PROTOCOLO RSVP

FROILAN AUGUSTO CARRANZA ROSSI
JHONNY URBANO RINCÓN LÓPEZ

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
PROGRAMA DE INGENIERÍA ELECTRONICA
CARTAGENA DE INDIAS
2006

PROTOCOLO RSVP

FROILAN AUGUSTO CARRANZA ROSSI
JHONNY URBANO RINCÓN LÓPEZ

Monografía, presentada para optar
al titulo de Ingeniero Electrónico

Director
Msc EDUARDO GOMEZ VASQUEZ

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
PROGRAMA DE INGENIERÍA ELECTRONICA
CARTAGENA DE INDIAS
2006

Nota de aceptación

Firma de presidente del jurado

Firma del Jurado

Firma del jurado

Cartagena Junio 6 de 2006.

CONTENIDO

TABLA DE FIGURAS

Pagina

INTRODUCCIÓN

Originalmente Internet, tal y como fue concebida, ofrecía sólo una simple QoS, basada en la entrega best-effort de datos punto a punto. En la actualidad, aplicaciones en tiempo real, como vídeo remoto, conferencias multimedia o realidad virtual, no funcionan bien bajo esta definición de red, debido a los retardos variables en colas y las pérdidas por congestión.

Antes de que estas aplicaciones sean ampliamente utilizadas, la infraestructura de red debe ser modificada para soportar calidad de servicio en tiempo real, la cual permita algún control sobre los retardos de paquetes extremo-extremo.

Además los operadores de red, solicitan disponer de la capacidad para controlar la repartición del ancho de banda de un enlace entre diferentes clases de tráfico, lo cual conlleva la necesidad de dividir el tráfico total en varias clases y asignar a cada una de éstas un mínimo porcentaje del ancho de banda total bajo condiciones de sobrecarga (compartición del enlace). Estas distintas clases pueden representar distintos grupos de usuarios o distintos protocolos.

Para solucionar este problema fue diseñado RSVP, un protocolo de reserva de recursos para servicios integrados de Internet. El protocolo de RSVP es usado por un *host* para pedir calidades específicas de servicio de la red para flujos de datos en aplicaciones particulares. RSVP es también usado por enrutadores para brindar calidad de servicio (QoS). Las solicitudes RSVP generalmente se dan en cada nodo a través de la trayectoria de datos, es en este donde se dan las reservas de los recursos.

RSVP pide recursos para flujos simples, lo hace en sólo una dirección. Por lo tanto, RSVP trata un remitente distinto a un receptor, aunque el mismo proceso de una aplicación puede actuar como ambos, un remitente y un receptor al mismo tiempo.

RSVP no es un protocolo de asignación de ruta; RSVP es diseñado para operar con los actuales y futuros protocolos de *unicast* y *multicast*. Un proceso de RSVP consulta la base de datos de asignación de ruta local para obtener rutas.

Dependiendo de selección del remitente RSVP, maneja diferentes estilos de reserva, que se encontraran en la red remitentes fijos, o muchos remitentes arbirtrarios.

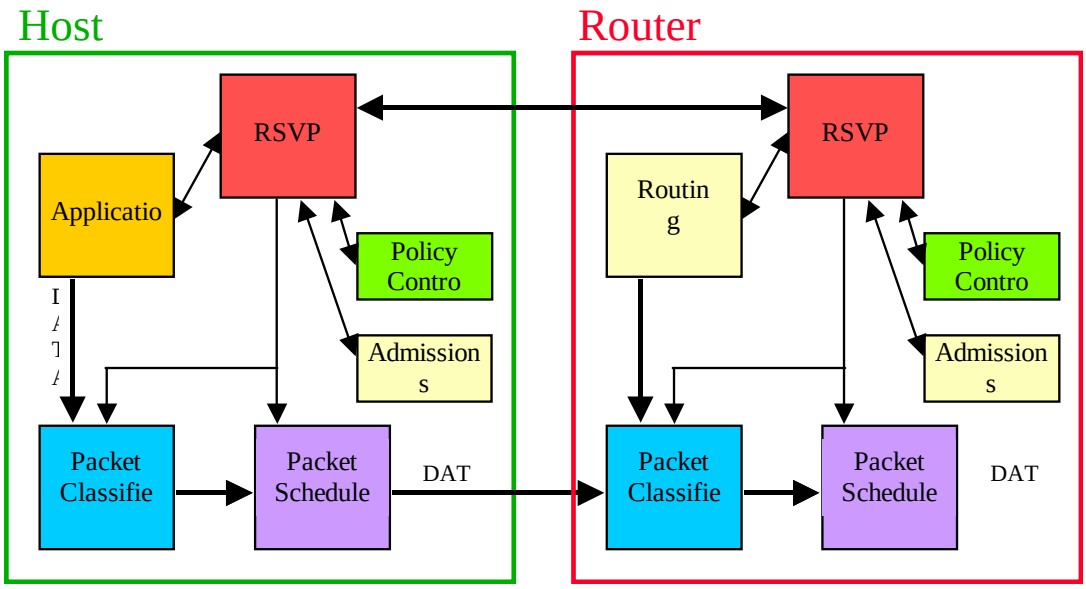
RSVP adopta el principio de reserva iniciada por el receptor, indicando que es el receptor en lugar del emisor quien inicia la reserva de recursos. Este principio es similar en esencia a muchos algoritmos de encaminamiento multidifusión donde cada receptor se una al grupo de multicast, o lo abandona, independientemente, sin afectar a otros receptores en el grupo. La principal motivación de adoptar este principio es que RSVP está diseñado principalmente para soportar conferencias con multiples localizaciones con receptores diferentes. En este entorno el receptor sabe realmente cuanto ancho de banda necesita.

1. DESCRIPCIÓN DEL PROTOCOLO RSVP

Como las ejecuciones de enrutamiento y protocolos de administración, una ejecución de RSVP típicamente se ejecutará en la base, no en el camino de despacho de datos.

En el caso de *multicast*, por ejemplo, un *host* envía mensajes IGMP[1] para unir un grupo de *multicast* y entonces este envía mensajes RSVP para reservar recursos a lo largo del camino de entrega de ese grupo. Los protocolos de enrutamiento determinan donde los paquetes se remitieron; RSVP es interesado sólo en brindar QoS a esos paquetes que se remitieron en conformidad con la ruta.

Figura 1: RSVP en *host* y enrutadores



Fuente RFC 2205 “Resource Reservation Protocol Version 1”

[1] Internet Group Management Protocol

Como se muestra en la figura 1 la calidad del servicio es implementada para un particular flujo de datos por mecanismos colectivamente llamado "control de tráfico[2]" que pueden encontrase en host y en enrutadores. Estos mecanismos incluyen:

(1) Un clasificador de paquetes
(2) Un control de admisión
(3) Un programador de paquetes

El clasificador de paquetes determina la clase de QoS (ancho de banda) para cada uno de los paquetes. Para cada interfaz de salida, El programador de paquetes logra el QoS prometido.

Durante la reserva, una solicitud de QoS de RSVP es llevada a dos módulos de decisión: "control de admisión" y "control de política".
El control de admisión determina si el nodo tiene suficiente recursos disponibles para suministrar el QoS pedido. El control de política determina si el usuario tiene permiso administrativo para hacer la reserva. Si ambos verificaciones tienen éxito, los parámetros son puestos en el clasificador de paquete el y en la interfaz de nivel de enlace, para obtener la QoS deseada. Si cualquier verificación fracasa, el programa de RSVP retorna una notificación de error a la aplicación que originó la solicitud.

[2] RFC 2205 “Resource Reservation Protocol Version 1” Available from Internet: http://www.faqs.org/rfcs/rfc2205.html

1.1 FLUJOS DE DATOS

RSVP define una "sesión" para establecer un flujo de datos con un destino particular y un protocolo de capa de transporte. RSVP trata cada sesión independientemente.

Una sesión de RSVP es definida por el triple: (*DestAddress, ProtocolID, Dstport*[3]*). DestAddress*, identifica la dirección IP de destino del paquete datos, ya que puede ser una dirección de *unicast* o *multicast*. *ProtocolID* reconoce el ID del protocolo IP. El parámetro de *Dstport* identifica, el puerto de destino, *Dstport* puede ser definido por un puerto de destino de UDP/TCP, o por un campo equivalente en otro protocolo de transporte, o por alguna información de una aplicación especifica.

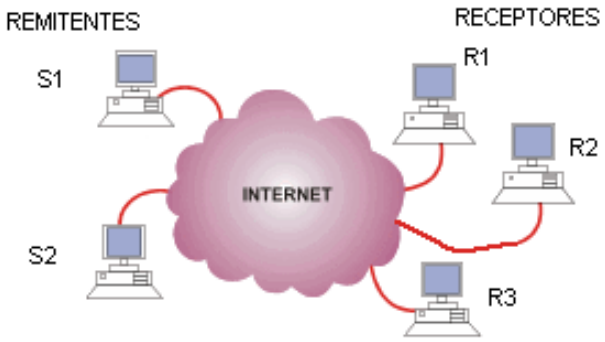
Aunque el protocolo RSVP es diseñado para ser fácilmente extensible (cualquier tecnología de red) para mayor generalidad, el protocolo básico solo soporta como puertos a UDP y TCP. Note que no es estrictamente necesario incluir *Dstport* en la definición de sesión, cuando *DestAddress* es un *multicast* desde diferentes sesiones las cuales tienen direcciones de *multicast* diferentes. Sin embargo, *Dstport* es necesario para permitir más de una sesión de *unicast* dirigida al mismo *host* receptor.

La figura 2 ilustra el flujo de paquetes de datos en una sesión sencilla de RSVP, asumiendo una distribución de datos por *multicast*. Las lineas indican el flujo de datos desde los emisores S1 y S2 hasta los receptores R1, R2, y R3 y la nube representa la malla de distribución creada por el enrutador de *multicast*. La distribución de *multicast* remite una copia de cada uno de los paquetes de datos

[3] STALLINGS, William. Comunicaciones y redes de computadores. 6ª edición 2000. p 551

de un remitente a cada receptor; una sesión de distribución de *unicast* tiene un receptor . Cada remitente puede estar en un único *host* de Internet.

Figura 2: Sesión de Distribucion de *Multicast*



Fuente: El Autor de la Monografia

Para transmisión *unicast*, habrá un *host* de destino sencillo pero pueden existir remitentes múltiples; RSVP puede preparar reserva para transmisión multipunto o punto-punto.

1.2 MODELOS DE RESERVA

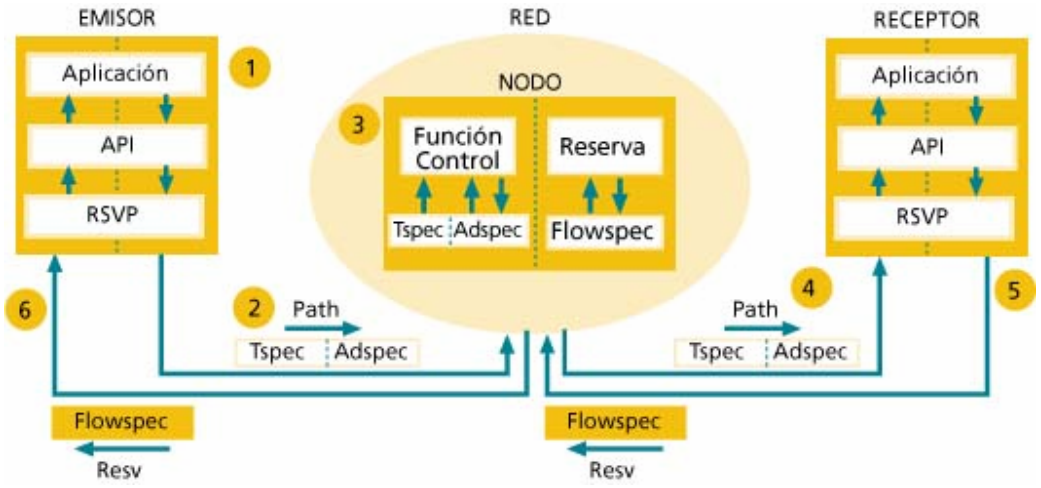
Una solicitud de reserva RSVP elemental consiste de un *flowspec* junto con un *filter spec*; este par es llamado "descriptor de flujo[4]". El *flowspec* especifica un QoS deseado. El *filter spec*, define el conjunto de paquetes de datos que van a recibir el QoS definido por el *flowspec* (el descriptor de flujo es enviado como se muestra en la figura 3 en un mensaje de Resv). El *flowspec* es usado para poner parámetros en el programador de paquetes de un nodo de otro mecanismo de

[4] RFC 2205, Op Cit.

nivel de enlace mientras que el *filter spec* es usado para poner parámetros en el clasificador de paquetes. Los paquetes de datos que son dirigidos a una sesión particular que no reconoce cualquier *filter spec*, son manejados por el tráfico de best-effort.

El *flowspec* en una reserva generalmente incluirá una clase de servicio y dos conjuntos de parámetros numéricos: (1) un "Rspec"(el R para reserva) este define el QoS deseado, y (2) un " Tspec" (el T para tráfico) este describe el flujo de datos. Los formatos y los contenidos de Tspecs y Rspecs son determinados por el modelo de servicios integrados.

Figura 3: Transmisión y Recepción del descriptor de flujo



Fuente: http://www.rediris.es/rediris/boletin/46-47/ponencia11.html

El formato exacto de un *filter spec* depende de que versión de IP se este usando IPv4 o IPv6. En la mayoría de casos el *filter spec* puede escoger subconjuntos arbitrarios de los paquetes en una sesión dada. Tales subconjuntos pueden ser definidos en términos de los remitentes (la dirección IP de remitente, y el puerto de la fuente de transmisión), en términos de un protocolo a nivel más alto, o

generalmente desde el punto de vista de cualquier campo en cualquier cabecera de un protocolo en el paquete. Por ejemplo, el *filter spec* puede ser usado para escoger subflujos diferentes de un *video stream* codificado jerárquicamente seleccionando campos en una cabecera de nivel de aplicación. Para simplificar el proceso (y para minimizar la violación de niveles), el *filter spec* básico tiene una forma muy limitada: la dirección IP del remitente y opcionalmente el numero de puertos SrcPort de UDP/TCP.
Debido a que el numero de puertos UDP/TCP son usados para la clasificación de paquetes, cada enrutador debe ser capaz de examinar estos campos.

En cada nodo intermedio, una reserva solicita dos acciones generales, como sigue:

1. Hacer una reserva en un enlace

El proceso de RSVP pasa el recurso al control de admisión y al control de política. Si cada test fracasa, la reserva es rechazada y el proceso de RSVP retorna un mensaje de error para el receptor(es). Si ambos tienen éxito, el nodo configura el paquete clasificador para escoger los paquetes de datos definidos por el *filter spec*, y este interactúa con el nivel de enlace apropiado para obtener el QoS deseado definido por el *flowspec*.

2. Remitiendo la solicitud *upstream*

Una solicitud de reserva es propagada *upstream* hacia el remitente. El conjunto de *host* remitentes que van a hacer la solicitud de reserva dada es propagado con el llamado "scope" de esa solicitud.

La solicitud de reserva que un nodo remite *upstream* puede diferir de la solicitud que recibió de *downstream*, por dos razones. El mecanismo de control de tráfico puede modificar el *flowspec* salto por salto. Hay que tener en cuenta que, las

reservas *downstream* de diferentes ramas de un árbol de *multicast* de el mismo remitente (o el conjunto de remitentes) debe ser "combinadas" como reservas de viaje *upstream*.

