depende del protocolo de capa 2 que se utilice.

7.6.5.4 Interfaz de Salida (*OUT-Int*)

Es utilizado para identificar la interfaz de salida de un requerimiento y la dirección IP del nodo destino.

7.6.5.5 Razón (*Reason*)

Es utilizado en los mensajes de Eliminación de Requerimiento (DRQ) e indica la razón por la cual se terminó una sesión. Entre las principales causas se cuentan: decisión administrativa, inactividad, cambio de ruta y recursos insuficientes.

7.6.5.6 Decisión (*Decision*)

Contiene la decisión tomada por el PDP la cual puede ser instalar, remover o mantener un estado.

7.6.5.7 Decisión LDPD (LPDP *Decision*)

Transporta la decisión efectuada por el PDP local, esta puede ser, igual que en el objeto anterior, instalar, remover o mantener un estado.

7.6.5.8 Error (*Error*)

Especifica el error que se ha producido en el protocolo COPS. Algunas causas de errores son: indicador defectuoso, formato de mensaje defectuoso, tipo de cliente no soportado, falla de comunicación, entre otros.

7.6.5.9 Información Específica del Cliente (*ClientSI*)

Contiene información específica del cliente útil para realizar requerimientos y reportes.

7.6.5.10 *Timer "Keep Alive"* (KA *Timer*)

Especifica el máximo intervalo de tiempo para enviar o recibir mensajes del protocolo.

7.6.5.11 Identificación del PEP (PEPID)

Transporta la identificación del cliente PEP utilizada por el PDP.

7.6.5.12 Tipo de Reporte (*Report-Type*)

Este objeto es utilizado por el PEP para reportar al PDP si la decisión tomada se implementó con éxito o si existió alguna falla en el proceso.

7.6.5.13 Dirección del PDP Redireccionado (*PDP Redirect*)

Indica al PEP la dirección IP y el puerto TCP de un PDP alterno (respaldo). Este objeto puede utilizarlo opcionalmente el PDP cuando se dispone a cerrar una sesión con el PEP.

7.6.5.14 Dirección del último PDP (*LastPDPAddr*)

Especifica la dirección IP del último PDP utilizado exitosamente por el PEP. Sirve para conmutar rápidamente entre PDP cuando ha existido alguna falla.

7.6.5.15 Timer de Contabilidad (*AccTimer*)

Es utilizado por el PDP para enviar al PEP mensajes de decisión no solicitados; especifica el tiempo del intervalo mínimo entre mensajes periódicos.

7.6.6 UTILIZACIÓN DEL PROTOCOLO COPS EN UNA RESERVACIÓN RSVP

El funcionamiento del protocolo COPS es muy variado, para ilustrarlo se detalla una aplicación específica cuando se lo utiliza en una reservación RSVP. Los pasos a seguirse cuando se desea hacer una reservación con la ayuda de COPS son los siguientes (obsérvese la figura 7.7):

1. El *router* 1 se registra con el servidor de políticas y le comunica cuales políticas puede hacer cumplir.

2. El cliente RSVP hace un requerimiento de reservación de ancho de banda a través del camino.

3. El *router* 1 consulta a su control de admisión para determinar si el requerimiento puede ser cumplido.

4. El *router* 1 (con LPDP) toma una decisión local de políticas para permitir la reservación.

5. El LPDP registra en el PDP el flujo, le comunica la decisión y mantiene el estado.

6. El *router* 2 (sin LPDP) consulta el control de admisión para determinar si el requerimiento puede ser cumplido.

7. El *router* 2 registra en el PDP el flujo y pide que tome una decisión de políticas.

8. El *servidor* COPS chequea la información de políticas y toma una decisión.

9. El *router* 2 recibe la decisión y permite la reservación.

10. Los mecanismos de organización de tráfico aplican la reservación para el flujo especificado, la cual está sujeta a cambios por parte del servidor COPS.

11. Cuando el flujo es liberado, el PEP informa lo sucedido al servidor COPS.

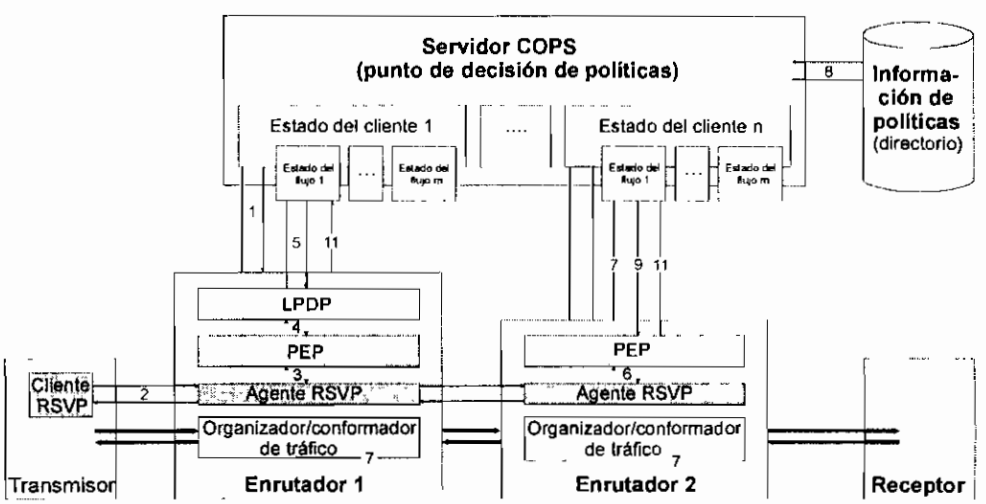

Figura 7.7 Transacción COPS en una negociación RSVP. [6]

Se observa que un modelo general de administración de tráfico con calidad de servicio tiene los siguientes componentes:

- Diferenciación del tráfico que cursa por la red:
  - Reservación explícita de recursos para un flujo mediante la utilización de protocolos como RSVP, ATM QoS, entre otros.
  - Diferenciación explícita mediante el marcaje de los paquetes, por ejemplo con IPTOS y DiffServ.
  - Configuración preferencial a cargo del administrador, el cual puede crear listas de acceso en cada router.
- Decisión de cada router sobre la forma de manipular el tráfico:
  - Decisión local, mediante algoritmos de manipulación de paquetes.
  - Decisión tomada por alguna autoridad de la red, por ejemplo mediante servidores COPS o *RADIUS*.
  - Decisión tomada basándose en un conjunto de reglas preconfiguradas por el administrador de la red, quien hace asociaciones estáticas entre los flujos y mecanismos que lo manipulan.
- Manipulación del tráfico en cada router:
  - Algoritmos de encolamiento.
  - Algoritmos de descarte de paquetes.
  - Etc.

7.7 CONSIDERACIONES ADICIONALES

Como se mencionó, un servidor de políticas permite proveer a una red de diferentes niveles de servicio a diversas clases de usuarios y aplicaciones,

dependiendo de la disponibilidad de recursos tales como ancho de banda y prontitud.

A continuación se dan ejemplos prácticos de políticas implementadas en una red, que pueden aplicarse tanto en el Internet como en una intranet privada. Se hace una diferenciación entre políticas estáticas y dinámicas.

Ejemplos de políticas estáticas:

- Permitir o negar el acceso de un grupo de usuarios en algún momento del día.
- Otorgar el más alto nivel de prioridad a aplicaciones particulares, por ejemplo a una videoconferencia gerencial.
- Asignar a un usuario una dirección de correo eléctrico única, independiente del lugar donde se encuentre.
- Realizar asignaciones explícitas de ancho de banda, por ejemplo entregar el 25% del ancho de banda total al departamento de contabilidad de la empresa.

Ejemplos de políticas dinámicas [1]:

- Negar el uso de algún recurso cuando se ha alcanzado cierto umbral de utilización, por ejemplo, negar el acceso web si la utilización de ancho de banda excede el 50%.
- Establecer umbrales al recurso asignado a cada entidad, y negar el servicio si se sobrepasa este umbral.

[1] Políticas dinámicas.- son aquellas que se invocan bajo ciertas características de la red.

Capítulo 8

MONITOREO

CAPÍTULO 8

MONITOREO

8.1 INTRODUCCIÓN

Una vez que se habilita QoS en una Red mediante algunas de las Arquitecturas estudiadas anteriormente (para mayor información remítase a los capítulos 4, 5 y 6) es necesario e imprescindible desarrollar y conocer técnicas que permitan comprobar cómo está trabajando esa red.

8.2 PARÁMETROS A MONITOREAR

Existe un amplio número de parámetros útiles para caracterizar el comportamiento de un servicio, y que por lo tanto deben monitorearse continuamente.

Uno de los conjuntos de métricas y de parámetros muy bien estructurados son los que maneja la tecnología ATM, debido a que ésta fue implementada basándose en el trabajo de un comité como lo es el *ATM Forum* en lugar de estudios aislados y de la ausencia de un código de trabajo, como la filosofía del IETF.

Muchas de las métricas y de los parámetros ATM son útiles para caracterizar una red. En muchos ambientes es común encontrar todavía el núcleo LAN basado en ATM, debido a que era la única forma de conseguir hace algunos años atrás 155 Mbps [10], limitación que hoy en día se ha resuelto con Gigabit Ethernet.

La caracterización fundamental de un servicio es mediante el conocimiento de la latencia, el *jitter*, la confiabilidad, la capacidad de absorber ráfagas y el volumen del tráfico. Al conocer para un servicio cada uno de estos parámetros, se podrá anticipar el comportamiento de la red.

En la tabla 8.1, se presenta una asociación de estos parámetros para diversos protocolos orientados a conexión.

Parámetro	TCP	Frame Realy	ATM	IntServ
Latencia	RTT		CTD	ADSPEC
Jitter	Desviación estándar de RTT		CDV	ADSPEC
Elección de Descarte	bits TOS	bit DE	bit CLP	Manipulación *best effort*
Ráfagas	Tamaño actual de la ventana	CIR excedido	PCR	*Flowspec*
Tasa de Error	Número de ACKs recibidos por el último segmento transmitido		CER	

Tabla 8.1 Parámetros de Servicio para varios protocolos. [6]

8.2.1 MONITOREO DEL RETARDO O LATENCIA

La latencia es uno de los parámetros de mayor importancia en el tránsito de los paquetes a través de una red. La idea más simple de una latencia está relacionada con la habilidad que posee el servidor para responder. A fin de cuantificar una verdadera latencia, se suele implementar en la red tres clases de pruebas [6].

PRUEBA 1:

PING: una prueba de este tipo mide la latencia de la red (considerando la ida y vuelta de un paquete, RTT *Round-Trip Time*) , además de dar muestras claras de la habilidad del servidor de responder al mencionado *PING*. El funcionamiento detallado de esta prueba se presenta en el numeral 8.5.1 de este capítulo.

PRUEBA 2:

Prueba TCP: esta prueba incluye la configuración de una sesión TCP y la medición de la respuesta de tiempo, y otorga resultados específicos sobre el proceso realizado en el servidor.

PRUEBA 3:

Prueba de Aplicación: esta prueba indica el verdadero comportamiento (comportamiento final) del servicio.

8.2.2 MONITOREO DEL JITTER

El *Jitter* o la variación del retardo es un parámetro muy importante para aplicaciones en tiempo real. Existen varias propuestas para medir el *Jitter*, una de ellas proviene del ITU (*International Telecommunication Union*) y requiere la inyección de paquetes a intervalos regulares de tiempo para medir, luego, la variabilidad en los tiempos de arribo. De esta forma el *jitter* corresponde al Rango Intercuartil IQR[1] (*Inter Quartile Range*) de la distribución de frecuencias de las mediciones del tiempo de respuesta.

Un segundo método para el cálculo del *Jitter* proviene del IETF (*Internet Engineering Task Force*), el mismo es: si R_i es el tiempo de respuesta obtenido en el envío del paquete *i*, entonces el *jitter* corresponde al Rango Intercuartil de la distribución de frecuencias de dR_i, donde $dR_i = R_i - R_{i-1}$. Ejemplos ilustrativos de estos dos métodos se encuentran más adelante en el numeral 8.5.1.2.

Altos niveles de *jitter* tienden a indicar fluctuación en la profundidad de las colas, lo cual indica un tratamiento de tráfico muy pobre en puntos de congestión de la red, que para ser controlado pueden implementarse los algoritmos de control de congestión analizados en el capítulo 3.

En tanto que un nivel bajo de *jitter* para una clase particular de tráfico es una muestra clara de la implementación de una diferenciación o tratamiento especial al tráfico en mención, debiendo tomarse muy en cuenta cuando estos valores sean sumamente pequeños, ya que pueden provocar que otras clases de tráfico sean servidas con valores de *jitter* muy exagerados y por lo tanto su desempeño sea pésimo.

---

[1] IQR: Medida estadística que corresponde al rango residual entre el 25% y 75% dentro del cual se ubican los datos, pues se considera que solo la mitad de los datos afectan la medida.

8.3 MONITOREO EN DIFERENTES PROTOCOLOS

Los distintos protocolos utilizados en las redes de transmisión de datos suelen utilizar cada uno de ellos diferentes tipos de métricas para tener conocimiento de los valores de parámetros como el retardo, el porcentaje de paquetes perdidos, la velocidad de datos, etc. En la siguiente sección se analiza las métricas o formas que utilizan cada uno de los protocolos a fin de controlar los parámetros antes mencionados.

8.3.1 PROTOCOLO TCP

Utiliza el descarte de paquetes y el deterioro de la respuesta de tiempo, para detectar congestión. Una vez que ésta es detectada el protocolo reacciona reduciendo la ventana de transmisión de datos, es decir reduciendo el número de paquetes circulantes entre origen y destino para esa sesión TCP.

Las métricas que caracterizan una sesión TCP son: el tamaño actual de la ventana de transmisión, un promedio de paquetes perdidos y el tiempo RTT.

8.3.2 PROTOCOLOS BASADOS EN RESERVACIONES

Los sistemas que basan su accionamiento en reservación de recursos, tal como el protocolo RSVP, mantienen un acuerdo centralizado sobre la capacidad existente como también una lista de reservaciones de flujos actuales mediante el control de admisión y de las negociaciones PEP/PDP, explicadas en el capítulo anterior.

Así por ejemplo para una red que habilite Servicios Integrados mediante el Protocolo de Reservación de Recursos (RSVP) y que su administración esté centralizada, tendrá en el servidor COPS el dispositivo en el cual se encuentre la mayor cantidad de información sobre el estado de Calidad de Servicio de la mencionada red.

8.3.3 PROTOCOLO ATM

ATM incluye un gran número de métricas que son utilizadas para caracterizar el servicio que presta a cierta clase de tráfico y que suelen ser acordadas entre el

proveedor y el usuario de la red. Este tipo de métricas depende de la clase de servicio que se haya contratado.

"A fin de hacer posible la realización de contratos de tráfico concretos, el estándar ATM define una serie de parámetros de Calidad de Servicio cuyos valores los pueden negociar el cliente y la portadora. Los tres primeros parámetros especifican la rapidez a la que quiere transmitir el usuario.

- Tasa pico de celdas (PCR, *Peak Cell Rate)*: es la rapidez máxima con que el transmisor planea enviar sus celdas. Este parámetro puede ser menor que lo permitido por el ancho de banda de la línea. Si el transmisor planea sacar celdas cada 4 µseg, su PCR es de 250 000 celdas/seg, aún si el tiempo real de transmisión de las celdas puede ser 2.7 µseg.

- Tasa sustentable de celdas (SCR, *Sustained Cell Rate)*: es la tasa esperada o requerida de celdas promediada en un intervalo de tiempo grande. La razón PCR/SCR es una medida de las ráfagas de tráfico.

- Tasa mínima de celdas (MCR, *Minimum Cell Rate*): es la tasa mínima de celdas/seg que el cliente considera aceptable.

- Tolerancia de variación de retardo de celdas (CVDT, *Cell Variation Delay Tolerance*): indica la cantidad de variación que habrá en los tiempos de transmisión de las celdas; se especifica en forma independiente tanto para la velocidad PCR como para SCR.

Las características de la red, cuantificables en el receptor son descritas por los siguientes tres parámetros, mismos que son negociables.

- Razón de pérdida de celdas (CLR, *Cell Loss Ratio*): mide la fracción de las celdas transmitidas que no se entregan en absoluto o que se entregan demasiado tarde, siendo por lo tanto inservibles para servicios en tiempo real, por ejemplo.

- Retardo de transferencia de celdas (CTD, *Cell Transfer Delay*): es el tiempo promedio de tránsito de las celdas desde el origen al destino.

- Variación en el retardo de celdas (CDV, *Cell Delay Variation*): es un parámetro que mide la uniformidad con que se entregan las celdas.

Normalmente el cliente y la portadora acuerdan establecer un valor de CTD, es decir establecen la tardanza con la que puede entregarse una celda y aún considerarla correctamente entregada. El valor de CLR en cambio se escoge de tal manera que la fracción de celdas rechazadas por llegar demasiado tarde, sea del orden de 10^{-10} o menos.

Los tres últimos parámetros QoS especifican características de la red y generalmente no son negociables:

- Razón de Errores de Celdas (CER, *Cell Error Ratio*): Es la fracción de las celdas que se entregan con uno o más bits equivocados.

- Tasa de bloques de celdas con errores severos (SECBR, *Severely-Errored Cell Block Ratio*): Es la fracción de bloques de N celdas en los que M o más celdas contienen errores.

- Tasa de mala inserción de celdas (CMR *Cell Missinsertion*): es la cantidad de celdas/seg que se entregan al destino equivocado debido a un error no detectado en la cabecera" [10].

8.4 ACUERDOS DE NIVEL DE SERVICIO (SLAs, *Service Level Agreements*)

Los acuerdos de nivel de servicio (SLAs) son contratos celebrados entre el cliente y su proveedor de servicio, mediante el cual se especifica el tratamiento que recibirá de la red el tráfico del cliente. Contienen un conjunto de parámetros que definen por un lado, las condiciones que debe cumplir el tráfico del cliente, y por otro los recursos que asignará la red del proveedor de servicios si se cumplen las condiciones. En la actualidad los SLAs son firmados especificando valores acordados de parámetros tales como disponibilidad, latencia, niveles de ráfagas, *jitter*.

Una organización (cliente) que contrata un servicio mediante un SLA desea tener siempre la certeza de que el ISP está sirviendo el tráfico de sus aplicaciones bajo los parámetros acordados, para ello muchas organizaciones utilizan promedios de múltiples *PINGs*. Siendo cada día más el número de empresas que utilizan estadísticas de respuesta de tiempo generadas por los *PINGs* para medir el rendimiento de sus aplicaciones, tal que el promedio de empresas que recurren a esas estadísticas se ha incrementado en un 100% entre Diciembre de 1998 y Junio de 1999. [6]

Además es obligación de los proveedores de servicio, mantener disponibles documentos en los cuales se detallen aspectos como los siguientes:

- Interpretación de los reportes y de sus estadísticas.
- Técnica utilizada para la recolección de datos.
- Recomendaciones para optimizar la red a fin de limitar lo siguiente:
  - Inversión de capital (*router*s, FRADs, etc.).
  - Costos de transmisión recurrentes.
  - Optimización del ancho de banda, realizando la identificación de usuarios no autorizados o de aplicaciones que estén monopolizando los recursos.
- Indicar que el servicio se degradará por alguna razón, antes de que esto afecte a los usuarios.

Para ilustrar un Acuerdo de Nivel de Servicio y la forma de determinarlo considérese el siguiente ejemplo, en el cual se oferta un servicio de video *MPEG2* libre de *jitter*:

> "... libre de *jitter* implica la necesidad de eliminar completamente las variaciones de los tiempos de envío entre los paquetes de video - variaciones del retardo de propagación. Sin embargo, el proveedor

de servicios puede estimar un tamaño razonable para los *buffers* de recepción, los cuales disminuyen el *jitter* y permiten alguna flexibilidad. Ya que la aplicación es un flujo unidireccional de video continuo, puede tolerar el retardo de reproducción (*playback*) resultante, a diferencia de una conversación bidireccional.

La calificación de video *MPEG2* caracteriza una imagen con resolución y velocidad de imagen de alta calidad, lo cual indica necesidad de gran ancho de banda. Sin embargo, no especifica un tamaño de imagen, lo que significa que no se define exactamente el requerimiento de ancho de banda. Se asume calidad de televisión digital DTV, resolución de 576x720 y una velocidad de imagen de 60 imágenes por segundo, lo cual genera un flujo de video comprimido de alrededor de 4 Mbps. Así, se requiere un *buffer* de recepción de alrededor de 16 Mb para permitir 4 segundos de *jitter*.

Estos puntos ilustran la necesidad de políticas en un Acuerdo de Nivel de Servicio bilateral que se traduzca, sin ambigüedades, en acciones específicas –en este caso, los parámetros del tráfico de la red son cuantificables- y tal vez incluyan requerimientos del usuario final como el tamaño del *buffer*. Se ha identificado los siguientes valores para una red de Servicios Integrados:

Tipo de Servicio:	Carga Controlada.
Máxima tasa de transmisión por flujo:	4000 kbps.
Máxima tasa de transmisión pico por flujo:	8000 kbps.
Máximo *token bucket* [2] por flujo:	64 kb.
Retardo mínimo:	2000 milisegundos.
Máxima duración del flujo:	180 segundos." [56]

[2] *Token Bucket*: parámetro definido en los Servicios Integrados (IETF) que cuantifica la característica de ráfagas de la fuente y que sirve para dimensionar los *buffers* de recepción.

En el ejemplo indicado, se determinan los parámetros principales, sin embargo, un ejemplo más completo incluirá el costo y el tiempo permitido para el servicio.

8.5 MONITOREO CON HERRAMIENTAS TRADICIONALES

8.5.1 PING (Packet InterNet Groper)

El *PING* es una de las herramientas más utilizadas para monitorear y detectar errores de comunicación en una red IP, esta utilidad generalmente está preinstalada en casi todas las plataformas. El *PING* puede ser utilizado para medir el tiempo de respuesta, el porcentaje de pérdida de paquetes, la variación del tiempo de respuesta y puede determinar también si un dispositivo de red es alcanzable para algún otro dispositivo de red.

Su nombre se deriva del inglés *Packet InterNet Groper* que significa paquete explorador de Internet, ello se debe a que en una red IP, el programa *PING* hace que un dispositivo de red o *host* origen envíe un paquete de datos pequeño hacia un dispositivo de red destino y luego espere en respuesta otro paquete de similares características.

El *PING* se implementa utilizando el Protocolo de Mensajes de Control de Internet (ICMP, *Internet Control Message Protocol*), el cual a su vez utiliza el soporte básico del Protocolo Internet IP como si fuera un protocolo de nivel superior, sin embargo, ICMP es parte integral de IP y debe ser implementado en cada módulo IP.

Los mensajes del protocolo ICMP sirven para notificar algún inconveniente en la comunicación de la red y no para hacer de IP un protocolo confiable. Se han definido 11 tipos de mensajes especificados en el RFC 792, sin embargo los más útiles para el *PING* son los mensajes Eco, *Timestamp* y Destino Inalcanzable cuyo formato y descripción se explica en la sección 8.5.1.4.

8.5.1.1 Funcionamiento

Como se mencionó anteriormente, el *PING* permite identificar problemas en el envío de los paquetes tales como pérdida, duplicación o arribo en desorden, para

ello el *PING* utiliza un número secuencial en cada paquete que se envía, y lo chequea posteriormente a la llegada del paquete correspondiente. Sin embargo que el *PING* puede detectar duplicación, retardo o daño de un paquete no especifica el lugar donde ocurre a pesar que ello se puede deducir.

El *PING* también permite detectar si los datos que contiene el paquete arriban con errores, para ello chequea la suma uno complemento de las palabras de 16 bits en que se divide el paquete, este valor se almacena en el campo *Checksum* (suma de comprobación) correspondiente (ver Formato de Mensajes ICMP - Mensaje Eco).

Un dato muy importante que proporciona el *PING* es el tiempo de respuesta, este tiempo se contabiliza desde que se transmite el paquete Eco hasta que se recibe el paquete Respuesta al Eco, es decir incluye la transmisión, propagación y procesamiento del paquete en un viaje de ida y vuelta (ver Formato de Mensajes ICMP – *Timestamp*).

Se puede observar que el menor tiempo de respuesta obtenido será aproximadamente igual al tiempo de viaje (ida y vuelta) RTT (*Round Trip Time*) de un paquete. El tiempo de respuesta es importante pues permite identificar el retardo y su fluctuación, parámetros fundamentales que caracterizan un enlace.

8.5.1.2 Ejemplo de sesiones *PING*

Para ilustrar la utilización de la utilidad *PING* considérese la red de acceso de un ISP, Ramtelecom, cuyos servidores en Ecuador están ubicados en Quito los cuales se enlazan con sus servidores en Miami (Estados Unidos) a través de un enlace satelital asimétrico; finalmente, se accede al Internet a través de una última milla de fibra óptica que los comunica con el ISP de primer nivel UUNET, cuya red de comunicaciones se encuentra en la figura 8.2. La red de acceso de Ramtelecom se muestra en la figura 8.1.

Dos puntos interesantes de la red de acceso de Ramtelecom (ver figura 8.1) son su red de área local en Quito (63.71.198.0) y la red de área local en Miami (216.219.62.0). Realizando un *PING* desde el PC (*Personal Computer*) de un

usuario *dial-up* hacia los servidores de Ramtelecom en Quito se puede obtener datos que caractericen el enlace *dial up* de 28.8 kbps; efectuándolo hacia los servidores en Miami se puede obtener parámetros que describan el enlace satelital.

Para la realización de las pruebas se utiliza el programa de utilidades de Internet WS_*PING*ProPack de Ipswitch disponible en http://www.ipswitch.com; se efectúan desde un computador personal con procesador Intel Pentium II de 350 MHz, 64 MB de memoria RAM (*Random Access Memory*) y sistema operativo Windows 98.

La primera prueba consiste en la realización de *PING*s desde el computador de un usuario hacia el servidor de acceso a la red NAS (*Network Access Server*) con dirección IP 63.71.198.3, los resultados de la prueba se presentan en la figura 8.3.

