Evaluación de Prestaciones de RSVP Extendido para un Escenario con Garantías de Distribución

Marcos Postigo Boix y José L. Melús Moreno
Departamento de Ingeniería Telemática
Universidad Politécnica de Cataluña …
C. Jordi Girona 1-3. Campus Nord. 08034 Barcelona.
marcos.postigo@entel.upc.edu, teljmm@entel.upc.edu

***Resumen*- In some guaranteed data delivery scenarios, servers can quickly and carefully allocate the available bandwidth among the requests of the users to reduce rejections. This is particularly important when reservations are higher than minimum required delivery rate (semi-elastic reservations). As an example, when semi-elastic reservations utilize all the available server bandwidth and a new flow reservation arrives, it is useful to reallocate current reservations to accept the new one. The RSVP protocol is receiver oriented and it is in charge of setting up these reservations. However, in some cases, to reallocate bandwidth in a receiver oriented way could delay the required sender reservation adjustments. This paper presents a new extension of the RSVP signaling messages that allows the server to adjust and modify these reservations. The performance evaluation of the extended and the native RSVP signaling protocols when used to manage the access bandwidth of a semi elastic flows server shows the benefits of the extensions. In particular, the use of resource reservation is reduced because of a more efficient bandwidth usage. In addition, the blocking probability is also reduced because of a more flexible bandwidth reallocation situation. The authors believe that this procedure is an intermediate step to improve the lack of flexibility that RSVP presents.**

***Palabras Clave*- Sender-initiated resource reservation, RSVP, signaling protocol messages, semi-elastic flows.**

I. INTRODUCCIÓN

El IETF escogió como estándar el protocolo "*resource ReSerVation Protocol*" (RSVP) [1][2][3][4]. Sin embargo, la especificación de RSVP tiene problemas significantes en muchos contextos. Otras contribuciones de diferentes autores proponen extensiones a RSVP para solucionar esas carencias. En [5], los autores presentan extensiones experimentales de RSVP para el empleo de clientes remotos que tienen acceso a un demonio RSVP y el empleo de reservas en una pasada. La referencia [6] presenta una versión ligera de RSVP que elimina la capacidad multicast y separa claramente los datos de los mensajes de señalización. La referencia [7] define una nueva extensión de RSVP llamada "*On Board RSVP protocol*" que apoya eficazmente la reserva de recursos extremo a extremo para la movilidad de los routers.

En [8] los autores proponen nuevas extensiones para RSVP que proporcionan servicios en tiempo real en entornos IPv6 móviles y jerárquicos.

En [9] los autores muestran un esquema de reserva de recursos dinámico para reducir la interrupción del servicio de aplicaciones en tiempo real debido a la frecuente movilidad de los terminales en las redes inalámbricas. La referencia [10] presenta una extensión de RSVP para proporcionar un "*Dynamic RSVP*" (DRSVP). Esta aproximación ajusta dinámicamente los recursos reservados en los nodos sin demasiado esfuerzo. Además, en [11] los autores extienden DRSVP para soportar "*soft-handoffs*" en redes IP móviles e inalámbricas. En [12] los autores también proponen un protocolo de reservas de recursos con soporte móvil (Mobile RSVP) que crea un camino móvil que se puede adaptar a la topología de encaminamiento según el movimiento de los nodos.

En [13] se analiza una extensión de RSVP para garantizar la calidad de servicio de las aplicaciones móviles de Internet. Además, se propone un algoritmo para realizar un mejor uso de los recursos de la red.

En [14], los autores proponen nuevas extensiones de RSVP en MPLS. En [15] "*Security Service RSVP*" extiende RSVP para proporcionar un mecanismo que dinámicamente negocia la QoS entre emisores y receptores de las aplicaciones multicast en Internet. La referencia [16] define una extensión de ingeniería de tráfico para RSVP conocida como RSVP-TE para un dominio administrativo y los mensajes de señalización siguen un camino de conmutación de etiquetas MPLS.