Cuando un receptor origina una solicitud de reserva, puede también pedir un mensaje de confirmación para indicar que su solicitud estaba (probablemente) instalada en la red. Una reserva exitosa se propaga *upstream* a lo largo del árbol de *multicast* hasta que este alcance un punto donde una existente reserva sea igual o mayor que la que se este pidiendo. En ese momento, la solicitud que esta llegando es fusionada con la reserva en el lugar apropiado y no necesita ser enviada; el nodo puede enviar entonces un mensaje de confirmación de reserva de vuelta al receptor.

El modelo básico de reserva RSVP es "one pass": un receptor envía una solicitud de reserva *upstream*, y cada nodo en el camino acepta o rechaza la solicitud. Este plan no proporciona ninguna vía fácil para que el receptor encuentre el servicio end–to-end resultante. Por lo tanto, RSVP soporta un servicio de un paso conocido como "One Pass With Advertising[5]" (OPWA). Con OPWA, RSVP controla los paquetes que son enviados *downstream*, siguiendo los caminos de datos, para recoger la información que pueda ser usada para predecir el QoS end-to-end.

RSVP le muestra los resultados al *host* receptor y tal vez a las aplicaciones del receptor. Los resultados se pueden usar entonces por el receptor para construir, o para ajustar con dinamismo, una solicitud de reserva apropiada.

[5] Ibid.

1.3 ESTILOS DE RESERVA

Una solicitud de reserva incluye un conjunto de opciones que son colectivamente llamados estilos de reserva.
Una opción de reserva concierne el tratamiento de estas para remitentes diferentes dentro de la misma sesión: estableciendo una distinta reserva para cada *upstream* remitente, o de otro modo, hacer una sencilla reserva que es "dividida" entre todos los paquetes de remitentes seleccionados.

Otra opción de reserva controla la selección de remitentes; puede ser una lista explícita de todos los remitentes escogidos, o un *"wildcard"* que escoge implícitamente todos los remitentes de la sesión. En una reserva de selección de remitente explícita, cada *filter spec* debe hacer elegir exactamente un remitente, mientras que en una selección de remitente de *wildcard* ningún *filter spec* es necesitado. Como se muestra en la figura 4 dependiendo del tipo de remitente es requerido un estilo de reserva.

Figura 4: Atributos y estilos de reserva

Selección Remitente	Reservas:	
	Distinto	Dividido
Explícito	Filtro Fijo Estilo (FF)	Explícito Dividido Estilo (SE)
Wilcard	(Ninguno Definido)	Filtro de Wildcard Estilo (WF)

Fuente: http://www.faqs.org/rfcs/rfc2205.html

- El filtro de *Wildcard* o el estilo (WF)

El estilo de WF implica las opciones: reserva "dividida" y selección de remitente *wildcard*. Así, una reserva de estilo de WF crea una reserva sencilla partida por flujos de todos los remitentes *upstream*. Esta reserva puede ser imaginada como un "tubo" dividido, cuyo "tamaño" es del largo del recurso requerido por todos los receptores, independiente del número de remitentes que estén usándolo. Una reserva de estilo de WF es propagada *upstream* hacia todos los *host* remitentes y automáticamente se extienda a nuevos remitentes que van apareciendo.
Simbólicamente, podemos representar una solicitud de reserva de estilo de WF por:

WF(* {Q})

donde el asterisco representa selección de remitente de wildcard y la Q representa el *flowspec*.

- El filtro fijo o estilo (FF)

El estilo de FF implica las opciones: reservas "distintas" y selección de remitente " explícitas". Así, un estilo de FF elemental crea una reserva distinta para paquetes de datos de un remitente particular, no dividiéndolos con otros paquetes de remitentes para la misma sesión.
Simbólicamente, podemos representar una reserva de FF elemental así:

FF (S{Q})

donde la S es el remitente escogido y Q es el correspondiente *flowspec*; el par forma un descriptor de flujo. RSVP permite múltiples reservas de estilo de FF elementales que pueden ser pedidas al mismo tiempo, usando una lista de descriptores de flujo:

FF (S1{Q1}, S2{Q2} ,...)

La reserva total en un enlace para una sesión dada es la suma de Q1, Q2 ,... para todos los remitentes pedidos.

- *Shared explicit* o estilo (SE).

El estilo SE implica las opciones: reserva "dividida" y selección de remitentes "explícita". Así, una reserva de estilo de SE crea una reserva sencilla dividida por remitentes *upstream* escogidos. A diferencia del estilo de WF, el estilo de SE permite a un receptor especificar explícitamente el conjunto de remitentes para incluirse.

Podemos representar una solicitud de reserva de SE que contiene un *flowspec* Q y una lista de remitentes S1, S2,... por:

SE (S1, S2,...) {Q})

Las reservas divididas, creadas por WF y SE, son apropiadas para esas aplicaciones de *multicast* en donde es improbable en que múltiples fuentes de datos puedan transmitir simultáneamente. El *audio Packetized* es un el ejemplo de una aplicación adecuada para reservas divididas; desde entonces un número limitado de las personas habla al instante, cada receptor podría emitir una solicitud de reserva de WF o SE para dos veces el ancho de banda requerido para un remitente. Por otra parte, el estilo de FF, que crea reservas distintas para los flujos de remitentes diferentes, está apropiado para señales de video.

Los reglamentos de RSVP desaprueban la fusión de reservas divididas con las reservas distintas, estos modos son fundamentalmente incompatibles. Desaprueban también la fusión entre la selección del remitente explícito con la

selección de remitente de wildcard, ya que esto podría producir un servicio inesperado para un receptor que especificó selección explícita.
A causa de estas prohibiciones los estilos WF, SE, y FF son mutuamente incompatibles.
Cuando una aplicación pregunta por WF, el proceso RSVP en el *host* receptor puede usar estado local para crear una reserva de SE equivalente que listó explícitamente en todos los remitentes. Sin embargo, una reserva de SE obliga a el clasificador de paquete en cada uno de los nodos a escoger explícitamente cada remitente en la lista, mientras que un WF permite que el clasificador de paquete haga un simplemente un "*wild card"* entre la dirección del remitente y el puerto. Cuando existe una lista grande de remitentes, una reserva de estilo WF puede dar por lo tanto un resultado considerablemente menor que una reserva de estilo de SE equivalente. Para esto ambos SE y WF están incluidos en el protocolo.

1.4 EJEMPLOS DE ESTILOS

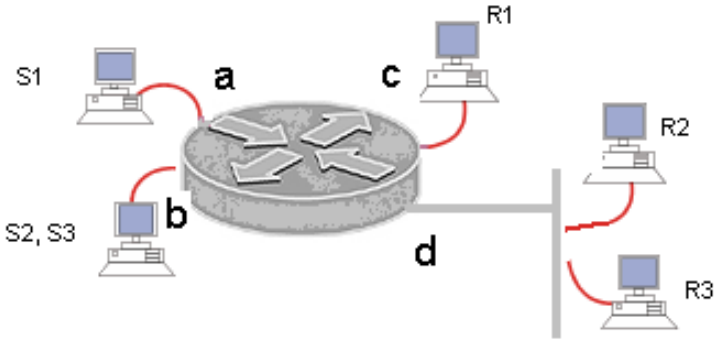
Esta sección presenta ejemplos de cada uno de los estilos de reserva y muestra los efectos de la fusión entre ellos.

La figura 5 ilustra un enrutador con dos interfaces de entrada,(a) y (b), por donde los flujos llegarán, y dos interfaces de salida, (c) y (d) por las cuales cada dato será enviado. Esta topología será supuesta en los ejemplos que siguen. Hay tres remitentes *upstream*; paquetes desde los remitentes S1, S2 y S3 llegue por salto previo hasta (a) de ((b), respectivamente). Hay también tres *downstream*

receptores; paquetes con destino a R1 (R2 y R3) que son enrutados hasta la interfaz (c) ((d), respectivamente). Se asume además que la interfaz d esta conectada a una broadcast LAN.

Se debe especificar también las rutas de *multicast* dentro del nodo de la Figura 5. Asuma primero que los paquetes de datos de cada S_i mostrado en la Figura 5 son encaminados a ambas interfaces de salida. Bajo esta suposición, las figuras 6, 7, y 8 ilustran un filtro de Wildcard, un Filtro fijo, y las reservas explícitas divididas, respectivamente.

Figura 5: Configuración del enrutador

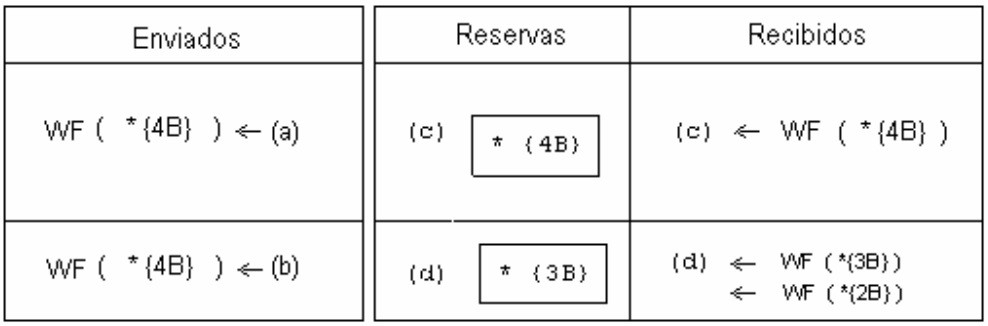


Fuente: El Autor de la Monografia

Para simplicidad, estos ejemplos muestran los *flowspec*s como múltiplos unidimensionales de cierta cantidad de recurso de base B. La columna de receptores exhibe la reserva de RSVP recibida en las interfaces (c) y (d), y la columna de reservas muestran la reserva resultante que se manifiesta para cada interfaz. La columna de remitente muestra las solicitudes de reserva que son enviadas *upstream* para (a) y (b). En la columna de reserva, cada caja representa un "tubo" reservado en el enlace de salida, con el descriptor de flujo correspondiente.

La figura 6, muestra el estilo de WF, ilustra dos distintas situaciones en que la fusión es exigida. (1) Cada uno de los dos saltos siguientes en la interfaz (d) da por resultado una reserva de RSVP separada, como se muestra; estas dos solicitudes deben ser fusionadas en un efectivo *flowspec*, 3B, esto es usado para hacer la reserva en la interfaz (d). (2) Las reservas en las interfaces (c) y la (d) deben ser fusionadas en orden para enviar las solicitudes de reserva *upstream*; como consecuencia, el *flowspec* más grande 4B es remitido *upstream* a cada salto previo.

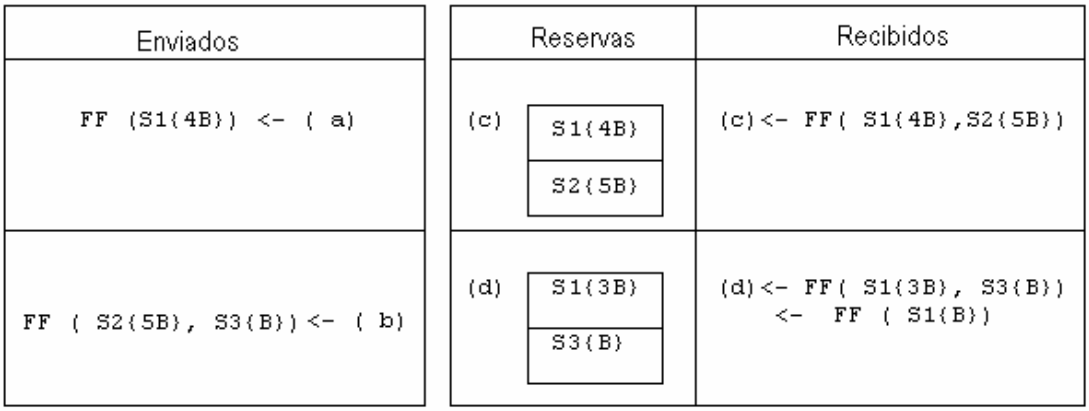
Figura 6: El filtro de Wildcard (WF) en el ejemplo de reserva



Fuente: http://www.faqs.org/rfcs/rfc2205.html

La figura 7 muestra el filtro fijo o las reservas de estilo (FF). Para cada interfaz de salida, existe una reserva separada para cada fuente que la ha estado requiriendo, pero esta reserva será dividido entre todos los receptores que han hecho la solicitud. El flujo descriptor para los remitentes S2 y S3, recibe las interfaces (c) y (d), está lleno (no hubo fusión) en la solicitud remitida para el salto previo desde (b). Por otra parte, los tres diferentes descriptores de flujo especificando el remitente S1 que es fusionado paulatinamente en la solicitud sencilla FF (S1{4B}) que se envía para salto previo desde(a).

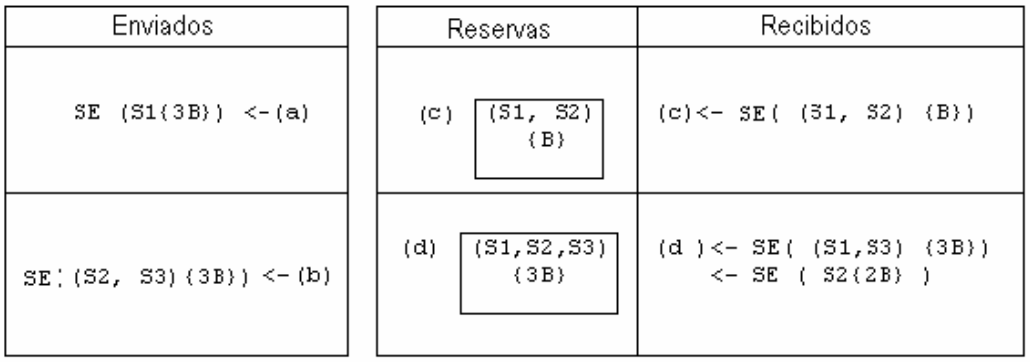
Figura 7: El filtro fijo (FF) en el ejemplo de reserva



Fuente: http://www.faqs.org/rfcs/rfc2205.html

La figura 8 muestra un ejemplo de el estilo SE. Cuando las reservas de estilo SE son combinados, el *filter spec* resultante es la unión de las especificaciones de filtro originales, y el *flowspec* resultante es el *flowspec* más grande.

Figura 8: Estilo de SE en el ejemplo de reserva

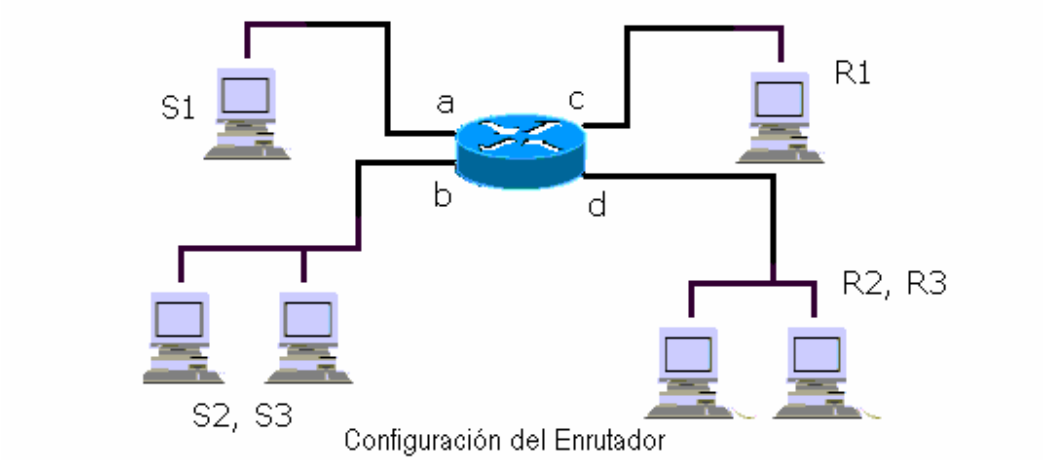


Fuente: http://www.faqs.org/rfcs/rfc2205.html

Los tres ejemplos mostrados sólo asumen que los paquetes de datos de S1, S2, y S3 son encaminados a ambas interfaces de salida. La parte superior de la figura 9 muestra otro enrutamiento asumido: los paquetes de datos de S2 y S3 no son remitidos para la interfaz (c), porque la topología de red proporciona un camino

más corto para estos remitentes hacia R1, no cruzando este nodo. La parte inferior de la figura 9 muestra reservas de estilo de WF bajo esta suposición. No existe ninguna ruta de (b) a (c), solo se considera la reserva en la interfaz (d).