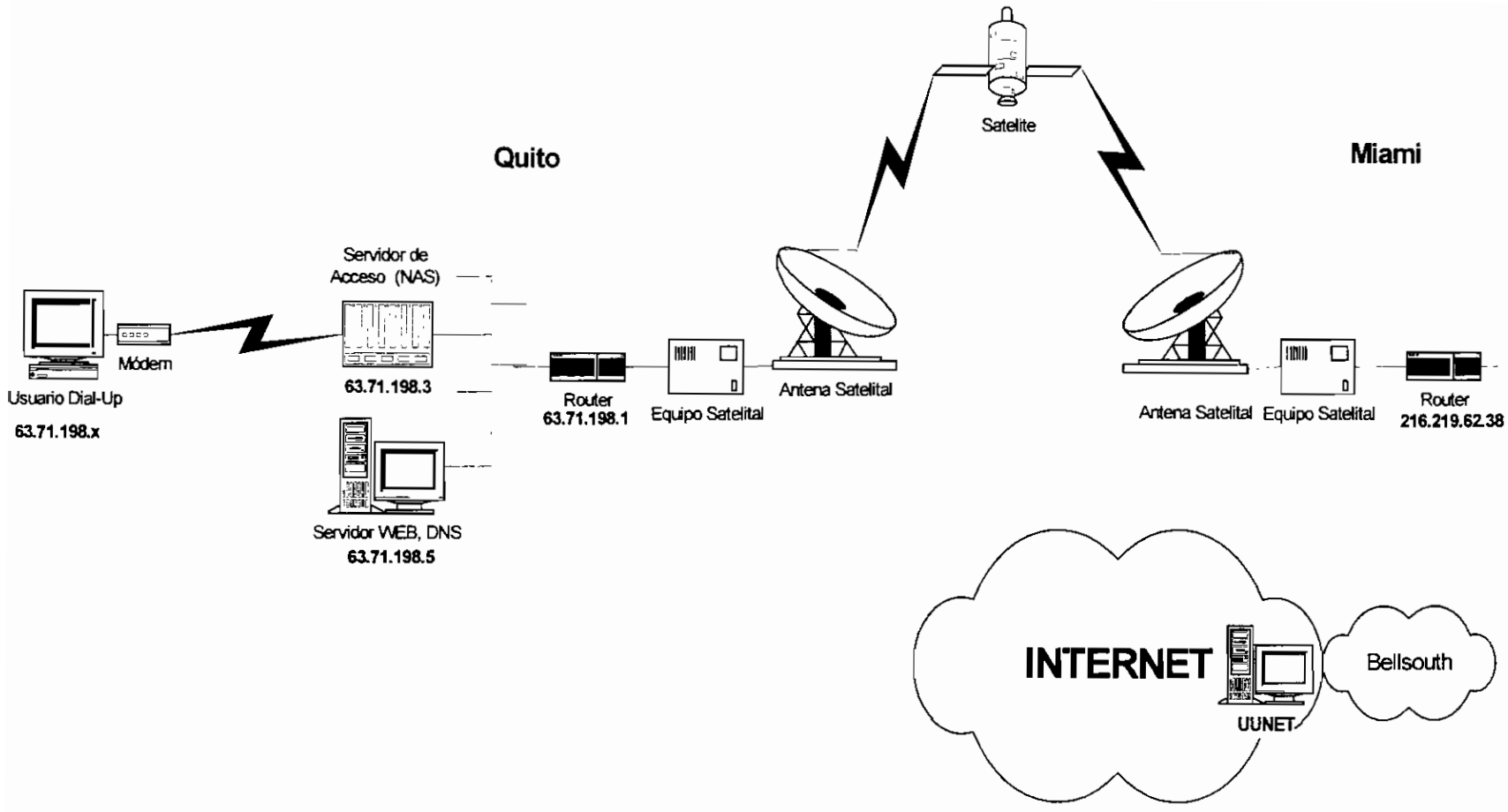

Figura 8.1 Red de Acceso de Ramtelecom

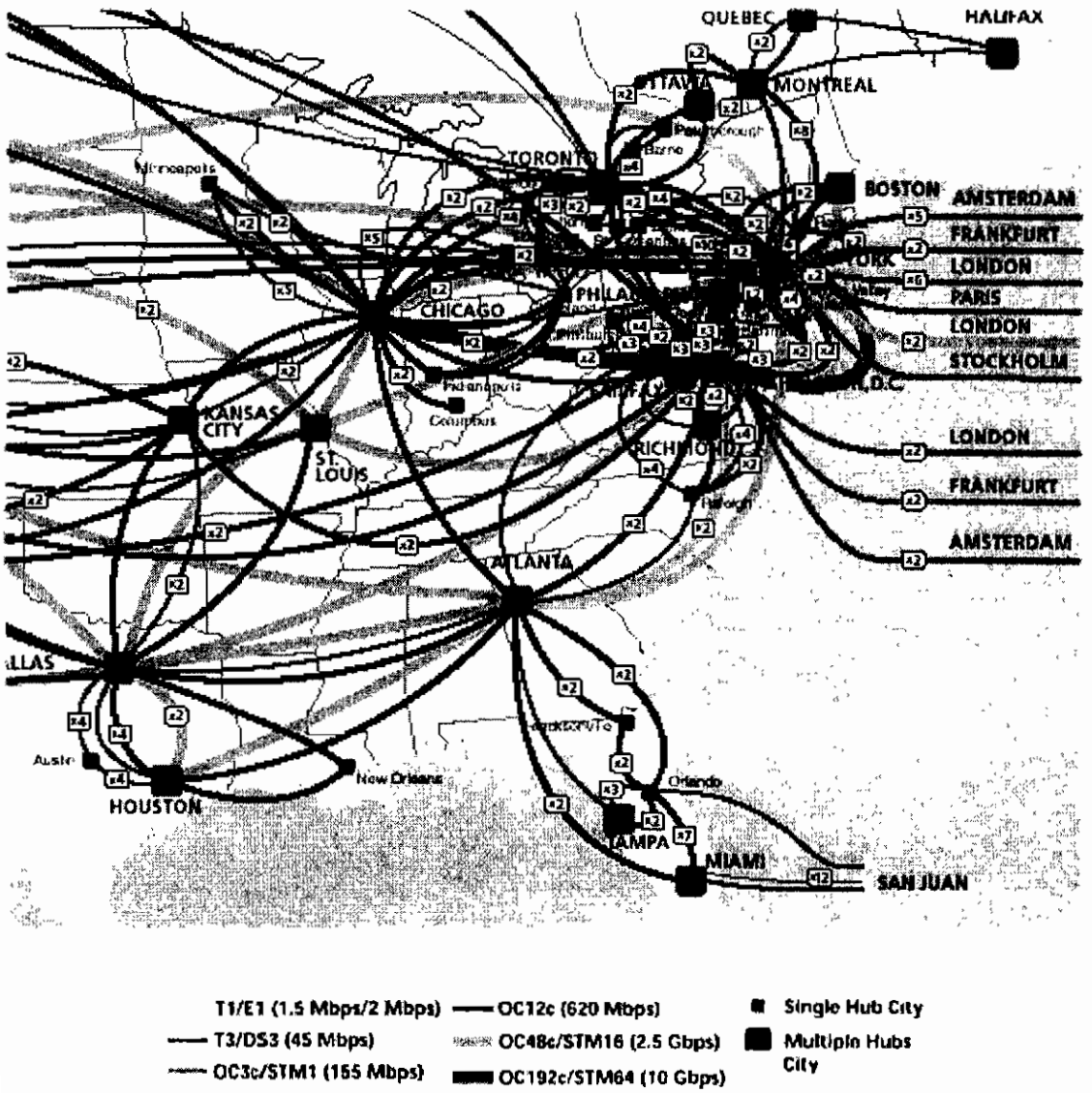

Figura 8.2 Red de UUNET en Norte América. [60]

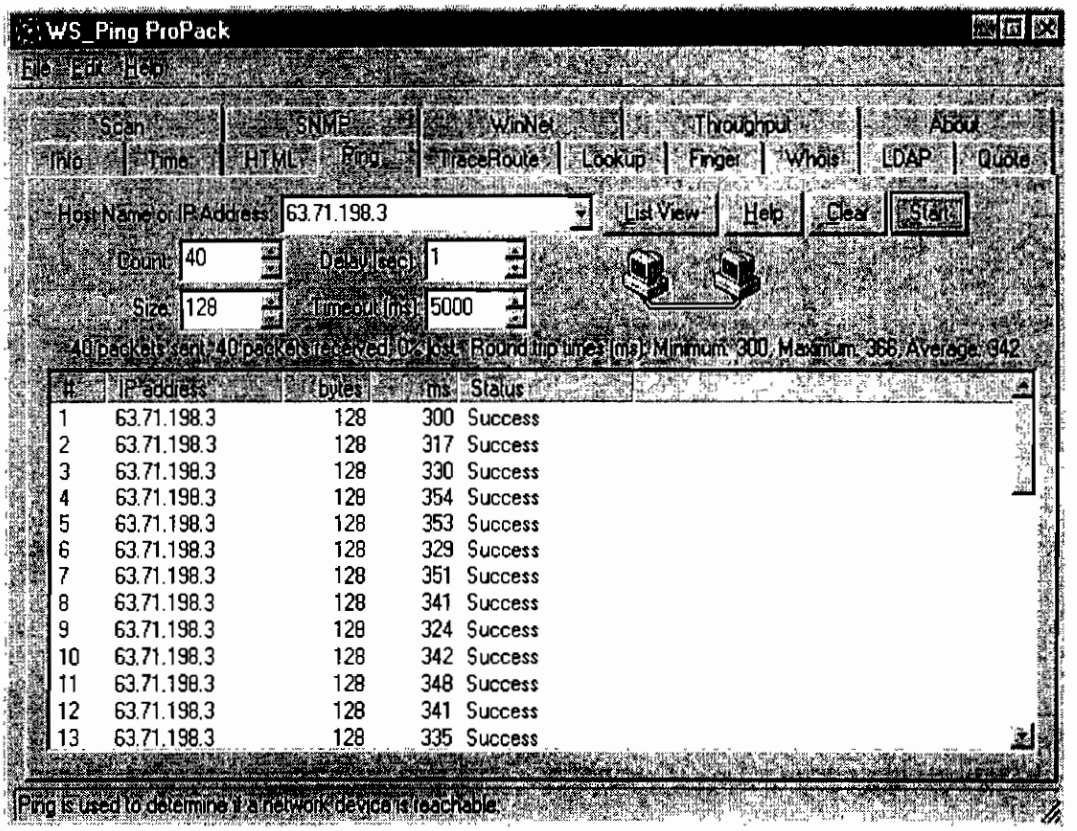

Figura 8.3 *PING* hacia el servidor de acceso de Ramtelecom.

Luego del establecimiento de la conexión a una tasa de transmisión de 28.8 kbps, se envían 40 paquetes ICMP de 128 bytes de tamaño, es decir un paquete IP de 148 bytes, incluidos los 20 bytes de la cabecera; se eligió este tamaño pues es el correspondiente a un típico paquete de voz sobre IP. Luego de realizadas las pruebas se obtiene un valor de retardo promedio de 342 milisegundos, el cual es alrededor de 100 milisegundos mayor que el retardo teórico definido en el capítulo 2 (sección 2.1.1.3). Además se observa que el retardo mínimo es 300 milisegundos y el máximo 366 milisegundos.

La segunda prueba consiste en realizar *PING*s hacia el servidor de Ramtelecom en Miami desde un usuario *dial-up*, los resultados de este experimento se detallan en la figura 8.4

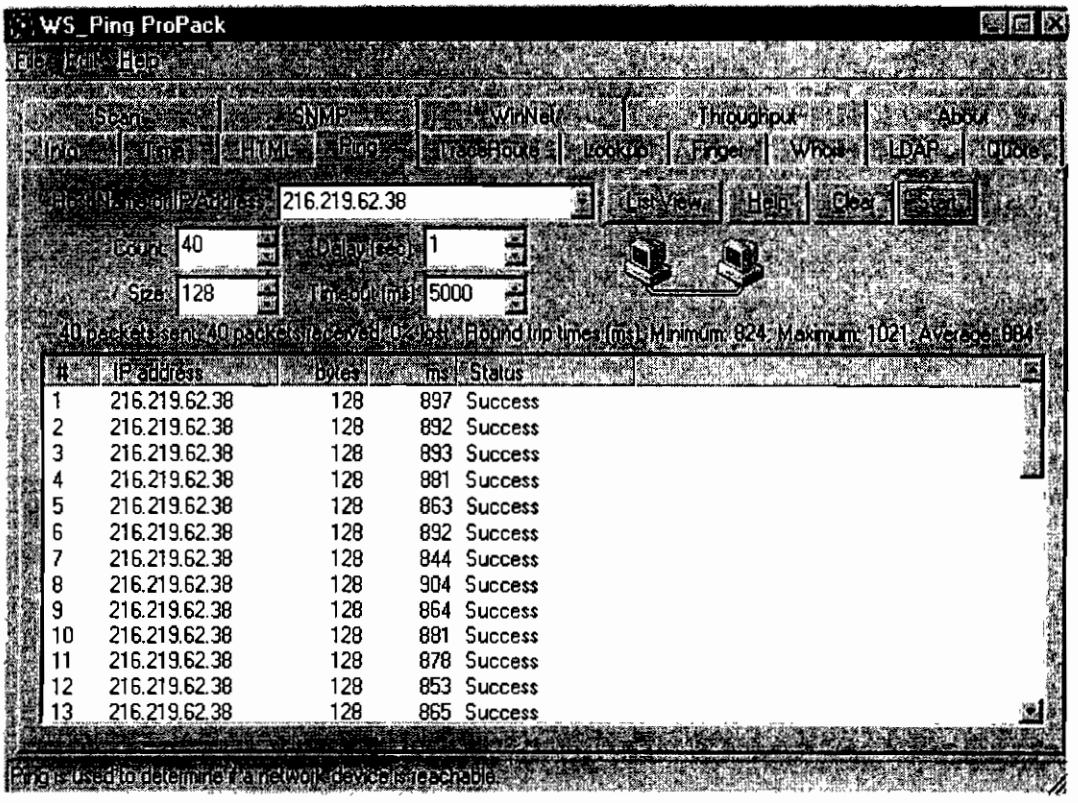

Figura 8.4 *PING* hacia un servidor de Ramtelecom en Miami

Al igual que la anterior, en esta prueba se enviaron 40 paquetes de 128 bytes ICMP (148 bytes, incluida la cabecera IP), los cuales atravesaron dos enlaces, el primero un enlace *dial-up* de 28.8 kbps entre el usuario y los servidores de Ramtelecom en Quito, y el segundo, un enlace satelital asimétrico entre los servidores de Ramtelecom ubicados en Quito y Miami.

Luego de efectuada las prueba se obtuvo un retardo promedio de 884 milisegundos, y los valores de retardo mínimo y máximo de 824 y 1021 milisegundos respectivamente.

A partir de los datos obtenidos se puede deducir el retardo en enlaces intermedios, por ejemplo el retardo en el enlace satelital corresponde a la diferencia entre el retardo hacia Miami y el retardo hacia Quito, es decir entre 884 y 342 milisegundos, obteniéndose un retardo de 542 milisegundos justificándose plenamente su valor, ya que se utiliza un satélite geoestacionario, como se demuestra en la ecuación 8.1.

$$retardo_{satelital} = 2 * \left(2 * \frac{36000Km}{300000Km/s} \right) = 0.48s = 448ms \qquad (8.1)$$

1	336	
2	310	-26
3	324	14
4	329	5
5	299	-30
6	342	43
7	329	-13
8	324	-5
9	312	-12
.....	.....	.....
990	345	
991	342	-3
992	341	-1
993	317	-24
994	335	18
995	350	15
996	317	-33
997	346	29
998	333	-13
999	324	-9
1000	317	-7

Tabla 8.2 Tiempos de respuestas

Para el cálculo del *Jitter* se considera el enlace *dial-up* hacia Ramtelecom con un tamaño de paquete ICMP de 128 bytes, pero en este caso se envían 1000 paquetes para obtener un valor estadístico más exacto, específicamente el correspondiente al Rango Intercuartil. La tabla 8.2 muestra algunos de los valores obtenidos.

Para calcular el valor del *jitter* de la forma sugerida por la ITU se considera la distribución de frecuencias de los tiempos de respuesta, es decir los correspondientes a la segunda columna de la tabla 8.2, obteniéndose un *jitter* o rango intercuartil (IQR) de 20 milisegundos. La distribución de frecuencias de estos datos se muestra en la figura 8.5.

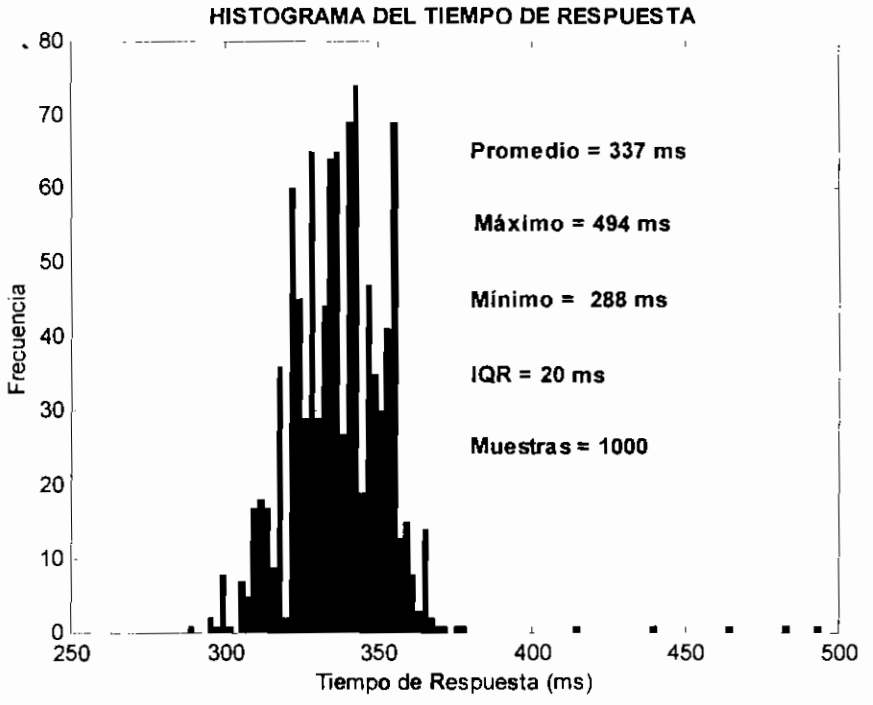

Figura 8.5 Histograma de los Tiempos de Respuesta.

Utilizando el método propuesto por el IETF el *jitter* corresponde al rango intercuartil de la distribución de frecuencias de la diferencia de los datos obtenidos, es decir la tercera columna de la tabla 8.2. El valor calculado para el *jitter* es de 25 milisegundos, la distribución de frecuencias se aprecia en la figura 8.6.

En todas las pruebas anteriores todos los paquetes ICMP llegaron nuevamente al origen, es decir hubo una pérdida del 0%.

Analizando los resultados obtenidos se concluye que el enlace de acceso a Internet de Ramtelecom presenta buenas características de retardo y *jitter* para efectuar comunicaciones interactivas tales como la telefonía a través de IP, sin embargo se debe tomar en cuenta que si en el momento de efectuarse la llamada sobre IP se ejecuta, desde el mismo computador, una sesión FTP, la calidad de la llamada decaerá notablemente.

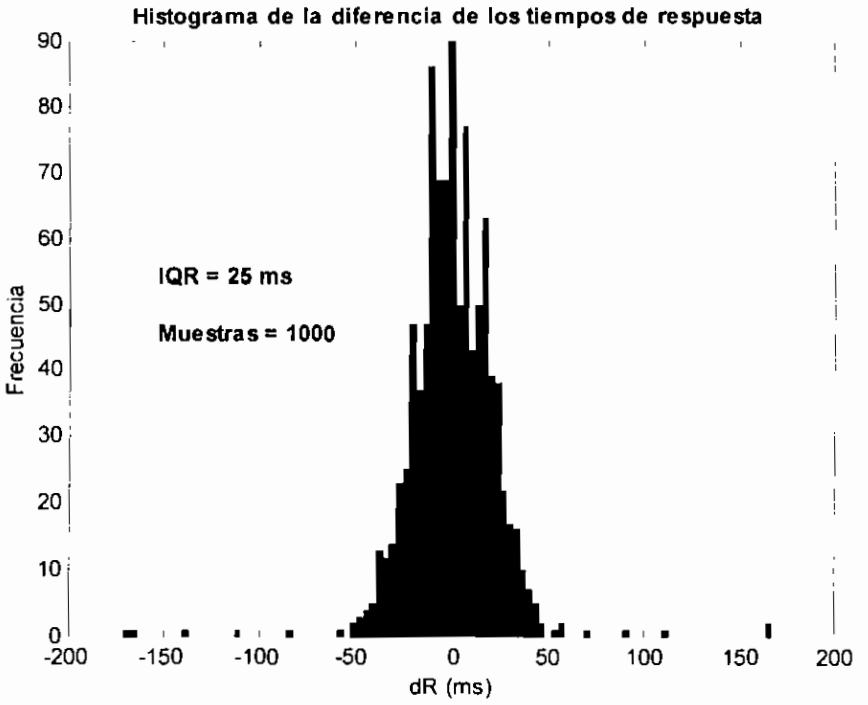

Figura 8.6 Histograma de la diferencia de los Tiempos de Respuesta

8.5.1.3 *PING* Y QoS

Al igual que los paquetes IP comunes, los mensajes ICMP pueden utilizar el byte TOS (*Type of Service*) de la cabecera IP para identificar los datos que requieren algún nivel de calidad de servicio; es más, la mayoría de sistemas operativos soportan los valores TOS en el comando *PING*.

Esta importante ayuda de enviar mensajes ICMP con algún valor TOS debe manipularse con cuidado para no interpretar los resultados en forma inadecuada, por ejemplo cuando un *router* está configurado para tratar los mensajes ICMP como mensajes de control de red, éste tiene la capacidad de alterar el valor TOS y así se puede obtener excelentes tiempos de respuesta para el *PING* cuando en realidad el tráfico referido está experimentando congestión.

8.5.1.4 Formato de Mensajes ICMP

8.5.1.4.1 Mensaje Eco

El mensaje Eco del protocolo ICMP está conformado por la cabecera IP básica y un cuerpo de longitud variable, como se observa en la figura 8.7.

CABECERA IP

Versión (Vers) [4 bits]: Indica la versión del Protocolo Internet, la referida es la versión 4.

Longitud de la cabecera Internet (IHL) [4 bits]: Especifica la longitud de la cabecera IP en palabras de 32 bits.

Tipo de Servicio (TOS) [8 bits]: Permite asignar al paquete un nivel de tipo de servicio.

Identificador de fragmento [16 bits]: Identifica el grupo al cual pertenece este fragmento (si existió fragmentación).

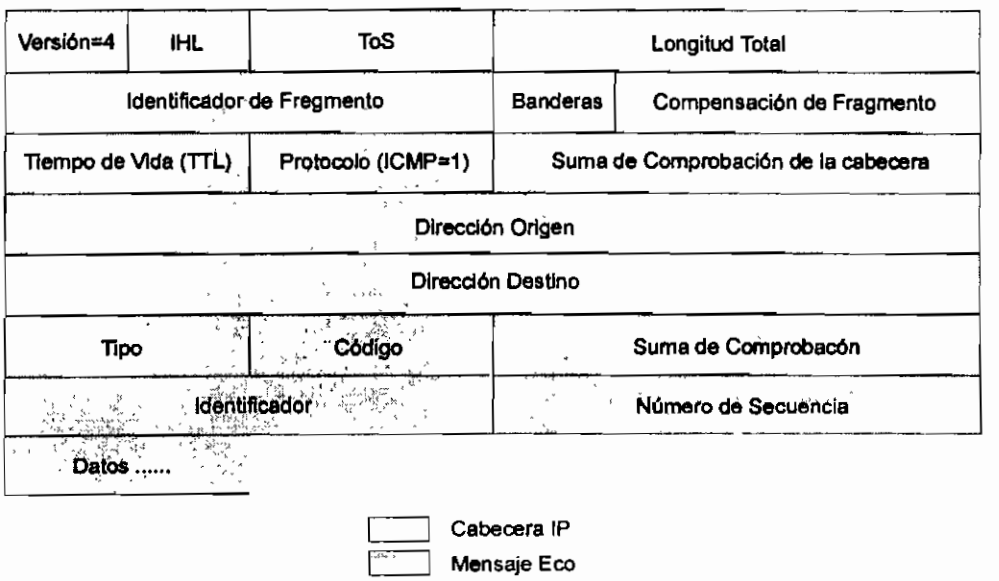

Figura 8.7 Formato del paquete *PING.* [61]

Banderas [4 bits]: Asigna banderas cuando se ha realizado la operación de fragmentación.

Compensación de Fragmento [12 bits]: Identifica la posición relativa del fragmento dentro del paquete original.

Tiempo de Vida [8 bits]: Especifica el tiempo de vida del paquete IP en segundos; como este campo se decrementa en cada dispositivo en el cual se procesa el datagrama, su valor debe ser por lo menos tan grande como el número de dispositivos de red por los cuales ha de atravesar el datagrama.

Protocolo [8 bits]: Indica el protocolo al cual pertenecen los datos que encapsula el datagrama IP en su carga útil; en este caso, su valor es 1 y corresponde al protocolo ICMP.

Suma de Comprobación de la cabecera [16 bits]: Es una palabra de 16 bits que corresponde al uno complemento de la suma uno complemento de todas las palabras de 16 bits de la cabecera IP. Para el cálculo, el valor inicial que toma este campo es cero.

Dirección Origen [32 bits]: Indica la dirección del dispositivo de red o *host* que genera el mensaje ICMP.

Dirección Destino [32 bits]: Indica la dirección del dispositivo de red o *host* hacia el cual se dirige el mensaje ICMP.

ECO Y RESPUESTA AL ECO

Cuando un dispositivo de red envía un mensaje Eco hacia un dispositivo de red destino, en respuesta espera un mensaje Respuesta al Eco, en el cual los campos correspondientes a las direcciones IP origen y destino simplemente se intercambian; además, el campo Tipo cambia su valor a 0 y se recalcula el campo suma de comprobación.

Los campos del cuerpo de éstos mensajes son:

Tipo [8 bits]: Especifica el tipo del mensaje; si es un mensaje Eco su valor será 8 y si es un mensaje Respuesta al Eco su valor será 0.

Código [8 bits]: El campo código no se utiliza en este tipo de mensajes ICMP, su valor simplemente es 0.

Suma de Comprobación [16 bits]: Este campo es una palabra de 16 bits utilizada para detección de errores, corresponde al uno complemento de la suma uno complemento del mensaje ICMP (inicia en el campo Tipo). Para el cálculo de este campo su valor inicial debe ser cero. Si la longitud total del mensaje es impar (el tamaño del mensaje puede ser un número de bits múltiplo de 8 pero no siempre de 16) se debe rellenar la última palabra con un octeto de ceros.

Identificador [16 bits]: Este campo puede tomar el valor de 0 pero también puede ser utilizado como un puerto TCP o UDP para identificar una sesión.

Número de Secuencia [16 bits]: Este campo puede tomar el valor de 0 pero también puede ser incrementado en cada mensaje Eco si se envía un conjunto de paquetes. El valor que tome este campo en el mensaje Eco debe ser mantenido en el mensaje Respuesta al Eco correspondiente.

Datos [variable]: La longitud y valor que tomen los datos son variables, simplemente se debe cumplir la condición de mantener los datos enviados en el mensaje Eco en su correspondiente mensaje Respuesta al Eco.

8.5.1.4.2 Timestamp y Respuesta al Timestamp

El formato del mensaje *Timestamp* es similar al anterior, la única diferencia es la adición de tres campos a continuación del campo correspondiente al número de secuencia; los tres campos que se añaden se indican en la figura 8.8.

El dispositivo de red o *host* que origina el mensaje coloca un *Timestamp*, mientras el dispositivo de red o *host* que recibe el mensaje mantiene los datos y añade otro *Timestamp*. Los tres campos que forman el *Timestamp* son palabras de 32 bits

cuyo valor corresponde a la Hora Universal (UT, *Universal Time*)[3] medida en milisegundos. Si la Hora Universal no está disponible, se puede insertar algún otro tiempo pero se debe indicar que su valor no es estándar haciendo que el bit más significativo sea uno.

Figura 8.8 Mensaje *Timestamp*. [61]

Los campos que definen el Timestamp son:

Timestamp Original [32 bits]: Indica el instante en el cual el transmisor manipula el mensaje por última vez antes de transmitirlo.

Timestamp de Recepción [32 bits]: Indica el instante en el cual el receptor manipula por primera vez el paquete cuando este ya ha sido recibido.

Timestamp de Transmisión [32 bits]: Indica el instante en el cual el receptor manipula el mensaje por última vez antes de enviarlo de vuelta.

El dispositivo de red que origina el mensaje llena estos tres campos con el mismo valor correspondiente al campo *Timestamp* Original (t_1); el dispositivo de red que recibe el mensaje llena los campos *Timestamp* de Recepción (t_2) y *Timestamp* de Transmisión (t_3); finalmente, cuando el mensaje es devuelto a su origen, el transmisor toma en cuenta el tiempo de llegada del paquete (t_4).

El tiempo de respuesta presentado en el *PING* se calcula a partir de la ecuación 8.2, este tiempo incluye el tiempo de procesamiento en el host destino.

$$t_{resp} = t_4 - t_1 \qquad (8.2)$$

[3] Hora Universal UT (*Universal Time*): Hora oficial del Reino Unido, formalmente llamada Hora del Meridiano de Greenwich.