En este artículo, nos centramos en la extensión de RSVP que ayuda a un emisor a reorganizar las reservas de forma adecuada cuando su ancho de banda para reservas está totalmente reservado y llegan nuevas reservas al servidor. El principal problema de RSVP en este contexto es que las reservas se inician en el receptor. En algunos escenarios unicast, especialmente cuando el emisor necesita renegociar las reservas de recursos (ReR), es importante ofrecer al servidor la posibilidad de ajustar las reservas directamente, sin la intervención del receptor. Esto puede reducir notablemente el retardo de renegociación. Los flujos semi-elásticos, a diferencia de los flujos inelásticos, no tienen requerimientos estrictos de QoS y pueden permitir la reducción de los recursos reservados e incluso la liberación de dichos recursos sin cambiar sus niveles de QoS esperados. Este es el caso, por ejemplo, de la transmisión de video pregrabado, que puede transmitirse a una tasa mayor que la

tasa de lectura del cliente. Si en algún momento de la transmisión la tasa de envío debe reducirse, el cliente puede continuar leyendo los datos que previamente ha almacenado en su memoria.

Para los flujos semi-elásticos es esencial una gestión correcta de las reservas cuando el ancho de banda disponible cambia dinámicamente. En [17], los autores definen escenarios donde la disponibilidad de ancho de banda es muy variable, como en las redes inalámbricas, y la reserva de ancho de banda se puede beneficiar de las renegociaciones. Consecuentemente, los gestores del ancho de banda pueden elaborar esquemas que maximicen el beneficio para todas las conexiones basándose en la elasticidad de los flujos de datos.

La referencia [18] presenta una solución que usa la planificación y el control de admisión para mejorar las prestaciones para usuarios Premium. El servidor clasifica las peticiones Premium como de alta prioridad y les da un trato preferente. En [19] los autores proponen varios métodos para incrementar la cantidad máxima de clientes que un servidor puede aceptar degradando la calidad de los videos distribuidos. En [20] y [21] los autores proponen acciones similares para distintos escenarios y aplicaciones. En [22] se comparan diferentes mecanismos para distribuir los mensajes de control RSVP y en [23] los autores estudian las interacciones entre los parámetros de temporización de RSVP y las métricas de prestaciones de red como un problema multi-objetivo.

En 2001, el IETF formó un nuevo grupo de trabajo, el *Next Steps In Signaling* (NSIS). Como resultado se desarrolló una nueva arquitectura extensible de señalización IP de dos capas [24]. En [25], los autores discuten protocolos aplicaciones de los protocolos, en particular, se resalta el protocolo de señalización de calidad de servicio (QoS). Las referencias [26][27][28] del NSIS *Working Group* definen los protocolos de señalización iniciados por el emisor como parte del marco de trabajo de NSIS. En [29] presentamos un método para distribuir flujos de información prealmacenada a varios clientes. Usando ReR (Reservas de Recursos), el emisor envía datos en unicast de forma más rápida de lo que los clientes requieren. Cuando la memoria de los clientes alcanza un umbral, no es necesario mantener el modo ReR y cambia a Best-Effort (BE) reduciendo el coste de transmisión e incrementando la eficiencia de las reservas de recursos. De forma similar, el cliente solicita el modo de transmisión ReR, si los datos alcanzan un umbral mínimo. En este escenario multicliente-servidor es esencial controlar el ancho de banda de acceso reservado, ya que un nueva reserva puede ser rechazada si todos los recursos disponibles están reservados.

En este artículo, describiremos la inclusión de nuevos mensajes RSVP y su procesamiento para prevenir rechazos de reservas de recursos en el servidor. Estos nuevos mensajes permiten la modificación de las reservas previas antes de que se acepte una nueva reserva. Estas modificaciones de RSVP pueden considerarse como un paso intermedio para mejorar la falta de flexibilidad de RSVP. Este es un paso previo para obtener la flexibilidad que NSIS ofrecerá cuando esté ampliamente implementado [27][28].

El resto del artículo se organiza de la siguiente forma: la sección 2 define los nuevos mensajes RSVP; la sección 3 describe como el demonio RSVP procesa los mensajes y muestra su uso en escenario homogéneo con varios clientes y un servidor de flujos semi-elásticos; la sección 4 compara la evaluación del protocolo RSVP nativo con el RSVP extendido; finalmente, la sección 5 concluye el artículo.

II. Nuevas Extensiones RSVP y su Implementación

Cuando se usa el protocolo RSVP, los receptores se encargan de establecer y mantener las reservas. De esta forma, si el emisor (servidor) necesita modificar o liberar la reserva, no puede hacerlo directamente. Así, el emisor debe indicar cualquier cambio al receptor que finalmente será quien cambie la reserva.