Figura 9: Asignación de ruta parcial (Ejemplo de reserva de WF)



Enviados	Reservas	Recibidos
WF(*{4B})<-(a)	(c) * {4B}	(c) <- WF(*{4B})
WF(*{3B})<- (b)	(d) * {3B}	(d) <- WF(* {3B}) <- WF(* {2B})

Fuente: http://www.faqs.org/rfcs/rfc2205.html

2. MECANISMOS DEL PROTOCOLO RSVP

Los mecanismos del protocolo RSVP proporcionan una modo sencillo para crear y mantener reserva distribuida a través de una malla de *multicast*, o por la trayectoria de *unicast*. RSVP transfiere y manipula QoS y los parámetros de control de política como datos opacos, ofreciéndoles un control de tráfico apropiado para su interpretación.

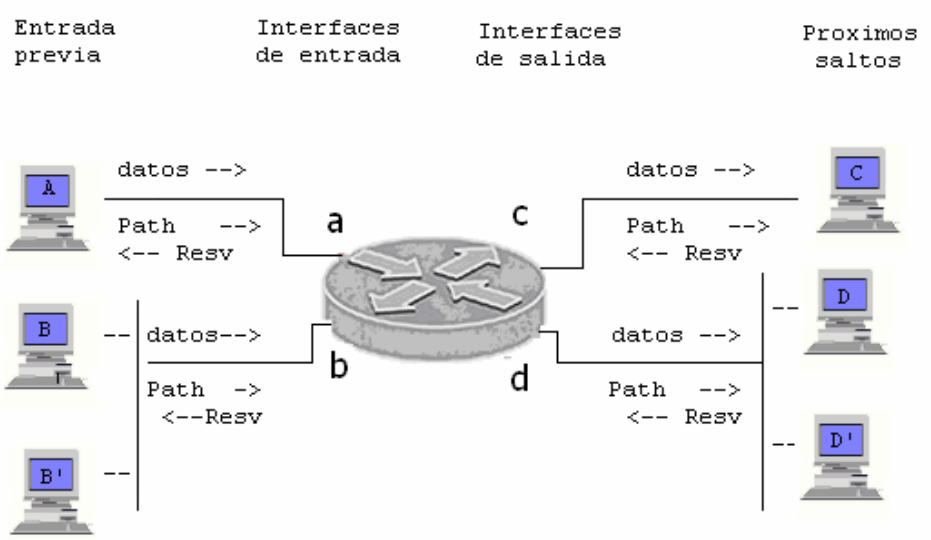
Tanto el miembro de un gran grupo de *multicast* como la topología de árbol resultante de *multicast*, pueden probablemente cambiar con el tiempo, el diseño de RSVP asume que los estados actuales de RSVP y el control de tráfico van a ser construidos y destruidos periódicamente en enrutadores y *host*s. Para esto, RSVP establece un "*soft state*[6]"; es decir, RSVP actualiza periódicamente los mensajes para mantener el estado a lo largo del camino reservado.

2.1 MENSAJES RSVP

La Figura 10 ilustra el modelo de RSVP en un enrutador nodo. Cada flujo de datos llega de un "salto previo" por su correspondiente interfaz de entrada " y sale por una o más "interfaces de salida". La misma interfaz puede actuar en ambos papeles entrada y salida para flujos de datos diferentes en la misma sesión.
Múltiples saltos previos y/o o saltos siguientes pueden alcanzarse por una interfaz física dada; por ejemplo, la figura implica que D y D' está unido a (d) con una LAN difundida (broadcast).

[6] RSVP, Resource ReSerVation Protocol. Available from Internet http://www.networksorcery.com/enp/protocol/rsvp.html

Figura 10: Enrutador usando RSVP



Fuente: El Autor de la Monografia

Existen dos tipos de mensaje de RSVP fundamentales: *Resv* y *Path*.

Cada *host* de receptor envía la reserva de RSVP (*Resv*) a través de mensajes *upstream* hacia los remitentes. Estos mensajes deben seguir exactamente el camino contrario al que los paquetes de datos usaran, *upstream* para todos los *host* remitentes, incluidos en la selección de remitentes. Ellos crean y mantienen el "estado de reserva" en cada nodo a lo largo de la trayectoria. Los mensajes de *Resv* finalmente deben ser dado al remitente, de modo que los *host* puedan preparar tráfico apropiado y un control de parámetros para el primer salto.
Cada *host* remitente de, RSVP transmite *downstream* el mensaje de *Path* de RSVP a lo largo de la ruta *uni-/multicast* proveída por los protocolos de asignación

de ruta, siguiendo la trayectoria de datos. Estos mensajes de *Path* almacenan el "estado de camino" en cada nodo a lo largo de la vía. Este "estado de camino" incluye al menos la dirección IP de *unicast* del nodo de salto previo, el cual es usado para encaminar los mensajes de *Resv* salto-por salto en la contraria dirección. (En lo sucesivo, ciertos protocolos de asignación de ruta pueden suministrar la información de despacho de camino directamente inversa, reemplazando la función de asignación de ruta inversa del *Path* state).

Un mensaje de *Path* además de la dirección de salto previa, contiene la información siguiente

- Plantilla de remitente (Sender_Template)

Un mensaje de *Path* es requerido para llevar una plantilla de remitente, que describa el formato de paquetes de datos que el remitente puede originar. Esta plantilla es en la forma de un *filter spec* que puede ser usado para seleccionar los paquetes de los remitentes de otros en la misma sesión en el mismo enlace.

Las plantillas de remitente tienen exactamente el mismo poder expresivo y el formato como el *filter spec* que aparece en los mensajes *Resv*.
Por lo tanto una plantilla de remitente puede especificar sólo la dirección IP del remitente y opcionalmente el puerto de remitente de UDP/TCP, y este asume el protocolo ID especificado para la sesión.

- Remitente Tspec

Un mensaje de *Path* es requerido para llevar un remitente Tspec, que defina las características de tráfico del flujo de datos que el remitente generará. Este Tspec

es usado por control de tráfico para impedir demasiada reserva, y tal vez innecesarias faltas de control de admisión.

- Adspec

Un mensaje de *Path* puede llevar un paquete de OPWA poniendo un aviso de información, conocido como un " Adspec". Un Adspec recibido en un mensaje de *Path* es pasado al control de tráfico local, que retorna un actualizado Adspec; la versión actualizada es entonces remitida en mensajes de *Path* enviados *downstream*.

Los mensajes de *Path* son enviados con la misma fuente y direcciones de destino como los datos, de modo que serán encaminados correctamente por los nubes de no-RSVP (ver la sección 2.9). Por otra parte, Los mensajes de *Resv* son enviados salto por salto; cada nodo parlante del RSVP remite un mensaje de *Resv* a la dirección de *unicast* de un salto previo.

2.2 COMBINAR LOS *FLOWSPEC*S

Un mensaje de *Resv* remitido a un salto previo lleva un *flowspec* este es el "más grande" de los *flowspec*s pedidos por los saltos próximos para cada flujo de datos que será enviado, se dice que los *flowspec*s han sido " combinados". Los ejemplos mostrados en la sección 1.4 ilustran otro caso de combinación, cuando la reserva es múltiple ya que se requiere de diferentes saltos próximos de la misma sesión y con el mismo *filter spec*, pero RSVP debe instalar solo una reserva en esa interfaz. Aquí de nuevo, la instalada reserva debe tener un *flowspec* efectivo

que es el "más grande " de los *flowspec*s pedidos por los próximos saltos diferentes.

Note que los *flowspec*s son generalmente vectores multi-dimensionales; ellos pueden contener ambos componentes de Tspec y Rspec, cada uno puede ser multi-dimensional. Por lo tanto, no puede ser estrictamente posible ordenar dos *flowspec*s. Por ejemplo, si una llamada de solicitud para un ancho de banda más alto y otras llamadas requieren un retraso en cada salto, una no es "más grande" que la otra. En tal caso, en lugar de tomar la más grande, el servicio específicos de rutinas de fusión debe ser capaz de retornar un tercer *flowspec* que es al menos tan grande como cada una; matemáticamente, este es el "límite superior menor[7]" (LUB Least Upper Bound). En algunos casos, un *flowspec* mas pequeño es necesitado; este es el "Limite inferior máximo[8]" (GLB Gratest Lower Bound).

Los pasos siguientes son usados para calcular el *flowspec* efectivo (Re, Te) para ser instalado en una interfaz. El Te es el Tspec efectivo y Re es el Rspec efectivo.

1. Un *flowspec* efectivo es determinado por la interfaz de salida. Depende de la tecnología de capa de enlace, esto puede requerir combinar *flowspec*s de saltos próximos diferentes; lo que significa que se computan los efectivos *flowspec*s como el LUB de los mismos. Note que lo que los *flowspec*s para combinar son determinados por los medios de nivel de enlace (vea la sección 3.11.2), mientras que la forma en que se combinan la determina modelo de servicio en uso.

2. Un cálculo específico del servicio del *Path*_Te, es la suma de todo los Tspecs que sirven como substituto en los mensajes de *Path* de diferentes saltos previos.

[7] **GARCIA, Tomas Jesús.** Alta velocidad y Calidad de servicio en redes IP. 1ª edición 2002. p 130

[8] Ibit. p.130.

3. *Resv*_Te y *Path*_Te son pasadas al control de tráfico.
El control de tráfico computará el *flowspec* efectivo como el "mínimo" *Path*_Te y *Resv*_Te, en un servicio de manera dependiente.

2.3 *SOFT STATE*

RSVP toma un "*Soft state*" para manejar el estado de reserva en enrutadores y *host*s. RSVP *soft state* es creado y periódicamente actualizado por los mensajes *Path* y *Resv*. El estado es borrado si no llega ningún mensaje de actualización antes que expire el intervalo de *"cleanup time out*[9]*"*. El estado también puede ser borrado por un mensaje de "demolición". A la expiración de cada período de "*refresh time out"* y después de un cambio de estado, RSVP examina su estado para construir y enviar mensajes *Path* y *Resv* actualizados a los siguientes saltos.

Los mensajes *Path* y *Resv* son idempotentes. Cuando una ruta cambia, el próximo mensaje de *Path* inicializará el estado de camino en la nueva ruta, y los mensajes de *Resv* futuros establecerán la reserva; el estado en el segmento de "libre ahora" de la ruta la cronometrará.

Así, un mensaje "nuevo" o "actualizado" es determinado separadamente en cada nodo, dependiendo de la existencia de estado en ese nodo.

RSVP envía sus mensajes como datagramas IP sin fiabilidad. La transmisión periódica de mensajes actualizados por *host*s y enrutadores son estimadas para manejar la pérdida ocasional de un mensaje RSVP. Si el efectivo Time out cleanup está seteado en K veces el periodo de tiempo de actualización, entonces

[9] RFC 2205. Op Cit

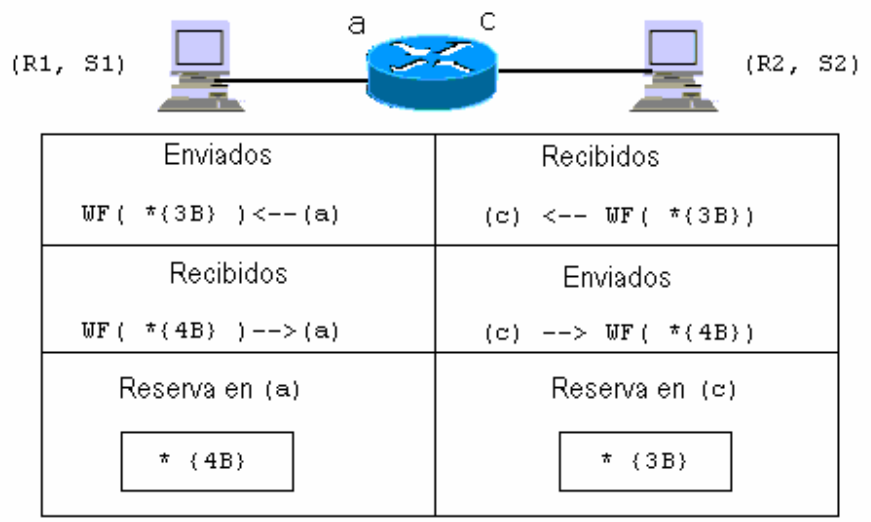
RSVP puede tolerar K-1 perdidas de paquetes sucesivos sin pasar al estado de borrarlos. El mecanismo de control de trafico de la red puede estar estáticamente configurado para otorgar cierto ancho de banda mínimo a los mensajes de RSVP para protegerlos de pérdidas por congestión.

El estado mantenido por RSVP es dinámico; para cambiar el conjunto S_i de remitentes o para cambiar cualquier requerimiento de QoS, un *host* empieza simplemente a enviar mensajes de *Path* y/o *Resv*. El resultado será un ajuste apropiado en el estado de RSVP en todos los nodos a lo largo del camino.

El estado continuo, se refresca salto por salto para permitir cualquier fusión.
Cuando el estado recibido difiere del estado guardado, este se actualiza. Si esta actualización da por resultado la transformación del estado, que luego actualiza todos los mensajes, éstos deben generar y remitir inmediatamente, de modo que los cambios puedan ser propagados end-to-end sin retardo. Sin embargo, la propagación de un cambio se detiene cuando se alcanza un punto donde de la fusión no resulta ningún cambio de estado. Esto minimiza el control de tráfico de RSVP y es esencial para atender a grandes grupos de *multicast*.
El estado que es recibido por una interfaz particular I no debe ser remitido por la misma interfaz. Al contrario, los estados que van a ser remitidos por la interfaz I, deben ser usados solamente para estados que llegan de interfaces diferentes. Un ejemplo trivial de esta regla se ilustra en la figura 11, que muestra un enrutador con un remitente y un receptor en cada interfaz (y asume un salto previo por interfaz). Interfaces (a) y (c) sirven ambas de salida y de entradas para esta sesión. Ambos receptores están haciendo las reservas de estilo wildcard, en que los mensajes de *Resv* son remitidos a todos los saltos previos para los remitentes en el grupo, a excepción del salto próximo de donde ellos vinieron. El resultado es reservas independientes en las dos direcciones.

Existe una regla adicional que rige el despacho de mensajes *Resv*: El estado que recibe los mensajes de *Resv* de una interfaz de salida Io puede ser enviada a una interfaz de entrada Ii solo si el mensaje de *Path* de Ii son remitidos a Io

Figura 11: Reservas independientes



Fuente: http://www.faqs.org/rfcs/rfc2205.html

2.4 DEMOLICIÓN (*TEARDOWN*)

Los mensajes de "demolición" de RSVP quitan el estado de *Path* o *Resv* inmediatamente. Aunque no es explícitamente necesario para arrancar una reserva vieja, es recomendable que todos los *host* remitentes envíen un requerimiento de demolición tan pronto una aplicación termine.

Existen dos tipos de mensaje de demolición de RSVP, *PathTear* y *ResvTear*. Un mensaje de *PathTear* viaja hacia todos los receptores *downstream* desde su punto de inicio y borra el estado de *Path*, como todos los estados de reserva dependiente, a lo largo de la vía. Un mensaje *ResvTear* borra el estado de reserva y viaja hacia todos los remitentes *upstream* desde su punto de la inicio. Un mensaje *PathTear* (*ResvTear*) puede conceptualizarse como un antónimo de el mensaje de *Path* (mensaje *Resv*, respectivamente).