El tiempo correspondiente al viaje de ida y vuelta RTT del paquete no incluye el tiempo de procesamiento en alguno de los dos dispositivos de red involucrados, el mismo es calculado a partir de la ecuación 8.3 en la cual el primer término corresponde al tiempo de transmisión y propagación del mensaje desde el origen al destino, y el segundo término representa el tiempo de propagación y transmisión del mensaje desde el destino hacia el origen.

$$RTT = (t_2 - t_1) + (t_4 - t_3) \qquad (8.3)$$

8.5.1.4.3 Mensaje Destino Inalcanzable

Este mensaje es enviado por un *router* hacia la fuente de un datagrama cuyo destino no puede ser localizado pues no existe la información pertinente en la tabla de enrutamiento del *router*. Este mensaje también puede ser generado a pesar que el datagrama haya llegado a su destino, este caso se produce cuando el módulo IP del host destino no puede enviar el datagrama de regreso debido a que no están activos el protocolo o el puerto especificado.

El formato del mensaje se ilustra en la figura 8.9, se debe considerar que a éste se debe añadir la cabecera IP básica especificada en la figura 8.7.

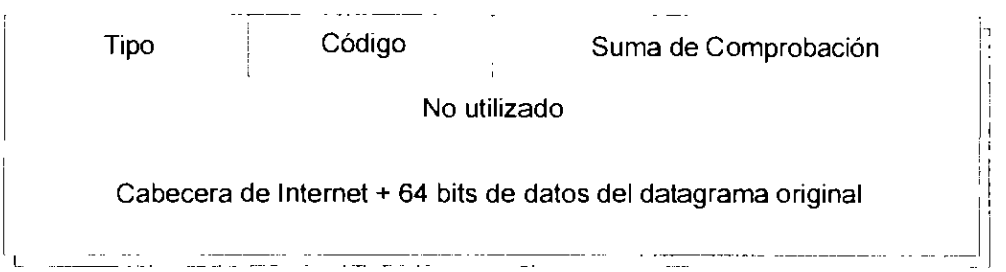

Figura 8.9 Formato del mensaje Destino Inalcanzable. [61]

Tipo [8 bits]: Especifica el tipo de mensaje ICMP, en el caso del mensaje Destino Inalcanzable su valor es 3.

Código [8 bits]: El campo Código indica la razón por la cual se generó el mensaje, las cuales pueden ser:

0 = Red inalcanzable;

1 = Host inalcanzable;

2 = Protocolo inalcanzable;

3 = Puerto inalcanzable;

4 = Se pide que no se fragmente el paquete pero éste debe ser fragmentado.

5 = Falla en la fuente de información para el enrutamiento.

Suma de comprobación [16 bits]: Se calcula en forma idéntica al campo *Checksum* del mensaje Eco.

El campo final contiene la cabecera IP más los primeros 64 bits de los datos del datagrama original. Estos datos utiliza el host origen para relacionar el mensaje con los procesos adecuados, por ejemplo, si el protocolo de la capa superior utiliza puertos, éstos se encontrarán dentro de los 64 bits. [61]

8.5.1.4.4 Mensaje Tiempo Cumplido

Este mensaje lo puede originar un *router* o un *host* cuando el campo del Tiempo de Vida del datagrama es cero; luego de descartar el datagrama. Este mensaje también se puede generar si se cumple el tiempo al momento de reensamblar el paquete debido a la pérdida de algún fragmento.

El formato de este mensaje es similar al correspondiente al mensaje Destino *Inalcanzable*, las únicas diferencias se producen en los campos Tipo y Código. El primer campo toma el valor de 11, mientras el segundo toma los valores de 0 o 1 si se ha excedido el tiempo de vida del paquete antes que llegue a su destino o si se excede el tiempo de reensamblaje.

8.5.2 TRAZADO DE RUTA (*TRACEROUTE*)

Otra herramienta poderosa para monitorear la conducta de la red es el *TRACEROUTE* o Trazado de Ruta, la cual permite conocer el número de saltos existente entre dos dispositivos de red y las características de la ruta.

Para su funcionamiento, el *TRACEROUTE* utiliza el campo correspondiente al tiempo de vida (TTL, *Time To Live*) del paquete IP, el cual indica

aproximadamente el número de saltos (*routers*) que cumple el paquete antes de expirar, y algunos mensajes del Protocolo ICMP (*Internet Control Messages Protocol*).

8.5.2.1 Funcionamiento

El *TRACEROUTE* direcciona los paquetes hacia un puerto UDP extraño de la máquina destino, el puerto UDP predeterminado es el 33434 [62]. Usualmente se envían a la vez grupos de tres paquetes para obtener varios valores de tiempos de respuesta.

Para los tres paquetes iniciales, se configura el campo TTL igual a 1 y se envía los paquetes hacia el destino, luego que el paquete ha atravesado el primer *router* (completando el primer salto), el campo TTL decrementa su valor desde 1 a 0. Al observar el campo TTL igual a 0, el *router* descarta el paquete y envía de vuelta hacia el host origen el mensaje de notificación ICMP correspondiente, indicando que se ha alcanzado el primer salto y el tiempo que ello tomó.

Se repite este procedimiento incrementando gradualmente los valores del campo TTL hasta que se alcance el host destino y se reciba en el origen un mensaje ICMP de puerto inalcanzable; ésta es la razón por la cual se utiliza un puerto extraño, pues ello provoca la respuesta del paquete en lugar de que sea absorbido por algún servicio aleatorio.

8.5.2.2 Ejemplo de la utilización de *TRACEROUTE*

Para ilustrar el uso del *TRACEROUTE* se presenta a continuación el trazado de la ruta desde un computador de un usuario con conexión dial-up suscrito al proveedor de Internet Ramtelecom hacia uno de los servidores de Net2Phone, empresa proveedora de telefonía sobre IP.

De acuerdo a la figura se puede observar que la ruta entre Ramtelecom y Net2Phone comprende 12 saltos, es decir existen 12 dispositivos de red por los cuales deben atravesar los paquetes correspondientes a una sesión establecida entre los dos sitios.

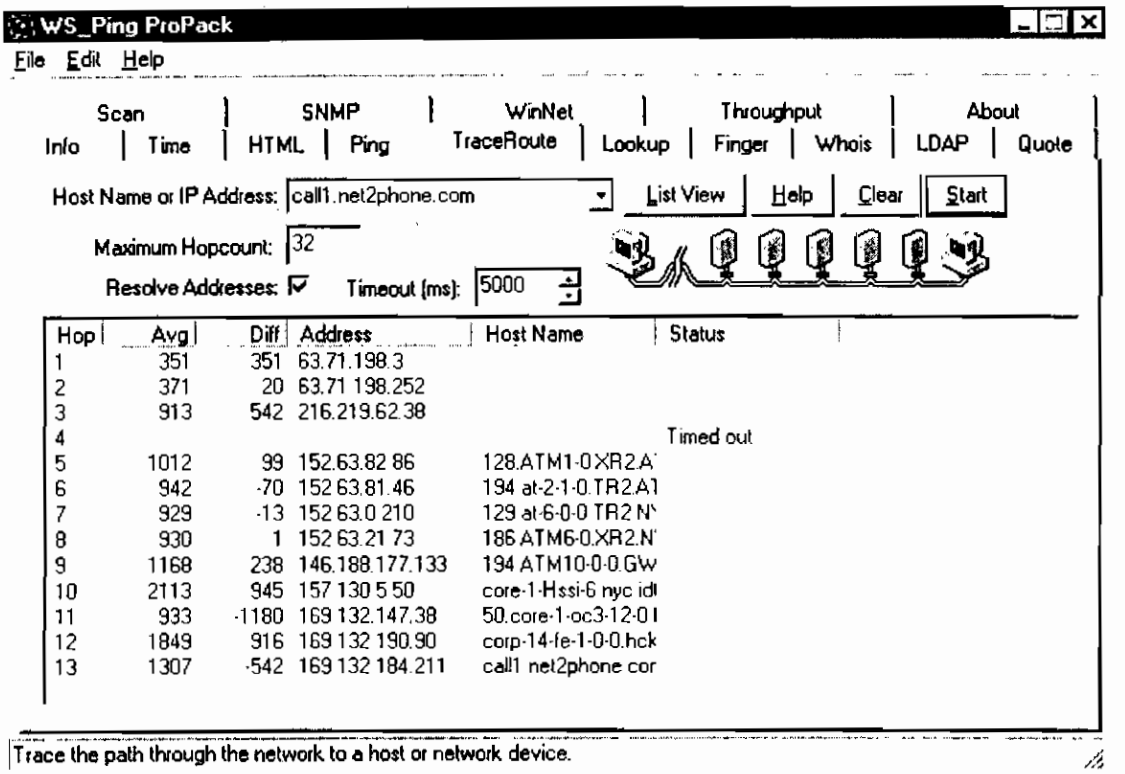

Figura 8.10 *TRACEROUTE* desde Ramtelecom hacia Net2Phone

8.5.2.3 *TRACEROUTE* Y QoS

El trazado de la ruta es importante, especialmente cuando se desea dar un tratamiento preferencial enrutando diferentes tipos de tráfico a través de diferentes rutas. Se debe considerar que las rutas a través de la red pueden cambiar debido a la tendencia creciente de seleccionar la ruta basándose en estados dinámicos del enlace tales como su rendimiento y su carga.

La dificultad en este caso radica en la señalización hacia la red para que trate los paquetes del *TRACEROUTE* de acuerdo a alguna clasificación. Si se pudiera crear un *TRACEROUTE* que establezca la precedencia IPTOS u otra configuración que especifique el tipo de servicio, entonces sería fácil conocer la ruta asignada a cierta clase de tráfico; sin embargo, debido al funcionamiento actual del *TRACEROUTE*, se necesita hacer ciertas mejoras en la infraestructura de red.

Conclusiones y Recomendaciones

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

- El aumento del número de usuarios y el aparecimiento de aplicaciones con altos requerimientos de recursos, hacen necesaria la implementación de una infraestructura de Calidad de Servicio en el Internet, tal que le permita entregar niveles de servicio adecuados a cada entidad (usuarios o aplicaciones) que lo soliciten; tratamiento que no podría realizarse con el actual modelo de servicio *best effort*.

- El objetivo básico de una red con Calidad de Servicio es permitir la diferenciación del tráfico de determinados usuarios o aplicaciones otorgándoles mejores características de rendimiento que el resto del tráfico. Este tratamiento preferencial implica además una diferenciación en cuanto al costo del servicio, siendo éste mayor mientras su desempeño se aleje más del básico servicio *best effort*.

- Implementar Calidad de Servicio en una red implica la administración efectiva de sus recursos mediante la utilización de mecanismos de gestión de tráfico, tales como WFQ, CBQ, RED, ECN, entre otros.

- Los parámetros que definen la Calidad de Servicio en Internet son: latencia o retardo, *jitter*, ancho de banda y confiabilidad. Para aplicaciones en tiempo real, las cuales día a día están ganando más espacio en Internet, los más críticos son el retardo y el *jitter*.

- Existen varias fuentes importantes de retardo. Una de ellas es el retardo de propagación presente en el enlace satelital, inevitable en países como el nuestro, en el cual la mayoría de proveedores acceden a Internet por medio de un enlace satelital hacia Miami. Otra fuente importante de retardo es el encolamiento producido en los dispositivos de red en condiciones de congestión. Finalmente, los módems analógicos tradicionales, presentes en la mayoría de hogares a nivel

mundial y casi en la totalidad en Ecuador, son la tercera causa importante de retardo.

- Se han desarrollado dos arquitecturas básicas para la implementación de calidad de servicio en una red: La Arquitectura de Servicios Diferenciados y la Arquitectura de Servicios Integrados. La primera, es una solución QoS de muy buenas características de escalabilidad, pero muy débil en cuanto a garantías de servicio extremo a extremo, debido a la sencillez de su modelo. La segunda, es una solución de QoS que garantiza un nivel de servicio extremo a extremo en base a un modelo de reservación de recursos, sin embargo, su implementación en los núcleos de gran capacidad genera serios problemas de escalabilidad.

- Las dos arquitecturas básicas pueden combinarse en una sola Arquitectura Híbrida *IntServ/DiffServ*, a fin de obtener un entorno extremo a extremo escalable y de buenas características de garantía de servicio.

- Las Arquitecturas *DiffServ* e *IntServ* permiten la configuración de los mecanismos de gestión de tráfico, que son los que realmente implementan un tratamiento diferenciado. La implementación particular depende de la capa enlace que se utilice, siendo una de las más robustas ATM, pues dispone de parámetros y características adecuadas para interpretar los servicios definidos por las arquitecturas.

- La implementación de las Arquitecturas de Calidad de Servicio en Internet implica realizar cambios tanto en los dispositivos de red actuales como en las formas de administración de la red. Esto no quiere decir que para su implementación se deba efectuar un cambio drástico en el Internet actual, sino por el contrario, cada proveedor de servicios podría implementar QoS en su dominio administrativo con el fin de ofrecer un servicio diferenciado a sus clientes, contribuyendo de esta forma a un desarrollo gradual del objetivo final: proveer QoS extremo a extremo.

- El sistema de políticas es parte fundamental en el desarrollo sustentable de un ambiente de Calidad de Servicio. La utilización de políticas adecuadas y su correcta implementación en una red con QoS hacen que su administración sea

más fácil, dinámica y general, tal que permita atravesar los dominios administrativos e interactuar con otras redes.

- Modernos protocolos de políticas, tales como COPS, utilizan infraestructuras de directorios centralizados y de propósito general. La ventaja de una infraestructura de este tipo es optimizar el rendimiento de los dispositivos de red, permitiéndoles obtener su información de configuración desde una única fuente externa en lugar de sobrecargarlos con extensa información.

- Existen diversas herramientas para monitorear el comportamiento de una red, algunas de las cuales son muy poderosas, pero a la vez, complejas y costosas. Esto ha motivado la utilización de herramientas tradicionales como el *PING*, una utilidad que a pesar de su extrema sencillez permite predecir, muy aproximadamente, la conducta de la red.

- Las Arquitecturas de Calidad de Servicio fueron diseñadas para trabajar en Internet, sin embargo, su filosofía, conceptos y elementos son igualmente válidos para todas las redes que utilicen el *stack* de protocolos TCP/IP.

RECOMENDACIONES

- Para una correcta implementación se recomienda considerar la interacción entre los parámetros que definen la Calidad de Servicio, pues el bajo rendimiento de alguno de ellos puede repercutir en el resto y por lo tanto en la apreciación final del servicio.

- Para mejorar el rendimiento y fidelidad de una aplicación sensible al retardo que utiliza un enlace *dial-up* para acceder a Internet, se recomienda deshabilitar las funciones de corrección de errores presentes en los módems, pues de esta forma disminuye el retardo introducido por éstos.

- Una infraestructura de Calidad de Servicio en Internet debe asegurar siempre un mínimo de recursos para el tratamiento adecuado del tráfico *best effort*. En presencia de tráfico de mayor prioridad, el tráfico *best effort* no debe ser dejado sin servicio.

- Si se tiene un acuerdo de nivel de servicio con algún proveedor, se debe monitorear constantemente la red con el fin de comprobar que los aspectos comprometidos en el contrato se cumplan a cabalidad. Los acuerdos de nivel de servicio deben ser claros, consistentes y específicos.

- Para el diseño de un sistema de Calidad de Servicio en una red, se recomienda tomar como primera opción una Arquitectura Híbrida, pues con ella se obtiene un equilibrio funcional entre escalabilidad y garantías de servicio.

- La mayor parte de las fuentes de información utilizadas en el desarrollo del presente trabajo están sujetas a modificaciones, motivo por el cual se recomienda revisar las versiones más actualizadas de las distintas referencias.

ANEXO 1

NIVELES ACEPTABLES E INACEPTABLES DE RETARDO, *JITTER* Y PÉRDIDA DE PAQUETES PARA TRÁFICO CONVERSACIONAL

A1.1 RETARDO

En comunicaciones de voz, los retardos en un sentido de la transmisión no deben exceder los 250 ms, pues la comunicación entre las partes se tornaría molestosa. Si el retardo excede los 250 ms, luego, debido a que el oyente está escuchando lo que dijo el hablante ¼ de segundo antes, puede pensar que hay un vacío en la conversación, a lo cual el oyente responderá solo si el hablante interrumpe el silencio y empieza a hablar nuevamente. Estas interrupciones tornan frustrantes a las conversaciones con altos niveles de latencia.

En general, el ser humano puede empezar a notar el retardo a partir de los 125 ms, sin embargo, la conversación no es afectada severamente sino hasta que se supera el límite de los 250 ms.

En una llamada de larga distancia estándar de 4800 km a través de la red telefónica convencional (PSTN), la latencia extremo a extremo es aproximadamente 40 ms, es decir, le toma alrededor de 45 ms al oyente escuchar lo que pronunció el hablante. Una llamada de larga distancia dentro de Estados Unidos, la cual en el peor de los casos cubre una distancia de 12800 km entre Seattle y Miami, introduce una latencia de alrededor de 100 ms. Una llamada internacional de 19200 km, puede experimentar un retardo de 180 ms si la mayoría de la distancia se cubre con cable de fibra óptica.

Las llamadas a través de la red celular pueden alcanzar retardos de 125 ms, aún en el caso de llamadas locales.

La siguiente tabla compara la latencia entre las redes PSTN y Level 3 (proveedor de servicios de voz sobre IP):

	Latencia (una via)		Porcentaje	L3 vs. PSTN
Escenario	PSTN	Red de Level 3	Diferencia	Puntaje según ITU
4800 route mile LD call	45 ms	110 ms	59% peor	3.5 vs. 3.7
12800 route mile LD call	100 ms	175 ms	44% peor	3.2 vs. 3.4
19200 route mile LD call	180 ms	255 ms	30% peor	2.8 vs. 3.0

Tabla 1. Comparación de los niveles de latencia entre PSTN y la red de Level 3.

Nótese que mientras se incrementa la distancia de la llamada, la diferencia en el retardo entre la red PSTN y la red de Level 3 empieza a decrecer, si se expresa como porcentaje. Esto se debe, a que mientras se incrementa la distancia, las dos redes experimentan los mismos retardos de transmisión, con la única diferencia en el retardo producto de la electrónica en cada *hub* o central telefónica.

En términos de percepción del usuario, se espera niveles de retardo importantes solamente en el rango de 11200 a 17600 km. Aunque las llamadas en este rango presentarán un retardo perceptible, éste no impedirá el flujo normal de la conversación, y por tanto, será aceptable para la mayoría de usuarios.

"Voice over IP and the Next Generation Network" Response to the ART consultation on Internet Telephony, April 14, 1999. http://www.art-telecom.fr/publications/level3.htm

El UIT-T *recomienda* los siguientes límites para el tiempo de transmisión en un sentido, en conexiones que cuenten con un control de eco adecuado (véase la Nota 1) conforme a la Recomendación G.131:

- de 0 a 150 ms: aceptable para la mayoría de las aplicaciones de usuario (véase la Nota 2);
- 150 a 400 ms: aceptable siempre y cuando las Administraciones conozcan la influencia del tiempo de transmisión en la calidad de transmisión de las aplicaciones de usuario (véase la Nota 3);

– por encima de 400 ms: inaceptable a efectos de planificación general de la red; se acepta, sin embargo, que este límite pueda ser rebasado en ciertos casos excepcionales (véase la Nota 4).

NOTAS

1 A fin de lograr una calidad de transmisión aceptable, podría ser necesario controlar la utilización de los equipos de control de eco que causen otros tipos de degradaciones, por ejemplo, recorte vocal o contraste de ruido.

2 Algunas aplicaciones de voz y datos altamente interactivas pueden experimentar degradación para valores inferiores a 150 ms. Por consiguiente, será desaconsejable todo aumento del tiempo de procesamiento en conexiones con tiempos de transmisión muy por debajo incluso de 150 ms, a menos que los beneficios obtenidos desde el punto de vista del servicio y de la aplicación sean evidentes.

3 Así, por ejemplo, se consideran aceptables conexiones internacionales con saltos de satélite cuyos tiempos de transmisión sean inferiores a 400 ms.

4 Como excepción, se puede citar por ejemplo: los saltos de satélite dobles inevitables, la utilización de satélites para restaurar rutas terrenales, las interconexiones del servicio fijo por satélite y el servicio celular digital, la videotelefonía por circuitos de satélite o las conexiones internacionales de gran longitud con dos sistemas celulares digitales conectados mediante facilidades terrenales de larga distancia.

"Tiempo de transmisión en un sentido", Recomendación del UIT-T G.114, 1996

A1.2 *JITTER*

El tamaño de los *buffers* afecta al *jitter* y a la latencia. Si las transmisiones de audio presentan niveles de *jitter* que degradan el servicio ofrecido a los usuarios, se debe incrementar la capacidad de los *buffers* para reducir el *jitter* a niveles aceptables.

Buffers demasiado grandes, sin embargo, pueden causar un incremento en la latencia. Un valor típico presente en los *buffers,* para contrarrestar el *jitter,* es 20 ms. Se debe aclarar que no existe un tamaño ideal de buffer, pues su valor varía de red a red.

La figura 1 muestra el gráfico de los tiempos de arribo para dos flujos de paquetes. Los paquetes pertenecientes a la curva roja tienen considerablemente menos *jitter* que los paquetes pertenecientes a la curva azul. Los administradores de red, utilizan el analizador de protocolos a menudo para medir el *jitter* y alterar la capacidad de los *buffers*. Durante las pruebas, también se puede monitorear la latencia para observar si los *buffers* producen retardos inaceptables en la recepción de paquetes de voz; si es así, se puede ajustar el tamaño de los *buffers* para obtener la mejor calidad de audio.

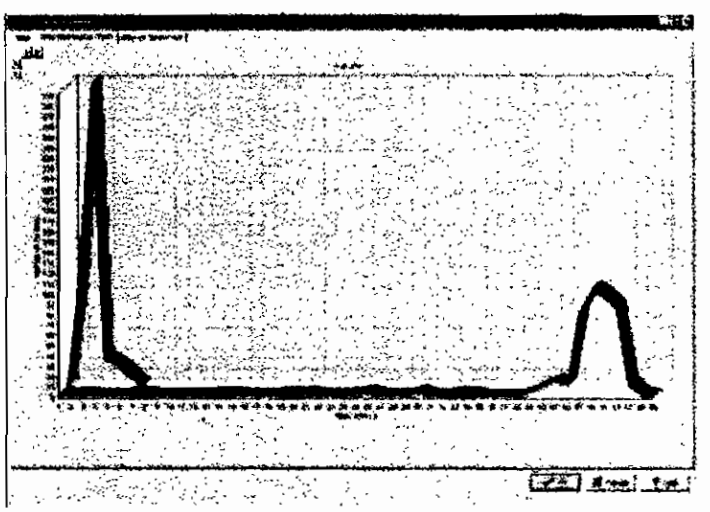

Figura 1. Medidas de *jitter* sobre paquetes de VoIP. (Cortesía de TTC)

Rowe, M., "Measure VoIP Networks for Jitter and Loss"
http://www.tmworld.com/articles/12_1999_VoIP.htm

A1.3 PÉRDIDA DE PAQUETES

Una elevada tasa de pérdida de paquetes también puede afectar el desempeño de una comunicación interactiva, típicamente, si supera el 5% de pérdida, es decir un paquetes perdido por cada 20 transmitidos. Si la pérdida de paquetes se

produce en ráfagas, ésta es más notorio; por ejemplo si se pierde un segundo de audio seguido por 19 segundos sin pérdidas.

Rowe, M., "Measure VoIP Networks for Jitter and Loss"
http://www.tmworld.com/articles/12_1999_VoIP.htm

El proveedor de servicios de VoIP Level 3, tiene una tasa de pérdida de paquetes menor a 1 en 10000 paquetes transmitidos (0.01%), causando una imperceptible diferencia en la calidad de reproducción de la voz. Debido a que cada paquete representa solo 10 ms de audio, la pérdida ocasional de un paquete no será percibida por los usuarios.

"Voice over IP and the Next Generation Network" Response to the ART consultation on Internet Telephony, April 14, 1999. http://www.art-telecom.fr/publications/level3.htm

ANEXO 2

ENCOLAMIENTO JUSTO PONDERADO: RENDIMIENTO

Para obtener algunos resultados cuantitativos sobre el rendimiento de un *router* FQ (*Fair Queuing*) referentes a la prontitud, o retardo, se considera clases muy restringidas de tráfico entrante en el cual hay solo dos tipos de fuentes. Existen fuentes de transferencia de archivos tipo FTP, las cuales siempre tienen paquetes listos que son transmitidos cuando lo permite el control de flujo de la fuente (el cual, por simplicidad, es el control de flujo de ventana corrediza), y fuentes interactivas tipo Telnet, las cuales producen paquetes intermitentemente de acuerdo a un proceso de generación no especificado.

Cuáles son las cantidades de interés? Una fuente FTP típicamente está transfiriendo una archivo grande, por lo tanto la cantidad de interés es el tiempo de transferencia del archivo, el cual para archivos grandes depende únicamente de la asignación de ancho de banda. Dada la configuración de las fuentes, esta asignación de ancho de banda puede ser calculada *a priori* utilizando la propiedad de justicia de los *switches* FQ. La cantidad de interés para las fuentes Telnet es el retardo promedio de cada paquete.

Considere un *switch* FQ con N fuentes FTP enviando paquetes de tamaño P_F, y un paquete Telnet de tamaño P_T que arriba al *switch* al tiempo t. Será asignado un número B tal que,

$$B = R(t) + P_T - \delta$$

de esta forma, la dependencia del retardo de encolamiento en las cantidades P_T y δ es únicamente a través de la combinación $P_T - \delta$. Denotaremos el retardo de encolamiento de este paquete con $\phi(t)$, el cual es una función periódica con periodo $N.P_F$. La cantidad de interés es el retardo de encolamiento promedio Δ:

$$\Delta \equiv \frac{1}{NP_F} \int_0^{NP_F} \phi(t)dt$$

Los números de finalización F_i^α para las *N* fuentes FTP pueden ser expresadas, posiblemente después de renumerar los paquetes, por $F_i^\alpha = (i + I^\alpha)P_F$ en donde *I* obedece a $0 \le I^\alpha < 1$. El retardo de encolamiento del paquete Telnet depende de la configuración de *I*, si $P_T < P_F$. Se puede mostrar que el retardo es limitado por los casos extremos $I^\alpha = 0$ para todo α e $I^\alpha = \alpha/N$ para $\alpha = 0, 1, ..., N-1$. Los valores de retardo para estos casos extremos son calculados directamente; por brevedad, se ha omitido el desarrollo y simplemente se ha desplegado los resultados abajo. El retardo de encolamiento promedio está dado por:

$$\Delta = A(P_T - \delta)$$

donde la función *A(P)*, el retardo con $\delta = 0$, es definida abajo (con el entero *k* y una pequeña constante ε, $0 \le \varepsilon < 1$, definida vía $P_T = P_F(k + \varepsilon) / N$).