En un escenario multicliente-servidor que utilice conexiones unicast con cada cliente, cuando los clientes solicitan simultáneamente flujos semi-elásticos, el servidor debe controlar y gestionar adecuadamente el ancho de banda de acceso utilizado por las reservas, para maximizar el número de flujos admitidos. Dado que el servidor es el único que conoce exactamente cuántas reservas están accediendo al servidor, debe ser el responsable de gestionar esos flujos simultáneos. Para garantizar esta funcionalidad, proponemos extender RSVP para permitir al servidor el modificar directamente las reservas. En esta sección, primero definiremos los mensajes RSVP nuevos necesarios y después describiremos la implementación de estas extensiones.

A. Nuevas Extensiones RSVP

Para asegurar al servidor el redistribuir el ancho de banda entre las reservas actuales, definimos tres nuevos mensajes: RESV_S, RESV_S_TEAR y RESV_S_ERROR. Viajan respectivamente en la dirección opuesta a los mensajes RESV, RESV_TEAR y RESV_ERR en RSVP. El mensaje RESV_S se encarga de ajustar las reservas y actualizar los valores de reserva de los mensajes RESV de refresco (Fig. 1). RESV_S_TEAR cancela las reservas y borra el estado específico de reserva. RESV_S_ERROR señaliza estados de error cuando se intenta cambiar una reserva con mensajes RESV_S. Avisa de un error cuando se procesa un mensaje RESV_S y se encamina salto-a-salto usando el estado de path, de forma parecida a los mensajes PATH_ERR.

Para informar de la llegada de los mensajes RESV_S y RESV_S_TEAR al receptor, se define la llamada RESV_S_EVENT. Esta se usa en el receptor para notificar a la aplicación local que el servidor ha cambiado la reserva. Otra llamada (RESV_CONTROL_EVENT) se utiliza para prevenir al emisor (servidor) de un rechazo de reserva, permitiendo a la aplicación la redistribución de ancho de banda antes de comenzar el procesado del mensaje RESV entrante (Fig. 1). Así, cuando no hay suficiente ancho de banda para aceptar una nueva petición el servidor podrá reducir o modificar, antes de aceptar la reserva, otras reservas semi-elásticas, para liberar suficiente ancho de banda para la nueva reserva. En cualquier caso, los niveles de QoS de todas peticiones se mantienen garantizados. Después de esto, se realiza el procesado normal del mensaje RESV.

Estas extensiones no cambian la operación normal de RSVP. Por tanto, los estados de reserva en los routers intermedios,

debido a sesiones RSVP se refrescan de igual forma que en RSVP.

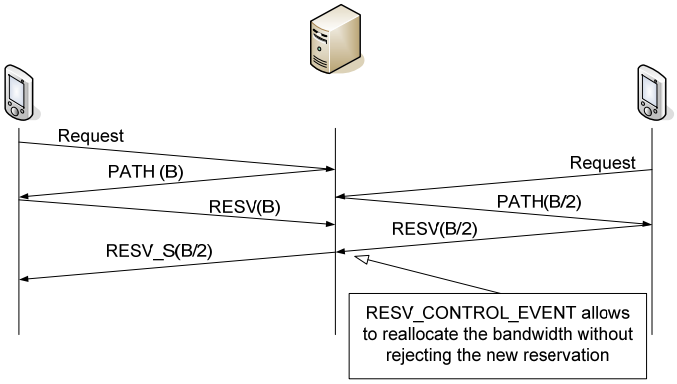

Fig. 1: Extension of RSVP with new messages (RESV_S message).