Una solicitud de demolición puede iniciarse, por una aplicación en un fin de sistema (remitente o receptor), o por un enrutador, al cual se le venció un tiempo de espera o tiene una prioridad en otro servicio. Una vez iniciada puede ser enviada, salto por salto sin retardo.

Una demolición borra el estado específico en el nodo donde es recibida. Como siempre, este cambio de estado será propagado inmediatamente al nodo próximo, pero sólo si hay una red cambiada después de combinarse. Como consecuencia, un mensaje de *ResvTear* reduce el estado de reserva (solamente) como sea posible.

Como todos otros mensajes de RSVP, las solicitudes de demolición no son dadas con seguridad. La pérdida de un mensaje de solicitud de demolición no causará una falla en el protocolo, porque el estado no usado eventualmente cronometrará aunque no está explícitamente borrado. Si un mensaje de demolición se pierde, el enrutador que no logró recibir este mensaje cronometrará, su estado e iniciará un nuevo mensaje de demolición más allá del punto de pérdida.
Asumiendo que la probabilidad de pérdida de un mensaje RSVP es pequeña, el mayor tiempo para borrar un estado raramente excede un periodo de actualización.

Debe ser posible demoler cualquier subconjunto de estados establecidos. Para estado de *Path*, la granularidad para la demolición es un sencillo remitente. Para estados de reserva, la granularidad es un individual *filter spec*. Por ejemplo, como se muestra en la figura 7, el receptor R1 envía un mensaje de *ResvTear* para el remitente S2 solamente (o para cualquier subconjunto de la lista del *filter spec*), dejando S1 en su lugar apropiado.

Un mensaje de *ResvTear* especifica el estilo y el filtro; cualquier *flowspec* es ignorado. Cualquier *flowspec* que esté en su lugar apropiado será borrado si todos sus *filter spec*s están demolidos.

2.5 ERRORES

Existen dos mensajes de error de RSVP, *ResvErr* y *PathErr*.
Los mensajes *PathErr* son muy simples; ellos se envían *upstream* hacia el remitente que creó el error, y no cambian el estado de camino en los nodos en los cuales ellos pasan. Existe solamente unas cuantas posibles causas de errores de *Path*.

Sin embargo, existen varias formas para que una reserva falle en un nodo a lo largo de una trayectoria. Un nodo puede decidir también si darle prioridad a una reserva establecida. La manipulación de los mensajes de *ResvErr* es algo compleja. Una falla en una solicitud puede ser el resultado de combinar un número de solicitudes, un error de reserva debe ser reportado a todos los receptores responsables. Además, combinar solicitudes heterogéneas, crean una dificultad potencial conocida como el problema "*killer reservation*[10]" en que una solicitud

[10] KEAGY, Scott. Integración de las redes de Voz y Datos. 1ª edición Cisco Systems 2001. p 452.

puede negar servicio para otra. Existen en la actualidad dos problemas de *killer-reservation*.

1. El primer problema *killer-reservation* (KR-I) se produce cuando existe ya una reserva Q0 en su lugar apropiado. Si otro receptor ahora hace una reserva más grande Q1 > Q0, el resultado de fusionar Q0 y Q1 puede ser rechazada por el control de admisión en algunos nodos *upstream*. Esto no debe negar servicio a Q0.

   La solución a este problema es simple: cuando el control de admisión fracasa para una solicitud de reserva, cualquier reserva existente se queda en su lugar apropiado.

2. El segundo problema de *killer-reservation* (KR-II) es lo contrario: el receptor sigue haciendo la reserva Q1 aunque el control de admisión está fracasando para Q1 en cierto nodo. Esto no debe impedir que un receptor diferente que ahora está estableciendo una reserva más pequeña Q0 tenga éxito si no se combinó con Q1.

   Para resolver este problema, un mensaje de *ResvErr* establece un estado adicional, llamado el "estado de bloqueo (*blockade state*)", en cada nodo por donde pasan. El estado de bloqueo en un nodo modifica el procedimiento de fusión para omitir el *flowspec* delincuente (Q1 en el ejemplo) desde la fusión, permitiendo que una solicitud más pequeña sea remitida y establecida. El estado de reserva de Q1 pasa a ser " bloqueado".

Una solicitud de reserva que suspende el control de admisión crea un estado bloqueado pero está en su lugar apropiado en los nodos *downstream* del punto de

falla. Se ha sugerido que estas reservas *downstream* de la falla representan reservas "devastadas" y deben ser de duración determinada de lo contrario se borran activamente. Sin embargo, las reservas *downstream* se quedan en su lugar apropiado, por las siguientes razones:

- Existe dos posibles razones para un receptor persistente en **a** falle una reserva: (1) está eligiendo recursos disponibles a lo largo del camino entero, o (2) quiere obtener el QoS deseado a lo largo del camino como sea posible.
  Desde luego en el segundo caso, y tal vez en el primer caso, el receptor querrá mantener las reservas que ha hecho *downstream* desde la falla.

- Si estas reservas *downstream* no eran retenidas, la sensibilidad de RSVP para obtener fallas transitorias podría ser deteriorada, por ejemplo suponga una bandera de ruta para una ruta alternativa que está congestionada, la reserva existente súbitamente falla, entonces rápidamente recupera la ruta original. El estado de bloqueo en cada enrutador *downstream* no debe remover el estado o previene esto actualizándose inmediatamente.

- Si no se actualiza las reservas *downstream*, ellas pueden cronometrar, para restaurarse cada Tb segundos (donde Tb es el intervalo del estado de bloqueo).

2.6 CONFIRMACIÓN

Para pedir una confirmación para una solicitud de reserva, un receptor Rj incluye en el mensaje de *Resv* un objeto de solicitud de confirmación que contiene la

dirección de IP de Rj. En cada punto de combinación, sólo el *flowspec* más grande y cualquier objeto de solicitud de confirmación adjunto son remitidos *upstream*. Si la reserva pedida a Rj es igual o más pequeña que la reserva en el nodo, su *Resv* no es remitida, y si tiene incluido una solicitud de confirmación, un mensaje de *ResvConf* es enviado a Rj. Esta solicitud no se remite más de una vez por cada requerimiento.

Este mecanismo de confirmación tiene las siguientes consecuencias:

- Una nueva solicitud de reserva con un *flowspec* más grande que cualquiera de una sesión normalmente dé por resultado o un *ResvErr* o un mensaje de *ResvConf* de vuelta al receptor de cada remitente. En este caso, el mensaje de *ResvConf* será una confirmación end-to-end.

- Recibir una *ResvConf* no da ninguna garantía. Asume las dos primeras solicitudes de reserva de los receptores R1 y R2 llegan al nodo donde se combinan. R2, cuya reserva fue la segunda en llegar a este nodo, puede recibir un *ResvConf* de ese nodo mientras que la solicitud de R1 no halla propagado por todo el camino a un remitente de igualación y este aun puede fracasar. Así, R2 puede recibir un *ResvConf* aunque no exista ninguna reserva end-to-end en su lugar apropiado; además, R2 puede recibir un *ResvConf* seguida por un *ResvErr*.

2.7 CONTROL DE POLITICA

Las solicitudes de QoS mediante RSVP permiten a un usuario particular para obtener acceso preferente a los recursos de red. Para impedir abusos, alguna forma de restricciones generalmente son requeridas en usuarios que hacen reservas. Por ejemplo, tales restricciones pueden realizarse por políticas de acceso administrativas, o puede depender de cierta forma de la realimentación de usuario o facturación virtual para los "costos" de una reserva. En todo caso, Una identificación de usuario confiable y la admisión selectiva se necesitará generalmente cuando una reserva es requerida.

El termino "control de política" es usado para los mecanismos requerido para soportar políticas de acceso y las restricciones para las reservas de RSVP.
Cuando una nueva reserva es pedida, cada nodo debe contestar dos preguntas: " ¿Están los recursos disponibles para encontrar esta solicitud?" y "¿se le tiene permitido a este usuario hacer esta reserva?" Estos dos decisiones son: la decisión de "control de admisión" y la decisión "control de política", respectivamente, y ambas deben ser favorable para que RSVP haga una reserva. Diferentes campos administrativos en la Internet pueden tener diferentes políticas de reserva.

La entrada para el control de política es mencionada como los "datos de política", que RSVP acarrea en POLICY_DATA. Los datos de política pueden incluir las credenciales que identifican usuarios o clases de usuarios, números de cuentas, límites, cuotas, etc. Como los *flowspec*s, los datos de política son opacos para RSVP, que simplemente pasan al control de política cuando son exigidos.

Similarmente, la fusión de los datos de política debe ser hecho por los mecanismos de control de política antes que por RSVP mismo. Note que los puntos de fusión para los datos de política están probablemente en los límites de los campos administrativos. Por lo tanto es necesario acarrear datos de política acumulados y no fusionados vía *upstream* por múltiples nodos antes de extender uno de estos puntos de fusión.

Los datos de política suministrados por el usuario llevado en un mensaje *Resv* presentan un problema potencial. Cuando un grupo de *multicast* tiene un gran número de receptores, será imposible o indeseable para acarrear todos los datos de política de receptores vía *upstream*. Los datos de política tendrán que combinarse administrativamente en lugares cerca de los receptores, para evitar datos de política excesivos.

2.8 SEGURIDAD

RSVP maneja los siguientes asuntos de seguridad.

- Integridad de mensaje y autenticación de nodo

  Las solicitudes de reserva corrompidas pueden llevar a hurto de el servicio por partidas desautorizadas o a la denegación de servicio causado por cerrar los recursos de la red. RSVP protege contra tales ataques con un mecanismo de autenticación salto por salto usando una función de hash codificada. El mecanismo es sustentado por la integridad de los objetos que van apareciendo en cualquier mensaje RSVP.

- Autenticación de usuario

  El control de política dependerá de autenticación positiva del usuario responsable para cada solicitud de reserva. Los datos de política pueden incluir por lo tanto usuario certificados criptográficamente protegidos.

Aún sin certificados de usuario mundialmente comprobables, puede ser posible proporcionar autenticación de usuario práctica en mucho casos estableciendo, una cadena de la confianza, usando el salto por salto el mecanismo de integridad describió anteriormente.

- Corrientes de datos seguras

  Los dos primeros asuntos de seguridad conciernen operación de RSVP. Un tercer asunto de seguridad concierne a las reservas de recurso para corrientes de datos seguras. En particular, el uso de IPSEC[11] (La seguridad en IP) en la corriente de datos presenta un problema para RSVP: si el transporte y las cabeceras de mayor nivel están codificados, el número de puertos generalizados de RSVP no pueden ser usados para definir una sesión o un remitente.

Para resolver este problema, una extensión de RSVP se ha definido en la cual el identificador de asociación de seguridad (IPSEC SPI) juega un papel ásperamente equivalente a los puertos generalizados.

[11] RFC 2207 “RSVP Extensions for IPSEC Data Flows” Available from Internet: http://www.faqs.org/rfcs/rfc2207.html

2.9 NUBES DE NO RSVP

Es imposible desplegar el RSVP (o cualquier nuevo protocolo) en el mismo momento a lo largo de toda la Internet. Además, es posible que RSVP nunca se despliegue en todas partes. RSVP debe proporcionar por lo tanto una correcta operación de protocolo aún cuando dos enrutadores capaces de suministrar RSVP se unen por un "nube" arbitraria de enrutadores de no-RSVP. Por supuesto, una nube intermedia que no soporta RSVP, no puede realizar una reserva de recursos. Sin embargo, si tal nube tiene suficiente capacidad, todavía puede proporcionar servicio en tiempo real útil.

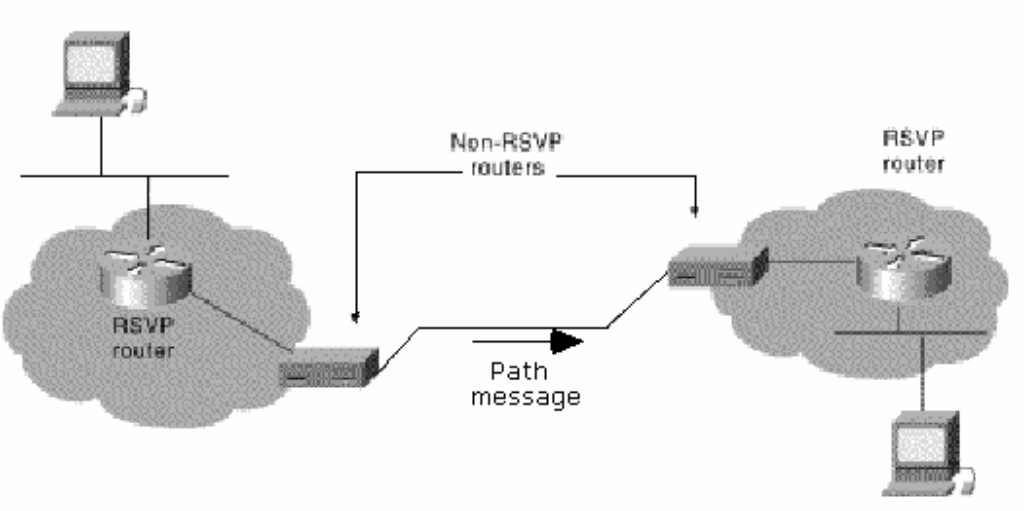
RSVP es diseñado para operar correctamente por la nube no-RSVP. Ambos enrutadores de RSVP y no-RSVP remiten los mensajes de *Path* hacia la dirección de destino usando su tabla local de enrutamiento de uni-*multicast*. Por lo tanto, el enrutamiento de los mensajes de *Path* no será afectado por los enrutadores de no-RSVP en el camino. Cuando un mensaje de *Path* atraviesa una nube de no-RSVP, esta lo lleva al próximo nodo capaz de RSVP con la dirección de IP del último enrutador capaz de RSVP antes entrar en la nube. Un mensaje de *Resv* se remite entonces directamente al próximo enrutador capaz de RSVP en el camino hacia la fuente (Como se ilustra en la figura 12).

Aunque RSVP opera correctamente por una nube de no-RSVP, los nodos no capaces de RSVP perturbarán por lo general el QoS requerido para un receptor. Por lo tanto, RSVP pasa un bit de bandera “NonRSVP”" a el mecanismo local de control de tráfico cuando existe un salto en el camino que no es capaz de RSVP en el camino hacia un remitente dado. El control de tráfico combina este bit de

bandera con sus propias fuentes de información, y remite esta fusión a lo largo del camino a receptores usando Adspecs.

Ciertas topologías de enrutadores de RSVP y enrutadores de no-RSVP pueden causar que los mensajes *Resv* lleguen al equivocado nodo capaz de RSVP, o que lleguen a la equivocada interfaz del nodo correcto. Un proceso de RSVP debe estar preparado para soportar cualquier situación. Si la dirección de destino no encuentra cualquier interfaz local y el mensaje no es un *Path* o un *PathTear*, el mensaje debe remitirse sin proceso adicional por este nodo. Para manejar el caso de la interfaz equivocada una "Interfaz lógica de manejo" (LIH) es usada. La información de salto previa incluida en un mensaje de *Path* no tiene sólo la dirección de IP del nodo previo sino también un LIH definiendo la interfaz de salida lógica; ambos valores son guardados en el estado de *Path*. Un mensaje de *Resv* que está llegando el nodo dirigido lleva ambos; la dirección de IP y el LIH de la interfaz de salida correcta.

Figura 12: Nube de No RSVP



Fuente: http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rsvp.htm#xtocid1

3. ESPECIFICACIÓN FUNCIONAL DE RSVP

Esta sección define las caracteristicas tecnicas de RSVP, en lo que respecta a tiempos, interfaces logicas. formatos de los mensajes etc.