Preferente

$$A(P) = N\left(P - \frac{P_F}{2}\right) \qquad \text{para } P \ge P_I$$

$$N\left(P - \frac{P_F}{2}\right) \le A(P) \le \frac{NP^2}{2P_F} \qquad \text{para } P_I \ge P \ge \frac{P_F}{2}\left(1 + \frac{1}{N}\right)$$

$$\frac{1}{2P_F}\left[\frac{P_F}{2} + N\left(P - \frac{P_F}{2}\right)\right]^2 \le A(P) \le \frac{NP^2}{2P_F} \qquad \text{para } \frac{P_F}{2}(1 + \frac{1}{N}) \ge P \ge \frac{P_F}{2}(1 - \frac{1}{N})$$

$$0 \le A(P) \le \frac{NP^2}{2P_I} \qquad \text{para } \frac{P_F}{2}\left(1 - \frac{1}{N}\right) \ge P$$

No Preferente

$$A(P) = N\left(P - \frac{P_F}{2}\right) \qquad \text{para } P \ge P_I$$

$$N\left(P - \frac{P_r}{2}\right) \le A(P) \le \left(\frac{P_r}{2}\right)\left\{1 + \frac{1}{N}\left[k^2 + k(2\varepsilon - 1)\right]\right\} \text{ para } P_r \ge P \ge \frac{P_r}{2}\left(1 + \frac{1}{N}\right)$$

$$\frac{P_r}{2} \le A(P) \le \left(\frac{P_r}{2}\right)\left\{1 + \frac{1}{N}\left[k^2 + k(2\varepsilon - 1)\right]\right\} \qquad \text{para } \frac{P_r}{2}\left(1 + \frac{1}{N}\right) \ge P$$

Un análisis más detallado puede hallarse en la referencia [A2-1]. Qué pasa en una red de *switches* FQ? Hay pocos resultados analíticos aquí, pero la referencia [A2-2] ha mostrado que para el servicio *round robin* estricto y únicamente fuentes FTP hay una asignación justa de ancho de banda cuando los tamaños de las ventanas son suficientemente grandes.

Keshav, S., "Congestion Control in Computer Networks" PhD Thesis, published as UC Berkeley TR-654 , September 1991. (Awarded the Sakrison Prize for the best PhD thesis in the EECS department, 1992.)

[A2-1] Demers, A., Keshav, S. and Shenker, S., "Analysis and Simulation of a Fair Queueing Algorithm", Journal of Internetworking Research and Experience, September 1990, 3-26;. also Proc. ACM SigComm, Sept. 1989, pp 1-12.

[A2-2] Hahne, E. , "Round Robin Scheduling for Fair Flow Control in Data Communication Networks", LIDS-TH-1631, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge.

ANEXO 3

CÁLCULO DE LA LONGITUD PROMEDIO DE LA COLA

Una compuerta RED usa un filtro pasabajos para calcular el tamaño promedio de la cola. De esta manera, el incremento momentáneo en el tamaño de la cola que resulta de una ráfaga de tráfico o de una congestión transitoria no produce un incremento significativo en el tamaño promedio de la cola.

El filtro pasabajos tiene un media móvil exponencial ponderada (EWMA, *Exponential Weighted Moving Average*):

$$avg \leftarrow (1\text{-}w_q)\ avg + w_q\ q$$

El peso w_q determina la constante de tiempo del filtro pasabajos. A continuación se analiza, los límites máximos y mínimos que podrían ser configurados para w_q.

A3.1 LÍMITE SUPERIOR PARA w_q

Si el valor de w_q es muy alto (cercano a 1), el procedimiento de promediar tomará en cuenta la presencia de congestión transitoria pues en la ecuación el primer término tiende a cero.

Para calcular el límite superior para w_q, se asume que la cola está inicialmente vacía, con un tamaño promedio de cola igual a cero, y que luego la cola incrementa su tamaño de 0 a L paquetes.

Luego que el paquete L ha llegado a la compuerta, el tamaño promedio de la cola es avg_L y está dado por:

$$avg_L = \sum_{i=1}^{L} i w_q (1 - w_q)^{L-i}$$

$$= w_q (1 - w_q)^L \sum_{i=1}^{L} i \left(\frac{1}{1 - w_q}\right)^i$$

$$= L + 1 + \frac{(1 - w_q)^{L+1} - 1}{w_q}$$

La derivación de esta ecuación usa la siguiente identidad :

$$\sum_{i=1}^{L} ix^i = \frac{x + (Lx + L - 1)x^{L+1}}{(1 - x)^2}$$

La figura 1 muestra el tamaño promedio de la cola para un rango de valores w_q y L. El eje x muestra w_q para valores desde 0.001 a 0.005, mientras el eje y muestra valores de L comprendidos entre 10 y 100. Por ejemplo, para w_q = 0.001, un incremento de 0 a 100 paquetes en el tamaño de la cola, produce un tamaño promedio de la cola igual a avg_L= 4.88 paquetes.

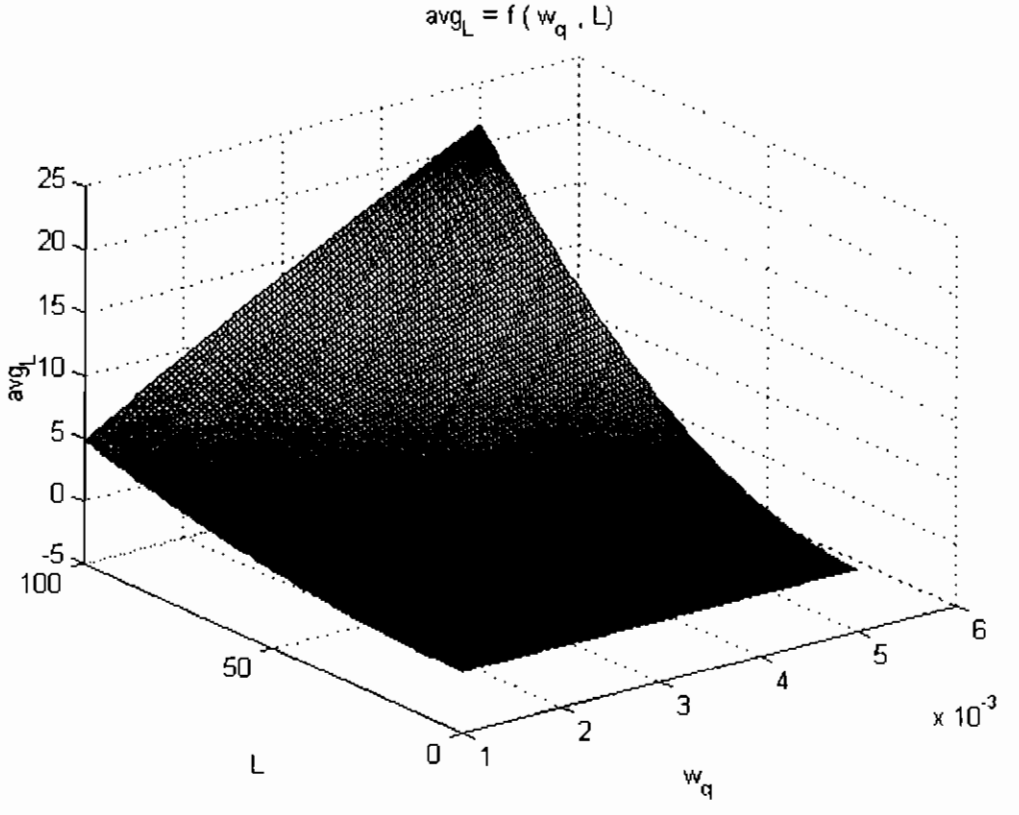

Figura 1. avg_L como función de w_q y L.

Fijando un mínimo umbral min_{th} y especificando, que se permitirá arribar a la compuerta, ráfagas de L paquetes el w_q debe ser seleccionado para satisfacer la siguiente ecuación:

$$avg_t < \min_{th}$$

y por lo tanto:

$$L + 1 + \frac{(1 - w_q)^{L+1} - 1}{w_q} < \min_{th}$$

Haciendo min_{th} = 5, y L = 50, por ejemplo, es necesario fijar el valor de $w_q \leq 0.0042$.

A3.2 LÍMITE INFERIOR PARA w_q

Las compuertas RED son diseñadas para mantener el tamaño promedio de cola calculado (*avg*) cercano siempre a un cierto umbral. Si w_q es configurado en un valor sumamente bajo el *avg* responderá muy lentamente a los cambios en el actual tamaño de la cola. En este caso la compuerta RED es incapaz para detectar estados iniciales de congestión.

Por ejemplo asumir, que la cola cambia de vacía a un paquete, y que, cuando los paquetes arriban y salen a la misma velocidad, esto supondrá un paquete encolado. Si se supone el tamaño promedio de la cola igual a cero, podrán arribar $-1/\ln(1 - w_q)$ paquetes hasta que el valor promedio de la cola alcanze el valor *avg* = 1-1/e = 0.63.

Para w_q = 0.001, pueden arribar 1000 paquetes; para w_q = 0.002 el arribo es permitido para 500 paquetes y para w_q = 0.003 pueden arribar 333 paquetes.

A3.3 CONFIGURACIÓN DE VALORES min_{th} y max_{th}

Los valores óptimos para min_{th} y max_{th} dependen de la profundidad de la cola, que se desee o necesite.

Si el tráfico que circula por la compuerta RED presenta comúnmente ráfagas, el min_{th} será lo suficientemente grande como para permitir que la utilización del enlace sea mantenido en un nivel aceptable.

El valor óptimo para max_{th} depende en parte del máximo promedio de retardo que pueda ser permitido por la compuerta.

Las compuertas RED funcionan de manera más eficiente cuando la diferencia entre max_{th} y min_{th} es más que el incremento típico en el tamaño promedio de cola calculado en un RTT. Una regla común es configurar max_{th} como al menos el doble de min_{th}.

Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, August 1993.

ANEXO 4

A4.1 ALGORITMO DE CUBETA CON GOTEO (*LEAKY BUCKET*)

Mecanismo de conformación de tráfico en el cual únicamente una cantidad fija de tráfico es admitida hacia la red. El exceso de tráfico se mantiene encolado hasta que pueda ser acomodado o descartado.

Imagínese una cubeta con un pequeño agujero en el fondo, tal como se ilustra en la figura 1 (a). Sin importar la rapidez con que entra agua en la cubeta, el flujo de salida tiene una tasa constante, *r*, cuando hay agua en la cubeta, y una tasa de cero cuando la cubeta está vacía.

También, una vez que se llena la cubeta, cualquier agua adicional que entra se derrama por los costados y se pierde (es decir, no aparece en el flujo por debajo del agujero).

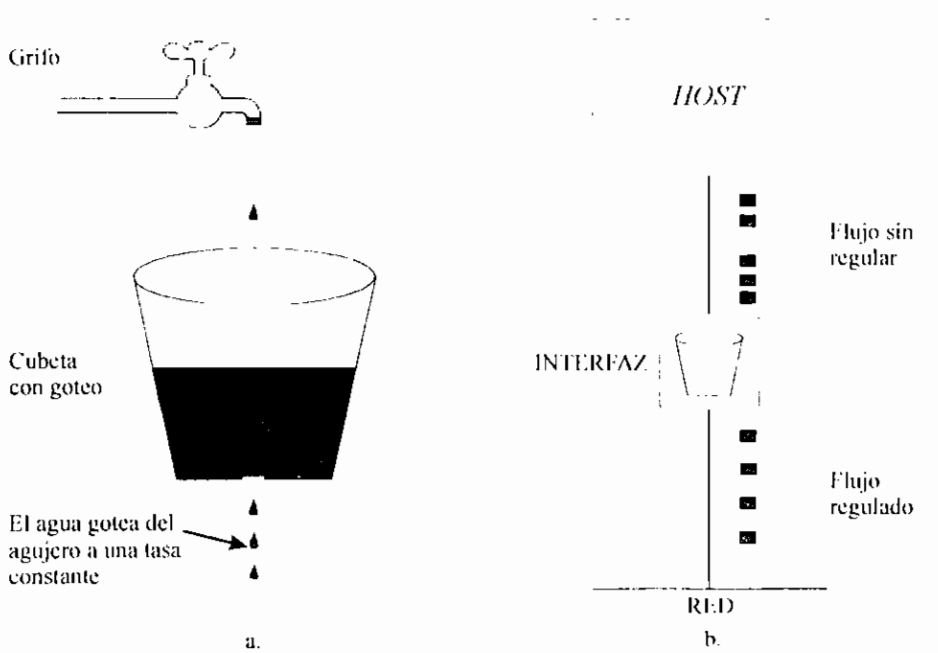

Figura 1. (a) Una cubeta con goteo, llena de agua. (b) Cubeta con goteo, llena de paquetes.

Puede aplicarse el mismo concepto a los paquetes, como se muestra en la figura 1 (b). Conceptualmente, cada *host* está conectado a la red mediante una interfaz

que contiene una cubeta con goteo, es decir, una cola interna finita. Si llega un paquete a la cola cuando está llena, se descarta el paquete. En otras palabras, si uno o más procesos del *host* tratan de enviar paquetes cuando ya está en cola la cantidad máxima, se descarta el paquete sin más. Este arreglo puede incorporarse en el *hardware* de la interfaz, o simularse a través del sistema operativo del *host*. Este algoritmo no es otra cosa que un sistema de encolamiento de un solo servidor con un tiempo de servicio constante.

El *host* puede poner en la red un paquete por pulso de reloj. Nuevamente, esto puede obligarse desde la tarjeta de interfaz o desde el sistema operativo. Este mecanismo convierte un flujo desigual de paquetes de los procesos de usuario dentro del *host* en un flujo continuo de paquetes hacia la red, moderando las ráfagas y reduciendo en buena medida las posibilidades de congestionamiento.

Cuando los paquetes son del mismo tamaño (por ejemplo, celdas ATM), este algoritmo puede usarse como se describe. Sin embargo, al usarse paquetes de tamaño variable con frecuencia es mejor permitir un número fijo de bytes por pulso, en lugar de un solo paquete. Por tanto, si la regla es de 1024 bytes por pulso, pueden recibirse por pulso un solo paquete de 1024 bytes, dos paquetes de 512 bytes, cuatro paquetes de 256 bytes, etc. Si el conteo de bytes residuales es demasiado bajo, el siguiente paquete debe esperar hasta el siguiente pulso.

La implementación del algoritmo de cubeta con goteo es fácil. La cubeta con goteo consiste en una cola finita. Al llegar un paquete, si hay espacio en la cola, se agrega a ella; de otro modo se descarta. En cada pulso de reloj se transmite un paquete (a menos que la cola esté vacía).

La cubeta con goteo que usa conteo de bits se implementa casi de la misma manera. En cada pulso se inicia un contador en *n*. Si el primer paquete de la cola tiene menos bytes que el valor actual del contador, se transmite y se disminuye el contador en esa cantidad de bytes. Pueden enviarse paquetes adicionales en tanto que el contador sea lo suficientemente grande. Al caer el contador por debajo de la longitud del siguiente paquete de la cola, se detiene la transmisión

hasta el siguiente pulso, en cuyo momento se sobrescribe el conteo de bytes residuales, perdiéndose.

Como ejemplo de cubeta con goteo, imagine que una computadora puede producir datos a razón de 25 millones de bytes/s (200 Mbps) y que la red también opera a esta velocidad. Sin embargo, los enrutadores pueden manejar esta tasa de datos sólo durante cortos intervalos. Durante intervalos grandes funcionan mejor con tasas que no exceden 2 millones de bytes/s.

Ahora suponga que los datos llegan en ráfagas de un millón de bytes con una ráfaga de 40 ms cada segundo. Para reducir la tasa promedio a 2 MB/s, podemos usar una cubeta con goteo de r = 2 MB/s y capacidad b = 1 MB. Esto significa que las ráfagas de hasta 1 MB pueden manejarse sin pérdidas de datos, ya que se distribuyen a través de 500 ms sin importar la velocidad a la que lleguen.

En la figura 2 (a) se observa la entrada de la cubeta con goteo operando a 25 MB/s durante 40 ms. En la figura 2 (b) se observa la salida drenándose a una velocidad uniforme de 2 MB/s durante 500 ms.

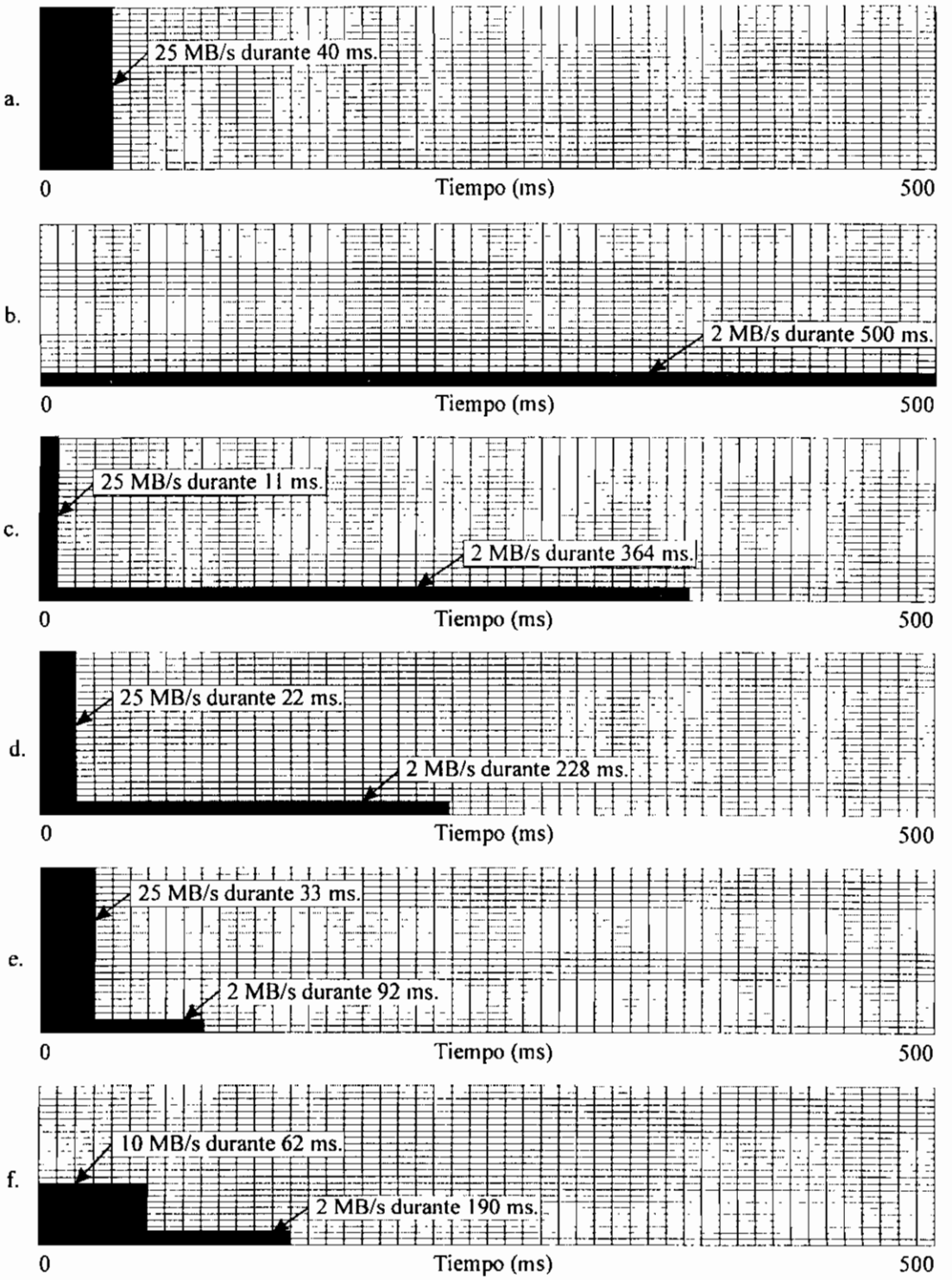

Figura 2. (a) Entrada a una cubeta con goteo. (b) Salida de una cubeta con goteo. (c)-(e) Salida de una cubeta con fichas con capacidades de 250 kB, 500 kB y 750 kB. (f) Salida de una cubeta con fichas de 500 kB que alimenta a una cubeta con goteo de 10 MB/s.

A4.2 ALGORITMO DE CUBETA CON FICHA (*TOKEN BUCKET*)

Es un mecanismo de conformación de tráfico en el cual una cantidad predeterminada de fichas dentro de la cubeta representa la capacidad permitida para cada clase de tráfico. Los paquetes son direccionados hasta que termine la cantidad disponible de fichas asignadas a ellos. Cuando se agotan las fichas, los paquetes pueden ser descartados o encolados hasta que la cubeta sea rellenada. Esto controla la tasa de transmisión (*r*, *rate*) y acomoda las ráfagas de tráfico (*b*, *bucket size*).

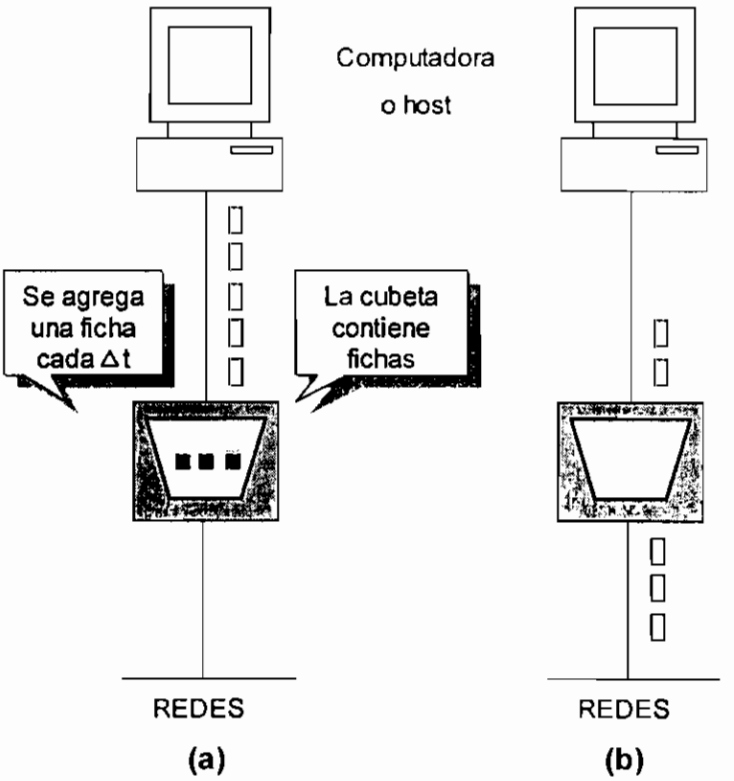

Figura 3. Algoritmo de cubeta con fichas. (a) Antes. (b) Después del ingreso al Algoritmo Token Bucket.

Las fichas son generadas por un reloj cada Δt segundos. En la figura 3 (a) se muestra una cubeta que contiene tres fichas, con cinco paquetes esperando ser transmitidos. Para transmitir un paquete, el transmisor debe capturar y destruir una ficha. En la figura 3 (b) se observa que han pasado tres de los cinco

paquetes, pero los otros dos están encolados, esperando la generación de dos o más fichas.

El algoritmo de cubeta con ficha ofrece una forma diferente de conformación de tráfico que el algoritmo de cubeta con goteo. Este último no permite que los *hosts* inactivos acumulen permisos para enviar posteriormente ráfagas grandes. El algoritmo de cubeta con ficha sí permite el ahorro, hasta el tamaño máximo de la cubeta, *n*. Esta propiedad significa que pueden enviarse a la vez ráfagas de hasta *n* paquetes, permitiendo cierta irregularidad en la corriente de salida y dando una respuesta más rápida a las ráfagas de entrada repentina.

Es posible una variación menor, en la que cada ficha representa el derecho de transmisión, no de un paquete, sino de *k* bytes. Solo puede transmitirse un paquete si hay suficientes fichas disponibles para cubrir su longitud en bytes. Las fichas fraccionarias se guardan para su uso futuro.

Los algoritmos de cubeta con goteo y cubeta con ficha pueden servir también para regular el tráfico entre los enrutadores, de la misma manera que se usan para regular la salida de un *host*, como en los ejemplos. Sin embargo, una diferencia clara es que una cubeta con ficha que regula a un *host* puede hacer que el *host* detenga el envío cuando las reglas dicen que debe hacerlo. Indicar a un enrutador que detenga la transmisión mientras sigue recibiendo entradas puede resultar en la pérdida de datos.

La implementación del algoritmo de cubeta con ficha simplemente es una variable que cuenta fichas. El contador se incrementa en uno cada Δt y disminuye en uno cada vez que se envía un paquete. Al llegar a cero el contador, ya no pueden enviarse paquetes. En la variante de conteo de bits, se incrementa el contador en *k* bytes cada Δt y disminuye en la longitud de cada paquete enviado.

En esencia, lo que hace la cubeta con ficha es permitir ráfagas, pero limitadas a una longitud máxima regulada. Véase un ejemplo en la figura 2 (c). Aquí se tiene una cubeta de fichas con 250 kB de capacidad. Las fichas llegan con una tasa tal que permite la salida a 2 MB/s. Suponiendo que la cubeta con ficha está llena cuando llega la ráfaga de 1MB, la cubeta puede drenarse a la velocidad máxima

de 25 MB/s durante unos 11 ms. Entonces tiene que desacelerarse hasta 2 MB/s hasta que se envíe la ráfaga de entrada completa.

El cálculo de la longitud de ráfaga con tasa máxima es un tanto complicado. No sólo es la división de 1 MB entre 25 MB/s, ya que, mientras se está enviando la ráfaga, llegan más fichas. Si t segundos es la duración de la ráfaga, b bytes es la capacidad de la cubeta con ficha, r bytes/segundo es la tasa de llegada de fichas, y p bytes/segundo es la tasa máxima de salida, podemos ver que una ráfaga de salida contiene un máximo de $b+r.t$ bytes. También se sabe que la cantidad de bytes en una ráfaga a velocidad máxima con duración t segundos es $p.t$. Por tanto, se tiene:

$$b + r.t = p.t$$

Para b = 250 kB, p = 25 MB/s y r = 2 MB/s, se tiene un tiempo de ráfaga t = 11 ms. En la figura 2 (d) y 2 (e) se muestra la cubeta con ficha para capacidades de 500 kB y 750 kB, respectivamente.

Un problema potencial con el algoritmo de cubeta con ficha es que permite ráfagas largas, aunque puede regularse el intervalo máximo de ráfaga mediante una selección cuidadosa de r y p. Con frecuencia es deseable reducir la tasa pico, pero sin regresar al bajo valor de la cubeta con goteo original.

Una manera de lograr tráfico más uniforme es poner una cubeta con goteo después de la cubeta con ficha. La tasa de la cubeta con goteo deberá ser mayor que r de la cubeta con ficha, pero menor que p. En la figura 2 (f) se muestra la salida de una cubeta con ficha de 500 kB seguida de una cubeta con goteo de 10 MB/s.

ANEXO 5

FORMATOS DE LOS MENSAJES RSVP

Un mensaje RSVP consiste de una cabecera común seguida por un cuerpo de longitud variable compuesto de varios objetos. A continuación se analiza el formato específico de cada mensaje RSVP, posteriormente se analiza en detalle la cabecera común y cada uno de los objetos. En la representación utilizada, los objetos entre corchetes son opcionales.

A5.1 MENSAJES

Los mensajes que utiliza el protocolo RSVP son los siguientes:

1. **Mensaje Path**

<*Path* Message> ::= <Common Header> [<INTEGRITY>]

<SESSION> <RSVP_HOP>
<TIME_VALUES>
[<POLICY_DATA> ...]
[<sender descriptor>]

<sender descriptor> ::= <SENDER_TEMPLATE> <SENDER_TSPEC>
[<ADSPEC>]

2. **Mensaje Resv**

<*Resv* Message> ::= <Common Header> [<INTEGRITY>]

<SESSION> <RSVP_HOP>
<TIME_VALUES>
[<*RESV*_CONFIRM>] [<SCOPE>]
[<POLICY_DATA> ...]
<STYLE> <flow descriptor list>

<flow descriptor list> ::= <empty> |
<flow descriptor list> <flow descriptor>

El objeto <flow descriptor list> depende del estilo de reservación utilizado, de la siguiente forma:

- Estilo WF:

<flow descriptor list> ::= <WF flow descriptor>
<WF flow descriptor> ::= <FLOWSPEC>

- Estilo FF:

<flow descriptor list> ::=
<FLOWSPEC> <FILTER_SPEC> |
<flow descriptor list> <FF flow descriptor>
<FF flow descriptor> ::=
[<*FLOWSPEC*>] <FILTER_SPEC>

- Estilo SE:

<flow descriptor list> ::= <SE flow descriptor>
<SE flow descriptor> ::=
<*FLOWSPEC*> <*filter spec* list>
<*filter spec* list> ::= <FILTER_SPEC>
| <*filter spec* list> <FILTER_SPEC>

3. Mensaje PathTear

<*PathTear* Message> ::= <Common Header> [<INTEGRITY>]
<SESSION> <RSVP_HOP>
[<sender descriptor>]
<sender descriptor> ::= (ver la definición anterior)

4. Mensaje ResvTear

<ResvTear Message> ::= <Common Header> [<INTEGRITY>]
<SESSION> <RSVP_HOP>
[<SCOPE>] <STYLE>
<flow descriptor list>
<flow descriptor list> ::= (ver la definición anterior)

5. Mensaje PathErr

<PathErr message> ::= <Common Header> [<INTEGRITY>]
<SESSION> <ERROR_SPEC>

[<POLICY_DATA> ...]