B. Implementación RSVP

La implementación de las extensiones RSVP en este artículo se realizan usando la versión 4.2a4 de la distribución de RSVP [30] del *Information Science Institute* (ISI). Se basa en este demonio RSVP para permitir la funcionalidad de estos nuevos mensajes y para asegurar la compatibilidad con versiones anteriores del ISI. Sin embargo, añade nuevas funcionalidades para permitir al servidor el modificar las reservas directamente. La distribución del ISI está diseñada para funcionar bajo Linux, FreeBSD, SunOS, Solaris y IRIX y está aceptada como una de las implementaciones de referencia. Además, el interfaz ente el demonio RSVP y las aplicaciones locales se realiza mediante la RAPI (*RSVP Application Programming Interface*) [31]. Brevemente, las funciones que tienen lugar en el procesado de los mensajes son:

1. *Procesado de mensajes RESV_S*

El mensaje RESV_S se procesa cuando el demonio RSVP llama a la función `accept_resv_s()` y comprueba si existe un estado de PATH previo antes de que se cambie la reserva. Si se confirma, el estado de reserva se establece llamando a la función `flow_reservation_s(),` y los parámetros del RSB (Reservation State Blocks) se comparan y actualizan con los del mensaje RESV_S. Finalmente, la función `Complete_ModFlowspec_s()` se encarga de aplicar la llamada RESV_S_EVENT, avisando a la aplicación local de que la reserva ha cambiado. Si no hay una reserva, se envía un mensaje RESV_S_ERROR y finaliza el proceso.

2. *Procesado del mensaje RESV_S_TEAR*

El mensaje RESV_S_TEAR se procesa cuando el demonio llama a la función `accept_resv_s_tear()` para confirmar que la sesión del paquete recibido está presente y para poder continuar con el proceso. En caso contrario finaliza.
La función `kill_RSB_s()` se encarga de eliminar la reserva que corresponde al RSB. Finalmente, llama a la función `LL_DelFlow()` que actualiza la función `Complete_DelFlow_s()` y realiza una llamada RESV_S_EVENT cuando se alcanza el receptor.

3. *Procesado de mensajes RESV_S_ERR*

El procesado de este mensaje es similar al del mensaje RESV_ERR, pero en este caso, es el emisor (servidor) en vez de receptor el que recibe la llamada desde el demonio.

4. *Procesado de mensajes RESV*

Extendemos el procesado de los mensajes RESV para permitir a las aplicaciones finales la gestión de su ancho de banda. Cuando no hay suficiente ancho de banda para usar el modo ReR en el enlace de acceso del servidor, esta reserva se rechaza y el emisor no puede realizar otras acciones para evitarlo. Mediante esta nueva extensión de RSVP el emisor detecta la llegada de nuevas peticiones de reserva antes de que se establezca en el enlace de acceso del emisor usando la llamada RAPI_RESV_CONTROL_EVENT. Si el emisor detecta que no hay recursos adecuados para acceder esta nueva petición, puede ajustar las otras reservas cambiando sus tasas o cancelando algunas de ellas si esto es aceptable. En cualquier caso, el emisor intenta evitar el mayor número de rechazos posible. La Fig. 2 describe el procedimiento que se sigue en el servidor cuando llega un nuevo mensaje RESV.

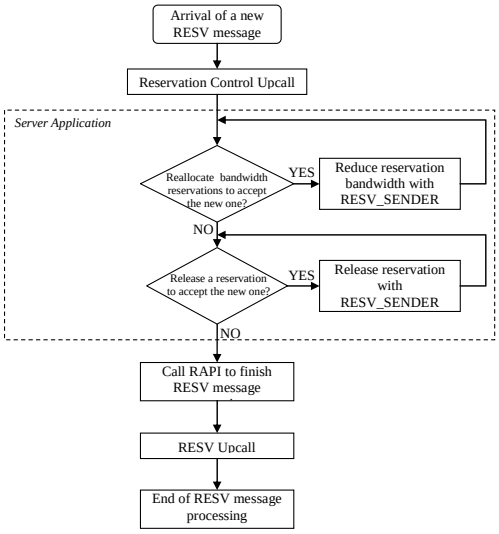

Fig. 2: RSVP extension. The arrival of new RESV message to the server

III. Caso de Aplicación: Flujos Semi-Elásticos

Esta sección presenta un caso de aplicación en el que se usan las extensiones RSVP. Hay otras muchas aplicaciones como las multimedia o las que necesitan transmisión en tiempo real que necesitan algún mecanismo para garantizar un determinado nivel de QoS para funcionar de forma adecuada. Los flujos semi-elásticos pueden usar las reservas de recursos para reducir el tiempo de espera o para garantizar la disponibilidad de datos en el receptor. Debido a la elasticidad de las reservas presentes, estas pueden reducirse o liberarse, si es necesario, sin afectar a la calidad de su transmisión, si el receptor dispone de suficientes datos disponibles.