Estas caracteristicas son tomadas en cuenta en el momento de subir el protocolo e instalarlo en la red, aquí se define tamaño (bit) y retardo de la llamada para pedir la reserva de los recursos.

3.1 FORMATOS DE LOS MENSAJES DE RSVP

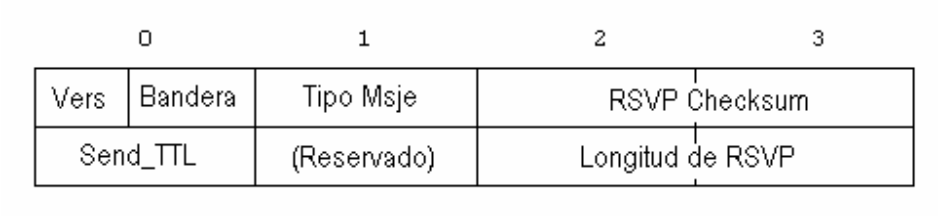
Un mensaje de RSVP consiste de una cabecera común (Figura 13), seguida por un cuerpo que contiene de un número variable de de longitud variable, conocido como "objetos". Las subdivisiones siguientes definen los formatos de la cabecera común, la cabecera de objeto estándar.

Para cada tipo de mensaje RSVP, existe un conjunto de reglamentos para la elección permisible del objeto. Estos reglamentos son especificados usando la forma Backus-Naur[12] (BNF) aumentado con corchetes rectángulos rodeando la subsucesion opcional. El BNF implica una orden para los objetos en un mensaje. Sin embargo, en muchos (pero no todos) casos, la orden de los objetos no hace ninguna diferencia lógica. Una implementación debe crear los mensajes con los objetos en orden mostrada aquí, pero debe aceptar los objetos en cualquier orden permisible.

[12] GARCIA, Tomas Jesús. Alta velocidad y Calidad de servicio en redes IP. 1ª edición 2002. p 250

3.1.1 Cabecera común

Figura 13: Cabecera común de RSVP



Fuente: http://www.faqs.org/rfcs/rfc2205.html

Los campos en la cabecera común son los siguientes:

Vers: 4 bits

Número de versión de protocolo. Esta es la versión 1.

Banderas: 4 bits

0x01-0x08: Reservado
Ningunos bits de bandera están definidos todavía.

Msg Type: 8 bits

1 = *Path*

2 = *Resv*

3 = *PathErr*

4 = *ResvErr*

5 = *PathTear*

6 = *ResvTear*

7 = *ResvConf*

RSVP *Checksum*: 16 bits

Este campo reemplaza por cero, los unos de la suma de los complementos a uno de los mensajes, para poder computar el *checksum*. Un valor de todos los bit en cero indica que el *checksum* no fue transmitido.

Send_TTL: 8 bits

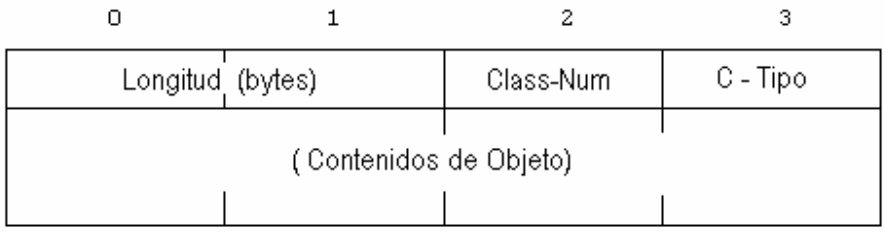
El valor IP TTL con que el mensaje fue enviado.

Longitud de RSVP: 16 bits

La longitud total de este mensaje RSVP en bytes, incluyendo la cabecera común y la longitud variable de objetos que siguen.

3.1.2 Formatos de objeto. Cada objeto consiste de unas o más palabras de 32-bit con una palabra de cabecera (Figura 14).

Figura 14: Cabecera de objeto



Fuente: http://www.faqs.org/rfcs/rfc2205.html

Una cabecera de objeto tiene los campos siguientes:

- Longitud

  Un campo de 16 bits que contiene la longitud de un objeto en bytes. Siempre debe ser un múltiplo de 4, o al menos 4.

- *Class-Num*

  Identifica la clase de objeto. Cada clase de objeto tiene un nombre. La implementación de RSVP debe reconocer las clases siguientes:

  - *NULL* (Nulo)

    Un objeto nulo tiene un Class-Num de cero, y su C-tipo es ignorado. Su longitud debe ser al menos 4, pero puede ser cualquier múltiplo de 4. Un objeto nulo puede aparecer en cualquier parte en una sucesión de objetos, y sus contenidos son ignorados por el receptor.

❖ SESIÓN

Contiene la dirección IP de destino (*DestAddress*), el protocolo IP , y cierta forma de puerto generalizado de destino, para definir una sesión específica para otros objetos que siguen. Son requeridos en cada mensaje de RSVP.

❖ *RSVP_HOP* (Saltos)

Lleva la dirección IP del nodo capaz de RSVP que ha enviado el mensaje y una interfaz lógica de salida. Los RSVP_Hops son referidos como un PHOP (" salto previo") mensajes *downstream* o un NHOP ("proximo salto") mensajes *upstream*.

❖ *TIME_VALUES* (Valores de tiempo)

Contiene el valor para el período de actualización R usado por el creador del mensaje; Requerido en cada mensaje de *Path* y *Resv*.

❖ ESTILO

Define el estilo de reserva más la información específica del estilo que no está en el *FLOWSPEC* o el FILTER_SPEC. Requerido en cada mensaje de *Resv*.

❖ *FLOWSPEC*

Define la QoS deseada, en un mensaje de *Resv*.

❖ *FILTER_SPEC*

Define un subconjunto de paquetes de datos de sesión que deben recibir el QoS deseado (especificado por un *FLOWSPEC*), en un mensaje de *Resv*.

❖ *SENDER_TEMPLATE* (Plantilla de Remitente)

Contiene la dirección IP del remitente y tal vez alguna información adicional demultiplexada para identificar un remitente. Requerido en un mensaje de *Path*.

❖ REMITENTE_*TSPEC*

Define las características de tráfico de un flujo de datos de un remitente. Requerido en un mensaje de *Path*.

❖ *ADSPEC*

Lleva los datos de OPWA, en un mensaje de *Path*.

❖ *ERROR_SPEC*

Especifica un error en un *PathErr*, o en un *ResvErr*, o una confirmación en un mensaje de *ResvConf*.

❖ DATOS DE POLÍTICA

Lleva la información que le permitirá a un módulo de política local decidir si una reserva asociada es administrativamente permitida. Puede aparecer en los mensajes de: *Path*, *Resv*, *PathErr*, o *ResvErr*.

❖ INTEGRIDAD

Lleva los datos criptográficos para autenticar el nodo original y para verificar los contenidos de este mensaje de RSVP.

❖ ALCANCE

Lleva una lista explícita de *host*s remitentes, hacia los cuales la información va a ser enviada. Puede aparecer en los mensajes de: *Resv*, *ResvErr*, o *ResvTear*.

- *RESV*_CONFIRM (Confirmación de *Resv*)
  Lleva la dirección IP de un receptor que pidió una confirmación. Puede aparecer en los mensajes de: *Resv* o *ResvConf*.

- C-Tipo
  Tipo de objeto, único dentro de Class-Num.

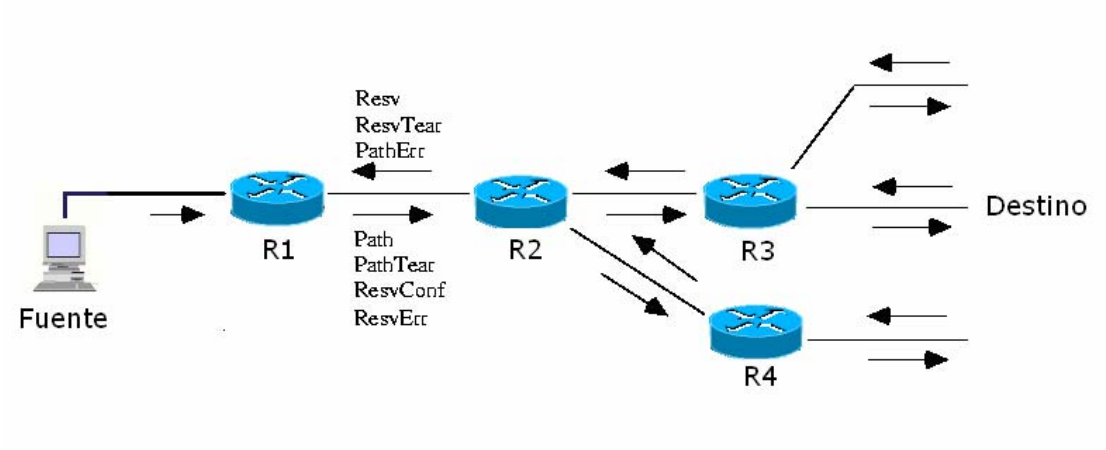
El máximo objeto contiene una longitud de 65528 bytes. El Class-Num y los campos de C-tipo pueden ser usados en conjunto como un número de 16-bit para definir un tipo único para cada objeto.

Los dos bits de orden alto del Class-Num son usados para determinar que acción debe tomar un nodo si este no reconoce el Class-Num de un objeto.

3.2 TRANSMITIENDO MENSAJES RSVP

Los mensajes de RSVP son enviados salto por salto entre enrutadores capaces del RSVP (Como se muestra en la figura 15) como datagramas IP "raw". Los datagramas de IP raw también intentando ser usados entre un fin de sistema y el primer/ultimo enrutador de salto, aunque eso es también posible para encapsular mensajes RSVP como datagramas de UDP para comunicación de fin de sistema. La encapsulación de UDP es necesitada para sistemas puedan hacer la red raw I/O.

Figura 15: Transmisión de los mensajes RSVP



Fuente: http://www.networksorcery.com/enp/protocol/rsvp.html.

Los mensajes de *Path*, *PathTear* y *ResvConf* deben ser enviados con la opción Alert IP del enrutador en sus cabeceras IP. Esta opción puede ser usada en el camino de despacho rápido de un enrutador de alta velocidad para detectar los datagramas que requieren un proceso especial.

Sobre la llegada de un mensaje RSVP “M “que cambia el estado, un nodo debe remitir la transformación de estado inmediatamente. Sin embargo, esto no debe enviar un mensaje fuera de la interfaz por el cual M llegó (que puede suceder si la ejecución simplemente dispara una actualización inmediata de todos los estado para la sesión).
Esta regla es necesaria para impedir las tormentas de paquetes en LANs difundidas.

En esta versión del spec, cada mensaje de RSVP debe ocupar exactamente un datagrama IP. Si este excede el MTU, tal datagrama será fragmentado por IP y reensamblado al nodo receptor. Esto tiene varias consecuencias:

- Un mensaje de RSVP sencillo no puede exceder el tamaño máximo del datagrama IP, aproximadamente de 64K bytes.

- Una nube de no-RSVP congestionada puede perder los fragmentos del mensaje individual, y como consecuencia se perderá el mensaje entero.

Las versiones futuras del protocolo proporcionarán soluciones para éstos problemas si resultan pesados. La más probable puede ser ejecutar la "fragmentación semántica", que es romper el estado de *Path* o de reserva para ser transmitidos en múltiples mensajes autocontenidos, cada uno de un tamaño aceptable.

RSVP usa un mecanismo periódico de actualización para recobrarse de pérdidas de paquetes ocasionales. Bajo sobrecarga de red, sin embargo, las pérdidas substanciales de los mensajes de RSVP pueden causar una falla en las reservas de recurso. Para controlar la demora y caída de paquetes, los enrutadores de RSVP deben configurarse para ofrecerles una preferida clase de servicio. Si los paquetes de RSVP experimentan perceptibles pérdidas cruzando una nube de no-RSVP congestionada, un valor más grande pueda ser usado para el factor K de tiempo de espera.

Algunos protocolos de enrutamiento *multicast* proveen los túneles de *Multicast* los cuales hacen encapsulación de IP de paquetes de *multicast* para la transmisión por enrutadores que no tienen capacidad de *multicast*. Un túnel de *multicast* (cuya configuración se muestra en la figura 16) parece una interfaz lógica de salida que está combinada con una interfaz física. Un protocolo de enrutamiento *multicast*

que soporta túneles describirá una ruta que usa una lista de lógicas interfaces. RSVP puede operar a través tales túneles de *multicast* de la manera siguiente:

1. Cuando un nodo N remite un mensaje de *Path* hacia una interfaz de salida lógica L, este incluye en el mensaje cierta codificación de la identidad de L, llamado " el manejo de la interfaz lógica" o LIH.
   El valor de LIH es llevado en el objeto RSVP_HOP.

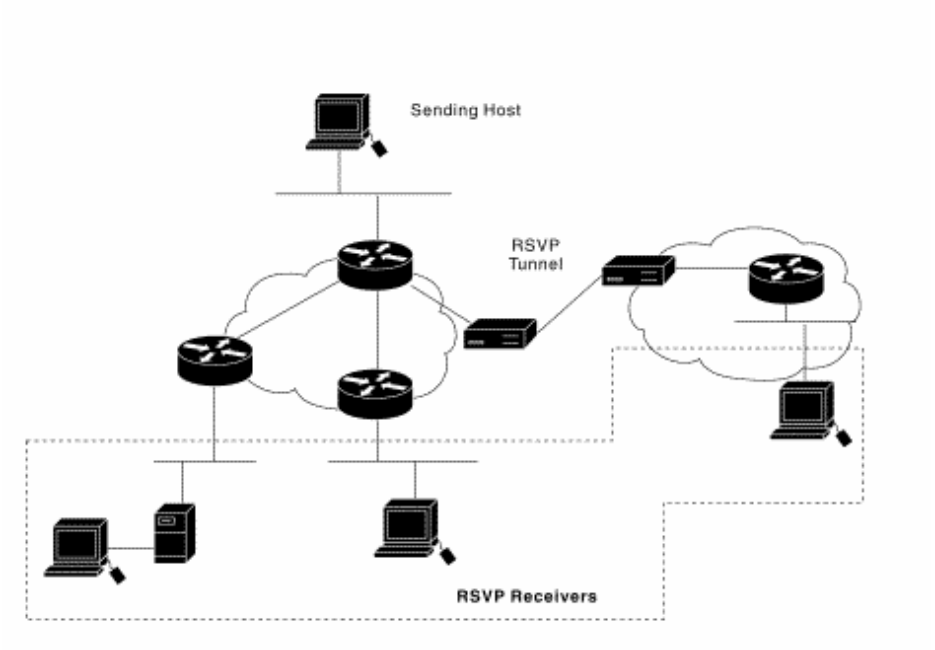
2. El próximo nodo N' almacena el valor de LIH en su estado de *Path*.

3. Cuando N' envía un mensaje de *Resv* N, este incluye el valor de LIH de el estado de *Path* (de nuevo, en el objeto de RSVP_HOP).

4. Cuando el mensaje de *Resv* llega a N, su valor de LIH proporciona la información necesaria para unir la reserva a la interfaz lógica apropiada. Note que N crea e interpreta el LIH.

Note que esto resuelve sólo el problema de asignación de ruta presentado por los túneles.
El túnel aparece para RSVP como una nube de no-RSVP. Para establecer reservas RSVP dentro del túnel, una maquinaria adicional será requerida.

Figura 16: Configuración del túnel de RSVP



Fuente: http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rsvp.htm#xtocid1

3.3 PARÁMETROS DE TIEMPO

Existen dos parámetros de tiempo relevantes para cada elemento de RSVP en un nodo: el período de actualización R entre la generación de actualizaciones sucesivas para el estado por el nodo vecino, y el tiempo de vida del estado local L. Cada mensaje RSVP *Resv* o *Path* puede contener un objeto de TIME_VALUES especificando el valor R que fue usado para generar estos mensajes de actualización. Este valor de R se usa entonces para determinar el valor de L

cuando el estado es recibido y almacenado. Los valores de R y L pueden variar salto por salto.