[<sender descriptor>]

<sender descriptor> ::= (ver la definición anterior)

6. **Mensaje ResvErr**

<ResvErr Message> ::= <Common Header> [<INTEGRITY>]

<SESSION> <RSVP_HOP>

<ERROR_SPEC> [<SCOPE>]

[<POLICY_DATA> ...]

<STYLE> [<error flow descriptor>

El objeto <error flow descriptor> depende del estilo de reservación empleado, como se muestra a continuación:

- Estilo WF:

  <error flow descriptor> ::= <WF flow descriptor>

- Estilo FF:

  <error flow descriptor> ::= <FF flow descriptor>

  Cada descriptor de flujo debe ser procesado independientemente, y si se genera un error, se enviará un mensaje *ResvErr* para cada uno.

- Estilo SE:

  <error flow descriptor> ::= <SE flow descriptor>

7. **Mensaje ResvConf**

<ResvConf message> ::= <Common Header> [<INTEGRITY>]

<SESSION> <ERROR_SPEC>

<RESV_CONFIRM>

<STYLE> <flow descriptor list>

<flow descriptor list> ::= (ver la definición anterior)

A5.2 FORMATO DE LA CABECERA RSVP

La cabecera RSVP se encuentra presente en todos los mensajes RSVP y corresponde a un campo de 8 bytes, analizado a continuación.

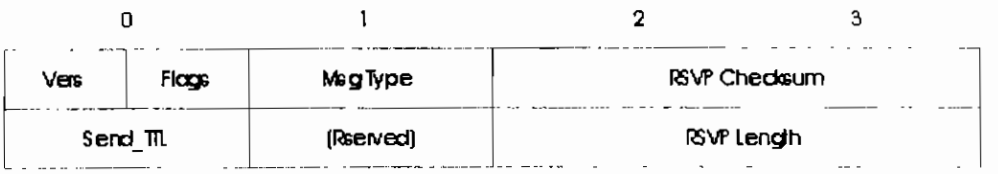

Figura 1 Formato de la cabecera RSVP.

Los campos en la cabecera común son:

Vers (Versión) [4 bits]: Número de la versión del protocolo. Esta especificación corresponde a la versión 1.

Flags (Banderas) [4 bits]: Aún no se especifica bits de banderas.

Msg Type (Tipo de Mensaje) [8 bits]: Identifica al mensaje, de la siguiente manera:

1 = Path
2 = Resv
3 = PathErr
4 = ResvErr
5 = PathTear
6 = ResvTear
7 = ResvConf

RSVP *Checksum* (Suma de comprobación) [16 bits]: Se calcula como el uno complemento de la suma uno complemento del mensaje, con el campo *checksum* reemplazado por ceros. Si todos los bits de este campo son ceros entonces se deduce que el *checksum* no fue transmitido.

Send_TTL [8 bits]: Es el valor IP TTL (*Time To Live*) del mensaje, se dará más detalles posteriormente.

RSVP *Length* (Longitud del mensaje) [16 bits]: Es la longitud total del mensaje RSVP en bytes, incluye la cabecera común y el campo de longitud variable correspondiente a los objetos.

A5.3 FORMATO DE LOS OBJETOS RSVP

Cada objeto consiste de una o más palabras de 32 bits, una de las cuales es la cabecera. El formato es el siguiente:

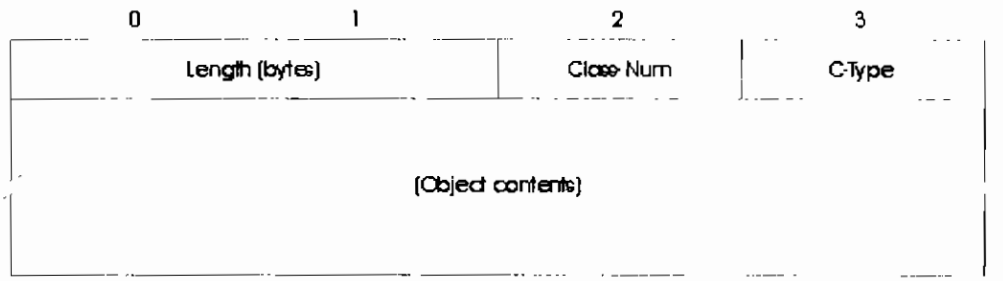

Figura 2 Formato de los Objetos RSVP.

La cabecera del objeto tiene los siguientes campos:

Length (Longitud) [16 bits]: Contiene la longitud total del objeto, en bytes; ésta puede ser 4 o múltiplos de 4.

Class-Num (Número de Clase) [8 bits]: Contiene el número correspondiente a la clase del objeto, el cual permite identificarlo.

C-Type (Tipo de Clase) [8 bits]: Cada objeto depende de la versión del protocolo IP que se utilice, de acuerdo a ello se define el tipo de clase del objeto, cuyo valor es 1 y 2 para IPv4 e IPv6 respectivamente.

A5.3.1 OBJETOS DEL PROTOCOLO RSVP

El protocolo RSVP reconoce las siguientes clases de objetos:

1. **NULL** [Class-Num = 0 ; C-Type = ignorado]

Este objeto puede transmitirse en cualquier momento y su contenido es ignorado por el receptor. Su longitud puede ser 4 o un múltiplo de 4 (bytes).

2. **SESSION** [*Class-Num* = 1]

Contiene la información necesaria para definir una sesión específica para los demás objetos que siguen, es necesario en todos los mensajes RSVP.

IPv4 / Sesión UDP [*C-Type* = 1]

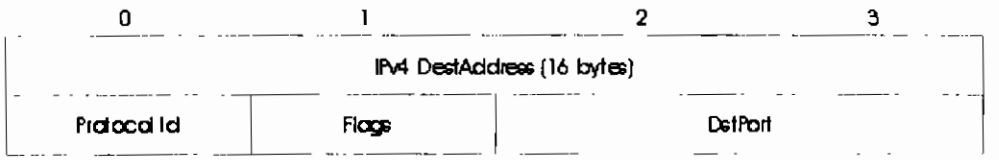

Figura 3 Objeto sesión (UDP) para IPv4.

IPv6 / Sesión UDP [*C-Type* = 2]

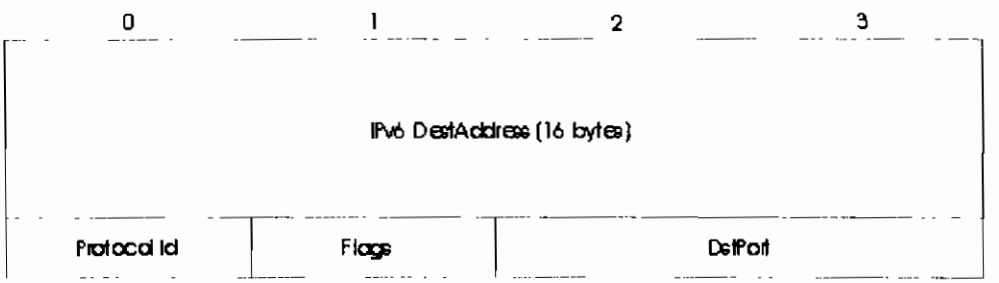

Figura 4 Objeto sesión (UDP) para IPv6.

DestAddress [4 o 16 bytes]: Es la dirección IP de destino de la sesión, la cual puede ser *unicast* o *multicast*.

Protocol Id [1 byte]: Es el identificador del protocolo IP del flujo de datos.

Flags [1 byte]: Está definida la bandera 0x01 (E_Police flag) para los mensajes *Path*. Esta bandera sirve para determinar donde se aplican las políticas de control de tráfico, si un transmisor no es capaz de proveer políticas de control de tráfico hace que el bit sea 1, y el primer nodo capaz de suministrar políticas de control de tráfico configura este bit a 0.

DstPort [1 byte]: Es el puerto UDP/TCP de destino de la sesión. Si no se usa, su valor será cero.

3. ***RSVP_HOP*** [Class-Num = 3]

Este objeto transporta la dirección IP del último nodo RSVP. El campo LIH (*Logical Interface Handle*) es creado por los mensajes *Path* para distinguir entre dos interfaces lógicas salientes y es posteriormente utilizado por los mensajes *Resv* para encontrar el destino y evitar pérdida de paquetes.

IPv4 [C-Type = 1]

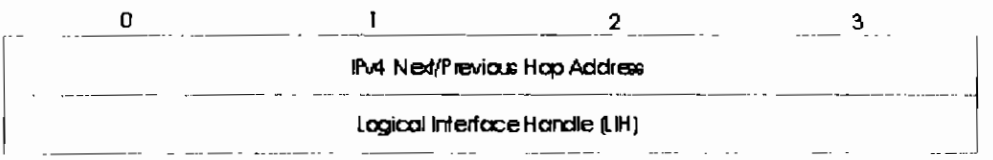

Figura 5 Objeto *RSVP_HOP* para IPv4.

IPv6 [C-Type = 2]

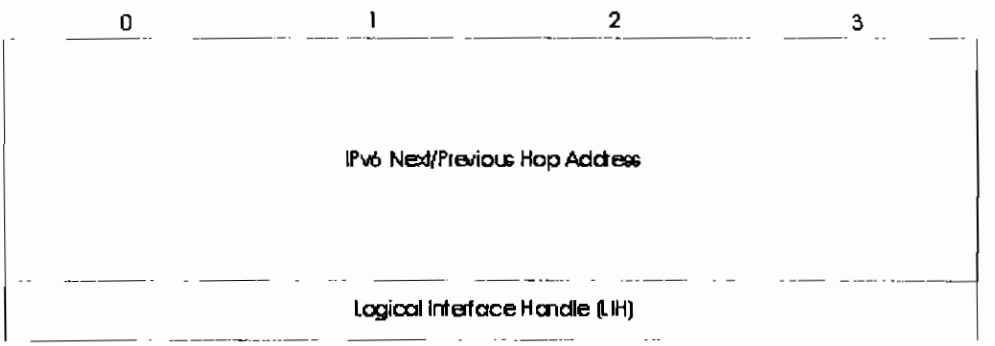

Figura 6 Objeto RSVP_HOP para IPv6.

4. ***TIME_VALUES*** [*Class-Num* = 5]

Contiene el periodo de refresco R en milisegundos. Es usado por la fuente del mensaje y su presencia es necesaria en todos los mensajes *Resv* y *Path*.

Figura 7 Objeto *TIME_VALUES*

5. ERROR_SPEC [*Class-Num* = 6]

Sirve para especificar el error en los mensajes *ResvErr* y *PathErr*, o la confirmación en el mensaje *Resv*Conf.

IPv4 [*C-Type* = 1]

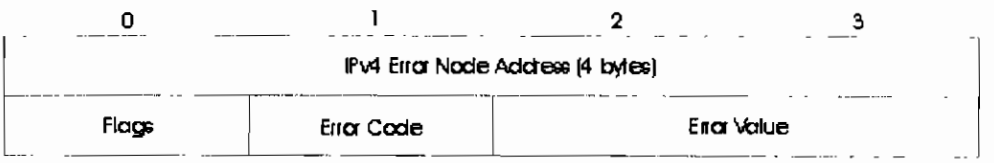

Figura 8 Objeto *ERROR_SPEC* para IPv4.

IPv6 [*C-Type* = 6]

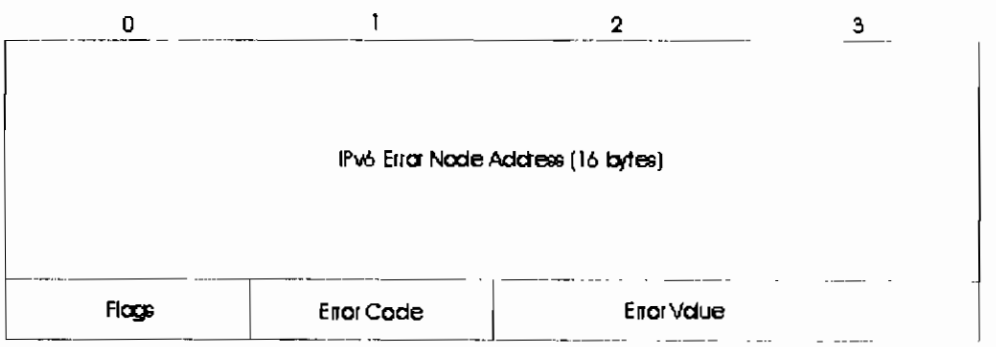

Figura 9 Objeto *ERROR_SPEC* para IPv4.

Los campos son los siguientes:

Error Node Address [4 o 16 bytes]: Especifica la dirección IP del nodo en el cual se produjo el error.

Flags [1 byte]: Se han especificado dos banderas: 0x01 (*InPlace*) y 0x02 (*NotGuilty*) que son utilizadas por los mensajes *ResvErr*. La primera se usa para indicar si existe (1) o no (0) reservación en el nodo que produjo el error. La segunda sirve para indicar que el *flowspec* que falló en la reservación fue estrictamente mayor que el *flowspec* requerido, es configurada sólo en la interfaz de la aplicación en el lado del receptor.

Error Code [1 byte]: Es la descripción del error. Los códigos definidos se presentan en la tabla 1.

Código	Significado
00	Confirmación
01	Falla en el control de admisión
02	Falla en el control de políticas
03	No existe información del camino para el mensaje Resv
04	No existe información del origen para el mensaje Resv
05	Conflicto en el estilo de reservación
06	Estilo de reservación desconocido
07	Conflicto en los puertos de destino
08	Conflicto en los puertos de origen
09	Reservado
10	Reservado
11	Reservado
12	Servicio reservado administrativamente
13	Clase de objeto desconocida
14	Tipo de objeto desconocido
15	Reservado
16	Reservado
17	Reservado
18	Reservado
19	Reservado
20	Reservado para API (Aplication Program Interface)
21	Error en el control de tráfico
22	Error del sistema de control de tráfico
23	Error en el sistema RSVP

Tabla 1 Códigos de Error

Error Value [2 bytes]: Provee información adicional del error.

6. ***SCOPE*** [*Class-Num* = 7]

Contiene el campo de la reservación, el cual es una lista de direcciones IP de los *hosts* transmisores hacia los cuales va dirigida la información. Este objeto puede aparecer en los mensajes *Resv*, *ResvErr* o *ResvTear.*

IPv4 [*C-Type* = 1]

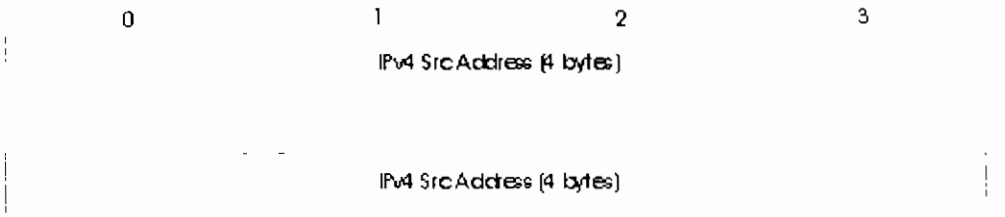

Figura 10 Objeto *SCOPE* para IPv4.

IPv6 [*C-Type* = 2]

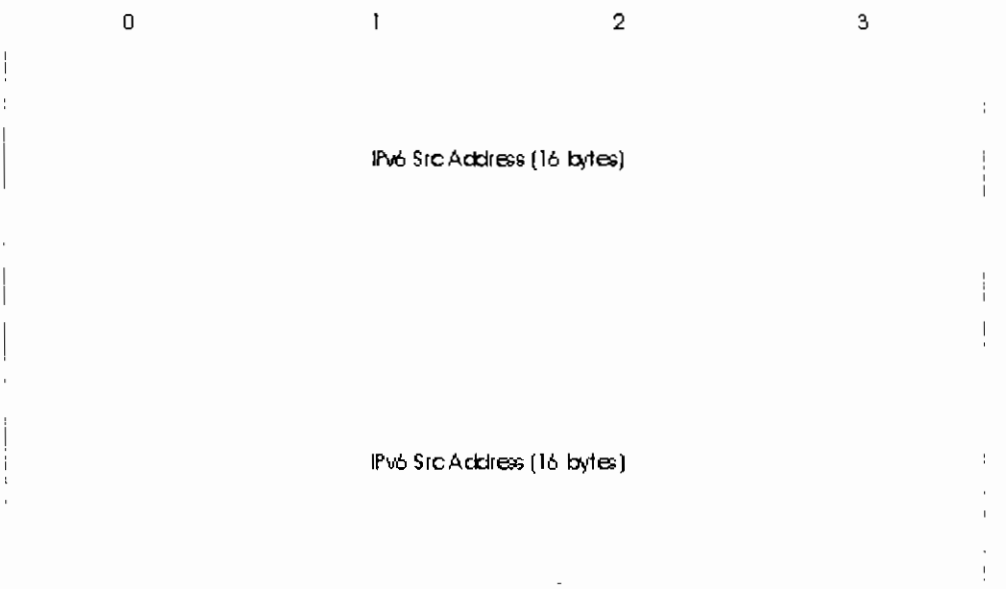

Figura 11 Objeto *SCOPE* para IPv6.

7. ***STYLE*** [*Class-Num* = 8]

Este campo permite definir el estilo de la reservación. El campo de 1 byte correspondiente a las banderas aún no está definido. El campo correspondiente al vector de opciones tiene tres subcampos; el primero, correspondiente a los 19 bits más significativos, está reservado para futuros estilos; el segundo, el cual contiene los dos bits siguientes, especifica si la reservación es o no compartida; y, el tercero, que comprende los tres bits menos significativos, especifica la forma de selección del emisor. Por lo tanto los valores binarios 10001, 01010 y 10010 representan los estilos WF, FF y SE respectivamente.

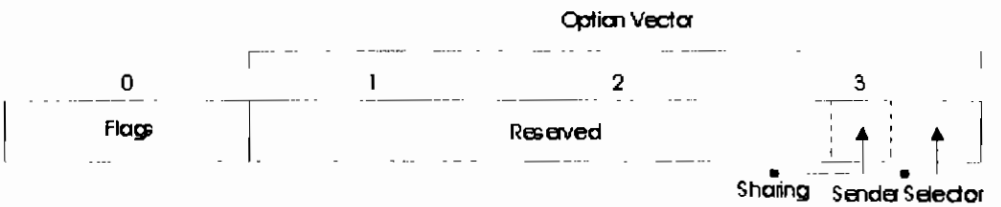

Figura 12 Objeto *STYLE*

8. ***FLOWSPEC*** [*Class-Num* = 9]

Este objeto permite definir las características de la calidad de servicio deseada, se utiliza con los mensajes *Resv*. El *FLOWSPEC* está formado por el TSpec y RSpec especificados anteriormente en el numeral 5.2.2.3.

El formato de este objeto para los servicios de carga controlada es:

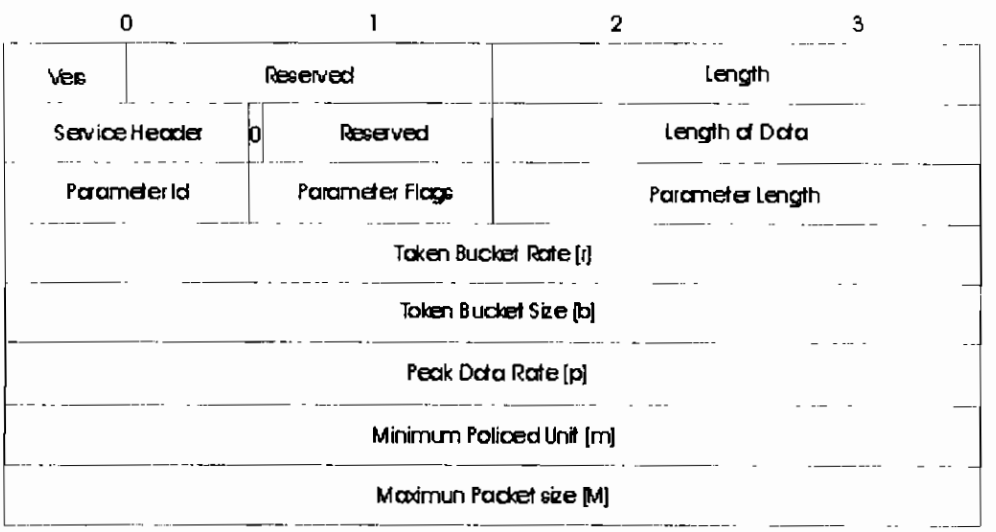

Figura 13 Objeto *FLOWSPEC* para el servicio de carga controlada.

El formato del objeto para los servicios garantizados está especificado en la figura 14.

Los campos del objeto son:

Vers [4 bits]: Es el número de versión del formato del mensaje, la actual es 0.

Lenght [2 bytes]: Especifica la longitud total del mensaje sin incluir la cabecera. Su valor es 7 y 10 palabras de 32 bits para los servicios de carga controlada y garantizado respectivamente.

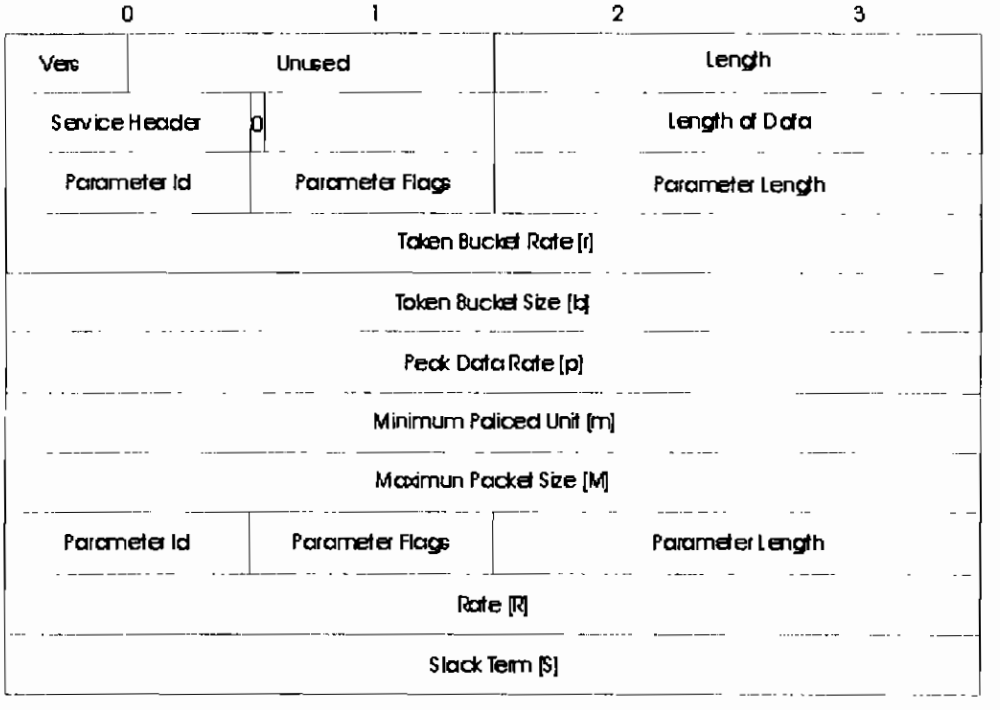

Figura 14 Objeto *FLOWSPEC* para el servicio garantizado

Service Header [1 byte]: Especifica el servicio que se dará al flujo de datos. Su valor puede ser 5 o 2 palabras para carga controlada o servicio garantizado respectivamente.

Length of Data [2 bytes]: Especifica la longitud de los datos correspondientes al servicio, sin incluir la cabecera. Para el servicio de carga controlada su valor es 6 palabras, y para el servicio garantizado su valor es 9 palabras.

Parameter Id [1 byte]: Identifica el parámetro; si se refiere al TSpec su valor es 127, y si se refiere al RSpec su valor es 130. Ambos parámetros se describen en el punto 5.2.2.3.

Parameter Flags [1 byte]: Este campo está destinado a las banderas que usará el parámetro descrito, las cuales aún no se han especificado.

Parameter Lenght [2 bytes]: Indica la longitud del parámetro sin incluir su cabecera. Su valor puede ser 5 o 2 palabras para TSpec y RSpec respectivamente.

Los parámetros *r*, *b*, *p*, *m*, *M*, *R* y *S* se describen en el punto 5.2.2.3.

9. *FILTER_SPEC* [*Class-Num* = 10]

Define el conjunto de paquetes que pertenecen a la sesión que recibirá el QoS. Este objeto es transportado por los mensajes *Resv*.

IPv4 [*C-Type* = 1]

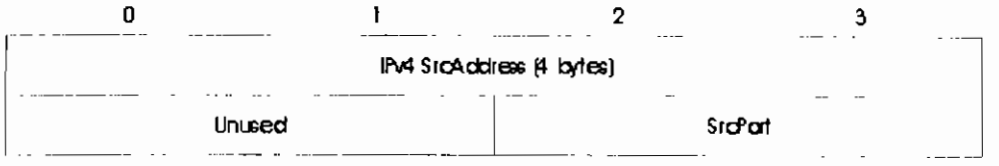

Figura 15 Objeto *FILTER SPEC* para IPv4.

IPv6 [*C-Type* = 2]

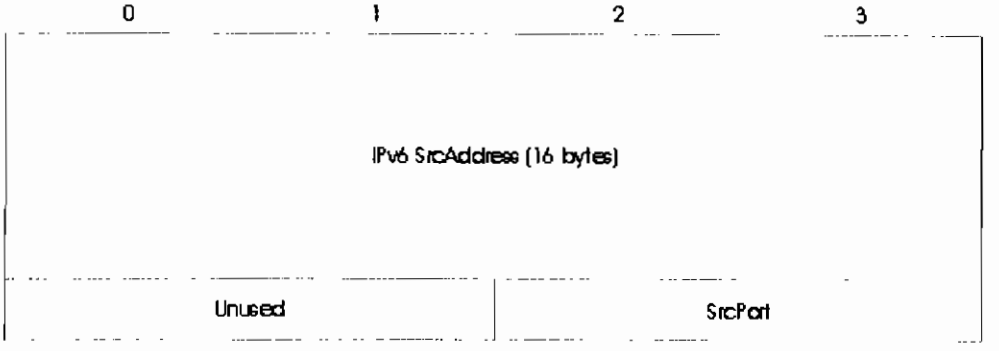

Figura 16 Objeto *FILTER SPEC* para IPv6.

Los campos de este objeto son:

SrcAddress [4 o16 bytes]: Especifica la dirección IP para el *host*s transmisor.

SrcPort [2 bytes]: Identifica el puerto UDP/TCP para el transmisor, si su valor es cero significa que este parámetro no está utilizado.