En nuestra propuesta, el servidor puede reservar recursos para enviar información a una tasa α_{ReR} que será mayor que la que necesita el cliente para leer, para así tener por adelantado más información de la que necesita [29] para así usar el servidor normal BE. Cuando esto ocurre, el cliente deja de recibir más información usando la reserva de recursos. Libera los recursos reservados que pueden usar otros flujos semi-elásticos que acceden al servidor, y continúan recibiendo datos en el modo BE a una tasa α_{BE} . Después, se explicará el algoritmo que gestiona el acceso múltiple de varios clientes al servidor en este caso.

A. Gestión del ancho de banda de acceso

Suponemos que el ancho de banda del servidor B_s b/s, está dividido en tres partes: para el servicio BE (B_{BE} b/s), para los servicios de reserva (B_{ReR} b/s), y para la señalización RSVP (B_{sig}). Además, la transmisión BE se asume que se carga con una tarifa plana, hacienda la transmisión BE más barata que la ReR. Los clientes son homogéneos, tienen la misma tasa de lectura (α_r b/s) y la tasa de reserva α_{Res} es siempre mayor que α_r .

Consideramos tres situaciones diferentes en la gestión del ancho de banda disponible para ReR: iniciar una nueva sesión, pasar a modo ReR y liberar una sesión.

B. Iniciar una sesión

La tasa máxima en el modo ReR depende del número de clientes conectados (*Num_clients)*. Si este número es menor que el número de reservas semi-elásticas simultáneas (*Max_resv)*, α_{Res} decrece con cada nueva conexión ya que se reparte entre todas las conexiones. Cuando *Max_resv* es 1, sólo se permite una conexión en modo ReR. Esto hace que los datos lleguen más rápido al cliente y por tanto, su memoria alcanza el umbral *Max* más rápido y se usará el modo BE durante más tiempo. Cuanto mayor es *Max_resv*, el ancho de banda asignado a las reservas simultáneas decrece, haciendo que los datos recibidos durante BE decrezca. Cuando llegan nuevos clientes, cada uno obtiene la misma parte de B_{ReR}. Si el número de clientes conectados excede *Max_resv*, α_{Res} no varía.

C. Paso al modo ReR

El servidor controla el ancho de banda asignado a cada reserva. Cuando el cliente solicita el paso al modo ReR, el servidor recibe una llamada RAPI_RESV_CONTROL_ EVENT. El servidor sabe que una reserva pendiente espera el acceso y por tanto, debe gestionar el resto de reservas para aceptar la nueva petición ReR. Si el servidor da servicio al número máximo de reservas simultaneas admitidas, envía un mensaje RESV_S_TEAR al cliente más antiguo en el modo ReR para cambiarle a BE y la reserva pendiente se completa. Alternativamente, si el máximo número de reservas simultáneas no se ha alcanzado, el servidor ajusta la nueva tasa de reserva para cada cliente usando el mensaje RESV_S. La Fig. 3 muestra este proceso.

D. Liberar una sesión

Cuando el cliente quiere finalizar su sesión, el servidor actualiza las reservas para cada cliente considerando el nuevo número de clientes conectados. Si los clientes restantes exceden *Max_resv*, la tasa máxima para cada cliente no varía. Por el contrario, la nueva tasa para cada cliente se calcula y se envía un mensaje PATH a cada uno incluyendo este nuevo valor.

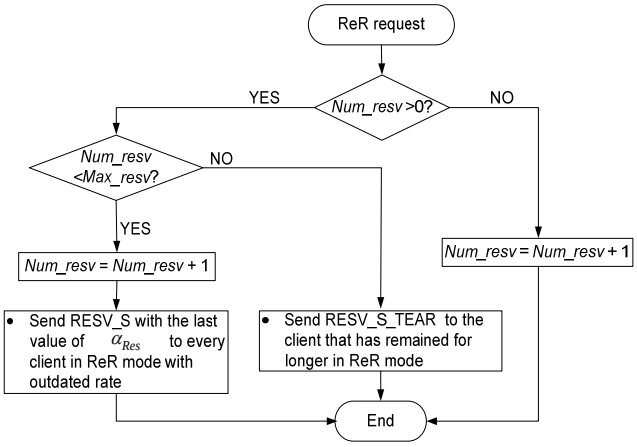

Fig. 3: Management of the server access bandwidth when a client asks to pass to ReR mode.