Con más detalle:

1. Se ha mostrado que los mensajes periódicos generados por nodos de red independientes pueden llegar a estar sincronizados. Esto puede llevar a una interrupción en los servicios de la red ya que como los mensajes periódicos compite con otro tráfico de red por enlace y despacho de recursos. Después que RSVP envía los mensajes periódicos de actualización, debe evitar mensajes de sincronización y asegurar que no ocurran.

   Por esta razón, el tiempo de actualización debe configurarse en el rango [0.5R, 1.5R].

2. Para evitar una pérdida prematura de estado, L[13] debe satisfacer L >= (K + 0.5)x1.5R, donde K es un pequeño numero entero. Entonces en el peor caso, K-1 mensajes sucesivos pueden perderse sin borrar el estado. Para computar un tiempo de vida L para una colección de estados valores diferentes de R, R0, R1,..., reemplace R por un máximo (Ri).

   Normalmente un valor K = 3 es sugerido. Sin embargo, puede ser necesario poner un valor de K más grande para saltos con grandes tazas de pérdida. K puede ser establecido por configuración manual por interfaz, o por cierta técnica adaptable que todavía no ha sido especificada.

[13] RFC 2205. Op Cit.

3. Cada mensaje de *Path* o de *Resv* lleva un objeto de TIME_VALUES que contiene el tiempo R usado para generar actualizaciones. El nodo remitente usa este R para determinar el tiempo de vida L de el estado almacenado, creado o actualizado por el mensaje (Resv o Path).

4. El tiempo R es escogido localmente por cada nodo. Si el nodo no utiliza la reparación local de reservas que se desorganizan por cambios de ruta, un R más pequeño hace acelerar la adaptación para cambios de ruta, mientras incrementa la cabecera RSVP. Un nodo puede ajustar por lo tanto con dinamismo un R efectivo para controlar el valor de la cabecera debido a los mensajes de actualización.

   El valor normal de R default son 30 segundos. Sin embargo, el valor implícito Rdef puede ser configurado por interfaz.

5. Cuando R es cambiado dinámicamente, existe un límite en cuan rápido puede incrementar. Específicamente, la relación de dos valores sucesivos R2/R1 no debe exceder 1 + Slew.Max.

   Normalmente, el valor de Slew.Max es 0.30. Con K = 3, un paquete puede perderse sin que se haya agotado el tiempo de estado mientras R aumente un 30% por ciclo de actualización.

6. Para mejorar robustez, un nodo puede enviar temporalmente actualizaciones más a menudo que R después de un cambio de estado (incluyendo el establecimiento del estado inicial).

7. Los valores de Rdef, K, y Slew.Max usados en una ejecución pueden ser fácilmente modificable por interfaz, ya que en la experiencia este toma valores diferentes.

3.4 INTERFACES DE RSVP

RSVP en un enrutador tiene interfaces para asignación de ruta y para control de tráfico. RSVP en un *host* tiene una interfaz para aplicaciones y también una interfaz para control de tráfico (si este existe en el *host*).

3.4.1 RSVP Interfaz/Aplicación Esta sección describe[*] una interfaz genérica entre una aplicación y un control del proceso RSVP. Los detalles de una interfaz real pueden ser operados por un sistema dependiente; la siguiente solo puede indicar las funciones básicas que van a ser ejecutadas. Algunos de estas llamadas causan que la información sea retornada asincrónicamente.

- Registro de Sesión (Register Session)

```
Call: SESSION (DestAddress, ProtocolID, Dstport
        [ , SESSION_object ]
      [ , Upcall_Proc_addr ] )  -> Session-id
```

Esta llamada inicia el proceso RSVP para una sesión, definido por *DestAddress* junto al *ProtocolID* y posiblemente un número de puerto *Dstport*. Si es exitosa, la llamada de Sesión retorna inmediatamente con un

[*] RFC 2205. Ibid.

identificador de sesión local Session_id, que puede ser usado en llamadas subsecuentes.

El parámetro Upcall_Proc_addr define la dirección de un procedimiento de upcall para recibir error asincrónico o un evento de notificación. El parámetro SESSION_object es incluido como un mecanismo de escape para soportar algunas definiciones mas generales de la sesión (“generales puerto de destino "), debe ser necesario en el futuro. Normalmente el SESSION_objet será omitido.

- Definición de Remitente (Define Sender)

```
Call: SENDER( Session-id
      [ , Source_Address ]  [ , Source_Port ]
      [ , Sender_Template ]
      [ , Sender_Tspec ]    [ , Adspec ]
     [ , Data_TTL ]        [ , Policy_data ] )
```

Un remitente usa esta llamada para definir, o para modificar, los atributos del flujo de datos. La primera Sender call registrada como Session_id causará que RSVP comience a enviar mensajes de *Path* para esta sesión; las llamadas posteriores modificarán la información de camino.

Los parámetros de remitente son interpretados como sigue:

❖ Source_Address

Esta es la dirección de la interfaz de donde el cual los datos serán enviados. Si este es omitido, una interfaz default será usada. Este parámetro es necesitado sólo en un *host* remitente de multihomed.

❖ Source_Port

Este es el puerto de UDP/TCP de el cual los datos serán enviados.

❖ Sender_Template

Este parámetro es incluido como un mecanismo de escape para soportar una definición más general del remitente. Normalmente este parámetro puede omitirse.

❖ Sender_Tspec

Este parámetro describe el tráfico de flujo de que será enviado.

❖ Adspec

Este parámetro puede especificarse para inicializar la computación de las propiedades de QoS a lo largo del camino.

❖ Data_TTL

Este es el parámetro (no implícito) IP Time-To-Live que está sirviendo como substituto en los paquetes de datos. Es necesitado para asegurar

que los mensajes de *Path* no tengan un alcance mas largo que los paquetes de datos de *multicast*.

- Policy_data

  Este parámetro opcional pasa datos de política para el remitente. Estos datos pueden suministrarse por un sistema de servicio, con la aplicación tratándolo como opaco.

- Reserva

```
Call: RESERVE( session-id, [ receiver_address , ]
                [ CONF_flag, ] [ Policy_data, ]
                 style, style-dependent-parms )
```

Un receptor usa esta llamada hacer o para modificar una reserva para la sesión registrada como el "session_id". La primera llamada de RESERVA iniciará la periódica transmisión de mensajes *Resv*. Una llamada de reserva posterior puede ser dada para modificar los parámetros de la anterior (pero note que los cambios en las reservas existentes pueden causar fallas en el control de admisión).

El parámetro opcional receiver_address puede usarse por un receptor en un *host* multihomed (o por el enrutador); es la dirección IP de la interfaz de un nodo. La CONF_flag [14]debe estar en on si una confirmación de una reserva es deseada, off si sucede otra cosa. El parámetro Police_data especifica los datos de política para el receptor, mientras que el parámetro de "style"

[14] RFC 2205. Ibid.

indica el estilo de reserva. El resto de los parámetros dependen del estilo; generalmente éstos serán apropiados *flowspec*s y *filter spec*s.

La llamada de RESERVA retorna inmediatamente. Seguida a esta un upcall de ERROR/EVENT asincrónico pueda ocurrir en cualquier momento.

- Liberación[15] (Release)

```
Call: RELEASE( session-id )
```

Esta llamada quita el estado RSVP para la sesión especificada por la session_id. El nodo entonces envía un mensaje de demolición apropiado y luego envia una actualización para esta session_id

- Error/Event Upcalls

La forma general de un upcall es como sigue:

```
Upcall: <Upcall_Proc>( ) -> session-id, Info_type,
                            information_parameters
```

Aquí "Upcall_Proc" representa el procedimiento de upcall cuya dirección fue substituida en la llamada de SESION. Este upcall puede ocurrir asincrónicamente en cualquier momento después de una llamada de SESION y antes de una llamada de LIBERACIÓN, para indicar un error o un evento.

[15] RFC 2205. Ibid.

Normalmente existen cinco tipos de upcall, distinguidos por el parámetro Info_type. La selección de los parámetros de información dependen del tipo.

1. Info_type = *PATH*_EVENT

Un upcall *Path* Event resulta de de recibir el primen mensaje de *Path* para esta sesión, indicando a una aplicación de receptor que tiene al menos un remitente activo, o si el estado de *Path* cambia.

```
Upcall: <Upcall_Proc>( ) -> session-id,
        Info_type=PATH_EVENT,
    Sender_Tspec, Sender_Template
      [ , Adspec ] [ , Policy_data ]
```

Este upcall presenta el Sender_Tspec, el Sender_Template, el Adspec, y cualquier dato de política de un mensaje de *Path*.

2. Info_type = *RESV*_EVENT

Un upcall de *R*[16]*esv* Event es disparado por el recibimiento de el primer mensaje *RESV*, o por la transformación de un estado de reserva previo, para esta sesión.

```
Upcall: <Upcall_Proc>( ) -> session-id,
        Info_type=RESV_EVENT,
```

[16] RFC 2205. Ibid.

```
Style, Flowspec, Filter_Spec_list
[ , Policy_data ]
```

Aquí el " *Flowspec* " será el QoS efectivos que ha sido recibido. Note que un mensaje de *Resv* de estilo de FF puede resultar en múltiples upcalls *RESV*_EVENT, uno por cada descriptor de flujo.

3. Info_type = *PATH*_ERROR

Un evento de error de *Path* indica un error en la información del remitente que fue especificada en una llamada de SENDER.

```
Upcall: <Upcall_Proc>( ) -> session-id,
        Info_type=PATH_ERROR,
        Error_code , Error_value ,
    Error_Node , Sender_Template
        [ , Policy_data_list ]
```

El parámetro Error_code definirá el error, y Error_value puede suministrar algunos datos adicionales del error. El parámetro Error_Node especificará la dirección IP del nodo que detectó el error. El parámetro Police_data_list si está presente, podrá contener cualquier objeto de POLICY_DATA del mensaje de *Path* que falló.

4.Info_type = *RESV*_ERRE

Un evento de error de *Resv* indica un error en un mensaje de reserva al que esta aplicación contribuyó.

```
Upcall: <Upcall_Proc>( ) -> session-id,
                    Info_type=RESV_ERROR,
                    Error_code , Error_value ,
                    Error_Node , Error_flags ,
                    Flowspec, Filter_spec_list
                    [ , Policy_data_list ]
```

El parámetro error_code definirá el error y el Error_value puede suministrar algunos datos adicionales. El parámetro Error_Node puede especificar la dirección IP del nodo que detectó el evento siendo anunciado.

Existen dos Error_Flags:

- InPlace

  Esta bandera puede estar en On para una falla del control de admisión, para indicar que allí estaba, y está, una reserva en su lugar apropiado en el nodo que falla. Esta bandera es puesta en el punto de falta y remitida en un mensaje *ResvErr*.

- NotGuilty

  Esta bandera puede estar en On para una falla en el control de admisión, para indicar que el *flowspec* pedido por este receptor era estrictamente menos que el *flowspec* que consiguió el error. Esta bandera es puesta un receptor API.

El Filter_spec_list y el *Flowspec* contienen los correspondientes objetos del descriptor de error de flujo. List_count especifica el número de *Filter spec*s en el Filter_spec_list. El parámetro Policy_data_list contendrá cualquier POLICY_DATA del mensaje de *ResvErr*.

5. Info_type = *RESV*_CONFIRM

   Un evento de confirmación indica que un mensaje de *ResvConf* fue recibido.

```
Upcall: <Upcall_Proc>( ) -> session-id,
                            Info_type=RESV_CONFIRM,
                            Style, List_count,
                            Flowspec, Filter_spec_list
                            [ , Policy_data ]
```

   Los parámetros son interpretados como en el upcall error de *Resv*.

Aunque los mensajes RSVP indicando eventos de *Path* o *Resv* pueden ser recibidos periódicamente, el API debe hacer el correspondiente upcall asincrónico a la aplicación sólo en la primera ocurrencia o cuando la información reporte cambios. Todos los eventos de error y confirmación deben reportarse a la aplicación.

3.4.2 Interfaz RSVP/Traffic Control. Es difícil presentar una interfaz genérica para el control de tráfico, porque los detalles de establecer una reserva dependen fuertemente de la tecnología particular del nivel de enlace en el uso de una interfaz.

La fusión de las reservas de RSVP es exigida debido a la entrega de datos en *multicast*, el cual replica los paquetes de datos para la entrega a diferentes nodos de salto próximo. En cada punto de réplica, RSVP debe combinar la reserva del correspondiente salto proximo computando el "máximo" de sus *flowspec*s. En un dado enrutador o *host*, uno o más de la tres localizaciones de réplica siguientes pueden usarse.

1. Capa de IP

   El despacho de IP *multicast* ejecuta réplicas en la capa IP. En este caso, RSVP debe combinar las reservas que están en ese lugar en las interfaces de salidas correspondientes a fin de remitir una solicitud *upstream*.

2. "La red"

   La réplica podría tomar lugar *downstream* desde el nodo, en una LAN difundida, en los switches de capa de enlace, o en una malla de enrutadores no capaces de RSVP (ver sección 2.8). En estos casos, RSVP debe combinar las reservas de los próximos saltos diferentes a fin de hacer la reserva en una interfaz de salida sencilla. Debe combinarse también las reservas de todas las interfaces de salida para remitirlas en una solicitud *upstream*.

3. Conductor de capa de enlace

   Para una tecnología multi-access, la réplica puede ocurrir en el conductor de nivel de enlace o tarjeta de interfaz. Por ejemplo, este caso podría levantarse cuando existe un punto de ATM[17] separado punto a punto ATM-VC hacia cada salto proximo. RSVP puede necesitar aplicar el control de tráfico independientemente para cada VC, sin combinarse los requerimientos de saltos próximos diferentes.

Por lo general, estas complejidades no impactan en el proceso del protocolo que es requerido por RSVP, a excepción de determinar exactamente que solicitudes de reserva necesitan combinarse. Puede ser deseable para organizar una ejecución de RSVP en dos partes: un centro que ejecute un proceso independiente de capa de enlace, y una capa de adaptación dependiente de la capa de enlace. Sin embargo, existe aquí una interfaz genérica que asume que la réplica puede ocurrir sólo en la capa de IP o en " la red ".

- Hacer una Reserva

```
Call: TC_Add Flowspec( Interface, TC_Flowspec,
                TC_Tspec, TC_Adspec, Police_Flags )
                    -> RHandle [, Fwd_Flowspec]
```

   El parámetro TC_*Flowspec* define la deseada QoS efectiva al control de admisión; su valor es computado como el máximo de los *flowspec*s de saltos próximos diferentes. El Parámetro TC_Tspec define el efectivo

[17] RSVP, Resource ReSerVation Protocol (on line). Network Sorcery, Inc, 2004. Available from Internet http://www.networksorcery.com/enp/protocol/rsvp.html.

remitente Tspec *Path*_Te (vea sección 2.2). El parámetro TC_Adspec define el efectivo Adspec. El parámetro Police_Flags lleva las tres banderas E_Police_Flag, M_Police_Flag, y B_Police_Flag[18];

Si esta llamada es exitosa, establece un nuevo canal de reserva correspondiente a RHandle; de otra manera, retorna un código de error. El número opaco RHandle es usado por el llamador para referencias subsecuentes para esta reserva. Si el servicio de control de tráfico actualiza el *flowspec*, la llamada retornará también el objeto actualizado como Fwd_*Flowspec*.