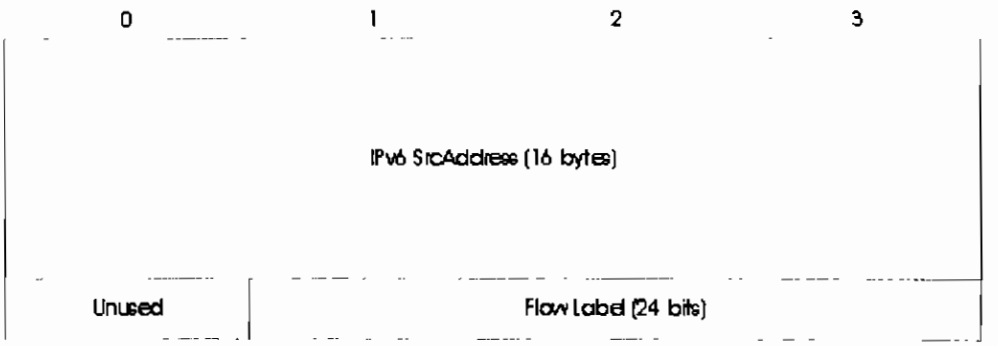

Figura 17 Objeto *FILTER SPEC* (opcional) para IPv6.

Flow Label [3 bytes]: Este campo es definido en IPv6, y su valor puede ser usado por el clasificador de paquetes para identificar, de una manera eficiente, los paquetes que pertenecen a un flujo en particular, caracterizado por una fuente y un destino.

10. ***SENDER_TEMPLATE*** [*Class-Num* = 11]

Contiene información que permite identificar a los *host*s transmisores. Su formato es idéntico que el objeto *FILTER_SPEC*, pero su uso se da en los mensajes *Path*.

11. ***SENDER_TSPEC*** [*Class-Num* = 12]

Define las características de tráfico que generará una fuente. Se utiliza en los mensajes *Path*. El formato es similar al del objeto *FLOWSPEC* de carga controlada (véase la Figura 13, siendo la única diferencia el valor del campo *Service Header*, el cual en este caso es 1 e indica que el objeto corresponde a los parámetros generales configurados por defecto.

12. ***ADSPEC*** [*Class-Num* = 13]

Transporta los datos de anuncio en los mensajes *Path*. Está formado por una cabecera general y por los fragmentos correspondientes a cada servicio (Figura 18). El primer fragmento es obligatorio y corresponde a la descripción de parámetros generales; los dos siguientes fragmentos son opcionales y se refieren a los servicios garantizado y de carga controlada respectivamente. La existencia de los dos últimos fragmentos depende de la capacidad o conocimiento de los

servicios que soporta el *host* emisor, si éste no soporta un determinado servicio simplemente no usa el campo correspondiente al mismo.

Los campos de este mensaje son los siguientes:

Vers [4 bits]: Especifica la versión del mensaje, su valor es 0.

Msg Length [2 bytes]: Indica la longitud total del mensaje sin incluir la cabecera. Su valor es N+M+9 palabras de 32 bits.

Service header [1 byte]: Define el servicio al cual pertenecen los parámetros que le siguen. Su valor puede ser 1, 2 o 5 dependiendo si los parámetros son generales, pertenecen al servicio garantizado o son del servicio de carga controlada respectivamente.

Length of Service (*General Parameter*) [2 bytes]: Indica la longitud, en palabras de 32 bits, que ocupa el fragmento correspondiente a un determinado servicio; su valor no incluye a la cabecera. El fragmento correspondiente a los parámetros generales tiene una longitud de 8 palabras; el correspondiente al servicio garantizado tiene una longitud de N - 1 palabras; y, el correspondiente al servicio de carga controlada tiene una longitud de M - 1 palabras de 32 bits.

Parameter Id [1 byte]: Identifica el parámetro, puede tomar el valor especificado en la tabla 2.

Servicio	Parámetro	Valor
Parámetros Generales	IS hop cnt	4
	Path b/w estimate	6
	Minimum *Path* Latency	8
	Composed MTU	10
[illegible]	[illegible]	[illegible]
	[illegible]	[illegible]
	[illegible]	[illegible]
	[illegible]	[illegible]

Tabla 2 Valores de los parámetros utilizados en el objeto *ADSPEC*.

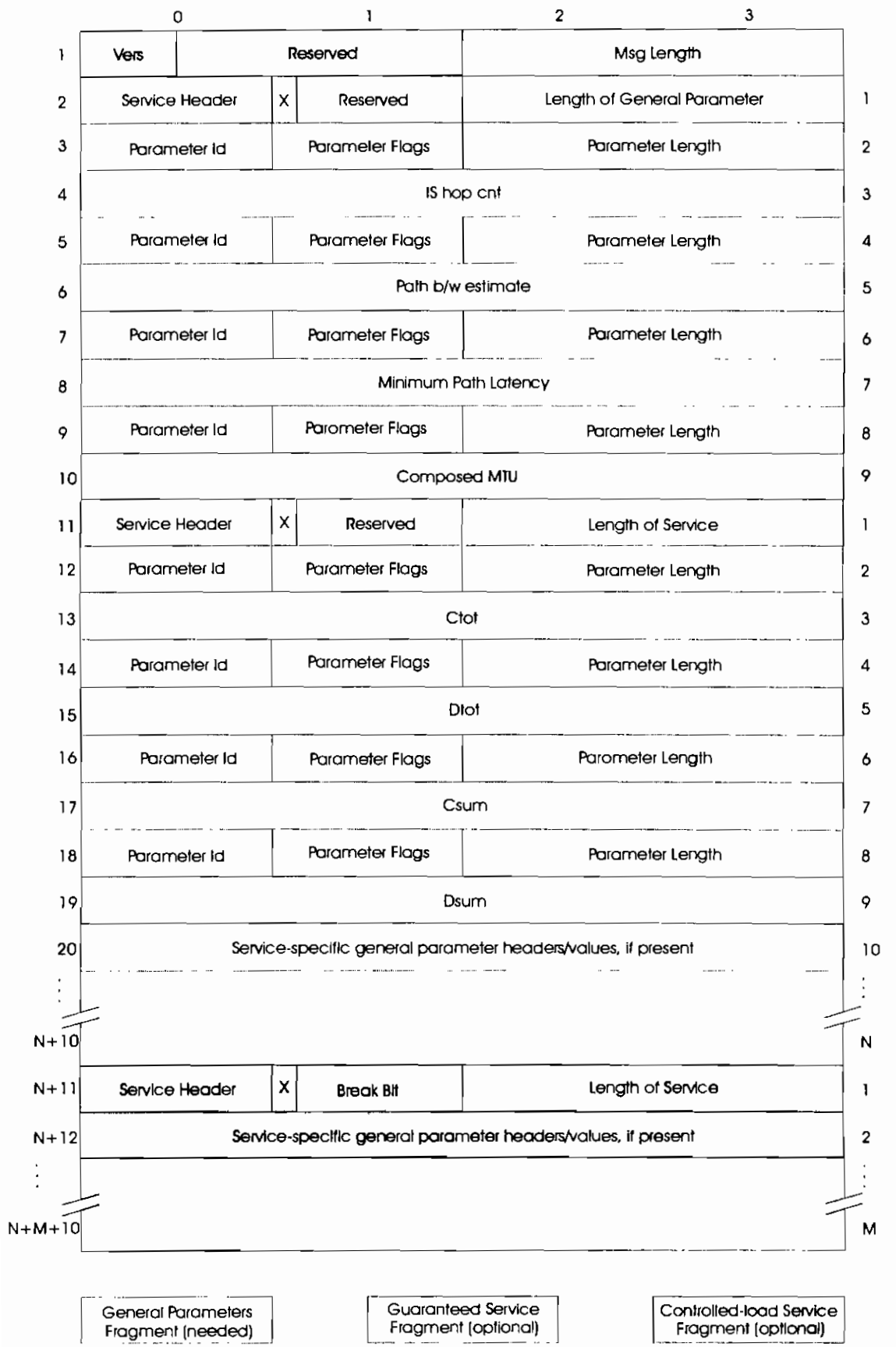

Figura 18 Objeto *ADSPEC*

Parameter Flags [1 byte]: Este campo corresponde a las banderas que pueden utilizar los parámetros. Aún no se especifican valores para ellas.

Parameter Length [2 bytes]: Especifica la longitud, en palabras de 32 bits, del parámetro sin incluir su cabecera. Su valor es 1 en todos los casos.

Los parámetros "*IS hop cnt*", "*Path* b/w estimate", "*Minimum Path Latency*" y "*Composed* MTU" determinan el número de saltos, un estimativo de b, la latencia y el MTU resultante del camino respectivamente.

Los parámetros C_{tot}, D_{tot}, C_{sum}, y D_{sum} se explican en la parte concerniente a los servicios garantizados, numeral 5.2.2.4.

13. ***POLICY_DATA*** [*Class-Num* = 14]

Este objeto fue creado para transportar datos que permitan conocer si un usuario está administrativamente permitido de hacer un requerimiento de calidad de servicio, sin embargo aún no se especifican parámetros para este objeto. Puede aparecer en los mensajes *Path*, *Resv*, *PathErr* y *ResvErr*.

14. ***RESV_CONFIRM*** [*Class-Num* =15]

Transporta la dirección IP del receptor que pide confirmación de su requerimiento, por lo tanto puede aparecer en los mensajes *Resv* o *Resv*Conf.

IPv4 [*C-Type* = 1]

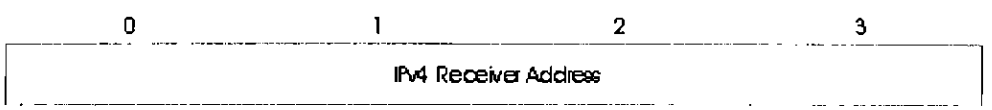

Figura 19 Objeto *RESV CONFIRM* para IPv4.

IPv6 [*C-Type* = 2]

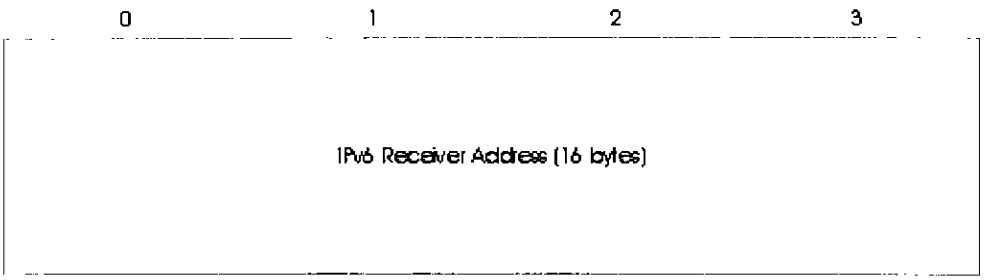

Figura 20 Objeto *RESV CONFIRM* para IPv6.

A5.4 INTERFACES DEL PROTOCOLO RSVP

El protocolo RSVP tiene interfaces para enrutamiento y control de tráfico en un *router*; e, interfaces para la aplicación y control de tráfico (si existe) en un *host*.

A5.4.1 Interfaz Aplicación / RSVP

Esta interfaz depende del sistema operativo que utilice la aplicación, sin embargo a continuación se exponen los delineamientos generales:

- ***Registro de Sesión***

Esta llamada inicia un proceso RSVP para generar una sesión, si se logra levantar la sesión, se le asigna a ésta una identificación (*Session-id*). El formato y los objetos utilizados se detallan a continuación; nuevamente los objetos entre corchetes son opcionales.

Call: SESSION(DestAddress , ProtocolId, DstPort

[, SESSION_object]

[, Upcall_Proc_addr]) -> Session-id

- ***Definición del transmisor***

Esta llamada permite al transmisor definir las características del tráfico que va a generar. El formato es el siguiente:

Call: SENDER(Session-id

[, Source_Address] [, Source_Port]

[, Sender_Template]

[, Sender_Tspec] [, Adspec]

[, Data_TTL] [, Policy_data])

- ***Reservación***

Esta llamada permite al receptor iniciar o modificar una reservación para una determinada sesión. El formato es el siguiente:

Call: RESERVE(session-id, [receiver address ,]

[CONF_flag,] [Policy_data,]

style, style-dependent-parms)

- ***Liberación de la Sesión***

Esta llamada libera el estado de reservación de la sesión especificada. Como consecuencia, el nodo puede enviar los correspondientes mensajes *teardown*. El formato es:

Call: RELEASE(session-id)

- ***Llamadas de Error y Eventos***

Estas llamadas utilizan transmisión asincrónica y su objetivo es informar el acontecimiento de eventos y errores sucitados durante la sesión. El formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id, Info_type,

information_parameters

Existen varios tipos de errores y eventos, los cuales se determinan en el campo "*Info_type*", éstos son:

PATH_EVENT

Se produce luego de la recepción o cambio del estado de la ruta (*Path* State), es indicativo de la existencia de transmisores listos para iniciar una reservación. Su formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id,

Info_type=*PATH*_EVENT,

Sender_Tspec, Sender_Template

[, Adspec] [, Policy_data]

RESV_EVENT

Se produce después de la recepción de un mensaje *Resv*, el cual puede iniciar o modificar una reservación. Su formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id,

Info_type=*RESV*_EVENT,
Style, *Flowspec*, Filter_Spec_list

[, Policy_data]

PATH_ERROR

Notifica la existencia de un error en la información del transmisor, la cual fue especificada en la llamada de registro de sesión. Su formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id,

Info_type=*PATH*_ERROR,

Error_code , Error_value ,

Error_Node , Sender_Template

[, Policy_data_list]

RESV_ERROR

Indica error en el mensaje de reservación, su formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id,

Info_type=*RESV*_ERROR,

Error_code , Error_value ,

Error_Node , Error_flags ,
Flowspec, Filter_spec_list

[, Policy_data_list]

RESV_CONFIRM

Este evento indica la recepción del mensaje *ResvConf*; su formato es el siguiente:

Upcall: <Upcall_Proc>( ) -> session-id,

Info_type=*RESV*_CONFIRM,

Style, List_count,

Flowspec, Filter_spec_list

[, Policy_data]

A5.4.2 Interfaz RSVP / Control de Tráfico

Esta interfaz depende de la tecnología de capa enlace que se utilice, sin embargo, a continuación se detallan los procesos generales.

- ***Hacer una reservación***

Esta llamada sirve para establecer una reservación denominada "*RHandle*" para cuyo propósito utiliza parámetros que definen, por ejemplo, el nivel de QoS efectivo (*TC_Flowspec*) y el tráfico efectivo (*TC_Tspec*). Su formato es el siguiente:

Call: TC_Add*Flowspec*(Interface, TC_*Flowspec*,

TC_Tspec, TC_Adspec, Police_Flags)

-> RHandle [, Fwd_*Flowspec*]

- ***Modificar Reservaciones***

Esta llamada sirve para modificar una reservación previamente establecida. Su formato es el siguiente:

Call: TC_Mod*Flowspec*(Interface, RHandle, TC_*Flowspec*,

TC_Tspec, TC_Adspec, Police_flags)

[-> Fwd_*Flowspec*]

- ***Suprimir Flowspec***

Esta llamada suprime una reservación existente. Su formato es el siguiente:

Call: TC_Del*Flowspec*(Interface, RHandle)

- ***Añadir Filter spec***

Esta llamada añade un *filter spec* a la reservación existente especificada por *RHandle*. Su formato es el siguiente:

Call: TC_AddFilter(Interface, RHandle,

Session , FilterSpec) -> FHandle

- ***Suprimir Filter spec***

Esta llamada suprime un *filter spec* especifico de la reservación *RHandle*; su formato es el siguiente:

Call: TC_DelFilter(Interface, FHandle)

- ***Actualizar OPWA***

Esta llamada se usa para calcular el *Adspec* en una interfaz saliente, además contiene un bit que indica si existen saltos previos que no soportan el protocolo RSVP. El formato es el siguiente:

Call: TC_Advertise(Interface, Adspec,

Non_RSVP_Hop_flag) -> New_Adspec

- ***Llamada Preferente***

Es usada para garantizar un nuevo requerimiento de reservación, para ello especifica la reservación y la razón por la cual se desea dar preferencia. Su formato es el siguiente:

Upcall: TC_Preempt() -> RHandle, Reason code

A5.4.3 Interface RSVP / Enrutamiento

El protocolo RSVP necesita el soporte especificado a continuación, el cual es proporcionado por los mecanismos de enrutamiento del nodo:

- ***Averiguación de la Ruta***

Los mensajes RSVP *Path* y *PathTear* necesitan el conocimiento de la ruta, dado por el protocolo de enrutamiento, para llegar al destino. El formato de la averiguación depende de la naturaleza del protocolo de enrutamiento, la que puede ser *unicast* o *multicast*; además la presencia de los campos opcionales depende del protocolo que se utilice. Los formatos son los siguientes:

Ucast_Route_Query([SrcAddress,] DestAddress,
Notify_flag) -> OutInterface

Mcast_Route_Query([SrcAddress,] DestAddress,
Notify_flag)
-> [IncInterface,] OutInterface_list

- ***Notificación de cambio en la ruta***

Esta notificación es dada por el protocolo de enrutamiento ante un pedido del proceso RSVP especificado con la bandera *Notify_flag true* en la averiguación de la ruta. A continuación se detalla el formato para protocolos *unicast* y *multicast* respectivamente.

Ucast_Route_Change( ) -> [SrcAddress,] DestAddress,
OutInterface

Mcast_Route_Change( ) -> [SrcAddress,] DestAddress,
[IncInterface,] OutInterface_list

- ***Descubrimiento de una lista de interfaces***

El protocolo RSVP debe conocer las interfaces reales y virtuales, con sus respectivas direcciones IP, y ser capaz de deshabilitar alguna interfaz para mensajes RSVP.

A5.4.4 Interfaz RSVP / Direccionamiento

El protocolo RSVP necesita el siguiente soporte de los mecanismos de direccionamiento y envío de paquetes:

- ***Receptor en modo desvío***

Cuando un mensaje *Path*, *PathTear* o *ResvConf* ha llegado a un nodo hacia el cual no fue direccionado, éste debe ser capaz de desviar el mensaje hacia un programa RSVP para su procesamiento y posterior envío.

- ***Especificación del Enlace Saliente***

El protocolo RSVP debe ser capaz de forzar un datagrama (*multicast*) hacia una interfaz saliente virtual o real independientemente del protocolo de enrutamiento. Esto es usado para enviar varias versiones de mensajes *Path* a diferentes interfaces de salida, y en algunos casos evitar lazos en el enrutamiento.

- ***Especificación de la dirección de origen y del TTL***

El protocolo RSVP debe ser capaz de especificar la dirección de origen y el tiempo de vida del mensaje TTL (*Time To Live*) usados en el envío de mensajes *Path*.

Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource Reservation Protocol (RSVP) – Version 1 Functional Specification", RFC 2205, September 1997.

Wroclawski, J., "The Use of RSVP with IETF Integrated Services", RFC 2210, September 1997

ANEXO 6

MPLS (*MultiProtocol Label Switching*)

A6.1 INTRODUCCIÓN

MPLS es un estándar del IETF que surgió para conceptuar diferentes soluciones de conmutación multinivel, propuestas por distintos fabricantes a mitad de los años 90. Como concepto, MPLS es a veces un tanto difícil de explicar, como protocolo es bastante sencillo. MPLS se puede presentar como un sustituto de la conocida arquitectura IP sobre ATM, también como un protocolo para hacer túneles (sustituyendo a las técnicas habituales de "*tunneling*"); o bien, como una técnica para acelerar el envío de paquetes e incluso para eliminar por completo el enrutamiento.

En realidad, MPLS realiza un poco de todo eso, ya que integra sin discontinuidades los niveles 2 (capa enlace de datos) y 3 (capa red), combinando eficazmente las funciones de control del enrutamiento con la simplicidad y rapidez de la conmutación de nivel 2. Es por esta razón que a MPLS se le considera parte constituyente de una llamada "capa 2,5".

Cuando un paquete de un protocolo de capa red no orientado a conexión (tal como IP) viaja de un enrutador al siguiente, cada uno de los enrutadores hace una decisión independiente del envío para aquel paquete. Esto es, cada uno de ellos analiza la cabecera del paquete y ejecuta un algoritmo de enrutamiento de capa red, por lo que cada enrutador independientemente selecciona el próximo salto para el paquete, basándose en el resultado del análisis de la cabecera y de la ejecución del algoritmo de enrutamiento.

La cabecera de los paquetes comúnmente contiene mucha más información que la necesaria para seleccionar el próximo salto. Selección que suele ser dividida en dos funciones. La primera de las cuales clasifica los paquetes entrantes en un

FEC[1] *(Forwarding Equivalence Classes)*, en tanto que la segunda función asigna cada FEC al próximo salto.

En un envío tradicional de datos bajo IP, un enrutador considera que dos paquetes forman parte del mismo FEC si existe algún prefijo de red X en la tabla de enrutamiento del mencionado enrutador, tal que X sea el prefijo de selección más alto para esa dirección de destino del paquete. Como el paquete atraviesa la red, en cada salto se examina el paquete y se lo asigna a un FEC.

En MPLS, un paquete es asignado a un FEC en particular cuando el mismo ingresa a la red MPLS, siendo esta asignación la única que se realiza durante el tránsito del paquete por la red. El FEC al cual el paquete es asignado se lo indica mediante una etiqueta, misma que es un valor de longitud fija. Cuando un paquete es enviado a su próximo salto, la etiqueta es colocada junto a éste, esto es los paquetes son "etiquetados" antes de ser enviados.

A saltos posteriores, no se realizan análisis de la cabecera de capa red presente en los paquetes, en lugar de ello se usa la etiqueta como una forma de indicador dentro de la tabla de enrutamiento mediante el cual se especifica el próximo salto y la nueva etiqueta.

A6.2 VENTAJAS DE MPLS

Dentro del envío de datos mediante MPLS, un paquete es asignado a un FEC una sola vez, no se realizan futuros análisis de cabeceras en los enrutadores subsecuentes de la red MPLS, y todo el proceso de encaminamiento de paquetes se lo realiza mediante etiquetas, como se analizará mas adelante. Esta funcionalidad de MPLS, ofrece ciertas ventajas sobre las redes de envío convencional de capa red, entre las cuales se puede citar las siguientes:

- El encaminamiento de datos mediante MPLS puede ser llevado a cabo por enrutadores LSRs (*Label Switching Routers*) que sean capaces de entender, manipular y reemplazar etiquetas, aún cuando éstos pueden ser no capaces de

[1] Un FEC es un conjunto de paquetes que se envían sobre el mismo camino a través de una red, aún cuando sus destinos finales sean distintos.

analizar las cabeceras de capa red de los paquetes circulantes por la red, o de no hacerlo a velocidades adecuadas. Lo que no sucede en la Arquitectura de Servicios Diferenciados en la que la presencia de un nodo no DiffServ podría dañar el esquema de ejecución diseñado.

- Un paquete que ingresa a la red por un enrutador específico puede ser etiquetado en forma distinta que si el mismo paquete hubiese entrado a la red por otro enrutador, y como resultado de esto la decisión de envío del paquete es realizada por el nodo de ingreso. Esto no puede realizarse con envío tradicional, puesto que en él la identidad del enrutador de ingreso de los paquetes no viaja junto con ellos.

- Las consideraciones a tomarse en cuenta para determinar como un paquete es asignado a un FEC pueden llegar a ser muy complejas, sin que esto afecte a la totalidad de los enrutadores que únicamente deben concentrarse en enviar paquetes etiquetados.

- Mediante MPLS es posible forzar a un paquete, seguir una ruta en particular la cual sea explícitamente seleccionada antes de que el paquete ingrese a la red, en lugar de que un algoritmo de enrutamiento dinámico seleccione la ruta a seguirse mientras el paquete viaja a través de la red.

A6.3 ETIQUETAS

Una etiqueta es un identificador de longitud fija, de significado local la cual es usada para identificar un FEC. Por lo tanto una etiqueta colocada sobre un paquete en particular representa el FEC al cual aquel paquete ha sido asignado y de ninguna manera la etiqueta es una codificación de la dirección de destino del paquete, aún cuando un paquete puede ser asignado a un FEC sobre la base de su dirección de destino de capa red.

Realmente, una etiqueta es similar a un identificador de conexión (como el VPI/VCI de ATM o el DLCI de Frame Relay). Al tener solamente significado local no modifica la información de la cabecera de los paquetes; tan sólo los encapsula, asignando el tráfico a los correspondientes FEC.

Con el objeto de entender y de clarificar de mejor manera los conceptos de etiquetas expresados en el párrafo anterior, a continuación se da un ejemplo acerca de la utilización de las mismas.

> "Si Ru y Rd son LSRs, ellos pueden llegar a establecer un acuerdo mediante el cual cuando Ru transmita un paquete a Rd, Ru etiquetará los paquetes con el valor L si y solo si el paquete pertenece a un particular FEC F. Esto es, ellos pueden establecer la etiqueta L a fin de representar el FEC F para todos aquellos paquetes que circulen de Ru a Rd.
>
> Esto se lo puede hacer debido a que la etiqueta es de significado local, y por tanto los enrutadores pueden asignar una etiqueta al FEC que ellos elijan. Es decir L es un valor arbitrario cuya " atadura " al FEC F es únicamente local para Ru y Rd.
>
> Serán etiquetados, todo aquellos paquetes que perteneciendo al FEC F circulen de Ru a Rd, y no solo los paquetes que sean generados por Ru o tengan como destino a Rd".

Algunos enrutadores analizan la cabecera de un paquete de capa red, no solo con el propósito de seleccionar el próximo salto del paquete, sino para determinar la "clase de servicio" a la cual será sujeta el paquete. MPLS no requiere que la precedencia o clase de servicio sea completa o parcialmente deducida de la etiqueta.

Pero, ante todo MPLS es el avance más reciente en la evolución de las tecnologías de enrutamiento y envío de paquetes en las redes IP, lo que implica una evolución en la manera de construir y gestionar estas redes.

Los problemas que presentan las soluciones actuales de IP sobre ATM, tales como la expansión sobre una topología virtual superpuesta, así como la complejidad de gestión de dos redes separadas y tecnológicamente diferentes, quedan resueltos con MPLS. Al combinar en uno solo lo mejor de cada nivel (la inteligencia del enrutamiento con la rapidez de la conmutación) MPLS ofrece nuevas posibilidades en la gestión de *backbones*, así como en la provisión de nuevos servicios de valor añadido. Para poder entender mejor las ventajas de la solución MPLS, vale la pena revisar antes los esfuerzos anteriores de integración de los niveles 2 y 3 que han llevado finalmente a la adopción del estándar MPLS.

A6.4 INTEGRACIÓN DE NIVELES 2 Y 3

A mediados de los años 90, IP fue imponiéndose como protocolo de red a otras arquitecturas en uso (SNA, IPX, AppleTalk, OSI, entre otras). Por otro lado, en esos mismos años los *backbones* IP que los proveedores de servicio (ISPs) habían empezado a desplegar estaban construidos en base a enrutadores conectados por líneas dedicadas T1/E1 y T3/E3. El crecimiento explosivo de la Internet había generado un déficit de ancho de banda en aquel esquema de enlaces individuales, por lo que la respuesta de los ISPs fue el incremento del número de enlaces y de la capacidad de los mismos. Del mismo modo, los ISPs se plantearon la necesidad de aprovechar mejor los recursos de red existentes, sobre todo la utilización eficaz del ancho de banda de todos los enlaces. Con los protocolos habituales de encaminamiento (basados en métricas del menor número de saltos), ese aprovechamiento del ancho de banda global no resultaba efectivo. Había que idear otras soluciones o alternativas de ingeniería de tráfico.

A6.4.1 IP SOBRE ATM

Como consecuencia, se impulsaron los esfuerzos para poder aumentar el rendimiento de los enrutadores tradicionales. Estos esfuerzos trataban de combinar, de diversas maneras, la eficacia y la rentabilidad de los conmutadores ATM con las capacidades de control de los enrutadores IP.

A favor de integrar los niveles 2 (enlace de datos) y 3 (red) estaba el hecho que los operadores de telecomunicaciones estaban desplegando infraestructuras de redes ATM, estas redes ofrecían entonces (1995-97) una buena solución a los problemas de crecimiento de los ISPs. Por un lado, proporcionaba mayores velocidades (155 Mpbs) y, por otro, las características de respuesta determinísticas de los circuitos virtuales ATM posibilitaban la implementación de soluciones de ingeniería de tráfico.