IV. Resultados de Evaluación

Hemos realizado de forma exhaustiva simulaciones para evaluar el caso de aplicación en Network Simulator 2 (ns-2). Se ha utilizado una implementación de RSVP para ns-2, RSVP/ns [32]. Para este trabajo, se mejora RSVP/ns con los nuevos mensajes y el algoritmo que permite al servidor la gestión de reservas de recursos en el caso de los flujos semi-elásticos. Se utilizan los valores recomendados por defecto para la temporización de mensajes RSVP [1].

La arquitectura multicliente-servidor del la Fig. 4 muestra la topología implementada. Para ser preciso, los clientes leen datos de sus memorias a la misma tasa (600 kb/s), y usan una memoria de tamaño 1 MB. Cada cliente abre sesiones desincronizadas y solicita un archivo de 300 MB. El tráfico de fondo se considera de tasa constante, incluido en la distribución del simulador NS. En media, para el modo BE, los clientes reciben a una tasa de 200 kb/s. El ancho de banda de acceso del servidor es de 10 Mb/s y el ancho de banda asignado para el modo ReR es de 8 Mb/s. Como la tasa de lectura del cliente es de 600 kb/s y el ancho de banda asignado para el modo ReR es 8 Mb/s, el servidor puede aceptar un número máximo de clientes de 13.

Las simulaciones se ejecutan independientemente 10 veces para evaluar los intervalos de confianza del 95%, aunque no se muestran en los resultados debido a que son suficientemente estrechos. Se realizan cerca de 10^5 solicitudes de sesión por cada simulación.

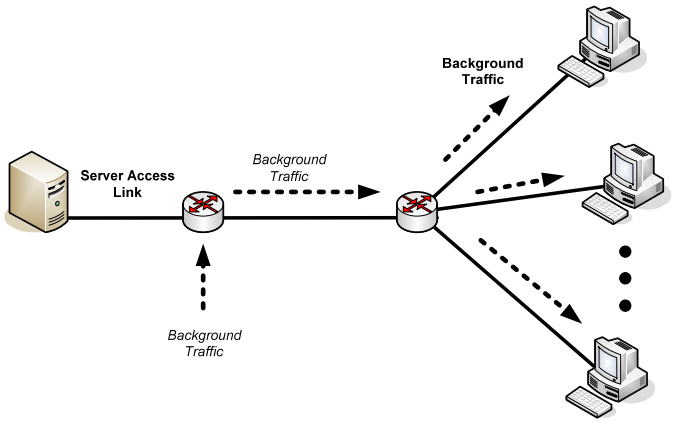

Fig. 4: Simulation and prototype scenario

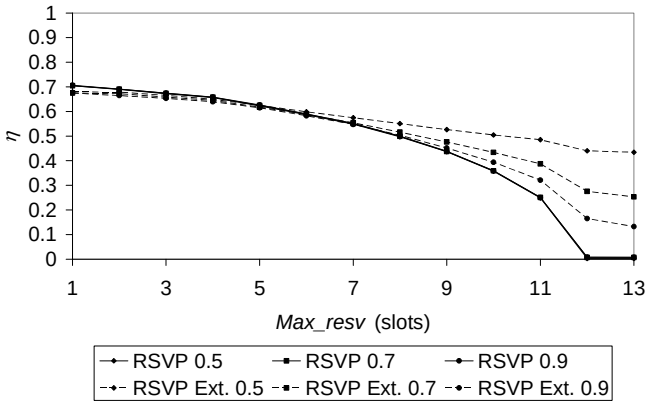

Fig. 5: Efficiency vs. the number of allowed simultaneous ReR connections for different values of offered traffic.

Se utiliza una implementación Tahoe de TCP para transferir los datos, y como los flujos usan TCP, tanto los clientes como el servidor heredan de la clase de ns-2 *TcpApp*, que proporciona el interfaz de aplicación con la capa de transporte TCP. En este caso, necesitamos adaptar el interfaz de *TcpApp* para permitir al servidor el control de la tasa de envío en modo ReR. En el Apéndice C de la referencia [29] se muestra la jerarquía de todas las clases C++ desarrolladas y las tareas necesarias. Los entornos simulados usan scripts OTcl que definen los parámetros adecuados como el número de clientes, los tamaños de los archivos y las tasas de lectura de los clientes.