- Modificar una reserva

```
Call: TC_Mod Flowspec( Interface, RHandle, TC_Flowspec,
                       TC_Tspec, TC_Adspec, Police_flags )
                                          [ -> Fwd_Flowspec
```

Esta llamada es usada para modificar una reserva existente. El parámetro TC_*Flowspec* es pasado al control de Admisión; si este es rechazado, el *flowspec* actual queda vigente. Los correspondientes *filter spec*s, no son afectados. Los otros parámetros son definidos como TC_Add*Flowspec*. Si se estan actualizando los *flowspec*s, la llamada retornará también el objeto actualizado como Fwd_*Flowspec*.

- Borrar *Flowspec*s

```
Call: TC_Del Flowspec( Interface, RHandle )
```

[18] RFC 2205. Op Cit.

Esta llamada borrará una reserva existente, incluyendo el *flowspec* y todos los asociados *filter spec*s.

- Agregar *Filter spec*s

```
Call: TC_AddFilter( Interface, RHandle,
                    Session , Filterspec ) -> FHandle
```

Esta llamada es usada para asociar un *filter spec* adicional con la reserva especificada por el RHandle dado, seguido de una exitosa llamada de TC_Add*flowspec*. Esta llamada retorna un filter handle FHandle.

- Borrar *filter spec*s

```
Call: TC_DelFilter( Interface, FHandle)
```

Esta llamada es usada para quitar un *filter spec*, especificado por FHandle.

- Actualizar OPWA

```
Call: TC_Advertise( Interface, Adspec,
                    Non_RSVP_Hop_flag ) -> New_Adspec
```

Esta llamada es usada por OPWA para computar el anuncio de salida New_Adspec para una interfaz especificada. El bit de bandera Non_RSVP_Hop debe ser establecido cuando RSVP detecta que el salto anterior incluye uno o más enrutadores no capaces de RSVP. TC_Advertise insertará esta información en un New_Adspec para indicar que un salto de non-integrated-service fue encontrado.

- Prioridad de Upcall

```
Upcall: TC_Preempt() -> RHandle, Reason_code
```

A fin de otorgar una nueva solicitud de reserva, el control de admisión y/o los módulos de control de política pueden darle prioridad una o más reservas existentes. Esto disparará un upcall de TC_Preempt a RSVP para cada reserva prioritaria, pasando el RHandle de la reserva y un subcódigo indicando la razón.

3.4.3 Interfaz RSVP/Routing. Una ejecución de RSVP necesita el siguiente soporte de los mecanismos de enrutamiento del nodo.

- Pregunta de Ruta (Route Query)

Para remitir mensajes de *Path* y *PathTear*, un proceso de RSVP debe ser capaz de preguntar al proceso(s) de asignación de ruta por las vías.

```
Ucast_Route_Query( [ SrcAddress, ] DestAddress,
                        Notify_flag ) -> OutInterface

    Mcast_Route_Query( [ SrcAddress, ] DestAddress,
                                Notify_flag )
              -> [ IncInterface, ] OutInterface_list
```

Dependiendo del protocolo enrutamiento, la pregunta puede o no puede depender de SrcAddress. IncInterface es la interfaz por la cual se estima que el paquete llegue; algunos protocolo de enrutamiento de *multicast* no la pueden proporcionar. Si la Notify_flag es verdadera, la asignación de ruta salvará el estado necesario para emitir una no solicitada notificación de cambios de ruta, siempre que la ruta especificada cambie.

Una pregunta sobre rutas *multicast* puede retornar una vacía OutInterface_list si no existen ningunos receptores *downstream* de un enrutador particular. Una pregunta de ruta puede retornar también un error de `No such route', probablemente a causa de una inconsistencia temporal en el enrutamiento (desde que un mensaje de *Path* o *PathTear* para la ruta pedida llegó a ese nodo). En cada caso, el estado local debe actualizarse como es pedido por el mensaje, que no puede ser remitido. Actualizar el estado local hará disponible inmediatamente el estado de *Path* para un nuevo receptor local, o puede demoler el estado de *Path* inmediatamente.

- Notificación de Cambios de Ruta

Si es pedida por una pregunta de ruta con el Notify_Flag verdadero, el proceso de enrutamiento puede proporcionar un callback asincrónico al proceso de RSVP que una ruta especificada ha cambiado.

```
Ucast_Route_Change( ) -> [ SrcAddress, ] DestAddress,
                                        OutInterface
Mcast_Route_Change( ) -> [ SrcAddress, ] DestAddress,
                [ IncInterface, ] OutInterface_list
```

- Interfaz de descubrimiento de lista (*Interface List Discovery*)

RSVP debe ser capaz de aprender cuales interfaces reales o virtuales están activas, con sus direcciones IP. Debe ser posible para RSVP deshabilitar lógicamente una interfaz. Cuando una interfaz es incapacitada por RSVP,

mensaje de *Path* es recibido en esa interfaz, y si un mensaje es recibido en esa interfaz, este debe ser descartado.

3.4.4 Interfaz RSVP/Packet I/O. Una ejecución de RSVP necesita el siguiente soporte del paquete I/O y los mecanismos de despacho del nodo.

- Modo Receptor Promiscuo para Mensajes RSVP

Los paquetes recibidos por el protocolo IP versión 4 o 6 pero que no están dirigidos a el nodo, deben desviarse al programa RSVP para que sean procesados, sin ser remitidos. Los mensajes RSVP que son desviados de esta manera incluirán mensajes *Path*, *PathTear*, y *ResvConf*. Estos tipos de mensaje llevan la opción Router Alert IP, que puede ser usada para escogerlos de un camino de despacho de alta velocidad. Alternativamente, el nodo puede interceptar todos los paquetes del protocolo versión 4 o 6.

En un enrutador o *host multihomed* [19] la identidad de la interfaz (real o virtual) en la que un mensaje desviado es recibido, así como la dirección IP de fuente e IP TTL con la cual llegó, también debe estar disponible al proceso RSVP.

- Especificación de Enlace de Salida

RSVP debe ser capaz de forzar un (*multicast*) datagrama para ser enviado a un enlace de salida específico real o virtual, desviando el mecanismo normal de enrutamiento. Un enlace virtual puede ser un túnel de *multicast*, por ejemplo. Una especificación de enlace saliente es necesaria para enviar

[19] GARCIA, Alberto León. Redes de comunicación, conceptos fundamentales y arquitecturas básicas. 1ª edición. p. 202.

versiones diferentes de un saliente mensaje de *Path* a interfaces diferentes, y para evitar los lazos en la ruta en algunos casos.

- Dirección de Fuente y Especificación TTL

  RSVP debe ser capaz de especificar la dirección IP de fuente e IP TTL para ser usados al enviar mensajes de *Path*.

- Alerta de Enrutador

  RSVP debe ser capaz de causar mensajes de *Path*, *PathTear*, y *ResvConf* para enviarse con la opción Router Alert IP.

3.4.5 Manipulaciones dependientes de servicio. *Flowspec*s, Tspecs, y Adspecs[20] son objetos opacos a RSVP; sus contenidos son definidos en los documentos de especificación de servicio. A fin de manipular estos objetos, el proceso de RSVP debe tener disponible las siguientes rutinas dependientes del servicio.

- Comparar *Flowspec*s

  ```
  Compare_Flowspecs( Flowspec_1, Flowspec_2 ) ->
                                          result_code
  ```

  El posible result_code indica: *flowspec*s iguales, *Flowspec*_1 mayor, *Flowspec*_2 mayor, los *flowspec*s son incomparables pero el LUB puede computarse, o los *flowspec*s son incompatibles.

[20] RFC 2205. Op Cit.

Note que comparando dos *flowspec*s se comparan implícitamente los Tspecs que contienen. Aunque el proceso RSVP no puede analizar un *flowspec* para extraer el Tspec, el puede usar la rutina Compare_*Flowspec*s para implícitamente calcular el *Resv*_Te (ver sección 2.2).

- Computar el LUB de los *Flowspec*s

```
LUB_of_Flowspecs( Flowspec_1, Flowspec_2 ) ->
                                    Flowspec_LUB
```

- Computar el GLB de los *Flowspec*s

```
GLB_of_Flowspecs( Flowspec_1, Flowspec_2 ) ->
                                      Flowspec_GLB
```

- Comparar Tspecs

```
Compare_Tspecs( Tspec_1, Tspec_2 ) ->
                                    result code
```

El posible result_code indica: Tspecs son iguales, o Tspecs son desiguales.

- Sumar Tspecs

```
Sum_Tspecs( Tspec_1, Tspec_2 ) ->
                                  Tspec_sum
```

Esta rutina es usada para computar el *Path*_Te (ver sección 2.2).

4. APLICACIONES DE RSVP

RSVP maneja dos clases de servicio: Servicios Garantizados[21] (Guaranteed Service) y Servicio de Carga Controlada[22] (Controlled-Load Service), las aplicaciones varian dependiendo del servicio que el cliente necesite**.**

Contar con RSVP tiene muchos benefios tales como:

- **Beneficios para las empresas**

  Las aplicaciones están consiguiendo ser cada vez más exigentes. Las denominadas críticas requieren cada vez más calidad, confiabilidad, y asegurar la puntualidad en la entrega. Un ejemplo claro son las aplicaciones de voz o vídeo, éstas deben ser manejadas cuidadosamente dentro de una red del IP para preservar su integridad. Además es necesario tener en cuenta que el tráfico no es predecible, ni constante, si no que funciona a ráfagas, produciéndose en ocasiones picos máximos de tráfico que son los causantes, en parte, de la saturación de la red. Ejemplos clarificadores de este tipo de tráfico es el producido por el mundo Web, el correo electrónico y las transferencias de ficheros, que son virtualmente imposibles de predecir. Los servicios de RSVP permiten a los administradores de red:

---

[21] RFC 2212 Specification of Guaranteed Quality of Service; Available from Internet: http://www.faqs.org/rfcs/rfc2212.

[22] RFC 2211 - Specification of the Controlled-Load Network Element Service; Available from Internet: http://www.faqs.org/rfcs/rfc2211.

- ❖ Manejar las aplicaciones sensibles al jitter, como las que manejan audio y vídeo.
- ❖ Manejar el tráfico sensible al retardo, como la voz en tiempo real.
- ❖ El control de pérdidas en los momentos en los que la congestión sea inevitable.

Beneficios para los proveedores de servicio

Claramente, las empresas y las corporaciones se están convirtiendo en negocios con requerimientos de “misióncrítica” sobre la red pública. Están delegando los servicios de sus redes a proveedores de servicio (outsourcing), lo que les permite centrarse más en el negocio interno y así reducir costosos capitales. Esto significa que los proveedores de servicio son quienes podrán ofrecer las garantías de calidad para el tráfico extremo-a-extremo (end-to-end) de la empresa. Las tecnologías de QoS permitirán a los proveedores de servicio ofrecer muchas más prestaciones, como el soporte del tráfico en tiempo real, o como la asignación específica de ancho de banda, que se suele especificar en los acuerdos de nivel de servicio (SLAs)

4.1 SERVICIOS GARANTIZADOS (GUARANTEED SERVICE)

Este servicio proporciona un nivel de ancho de banda y un límite en el retardo, garantizando la no existencia de pérdidas en colas. Está pensado para aplicaciones con requerimientos en tiempo real, tales como ciertas aplicaciones de audio y vídeo. Cada router caracteriza el SG para un flujo específico asignando un ancho de banda y un espacio en buffer.

4.1.1 Video en tiempo Real. El vídeo puede servirse como un fichero, o en tiempo real. A este última forma de enviar el vídeo se le conoce como streaming[23]. Streaming video, o vídeo en tiempo real, es la tecnología que permite la transmisión y recepción de imágenes y sonidos de manera continua a través de una red. A diferencia de otros formatos de audio y vídeo, en los que es necesario esperar que el archivo sea cargado en el equipo para su visualización, esta tecnología permite apreciar el contenido conforme se va teniendo acceso a la información del archivo. El servidor de streaming permite visionar el vídeo de forma continua porque hace uso de un buffer, donde van cargándose algunos segundos de la secuencia antes de que sean mostrados. Entonces cuando se detecta un periodo de congestión de red, se visualizarán los datos que tenemos ya almacenados en el buffer. De esta forma el cliente obtiene los datos tan rápido como el servidor y la red lo permitan. Hay pocos formatos hoy en dia que soporten este tipo de visualización progresiva, probablemente en el futuro próximo, el estandar para el streaming vídeo será en Advanced streaming format (ASF). El streaming puede decirse que funciona de forma inteligente ya que asegura al usuario que recibirá la más alta calidad posible dependiendo de la velocidad de conexión o de los problemas de conexión de la red. Tradicionalmente la congestión de la red forzaba al usuario a detener la visualización del vídeo almacenando en un buffer la información para posteriormente continuar mostrando la secuencia. Con los nuevos formatos de streaming como el MPEG-4[24], el cliente y el servidor pueden degradar la calidad de forma inteligente para asegurar una reproducción continua del vídeo.

---

23 Video sobre redes**;** Available from Internet http://www.monografias.com/trabajos10/vire/vire.shtml.

[24] Transmisión de Audio y Video por Internet; Available from Internet http://www.monografias.com/trabajos10/vire/vire

La trasmision de este tipo de formato requiere un gran ancho de banda el vídeo estándar utiliza 30 imágenes (tramas) por segundo, por tanto, 30 imágenes de 60 KB dan 1,8 millones de bytes por segundo, entonces RSVP utiliza el estilo de WF para reservar este ancho de banda, es recomendable porque la reserva es dividida, y permitirá a los otros usuarios tener una buena velocidad en las aplicaciones que esten usando simultaneamente.

4.1.2 VoIP Voz Sobre IP. El concepto original de VoIP [25] es relativamente simple: se trata de transformar la voz en "paquetes de información" manejables por una red IP (con protocolo Internet, materia que también incluye a las intranets y extranets). Gracias a otros protocolos de comunicación, como RSVP, es posible reservar cierto ancho de banda dentro de la red que garantice la calidad de la comunicación. La arquitectura de red de este servicio es mostrado en la figura 17.

- **Ventajas de la tecnologia de voz sobre IP**

  - Integración sobre su Intranet de la voz como un servicio más de su red, tal como otros servicios informáticos.
  - Las redes IP son la red estándar universal para la Internet, Intranets y extranets.
  - Estándares efectivos (H.323)
  - Interoperabilidad de diversos proveedores
  - Uso de las redes de datos existentes
  - Independencia de tecnologías de transporte (capa 2), asegurando la inversión. Menores costos que tecnologías alternativas (voz sobre TDM, ATM, Frame Relay)

[25] Descripcion tecnica detallada sobre Voz sobre IP (VOIP); Available from Internet: http://www.ilustrados.com/publicaciones/EpyVZEVkEEkDNZkcAb.php

Todo sobre VoIP

Las redes de voz y datos son esencialmente diferentes. Las redes de voz y fax, que emplean conmutación de circuitos, se caracterizan por:

- Para iniciar la conexión es preciso realizar el establecimiento de llamada.
- Se reservan recursos de la red durante todo el tiempo que dura la conexión.
- Se utiliza un ancho de banda fijo (típicamente 64 Kbps por canal de voz) que puede ser consumido o no en función del tráfico.
- Los precios generalmente se basan en el tiempo de uso.
- Los proveedores están sujetos a las normas del sector y regulados y controlados por las autoridades pertinentes (en nuestro caso, el Ministerio de Fomento y la Comisión del Mercado de las Telecomunicaciones).
- El servicio debe ser universal para todo el ámbito estatal.
- Por el contrario, las redes de datos, basadas en la conmutación de paquetes, se identifican por las siguientes características:
- Para asegurar la entrega de los datos se requiere el direccionamiento por paquetes, sin que sea necesario el establecimiento de llamada .
- El consumo de los recursos de red se realiza en función de las necesidades, sin que, por lo general, sean reservados siguiendo un criterio de extremo a extremo.
- Los precios se forman exclusivamente en función de la tensión competitiva de la oferta y la demanda.