El modelo de red "IP sobre ATM" (IP/ATM) pronto ganó adeptos entre la comunidad de ISPs, a la vez que facilitó la entrada de los operadores telefónicos en la provisión de servicios IP y de conexión a la Internet al por mayor.

El funcionamiento IP/ATM supone la superposición de una topología virtual de enrutadores IP sobre una topología real de conmutadores ATM, mediante la cual el *backbone* ATM se presenta como una nube central (el núcleo) rodeada por enrutadores en la periferia. Cada enrutador se comunica con el resto mediante los circuitos virtuales permanentes (PVCs) que se establecen sobre la topología física de la red ATM.

Los PVCs actúan como circuitos lógicos y proporcionan la conectividad necesaria entre los enrutadores de la periferia. Éstos, sin embargo, desconocen la topología real de la infraestructura ATM que sustenta los PVCs. Los enrutadores ven los PVCs como enlaces punto a punto entre cada par de ellos. En la figura 1 se representa un ejemplo en el que se puede comparar la diferencia entre la topología física de una red ATM con la de la topología lógica IP superpuesta.

Así en la figura 1, se puede notar que los enrutadores IP de la periferia "sienten" estar unidos a través de enlaces punto a punto mediante los PVCs, por ejemplo el enrutador 1 está en comunicación con el enrutador 2 mediante el PVC 1, el enrutador 2 con el enrutador 3 mediante el PVC 3, y finalmente el enrutador 3 intercambia información con el enrutador 1 a través del PVC 2.

La base del modelo IP/ATM está en la funcionalidad proporcionada por el nivel ATM, es decir, los controles de software (señalización y enrutamiento) y el envío de las celdas por hardware (conmutación). En realidad, los PVCs se establecen a base de intercambiar etiquetas en cada conmutador de la red, de modo que la asociación de etiquetas entre todos los elementos ATM determina los correspondientes PVCs. Las etiquetas tienen solamente significado local en los conmutadores y son la base de la rapidez en la conmutación de celdas. El gran potencial de esta solución de topologías superpuestas está en la infraestructura ATM del *backbone*; el papel de los enrutadores IP queda relegado a la periferia.

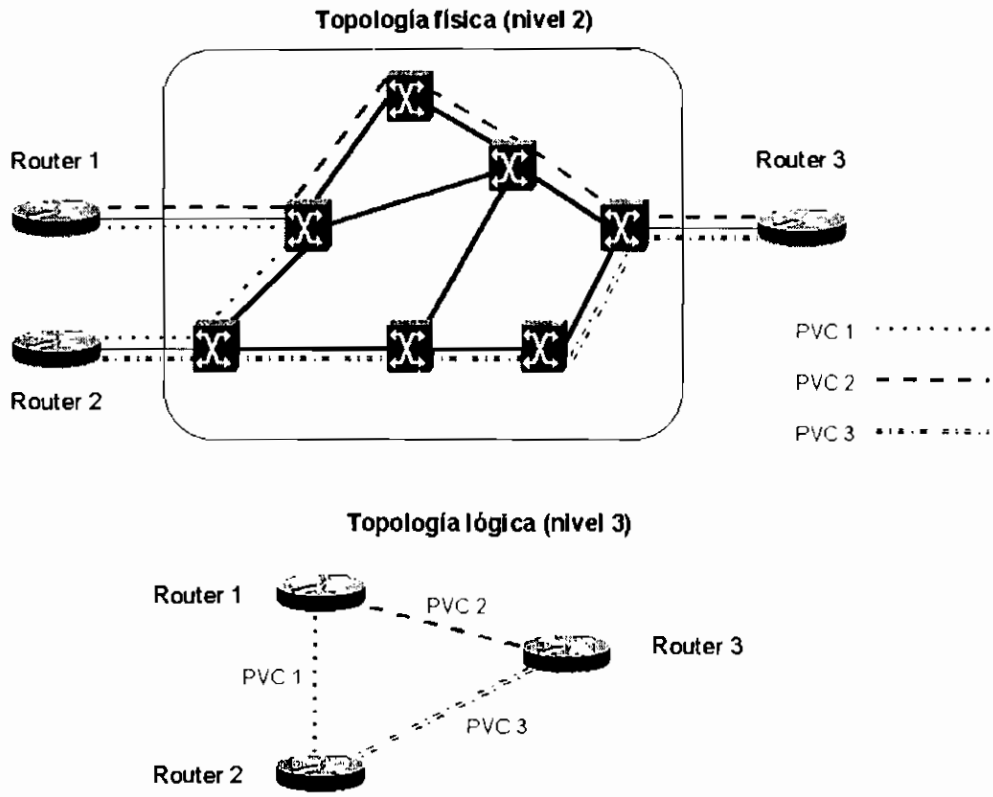

Figura 1. Topología física ATM (nivel 2) y topología lógica IP (nivel 3) superpuesta

En la figura 2 se representa el modelo IP/ATM con la separación de funciones entre lo que es encaminamiento IP en el nivel 3 (control y envío de paquetes) y lo que es conmutación en el nivel 2 (control/señalización y envío de celdas). Aún cuando se trata de una misma infraestructura física, en realidad existen dos redes separadas, con diferentes tecnologías, con diferente funcionamiento y, lo que quizás es más sorprendente concebidas para dos finalidades totalmente distintas.

La solución de superponer IP sobre ATM permite aprovechar la infraestructura ATM existente. Las ventajas inmediatas son el ancho de banda disponible a precios competitivos y la rapidez de transporte de datos que proporcionan los conmutadores. En los casos de ISPs de primer nivel, ellos poseen y operan el *backbone* ATM al servicio de sus redes IP. Los caminos físicos de los PVCs se calculan a partir de la necesidades del tráfico IP, utilizando la clase de servicio ATM UBR (*Unspecified Bit Rate*), ya que en este caso el ATM se utiliza solamente como infraestructura de transporte de alta velocidad (no hay necesidad de

apoyarse en los mecanismos inherentes del ATM para control de la congestión y clases de servicio).

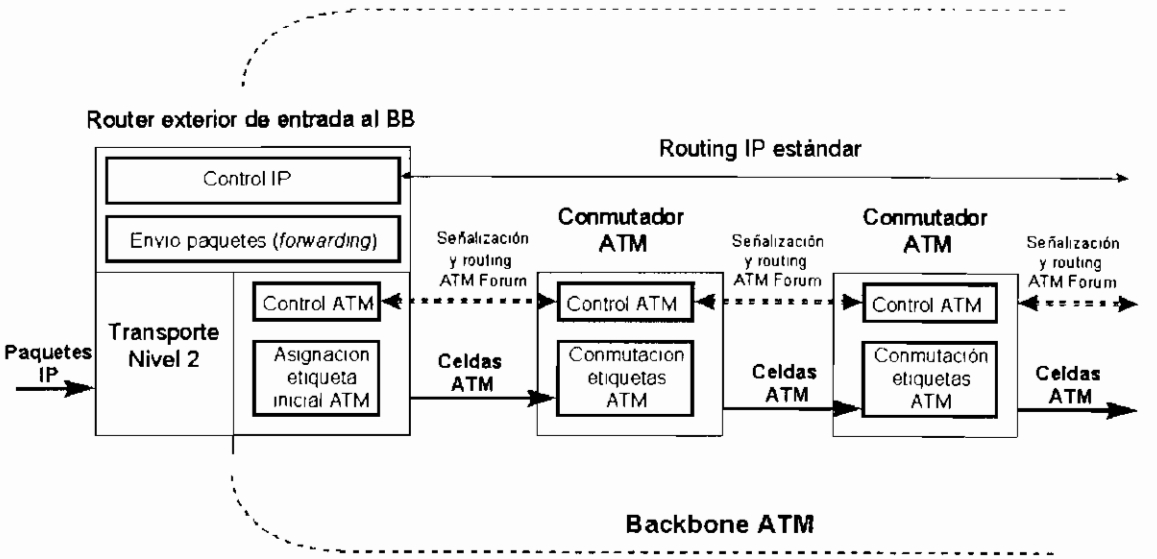

Figura 2. Modelo funcional IP sobre ATM

La ingeniería de tráfico se realiza proporcionando a los enrutadores los PVCs necesarios, con una topología lógica entre ellos totalmente mallada. Lo habitual es que entre cada par de enrutadores, exista un PVC principal y otro de respaldo.

Sin embargo, el modelo IP/ATM posee también sus inconvenientes, entre los cuales se tiene:

- Hay que gestionar dos redes diferentes, una infraestructura ATM y una red lógica IP superpuesta, lo que supone a los proveedores de servicio unos mayores costes de gestión global de sus redes.

- Existe, además, lo que se llama la "tasa impuesta por la celda", un *overhead* aproximado del 20% que el transporte de datagramas IP impone sobre las celdas ATM y que reduce en ese mismo porcentaje el ancho de banda disponible.

- Al estar concebida la solución IP/ATM sobre una topología completamente mallada se presentan, al aumentar el número de nodos IP, los típicos

problemas de crecimiento exponencial n x (n-1). Así por ejemplo en una red con 9 enrutadores externos con una topología virtual totalmente mallada sobre una red ATM, son necesarios 9 x 8 = 72 PVCs (uno en cada sentido de transmisión). Si se añade un décimo enrutador se necesitan 18 PVCs más para mantener la misma estructura (10 x 9 = 90). Adicionalmente el crecimiento exponencial de rutas involucra que el correspondiente protocolo IGP tenga que realizar un mayor esfuerzo.

Como conclusión, se puede decir que el modelo IP/ATM, si bien presenta ventajas evidentes en la integración de los niveles 2 y 3, lo hace de modo discontinuo, a base de mantener dos redes separadas. El MPLS, tal como se verá en las secciones siguientes, ofrece la ventaja de lograr esa misma integración de niveles sin discontinuidades.

A6.4.2 CONMUTACIÓN IP

La convergencia hacia IP de todas las aplicaciones existentes, junto a los problemas de rendimiento derivados de la solución IP/ATM, motivaron posteriormente (1997-98) a que varios fabricantes desarrollasen técnicas para alcanzar la integración de niveles en forma efectiva. Esas técnicas se conocieron como "conmutación IP" (*IP switching*) o "conmutación multinivel" (*multilayer switching*), su desarrollo originó una serie de tecnologías privadas, entre las que merecen citarse:

- IP Switching de Ipsilon Networks,
- Tag Switching de Cisco,
- Aggregate Route-Base IP Switching (ARIS) de IBM,
- IP Navigator de Cascade/Ascend/Lucent y
- Cell Switching Enrutador (CSR) de Toshiba

Tecnologías que condujeron finalmente a la adopción del actual estándar MPLS del IETF. El problema que presentaban tales soluciones era la falta de interoperatividad, ya que usaban diferentes tecnologías privadas para combinar la conmutación de nivel 2 con el encaminamiento IP (nivel 3).

Todas las soluciones de conmutación multinivel (incluido MPLS) se basan en dos componentes básicos comunes:

1) la separación entre las funciones de control (*routing*) y de envío (*forwarding*).

2) el envío de datos mediante la conmutación de intercambio de etiquetas.

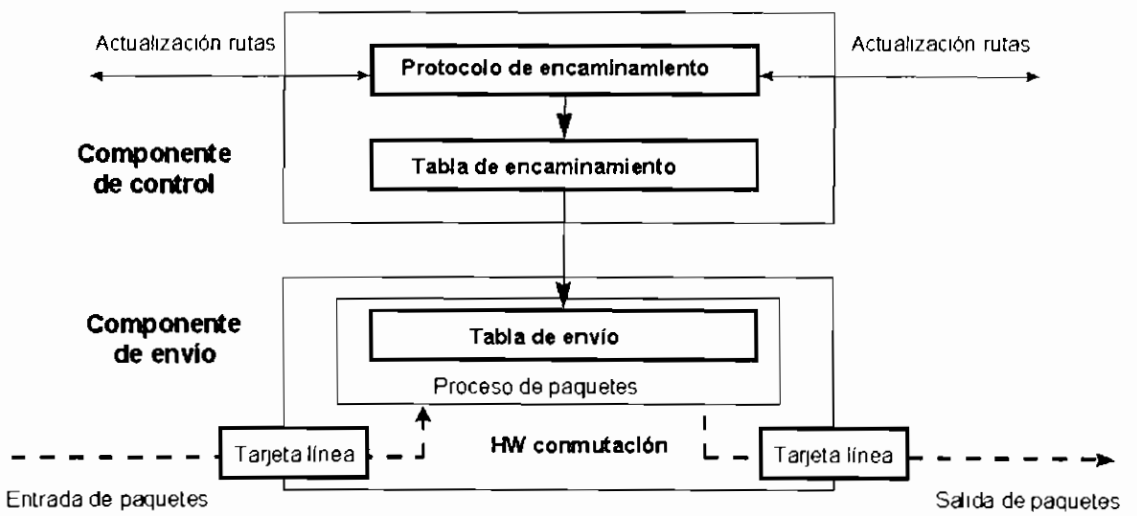

Figura 3. Separación funcional de encaminamiento y envío

En la figura 3 se representa la separación funcional de esas dos componentes, la componente de control utiliza los protocolos estándar de encaminamiento (OSPF, IS-IS y BGP-4) para el intercambio de información con los otros enrutadores a fin de llevar a cabo la construcción y el mantenimiento de las tablas de encaminamiento. Al llegar los paquetes, la componente de envío examina la información de la cabecera del paquete, busca en la tabla de envío la entrada correspondiente y dirige el paquete desde el interfaz de entrada al de salida a través del correspondiente hardware de conmutación.

Al separar la componente de control (encaminamiento) de la componente de envío, cada una de ellas se puede implementar y modificar independientemente. El único requisito es que la componente de encaminamiento mantenga la comunicación con la de envío mediante la tabla de envío de paquetes y actualice

la información. El mecanismo de envío se implementa mediante el intercambio de etiquetas, similar a lo visto para ATM. La diferencia está en que ahora lo que se envía por el interfaz físico de salida son paquetes "etiquetados". De este modo, se está integrando realmente en el mismo sistema las funciones de conmutación y de encaminamiento.

El algoritmo de intercambio de etiquetas permite así la creación de "caminos virtuales" conocidos como LSPs[1] (*Label-Switched Paths*), funcionalmente equivalentes a los PVCs de ATM y Frame Relay. En el fondo, lo que hace es imponer una conectividad entre extremos a una red no conectiva por naturaleza, como son las redes IP, pero todo ello sin perder la visibilidad del nivel de red .

Esta es la diferencia básica con el modelo IP/ATM. Al hablar de MPLS con más detalle se entenderán mejor estas características.

A6.4.3 LA CONVERGENCIA REAL, MPLS

Como se analizó anteriormente, el principal problema que presentaban las diversas soluciones de conmutación multinivel era la falta de interoperatividad entre productos privados de diferentes fabricantes. Además de ello, la mayoría de esas soluciones necesitaban ATM como transporte, pues no podían operar sobre infraestructuras de transmisión mixtas (Frame Relay, PPP, SONET/SDH y LANs). Se quería obtener un estándar que pudiera funcionar sobre cualquier tecnología de transporte de datos en el nivel de enlace. De aquí que el Grupo de Trabajo de MPLS que se estableció en el IETF en 1977 se propuso como objetivo la adopción de un estándar unificado e interoperativo.

A6.4.3.1 Descripción funcional de MPLS

La asignación e intercambio de etiquetas constituye la base funcional de MPLS, esto permite el establecimiento de los caminos LSP a través de la red. Los LSPs son *simplex* por naturaleza, por lo que el tráfico *dúplex* requiere el establecimiento de dos LSPs, uno en cada sentido. Cada LSP se crea a base de concatenar uno o más saltos (*hops*) en los que se intercambian las etiquetas, de modo que cada

[1] Un camino LSP es el circuito virtual que siguen por la red todos los paquetes asignados a la misma FEC

paquete se envía de un "conmutador de etiquetas" a otro, a través del dominio MPLS.

Al igual que en las soluciones de conmutación multinivel, MPLS separa las dos componentes funcionales de control (*routing*) y de envío (*forwarding*). Del mismo modo, el envío se implementa mediante el intercambio de etiquetas en los LSPs.

Sin embargo, MPLS no utiliza ninguno de los protocolos de señalización ni de encaminamiento definidos por el ATM Forum; en lugar de ello, en MPLS se utiliza el protocolo RSVP o bien el nuevo estándar de señalización LDP (*Label Distribution Protocol)*. Pero, de acuerdo con los requisitos del IETF, el transporte de datos puede ser cualquiera. Si éste fuera ATM, una red IP habilitada para MPLS es mucho más sencilla de gestionar que la clásica solución IP/ATM, pues ya no hay que administrar dos arquitecturas diferentes a base de transformar direcciones y las tablas de encaminamiento IP, en direcciones y conmutación ATM: esto lo resuelve el procedimiento de intercambio de etiquetas MPLS. El papel de ATM queda restringido únicamente al transporte de datos a base de la conmutación de celdas. Para MPLS esto es indiferente, ya que puede utilizar otros transportes como Frame Relay, o trabajar directamente sobre líneas punto a punto. En la figura 4 se presenta el esquema funcional de MPLS.

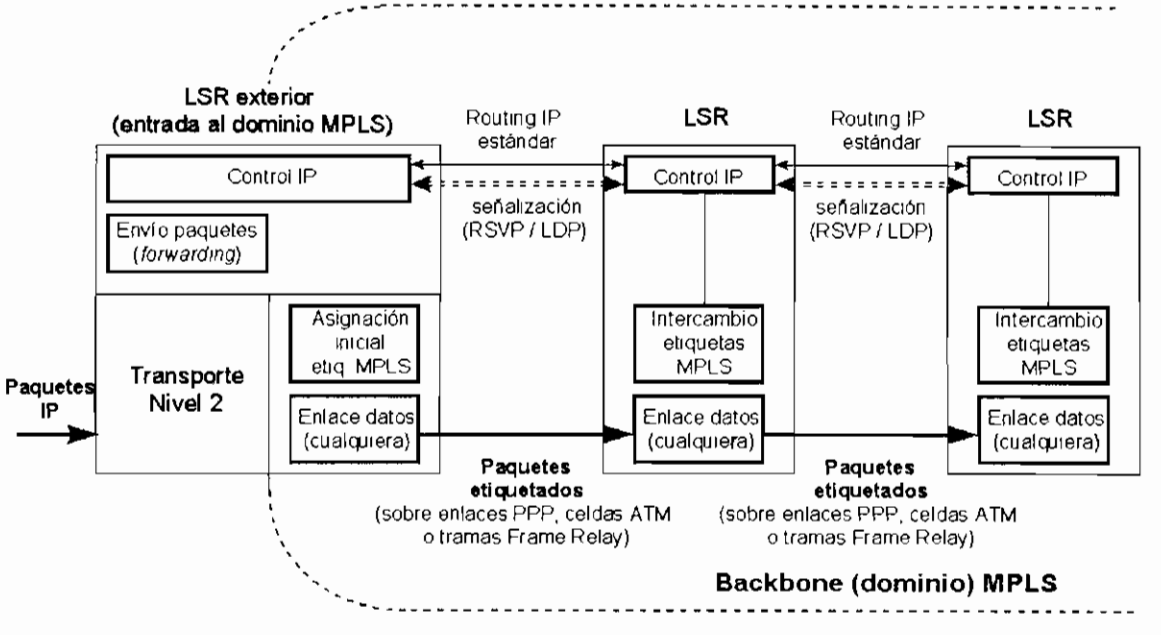

Figura 4. Esquema funcional de MPLS

Al primer LSR que interviene en un LSP se le denomina de entrada o de cabecera y al último se le denomina de salida o de cola. Los dos están en el exterior del dominio MPLS. El resto, entre ambos, son LSRs interiores del dominio MPLS (observar figura 4).

Un LSR es un enrutador que funciona a base de intercambiar etiquetas según una tabla de envío, la cual se establece a partir de la información de encaminamiento que proporciona la componente de control. Cada entrada de la tabla contiene un par de etiquetas entrada/salida que se utilizan para enviar a cada paquete hacia una interfaz preestablecida. En los LSR exteriores sólo hay una etiqueta, de salida en el de LSR de entrada al dominio MPLS y de entrada en el LSR de salida.

En la figura 5 se ilustra un ejemplo del funcionamiento de un LRS del núcleo MPLS: un paquete llega al LSR por el interfaz de entrada 3 con la etiqueta 45, entonces el LSR busca en su tabla de envío la próxima etiqueta que debería asignar y conmuta la etiqueta a 22 enviando el paquete hacia el interfaz de salida 4 que se comunicará con otro LSR.

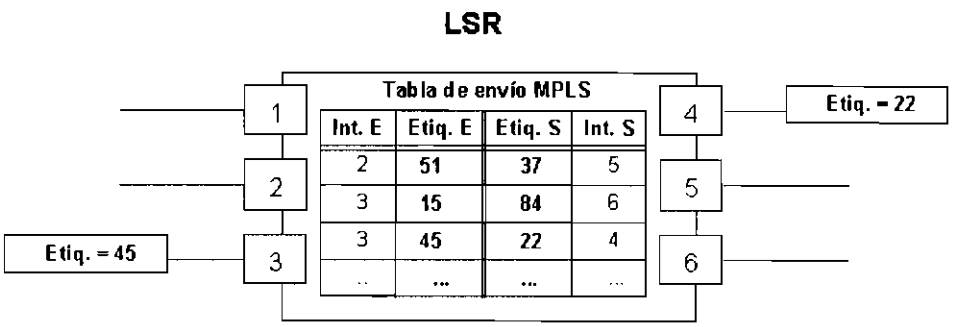

Figura 5. Detalle de la tabla de envío de un LSR

El algoritmo de intercambio de etiquetas requiere la clasificación de los paquetes a la entrada del dominio MPLS para poder realizar la asignación de etiquetas en su LSR de entrada. En la figura 6 el LSR de entrada recibe un paquete normal (sin etiquetar) cuya dirección de destino es 212.95.193.1. El LSR consulta la tabla de encaminamiento y asigna el paquete a la clase FEC definida por el grupo 212.95/16. De la misma forma, este LSR asigna al paquete una etiqueta (con valor 5 en el ejemplo) y lo envía al siguiente LSR del LSP.

En el interior del dominio MPLS los LSR ignoran la cabecera IP; solamente analizan la etiqueta de entrada, consultan la tabla correspondiente (tabla de conmutación de etiquetas) y la reemplazan por otra nueva.

Al llegar el paquete al LSR de salida, nota que el siguiente salto lo saca de la red MPLS; al consultar ahora la tabla de conmutación de etiquetas quita ésta y envía el paquete por enrutamiento convencional.

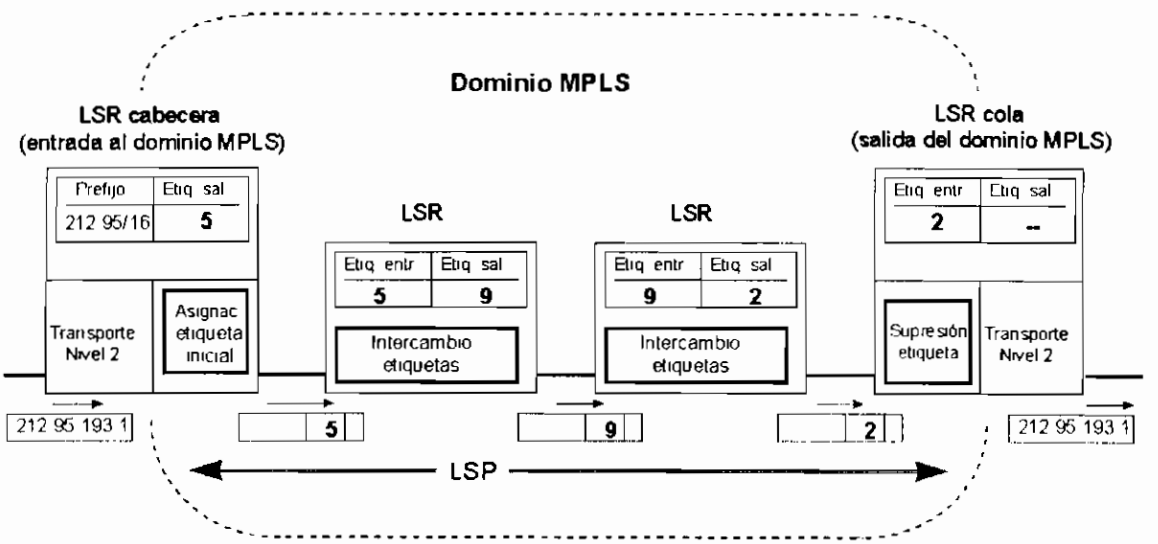

Figura 6. Ejemplo de envío de un paquete por un LSP

Como se puede notar, la identidad del paquete original IP queda enmascarada durante el transporte por la red MPLS que no "mira" sino las etiquetas que necesita para su envío por los diferentes saltos LSR que configuran los caminos LSP. Las etiquetas se insertan en cabeceras MPLS, entre los niveles 2 y 3. Según las especificaciones del IETF, MPLS debía funcionar sobre cualquier tipo de transporte: PPP, LAN, ATM, Frame Relay, etc.

Por ello, si el protocolo de transporte de datos contiene ya un campo para etiquetas (como ocurre con los campos VPI/VCI de ATM y DLCI de Frame Relay), se utilizan esos campos para efectuar la asignación de etiquetas MPLS. Sin embargo, si la tecnología de nivel 2 empleada no soporta un campo para etiquetas (p. ej. enlaces PPP o LAN), entonces se emplea una cabecera genérica MPLS de 4 octetos, que contiene un campo específico para la etiqueta y que se inserta entre la cabecera del nivel 2 y la del paquete (nivel 3).

En la figura 7 se representa el esquema de los campos de la cabecera genérica MPLS y su relación con las cabeceras de los otros niveles.

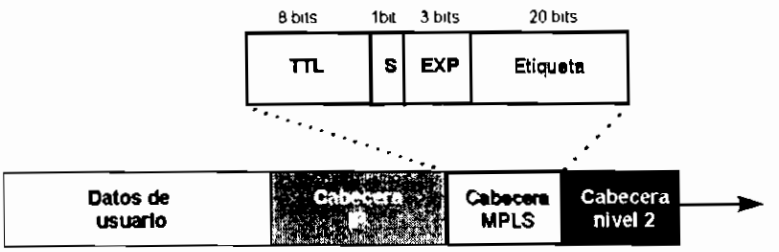

Figura 7. Estructura de la cabecera genérica MPLS

Según se muestra en la figura, una cabecera MPLS está formada por 32 bits que define los siguientes campos

- Etiqueta: formada por 20 bits, constituye la etiqueta asignada por los LSRs.
- EXP (experimental); es un campo de 3 bits para identificar la clase de servicio que un paquete podría solicitar. Su uso actualmente es experimental.
- Stack (S): es un bit que permite apilar etiquetas en forma jerárquica.
- TTL: es un campo formado por 8 bits que básicamente sustenta la funcionalidad estándar TTL de redes IP.

Barbera, J., "MPLS: Una Arquitectura de backbone para la Internet del siglo XXI". http://www.aul.es/biblio/libros/mi2000/Jose%20Barbera.htm.

Rosen, E., Viswanathan,A., and Callon, R., " MultiProtocol Label Switching Architecture". Internet Draft. Draft-ietf-mpls-arch-07.txt. July 2000.

GLOSARIO

A

ABR **Velocidad Disponible de Bit (*Available Bit Rate*):** Categoría de servicio de la arquitectura ATM diseñada para trabajar con ráfagas de datos, cuya gama de ancho de banda se conoce aproximadamente.

ADSL **Línea Asimétrica de Subscriptor Digital (*Asymmetric Digital Subscriber Line*):** Tipo de tecnología que permite la transmisión asincrónica de información digital a altas velocidades. Utiliza como medio de transmisión al par de cobre.