Hay varias medidas que hemos usado para comparar nuestra versión de RSVP extendida con RSVP nativo: eficiencia, número medio de conexiones activas, probabilidad de bloqueo y señalización extra.

A. Eficiencia

La eficiencia (η) representa el ratio entre los datos enviados usando BE (D_{BE}) y los datos totales ($D_{Total} = D_{ReR} + D_{BE}$). Una eficiencia de 1 significa que todos los datos se han enviado usando BE y una eficiencia de 0 que todos los datos se han enviado usando ReR.

$$\eta = \frac{D_{BE}}{D_{Total}} \tag{1}$$

La Fig. 5 muestra la eficiencia para diferentes valores del número de conexiones ReR simultáneas permitidas (*Max_resv*), cuando la carga ofrecida es de 0.5, 0.7 y 0.9. El comportamiento cuando *Max_resv* es 12 ó 13 muestra el beneficio de redistribuir el ancho de banda entre las conexiones restantes en el servidor. Cuanto menor es el tráfico ofrecido de sesiones, menos conexiones comparten el ancho de banda y consecuentemente, reciben mayores reservas de ancho de banda haciendo posible que aumente la eficiencia. Por el contrario, la eficiencia de RSVP se mantiene igual ya que el ancho de banda reservado por conexión es siempre el mismo y depende de *Max_resv*. La diferencia entre la eficiencia para RSVP extendido y nativo alcanza un 43% para un valor de *Max_resv* de 13.

B. Tasa de sesiones cursadas

Definimos la tasa de sesiones cursadas (*Carried Sessions Ratio*, CSR) como la relación entre las sesiones cursadas por el servidor (S_c) y la cantidad total de sesiones solicitadas por los clientes (S_{Total}).

$$CSR = \frac{S_c}{S_{Total}} \tag{2}$$

Esta tasa puede expresarse en dos partes: la primera puede asociarse al ratio BE cursado (*CbeR*) y la segunda al ratio ReR cursado (*CrerR*). La primera representa el ratio de tiempo que han estado en modo BE las sesiones cursadas y la segunda el de tiempo que han estado en modo ReR.

$$CSR = CbeR + CrerR \tag{3}$$

$$CbeR = CSR \cdot \eta \tag{4}$$

$$CrerR = CSR \cdot (1 - \eta) \tag{5}$$

La Fig. 6 muestra el *CSR* para diferentes valores de *Max_resv*, cuando la carga ofrecida vale 0.5, 0.7 y 0.9. El *CSR* con RSVP extendido se mantiene casi constante. Esto se debe a que las extensiones permiten al servidor aceptar el máximo número de clientes que puede servir. Este número sólo depende de la tasa de lectura del cliente y el ancho de banda de acceso. El tráfico ofrecido de sesiones también afecta al *CSR*. Cuanto mayor es, el *CSR* decrece. Esto se debe a que la cantidad de sesiones bloqueadas también se incrementa. El comportamiento de RSVP mejora cuando aumenta *Max_resv* ya que las sesiones aceptadas se incrementan.

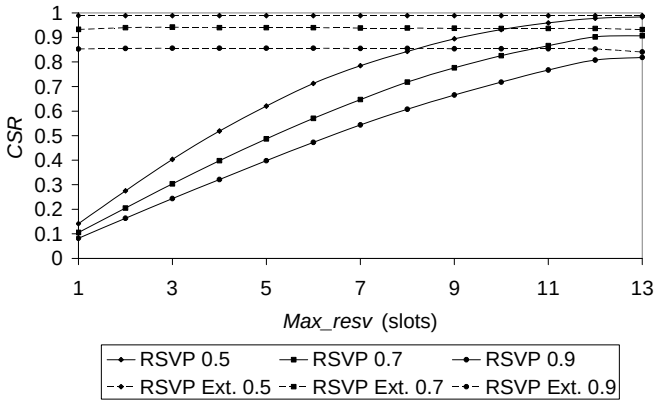

Fig. 6: Carried sessions ratio vs. the number of allowed simultaneous ReR connections for different values of offered traffic.