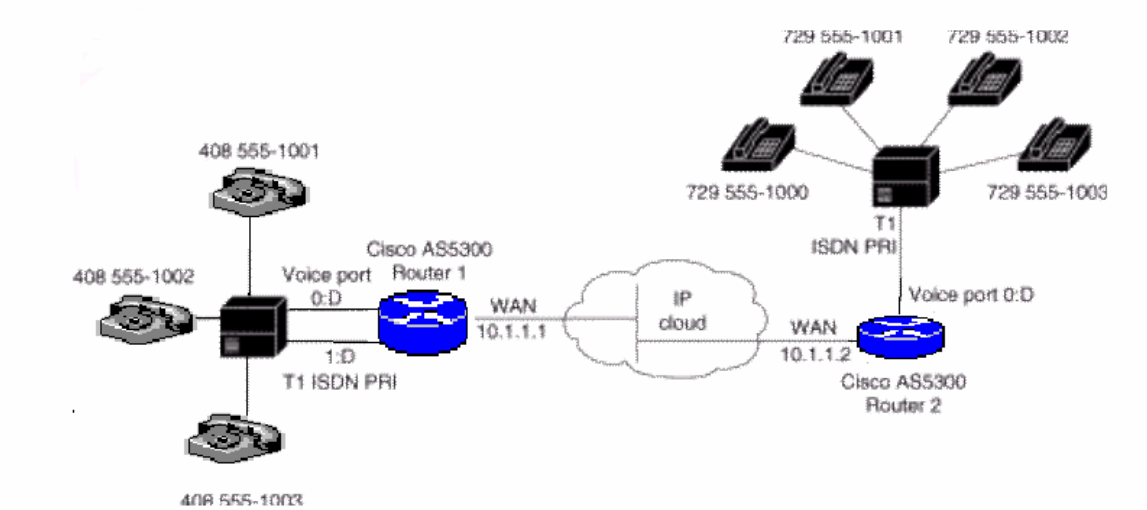
- Los servicios se prestan de acuerdo a los criterios impuestos por la demanda, variando ampliamente en cuanto a cobertura geográfica, velocidad de la tecnología aplicada y condiciones de prestación.
- Implementar una red convergente supone estudiar las diferencias existentes entre las características de las redes de voz y de datos, comprendiendo los problemas técnicos que implican dichas diferencias sin perder de vista en ningún momento la perspectiva del usuario final.

Requerimientos de una red para soportar VoIP

A continuación se mencionan aspectos importantes que se deben tener en la red IP para implantar este servicio en tiempo real.

- Manejar peticiones RSVP que es un protocolo de reservación de recursos.
- El costo de servicio debe estar basado en el enrutamiento para las redes IP.
- Donde se conecta con la red publica conmutada un interruptor de telefonía IP debe soportar el protocolo del Sistema de Señalización 7 (SS7). SS7 se usa eficazmente para fijar llamadas inalámbricas y con línea en la PSTN y para acceder a los servidores de bases de datos de la PSTN. El apoyo de SS7 en interruptores de telefonía IP representa un paso importante en la integración de las PSTN y las redes de datos IP.
- Se debe trabajar con un comprensivo grupo de estándares de telefonía (SS7, Recomendación H.323) para que los ambientes de telefonía IP y PBX/PSTN/ATM vídeo y Gateway telefónica puedan operar en conjunto en todas sus características

Figura 17. Configuración de Voz sobre IP



Fuente: http://www.ilustrados.com/publicaciones/EpyVZEVkEEkDNZkcAb.php

Para prestar este servicio RSVP utiliza el modelo de reserva de FF, ya que es una trasmision unicast y se necesita un ancho de banda fijo por toda la duracion de la llamada.

4.2 SERVICIO DE CARGA CONTROLADA (CONTROLLED-LOAD SERVICE)

A diferencia del SG este servicio no ofrece garantías en la entrega de los paquetes. Así, será adecuado para aquellas aplicaciones que toleren una cierta cantidad de pérdidas y un retardo mantenidos en un nivel razonable. Los routers que implementen este servicio deben verificar que el tráfico recibido siga las especificaciones dadas por el Tspec, y cualquier tráfico que no las cumpla será reenviado por la red, como tráfico best-effort.

Son ejemplo de estas aplicaciones, la transmision de audio y video digitalizados, se necesita un ancho de banda reservado para hacer una transmision eficiente pero pueden tener retardos no perceptibles, ya que en en este servicio RSVP utiliza el estilo de reserva SE, que es explicito dividido, conoce el remitente pero el canal es dividido, por esta razon hay retardos en la red, pero son despreciables; en cuanto a las perdidas no son perceptibles por la vista, ni por el oido humano (Audio y Video).

4.2.1 Transmision de video digital. La transmisión digital y la distribución de información audiovisual[26] permite la comunicación multimedia sobre las redes que soportan la comunicación de datos, brindando la posibilidad de enviar imágenes en movimiento a lugares remotos. Pero no es todo tan bonito a la hora de transmitirlo por red, debido a que nos encontramos con sucesos como lentitud entre la reproducción de imágenes, errores de transmisión, o perdidas de datos... Existen dos formas de transmisión de datos, analógico y digital. Una de las características del vídeo es que está compuesto por señales analógicas, con lo que se pueden dar las dos formas de transmisión. En los últimos años la transmisión de datos se ha volcado hacia el mundo digital ya que supone una serie de ventajas frente a la transmisión analógica. Al verse la información reducida a un flujo de bits, se consigue una mayor protección contra posibles fallos ya que se pueden introducir mecanismos de detección de errores, se elimina el problema de las interferencias, podemos disminuir el efecto del ruido en los canales de comunicación, conseguir codificaciones más óptimas y encriptado, mezclar con otros tipos de información a través de un mismo canal, y poder manipular los datos con ordenadores para comprimirlos, por ejemplo. Además si queremos difundir el vídeo por vías digitales tendremos que digitalizarlo, con lo que debe ser capturado en su formato analógico y almacenado

[26] Video sobre Redes Op Cit.

digitalmente logrando así que sea menos propenso a degradarse durante la transmisión.

Existen dos tipos de redes de comunicación, de conmutación de circuitos y de conmutación de paquetes. En la conmutación de circuitos, donde la comunicación está permanentemente establecida durante toda la sesión, un determinado ancho de banda es asignado para la conexión, y el tiempo de descarga del vídeo puede predecirse, pero tienen la desventaja de que las sesiones son punto a punto y limitan la capacidad de usuarios. En la conmutación de paquetes pueden acomodarse más fácilmente las conferencias multipunto. Aquí el ancho de banda esta compartido pero es variable, lo que supone una importante mejora puesto que, si el bit rate (o número de bits por segundo) es fijo la calidad de la imagen variará dependiendo del contenido de los fotogramas. Debe de cumplirse que el ancho de banda, la resolución, y la compresión de audio sean idénticos para cada cliente que recibe el vídeo, lo que dificulta la configuración del sistema.

4.2.2 Transmision de Audio digital. Mp3[27] o Mpeg 1 Layer 3 (Mpeg Group), este es el sistema de compresión de moda actualmente, es casi el de mejor ratio de compresión (dependiendo de la calidad de grabación) que oscila entorno a 10 :1. Se esta haciendo famoso no solo por su "cantidad" sino por su calidad de compresión, para cualquiera con oídos normales, son casi indistinguibles los ficheros comprimidos de los originales. Se trata de una compresión con perdidas, pero son casi inaudibles. Lo que ha hecho famoso a este sistema es el método de compresión, que se supone será modelo para los siguientes, no se trata de métodos matemáticos para comprimir los paquetes de audio (como en los métodos anteriores), sino que es un método dedicado al oído humano. El método de compresión *a groso modo* es el siguiente : nosotros, normalmente, cuando oímos música, los tonos graves y los tonos agudos (sobre todo los graves)

[27] Transmisión de Audio y Video por Internet; Available from Internet http://www.gui.uva.es/~laertes/nuke/index.php?option=com_content&task=view&id=46&Itemid=41

eliminan los tonos medios, es decir, un sonido muy grave (como un bajo, o la batería en bacalao) destroza o hace que dejemos de oír un tono medio o agudo (la voz del cantante, unas cuerdas, un sonido sintetizado, etc...) así que se logramos cuantificar cuanto dejamos de oír podríamos eliminar ese sonido (o atenuarle) con lo cual ya vamos reduciendo la canción (no hace falta guardar tantos sonidos, tantos matices). Esto es lo que hace un compresor de Mp3. De todas maneras si quereis mas informacion acerca de este formato, teneis algo menos resumido (y con más formatos), aqui con mas formatos de compresion Offline/Online tipo Mp3.

CONCLUSIONES

El protocolo de reserva de recursos (RSVP) se diseñó con un protocolo de señalización IP para el modelo de servicios integrados. RSVP lo puede usar una computadora para solicitar un recurso de QoS específica para un flujo particular, o lo puede usar un dispositivo de encaminamiento para proporcionar una QoS solicitada a lo largo del camino mediante el establecimiento de un estado apropiado.

Ya que el IP tradicional no tiene ningún protocolo de señalización, los diseñadores de RSVP tuvieron la libertad para construir el protocolo desde cero. RSVP tiene las siguientes características:

- Lleva acabo una reserva de recursos para las aplicaciones *unicast* y *multicast*, adaptando dinámicamente a los miembros de grupo que cambian y a las rutas cambiantes.
- Solicita recursos en una dirección desde un emisor a un receptor. La reserva de recursos bidireccional requiere que ambos sistemas finales inicien por separado las reservas.
- Obliga al receptor que inicie y mantenga la reserva de recursos.
- Mantiene un estado flexible en cada dispositivo de encaminamiento intermedio: una reserva de recursos en un dispositivo de encaminamiento se mantiene solamente durante un tiempo limitado, y, por tanto, el emisor debe periódicamente actualizar su reserva.
- No requiere que cada dispositivo de encaminamiento sea compatible RSVP. Los dispositivos de encaminamiento no compatibles RSVP utilizan la técnica del mejor esfuerzo.

- Proporciona diferentes estilos de reserva, de forma que las solicitudes se puedan mezclar en varias formas de acuerdo con las aplicaciones.
- Soporta IPv6 e IPv4.

Para habilitar la reserva de recursos en cada nodo, un proceso RSVP tiene que interactuar con otros módulos. Si el nodo es una computadora, entonces la aplicación que solicita un servicio de entrega QoS tiene que hacer una solicitud a un proceso RSVP, que a su vez pasa mensajes RSVP de un nodo a otro. Cada proceso RSVP pasa el control a sus dos módulos de control locales: el control de Política y el control de admisión. El control de política determina si se le permite a la aplicación hacer la reserva. Las cuestiones relevantes que se tienen que determinar son la autentificación, la contabilidad, y el control de acceso. El control de admisión determina si el nodo tiene suficientes recursos para satisfacer la QoS solicitada. Si ambos test tienen éxito, se establecen los parámetros en el clasificador y en el gestor de salida de paquetes para realizar la reserva. Si algunos de los tests falla en cualquier nodo, se devuelve un error de notificación a la aplicación origen.

En lenguaje RSVP se define una sesión como un flujo de datos identificado por su destino. Siendo más concretos, una sesión RSVP se identifica por su dirección IP destino, el numero de protocolo IP y, opcionalmente, por el puerto destino. La dirección IP destino puede ser *unicast* o *multicast*.

El puerto destino opcionalmente se puede especificar mediante un número de puerto TCP o UDP. Cuando la dirección destino es *multicast*, no es necesario, incluir un puerto de destino.

Una solicitud de reserva de RSVP consta de un *flowspec* y un *filterspec*. Los *flowspec*s se utilizan para establecer parámetros en el gestor de salida de paquetes del nodo. El *filterspec* especifica el conjunto de paquetes que puede utilizar la reserva en una sesión dada.

GLOSARIO

- Adspec

  Un Adspec es un elemento del dato en un mensaje de Path que lleva la información de OPWA.

- Control de Tráfico

  El conjunto entero de la maquinaria en el nodo que suple los requerimientos de QoS. El control de Tráfico incluye las funciones de: el clasificador de paquetes, el organizador de paquetes y el control de admisión.

- C-Type

  La clase de tipo en un objeto.

- DestAddress

  La dirección IP de destino; parte de la identificación de la sesión.

- Downstream

  Hacia el receptor de datos.

- ERROR_SPEC

  Objeto que lleva el reporte de error en un mensaje de PathErr o ResvErr.

- FF style

  Estilo de reserva Filtro Fijo, el cual tiene selección explicita de remitente.

- FilterSpec.

Define el conjunto de paquetes de datos que van a recibir el QoS especificado en un flowspec.

- Flowspec

  Define el QoS que va ser suministrado a un flujo de datos.

- Integridad

  Objeto en un mensaje de control de RSVP que contiene datos criptográficos para autenticar el nodo original y verificar los contenidos de un mensaje de RSVP.

- Merging

  Es el proceso mediante el cual se combinan los Flowspecs para remitirlos a la más cercana interfaz de entrada.

- NHOP

  Un objeto que lleva la información del próximo salto en un mensaje de control de RSVP.

- Nodo

  Un enrutador o un sistema de hosts.

- PHOP

  Un objeto que lleva la información del salto previo en un mensaje de control de RSVP.

- Rspec

  El componente de un flowspec que define la deseada QoS.

- SE style

  Estilo de reserve Shared Explicit, el cual tiene selección de remitente explicita y atributo dividido.

- TSpec

  Un conjunto de parámetros de tráfico que describen un flujo.

- Upstream

  Hacia la fuente de datos.

- WF style

  Estilo de reserva de Wildcard Filter, el cual tiene selección de remitente de Wilcard y atributo dividido.

BIBLIOGRAFIA

KEAGY, Scott. Integración de las redes de Voz y Datos. 1ª edición Cisco Systems 2001. 600p.
El capitulo 6 indica como RSVP juega un papel fundamental en la integración de redes, ya que evita congestiones en el trafico de información.

STALLINGS, William. Comunicaciones y redes de computadores. 6ª edición 2000. 800p.
La sección 16.3 explica brevemente RSVP, como parte de los servicios integrados de internet.

GARCIA, Alberto León. Redes de comunicación, conceptos fundamentales y arquitecturas básicas. 1ª edición. 750p.
La sección 10.5 define RSVP y además explica claramente los estilos de reserva mediante ejemplos de reserva de recursos.

GARCIA, Tomas Jesús. Alta velocidad y Calidad de servicio en redes IP. 1ª edición 2002. 300p
La seccion 4.8 brinda algunas aplicaciones sencillas de RSVP, como la transmision de datos de enorme tamaño como los videos.

Resource Reservation Protocol (on line). Cisco Systems, Feb 20, 2002. Available from Internet:
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rsvp.htm#xtocid1
Esta pagina Web, explica claramente, el concepto de nube de no RSVP y el Túnel de RSVP

RSVP, Resource ReSerVation Protocol (on line). Network Sorcery, Inc, 2004. Available from Internet
http://www.networksorcery.com/enp/protocol/rsvp.html.
Esta pagina Web, explica de forma breve y concisa el protocolo RSVP y la trasmision del descriptor de flujo.

Protocolo RSVP: Evolución y experiencias (on line). Sergi Sanchez, Xavi Masip, Jordi Domingo, Octubre 17 de 2003. Available from Internet: http://www.rediris.es/rediris/boletin/46-47/ponencia11.html
Esta pagina Web, aplica ciertas aplicaciones de RSVP en tiempo real.

RFC 2205 - Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification (on line). Bob Braden, Lixia Zhang, Steve Berson, Shai Herzog, Sugih Jamin, Ann Arbor. September 1997. Available from Internet: http://www.faqs.org/rfcs/rfc2205.html
Esta pagina Web explica la ficha tecnica del protocolo, incuyendo los mecanismos y la especificación fundamental.