ANSI **Instituto de Estándares Americano (*American National Standars Institute*):** Organización, miembro de la ISO, que promueve el desarrollo de estándares en los Estados Unidos.

ATM **Modo de Transferencia Asincrónica (*Asynchronous Transfer Mode*):** Tecnología de conmutación de circuitos a altas velocidades utilizando celdas de longitud fija de 53 bytes, que le permiten cursar diferentes tipos de tráfico.

B

Best Effort Tratamiento predeterminado otorgado al tráfico en una red *TCP/IP* en ausencia de alguna característica de calidad de servicio.

C

CBQ **Encolamiento Basado en Clases (*Class Based Queuing*):** Es un método QoS de dominio público que permite clasificar paquetes y encolarlos de acuerdo a criterios establecidos por un administrador con el objetivo de proveer diferentes tratamientos a cada una de las clases de tráfico.

CBR **Velocidad Constante de Bit (*Constant Bit Rate*):** Categoría de servicio de ATM, la cual pretende simular un enlace dedicado.

CDV **Variación de Retardo de Celdas (*Cell Delay Variation*):** Parámetro de Calidad de Servicio definido en ATM. Indica la variación en el retardo de las celdas en su tránsito por la red.

CER **Razón de Errores de Celdas (*Cell Error Ratio*):** Parámetro de Calidad de Servicio definido en ATM. Indica la fracción de las celdas que se entregan con uno o más bits equivocados.

CLI **Interfaz de Línea de Comandos (*Command Line Interface*):** Interfaz de usuario para el sistema operativo de algún dispositivo de red, que permite su configuración por medio de instrucciones definidas (comandos).

CLR **Razón de Pérdida de Celdas (*Cell Loss Ratio*):** Parámetro de Calidad de Servicio definido en ATM. Mide la fracción de las celdas transmitidas que no se entregan en absoluto o que se entregan demasiado tarde.

CMR **Tasa de Mala inserción de Celdas (*Cell Missinsertion Rate*):** Parámetro de Calidad de Servicio definido en ATM. Corresponde a la cantidad de celdas por segundo que se entregan al destino equivocado

debido a un error no detectado en la cabecera.

COPS **Servicios Comunes de Políticas Abiertas (*Common Open Policy Service*):** Es un estándar propuesto por el IETF que define un protocolo para aprovisionar QoS mediante control de admisión basado en políticas.

COS **Clase de Servicio (*Class Of Service*):** Método utilizado para administrar el ancho de banda, enmarcando al tráfico en pocas categorías de servicio.

CR-LDP **Protocolo LDP de enrutamiento basado en restricciones (*Constraint based Routing LDP*):** Extensión del protocolo LDP que provee la información necesaria para controlar el enrutamiento LSP y especifica algunas otras características requeridas para el establecimiento de la conexión, tales como velocidad de transmisión.

CSMA/CD **Acceso múltiple con detección de portadora y detección de colisiones (*Carrier Sense Multiple Access with Collision Detection*):** Técnica de acceso a un canal de transmisión compartido, en la cual un nodo verifica la ausencia de algún otro tráfico antes de transmitir.

CTD **Retardo de Transferencia de Celdas (*Cell Transfer Delay*):** Parámetro de Calidad de Servicio que se define en ATM. Representa el tiempo promedio de tránsito de las celdas del origen al destino.

CVDT **Tolerancia de Variación del Retardo de Celdas (*Cell Variation Delay Tolerance*):** Parámetro de Calidad de Servicio que se define en ATM. Indica la cantidad de variación que habrá en los tiempos de transmisión de las celdas.

D

Dial up Conexión de un ordenador a la red del Internet por medio del módem a través de una línea telefónica convencional. Una velocidad de transmisión típica es 28.8 kbps.

DiffServ Acrónimo utilizado par referirse a la Arquitectura de Servicios Diferenciados.

DLCI **Identificador de la Conexión de enlace de Datos (*Data Link Connection Identifier*):** Campo utilizado en Frame Relay para identificar un circuito particular.

DSBM **Administrador Designado del Ancho de Banda de la Subred (*Designated Subnet Bandwidth Manager*):** Dispositivo perteneciente a una subred que actúa como el Administrador del ancho de banda de la misma.

DSCP **Campo de Servicios Diferenciados (*DiffServ CodePoint*):** Campo de seis bits definido en la cabecera IP, utilizado para clasificar el tráfico y otorgarle características diferenciadas de tratamiento dentro de la red.

E

ECN **Notificación Explícita de Congestión (*Explicit Congestion Notification*):** Método propuesto por el IETF para indicar a los nodos de una red la presencia de congestión en el camino de datos.

F

FDDI **Interfaz de Datos Distribuidos por Fibra (*Fiber Distributed Data Interface*):** Estándar establecido por la ANSI que define una tecnología de red basada en fibra óptica.

FIFO **Primero en Ingresar Primero en Salir (*First Input First Output*):** Disciplina de encolamiento en la cual los paquetes dejan la cola en el mismo orden en el que arribaron.

FRAD **Dispositivo de Acceso Frame Relay (*Frame Relay Access Device*):** Dispositivo de hardware que interconecta circuitos Frame Relay y redes IP.

Frame Relay Protocolo de conmutación de circuitos, generalmente utilizado en redes WAN. Su velocidad de transmisión usualmente varía entre 56 kbps y 1544 kbps (T1).

G

Gateway Ordenador dedicado que interconecta dos o más redes y encamina los paquetes de una red hacia la otra.

GE **Gigabit Ethernet (*Gigabit Ethernet*)**: Red estándar de área local, definida en la norma IEEE 802.3, la cual permite alcanzar velocidades de transmisión de 1 Gbps.

GPS **Procesador Generalizado de Compartición (*Generalized Processor Sharing*):** Es una disciplina ideal de compartición de recursos mediante la cual se programa el servicio que se otorga a

varias sesiones de tráfico activas.

GSM **Sistema Móvil Global (*Global System Mobile*):** Sistema telefónico móvil digital, adoptado como estándar *de facto* en Europa.

H

HDLC **Control de Enlace de Datos de Alto Nivel (*Hihg-level Data Link Control*):** Protocolo que permite transmitir datos entre nodos de redes. Es uno de los protocolos más utilizados en la capa 2 del modelo de referencia OSI.

HFC **Sistema Híbrido de Fibra y Coaxial (*Hybrid Fiber Coaxial*):** Tecnología de telecomunicaciones que posibilita la utilización de segmentos de fibra óptica y cable coaxial para transportar aplicaciones que requieren un gran ancho de banda, tales como audio y video.

Host En telecomunicaciones máquina o computadora anfitrión que está participando en el sistema de intercambio de información digital. De hecho un *host* es un dispositivo electrónico de datos primarios.

I

ICMP **Protocolo de Control de Mensajes de Internet (*Internet Control Message Protocol*):** Protocolo utilizado para cursar datos de control y reportar errores en la transmisión de paquetes.

IEEE **Instituto de Ingenieros Eléctricos y Electrónicos (*Institute of Electrical and Electronics Engineers*)**: Organización de Ingenieros en electrónica, computación y comunicaciones, con sede en EE.UU.,

dedicada al estudio y desarrollo de estándares de comunicaciones.

IETF **Fuerza de Trabajo en Ingeniería Internet (*Internet Engineering Task Force*):** Organización que desarrolla y especifica los protocolos y estándares de Internet, tales como TCP/IP.

IGP **Protocolo de Compuerta Interior (*Interior Gateway Protocol*):** Protocolo para intercambio de información de enrutamiento entre compuertas pertenecientes a una red autónoma (por ejemplo LANs de una misma empresa).

IntServ Acrónimo utilizado para referirse a la Arquitectura de Servicios Integrados.

IPG **Intervalo Entre Paquetes (*Inter Packet Gap*):** Intervalo de tiempo definido en algunos mecanismos de compartición del medio de transmisión (tal como CSMA/CD), que especifica un período de tiempo en el cual los transmisores ocasionalmente deben efectuar una pausa en la transmisión de sus datos.

ISA **Arquitectura Estándar de Fabricación (*Industry Standar Architecture*):** Estándar muy antiguo utilizado para la transferencia de información entre tarjetas de expansión y la tarjeta principal (*motherboard*) en un computador.

ISDN **Red Digital de Servicios Integrados (*Integrated Services Digital Networks*):** Red integrada desarrollada a partir de la red telefónica digital, la cual provee conectividad digital extremo a extremo, permitiendole soportar un amplio grupo de servicios, a los cuales los usuarios tienen acceso por un conjunto limitado de interfaces estándares multipropósito.

ISP **Proveedor de Servicios del Internet (*Internet Service Provider*):**

Empresa que provee el acceso a Internet a usuarios individuales o corporativos por medio de equipos de conectividad.

ITU **Unión Internacional de Telecomunicaciones (*International Telecommunication Union*):** Organización internacional que establece y promueve estándares para telecomunicaciones.

J

Jitter Es la variación del tiempo de arribo entre paquetes consecutivos debido, básicamente, a la diferencia de velocidad con la que éstos atraviesan la red y al retardo variable experimentado en las colas de los enrutadores.

JPEG ***Joint Photographic Experts Group*:** Técnica estándar utilizada para la compresión de imágenes.

L

LAN **Red de Área Local (*Local Area Network*):** Son redes privadas localizadas dentro de un edificio o un campus de pocos kilómetros de extensión.

Latencia Tiempo que tarda un paquete en llegar al destino desde su origen, en su tránsito por la red.

LDAP **Protocolo Liviano de Acceso a Directorios (*Lightweight Directory Access Protocol*):** Protocolo utilizado para acceder a un directorio central basado en el estándar X.500.

LDP **Protocolo de Distribución de Etiquetas (*Label Distribution Protocol*):** Conjunto de procedimientos y mensajes mediante el cual un router de conmutación de etiquetas establece un camino a través de una red.

LSP **Camino Conmutado de Etiquetas (*Label Switched Path*):** Ruta establecida a través de una red mediante la traducción de la información de enrutamiento (de capa red) a caminos conmutados (de capa enlace de datos).

LSR **Enrutador de Etiquetas Conmutadas (*Label Switched Router*):** Enrutador que "entiende" y puede trabajar con etiquetas conmutadas.

M

MAC **Subcapa de Acceso al Medio (*Medium Access Control*):** Parte de la capa enlace de datos que admite funciones dependientes de la topología y utiliza los servicios de la capa física para proporcionar servicios a la subcapa de control de enlace lógico (LLC).

MBS **Máximo tamaño de Ráfaga (*Maximum Burst Size*):** Parámetro definido en ATM, para especificar un tamaño máximo de datos consecutivos (ráfaga) que se permite enviar a un transmisor.

MCR **Mínima Velocidad de Celda (*Minimum Cell Rate*):** Parámetro definido en ATM, que especifica una mínima velocidad de celdas a la cual un transmisor se compromete a enviar sus datos.

MTU **Máxima Unidad de Transmisión (*Maximum Transmission Unit*):** Paquete de mayor longitud (especificado en bytes) que puede ser enviado a través de una red de paquetes (como TCP/IP).

N

nrtVBR **Velocidad Variable de Bit inadecuada para aplicaciones en tiempo real (*non real time Variable Bit Rate*):** Categoría de servicio de la arquitectura ATM, adecuada para la transmisión de aplicaciones que necesitan una garantía de envío en lugar de bajo retardo.

O

OSI **Interconexión de Sistemas Abiertos (*Open Systems Interconnection*):** Es el estándar que define el modelo de red referencial de siete capas.

P

PBX ***Private Branch eXchange*:** Sistema privado de conmutación telefónica que usualmente se conecta a la red telefónica conmutada pública.

PCR **Tasa Pico de Celdas (*Peak Cell Rate*):** Parámetro de calidad definido en ATM; representa la máxima velocidad con la que el transmisor planea enviar datos.

PDP **Punto de Decisión de Políticas (*Policy Decision Point*):** Componente básico de un sistema de políticas; es la entidad encargada de tomar decisiones en base a un conjunto de reglas.

PEP **Punto de Ejecución de Políticas (*Policy Enforcement Point*):** Es la

entidad encargada de ejecutar o hacer cumplir las reglas en un sistema de políticas.

PHB **Comportamiento Por Salto (*Per Hop Behavior*):** Tratamiento de envío otorgado a una específica clase de tráfico, en base al valor del DSCP del paquete. Los routers y switches utilizan los PHBs para determinar prioridades entre varios tipos de tráfico.

POP **Punto de Presencia (*Point Of Presence*):** Punto de acceso a Internet. El tamaño de un ISP se mide en función de los POPs que posee.

PRR ***Packet Round Robin*:** Técnica utilizada por los algoritmos de gestión de tráfico que consiste en asignar algún servicio a los paquetes en una manera secuencial y cíclica.

PSTN **Red Telefónica Pública Conmutada (*Public Switched Telephony Network*):** Red pública de telefonía orientada principalmente a la transmisión de tráfico de voz.

Q

QAM **Modulación de Amplitud en Cuadratura (*Quadrature Amplitude Modulation*):** Método de modulación de señales digitales sobre una señal portadora de radiofrecuencia que entraña la codificación en amplitud y fase.

QoS Acrónimo utilizado para referirse a Calidad de Servicio implementada en una red. Tecnología que posibilita el surgimiento de redes de nueva generación.

R

RED **Detección Aleatoria Temprana (*Random Early Detection*):** Algoritmo utilizado para detectar y evitar congestión. Trata de anticiparse a la congestión monitoreando la cola de un router y descartando aleatoriamente paquetes cuando el tamaño de la cola ha superado un umbral.

RFC **Petición de Comentarios (*Request For Comments*):** Documento de carácter técnico relativo a Internet, expedido por el IETF.

Router **Enrutador:** Dispositivo asociado a una red que direcciona o enruta paquetes de datos hacia un destino deseado.

RSpec **Especificación de Recursos (*Resource Specification*):** Parámetro definido en la Arquitectura de Servicios Integrados para especificar los recursos asignados por la red a algún flujo que los haya solicitado.

RSVP **Protocolo de Reservación de Recursos (*Resource ReSerVation Protocol*):** Protocolo utilizado principalmente en la Arquitectura de Servicios Integrados para establecer y mantener reservaciones de recursos que permitan suministrar calidad de servicio a un flujo de datos.

RTP **Protocolo de Transporte en Tiempo Real (*Real Time Transport Protocol*):** Protocolo diseñado para proveer funciones de transporte de red extremo a extremo para aplicaciones que transmiten datos en tiempo real (voz y video) sobre servicios de red unicast o multicast.

rtVBR **Velocidad Variable de bit para aplicaciones en tiempo real (*real time Variable Bit Rate*):** Categoría de servicio de ATM, adecuada para aplicaciones que necesitan sincronización de tiempo entre la

fuente y el destino.

S

SBM **Administrador del Ancho de Banda de la Subred (*Subnet Bandwith Manager*):** Estándar propuesto por el IETF para manejar reservación y compartición de recursos, así como conmutación bajo el estándar IEEE 802.3 en redes de área local.

SCR **Tasa Sustentable de Celdas (*Sustained Cell Rate*):** Parámetro de calidad de servicio definida en ATM, representa la tasa mínima de celdas por segundo que el cliente considera aceptable.

SECBR **Tasa de Bloques de Celdas con Errores Severos (*Severely-Errored Cell Block Ratio*):** Parámetro de calidad de servicio definida en ATM, correspondiente a la fracción de bloques de N celdas en los que M o más celdas contienen errores.

SLA **Acuerdo de Nivel de Servicio (*Service Level Agreement*):** Es un contrato celebrado entre el cliente y un proveedor de servicio, mediante el cual se especifica el tratamiento que recibirá de la red el tráfico del cliente.

SMTP **Protocolo Sencillo de Transporte de Correo Electrónico (*Simple Mail Transport Protocol*):** Protocolo estándar en Internet para la transferencia de mensajes mediante el uso de servicios de correo electrónico desde un ordenador a otro. STMP especifica tanto la interacción entre los sistemas de correo y el formato de los mensajes de control para la transferencia de correo.

SNMP **Protocolo Sencillo de Administración de Red (*Simple Network***

***Management Protocol*)**: Protocolo de administración de red basado en UDP, empleado principalmente en redes TCP/IP. SNMP puede ser utilizado para monitorear, censar y controlar dispositivos de red.

SONET **Red Sincrónica Óptica (*Synchronous Optical Network*)**: Estándar para la transmisión sincrónica de datos sobre un medio óptico. Maneja velocidades de aproximadamente 52 Mb/s (OC-1) a 10 Gb/s (OC-192).

T

TCA **Acuerdo de Acondicionamiento de Tráfico (*Traffic Conditioning Agreement*)**: Acuerdo entre dominios DiffServ para realizar la traducción de los valores DSCP de los paquetes que salen de un dominio e ingresan al otro.

TCM **Modulación Codificada de Trellis (*Trellis Coded Modulation*)**: Conjunto de algoritmos que permiten mejorar la eficiencia respecto del esquema de modulación convencional realizando corrección de errores en el receptor, sin efectuar retransmisión.

TCP/IP **Protocolo de Control de Transmisión / Protocolo Internet (*Transmission Control Protocol / Internet Protocol*)**: Conjunto de protocolos utilizados en Internet para soportar servicios tales como acceso remoto (telnet), transferencia de ficheros (FTP) y correo electrónico (SMTP), entre otros.

Telnet Protocolo estándar en Internet que permite mantener una sesión en un sistema remoto.

TSpec **Especificación de Tráfico (*Traffic Specification*)**: Parámetro introducido en la Arquitectura de Servicios Integrados para definir las características del tráfico que generará una fuente.

U

UART **Transmisor – Receptor Asincrónico Universal (*Universal Asynchronous Receiver - Transmitter*):** Chip que controla el envío y recepción de datos desde y hacia un puerto serial.

UBR **Velocidad no especificada de bit (*Unspecified Bit Rate*):** Categoría de Servicio de ATM, que intenta emular el tradicional servicio *best effort* de TCP/IP.

UDP **Protocolo de Datagramas de Usuario (*User Datagrama Protocol*):** Protocolo no confiable no orientado a conexión que permite el envío de datos por medio de Internet.

UNI **Interfaz Usuario Red (*User Network Interface*):** Punto de interfaz entre usuarios finales de ATM y un *switch* privado ATM, o entre un *switch* privado ATM y un proveedor ATM público.

V

VBR **Velocidad Variable de Bit (*Variable Bit Rate*):** Categoría de servicio de ATM, adecuada para aplicaciones que generan ráfagas de datos y que necesitan sincronización de tiempo entre la fuente y el destino.

VCI **Identificador de Circuito Virtual (*Virtual Circuit Identifier*):** Código utilizado para identificar un circuito virtual de ATM.

VLAN **Red Virtual de Área Local (*Virtual Local Area Network*):** Red de área local que agrupa las estaciones de trabajo basándose en algún parámetro que no sea la ubicación geográfica, como por ejemplo, usuarios y aplicaciones.

VoIP **Voz sobre IP (*Voice over IP*)**: Tecnología para transportar voz digitalizada sobre redes IP. La mayor ventaja de VoIP es el ahorro en costo con respecto al sistema de telefonía internacional tradicional.

VPI **Identificador de Camino Virtual (*Virtual Path Identifier*)**: Código para identificar un camino virtual en una red ATM.

VPN **Red Privada Virtual (*Virtual Private Network*)**: Es una red privada de datos que mantiene la privacidad de la información mediante protocolos de túneles y procedimientos de seguridad.

W

WAN **Red de Área Extendida (*Wide Area Network*)**: Red que abarca una gran cobertura geográfica, por ejemplo un país o un continente.

WDM **Multiplexación por Division de longitud de Onda (*Wavelength División Multiplexing*)**: Mecanismo que permite que múltiples señales sean codificadas dentro de múltiples longitudes de onda.

WFQ **Encolamiento Justo Ponderado (*Weighted Fair Queuing*)**: Algoritmo de organización de tráfico que automáticamente categoriza el tráfico en flujos de alta y baja prioridad, basado en el volumen de paquetes que observa el *router* o *switch*.

WRR ***Weighted Round Robin***: Técnica utilizada por los algoritmos de gestión de tráfico que consiste en asignar algún servicio a los paquetes en una manera secuencial y cíclica. Se diferencia de PRR debido a que permite servir a una conversación más de una vez en cada ciclo.

REFERENCIAS

Los siguientes textos, folletos, RFCs, artículos de Internet y otras referencias contienen disposiciones que mediante su referencia en este texto, constituyen material de apoyo del presente Proyecto de Titulación. Al efectuar este trabajo, estaban en vigor las ediciones indicadas de las recomendaciones respectivas.

[1] Gatens, D., and Witowsky, W., "The Evolution Access Server (RAS) to a Universal Port-Enabled Platform". Texas Instruments. May 2000.

[2] Carpenter, B. and Kandlur, D., "Diversifying Internet Delivery", IEEE Spectrum Magazine, November 1999.

[3] IEEE Communications Magazine, September 1999.

[4] Coffman, K., and Odlyzko, A., "The Size and Growth Rate of The Internet", http://www.firstmonday.dk/issues/issue3_10/coffman/index.html

[5] De Miguel, T., "Nuevos Servicios de Videoconferencia sobre Internet", SATELEC99, 1999. http://www.etsit.upm.es/satelec/conferencias/ponencias-Tmiguel-completa.html.

[6] Croll, A., Packman, E., "Managing Bandwidth: Deploying QoS in Enterprise Networks", Prentice Hall, 1999.

[7] Papir, Z., Simmonds, A., "Competing for Throughput in the Local Loop", IEEE Communications Magazine, May 1999.

[8] Bormann, C., "The Multi-Class Extension to Multi-Link PPP", RFC 2686, September 1999.

[9] "IP QoS FAQ", QoS Forum, September 1999. http://www.qosforum.com

[10] Tanenbaum, A., "Redes de Computadores", Prentice Hall, 1997.

[11] http://wwwhost.ots.utexas.edu/ethernet/10quickref/ch7qr_7.html.

[12] Georgia Tech — Graphics, Visualization & Usability Center, Eighth WWW User Survey, Oct. 1997. http://www.gvu.gatech.edu/user_surveys/survey-1997-10/graphs/technology/Connection_Speed.html

[13] Goodman, B., "Internet Telephony and Modem delay", IEEE Network Magazine, May/June 1999.

[14] Defense Advanced Research Project Agency, "Transmission Control Protocol", RFC 793, September 1981.

[15] Batthi, N., and Friedrich, R., "Web Server Support for tiered services", IEEE Network Magazine Sep/Oct 99.

[16] Claypool, M. and Tanner, J., "The effects of Jitter on the perceptual quality of video", Worcester Polytechnic Institute.

[17] http://www.whatis.com

[18] Stallings, W.,"Redes de Computadoras". Quinta Edición.

[19] http://www.comtro.com/fibra.htm

[20] Grupo Conuemex.
http://www.club.telepolis.com/avpavp/coaxial.htm

[21] Jordan, R. y Shawwa, L., "Introducción a las redes de Computadoras", 1998.
http://www.istec.org/ace/ati98/leccion9/sld031.htm

[22] IEEE Communications Magazine, May 2000.

[23] Cisco CCIE Fundamentals: Network Design, "Designing Internetworks for Multimedia".
http://www.cisco.com/univercd/cc/td/doc/cisintwk/idg4/nd2013.htm

[24] Parameswaran, J., "End to End Quality of Service", 1998.
http://www.cnds.jhu.edu/courses/cs667/qos98

[25] Keshav, S., "Congestion Control in Computer Networks" PhD Thesis, published as UC Berkeley TR-654 , September 1991. (Awarded the Sakrison Prize for the best PhD thesis in the EECS department, 1992.)

[26] http://www.cisco.com

[27] Varma, A., and Stiliadis, D., "Hardware Implementation of Fair Queuing Algorithms for Asynchronous Transfer Mode Networks", IEEE Communications Magazine, December 1997.

[28] Risso, F., and Gevros, P., "Operational and Performance Issues of a CBQ router", University College London.

[29] Floyd, S. And Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", University of California, August 1993.

[30] Floyd, S., and Ramakrishnan, K., "A Proposal to add Explicit Congestion Notification ECN to IP", RFC 2481, January 1999.

[31] Braden, B., Clark, D., "Recomendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

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

[33] Bernet, Y., Smith, A., "A Conceptual Model for DiffServ routers", Internet Draft draft-ietf-diffserv-model-03.txt, May 2000.

[34] Salsano, S., and Mameli, R., "Quality of Service (QoS) in the Internet", Coritel, 1998.

[35] Defense Advanced Research Project Agency, "Internet Protocol", RFC 791, Semptember 1981.

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

[37] Jacobson, V., Nichols, K. and Poduri, K., "An Expedited Forwarding PHB", RFC 2598, June 1999.

[38] Heinanen, J., Baker, F., Weiss, W. and Wroclawski, J., "Assured Forwarding PHB Group", RFC 2597, June 1999.

[39] Nichols, K. and Carpenter, B., "Definition of Differentiated Services Behavior Aggregates and Roles for their Specification", Internet Draft draft-ietf-diffserv-ba-def-01.txt, February 2000.

[40] Dutta-Roy,A., "CABLE it's not just for TV", IEEE Spectrum Magazine, May 1999.

[41] Tomasi, W., "Sistemas de Comunicaciones Electrónicas", Prentice Hall, 1996.

[42] Braden, B., Clark, D., y Shenker S., "Integrated Service in the Internet Architecture: An Overview", RFC 1633, June de 1994.

[43] Parekh, A., "A generalized Processor Sharing Aproach to Flow Control in Integrated Services Networks", 1992.

[44] Wroclawski, J., "Specification of Controlled-Load Network Element Service", RFC 2211, September 1997.

[45] Shenker, S., Partridge, C. and Guerin, R., "Specification of Guaranteed Quality of Service" RFC 2212, September 1997.

[46] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource Reservation Protocol (RSVP) – Version 1 Functional Specification", RFC 2205, September 1997.

[47] Garrett, M. and Borden, M., "Interoperation of Controlled-Load Service and Guaranteed Service with ATM", RFC 2381, August 1998.

[48] Bormann, C., "Providing Integrated Services over Low bitrate Links", RFC 2689, Semptember 1999.

[49] Bormann, C., "The Multi-Class Extension to Multi-Link PPP", RFC 2686, September 1999.

[50] Bormann, C., "PPP in a Real-Time Oriented HDLC-like Framing", RFC 2687, September 1999.

[51] Bernet, ed. et al., "A Framework For Integrated Services Operation over DiffServ Networks", draft-ietf-issll-diffserv-rsvp-05.txt, May, 2000.

[52] Radhakrishna. C., "Use of RSVP for Differentiated Services Signaling and Admission Control", Internet Draft, draft-rx-diffserv-rsvp-sig-00.txt, June 2000.

[53] Bernet, Y., "Format of the RSVP DCLASS Object", Internet Draft, draft-ietf-issll-dclass-01.txt, October 1999.

[54] Williams, B., "Quality of Service Differentiated Services and Multiprotocol Label Switching", Ericsson Inc. March 2000.
http://www.ericsson.com/iptelephony

[55] Semeria, C., "Extensión de la señalización RSVP para Ingeniería de Tráfico MPLS". Juniper Networks. Septiembre 2000.
http://www.juniper.net/techcenter/techpapers

[56] QoS Forum, "Introduction to QoS Policies", 1999.

[57] Frank, A., "Lighweight Directory Access Protocol", Network Magazine, 1998.
http://www.networkmagazine.com/article/NMG20000727S0003

[58] Markey, B., "A System Administrator's View of LDAP". 1998

[59] Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and Sastry, A., "The COPS (Common Open Police Service) Protocol", RFC 2748, January 2000.

[60] UUNET
http://www.uunet.net

[61] Postel, J., "Internet Control Messages Protocol", RFC 792, September 1981.

[62] Traceroute: Cool Network Tool
http://www.pcwebopedia.com/

[63] Stanford, H., "Telecomunications for Managers ", Prentice Hall, Quinta Edición. Enero 2000.