Las figuras 7 y 8 muestran el *CbeR* y el *CrerR* para diferentes números de conexiones simultáneas aceptadas (*Max_resv*). En cuanto al comportamiento de las extensiones, las conexiones siempre usan más el modo BE que el ReR. También, el modo BE su usa menos a medida que *Max_resv* se incrementa como se explicó anteriormente para la eficiencia. Para RSVP nativo, podemos observar que el modo BE se usa menos que el ReR cuando *Max_resv* toma valores altos ya que con RSVP el servidor no puede redistribuir el ancho de banda.

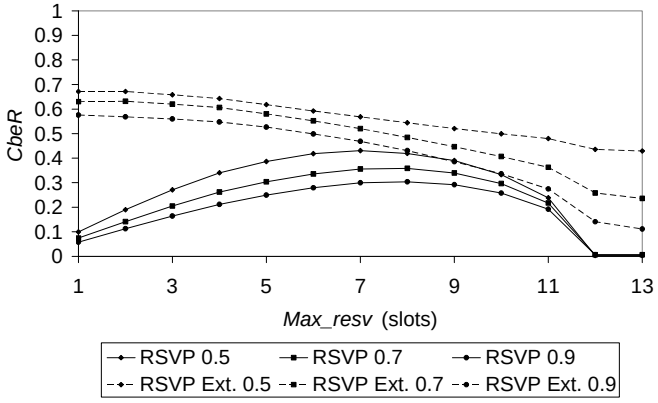

Fig. 7: Carried BE ratio vs. the number of allowed simultaneous ReR connections for different values of offered traffic.

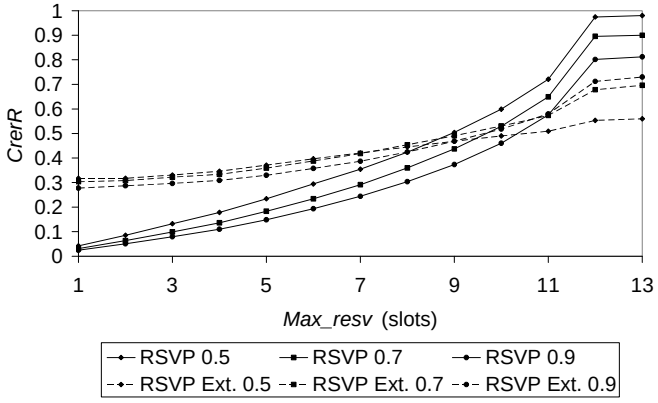

Fig. 8: Carried ReR ratio vs. the number of allowed simultaneous ReR connections for different values of offered traffic.

Comparando el comportamiento de RSVP extendido con el nativo, es destacable que la diferencia máxima del *CSR* se da para *Max_resv* igual a 1. En este caso, las sesiones que cursa un servidor usando RSVP nativo son un 80-90 % menos que con las extensiones (Fig. 6). Por otro lado, si *Max_resv* es igual a 13, el *CSR* para las dos versiones de RSVP dan valores similares, pero mirando a la Fig. 8, podemos observar que el RSVP nativo utiliza ReR durante más tiempo que el extendido.

C. Probabilidad de bloqueo

La probabilidad de bloqueo (P_b) se define como el ratio entre las sesiones cursadas y el total de sesiones recibidas.

$$P_b = 1 - \frac{Carried\_sessions}{Received\_sessions} \quad (6)$$

La Fig. 9 muestra la probabilidad de bloqueo para diferentes tráficos ofrecidos de sesiones, cuando el valor de *Max_resv* es 3, 7 y 11. Como las peticiones de nueva sesión se incrementan, la probabilidad de bloqueo también lo hace. Para RSVP la probabilidad de bloqueo es siempre mayor que para RSVP extendido ya que el *CSR* es menor. Los valores de probabilidad de bloqueo dependen del máximo número de clientes aceptados en el sistema, que se mantiene constante para un valor particular de *Max_resv* con el RSVP extendido. La Fig. 10 representa la probabilidad de bloqueo en función de la eficiencia para diferentes valores de tráfico ofrecido. Los resultados muestran como el RSVP extendió obtiene siempre mejor probabilidad de bloqueo y eficiencia en comparación con el RSVP nativo.

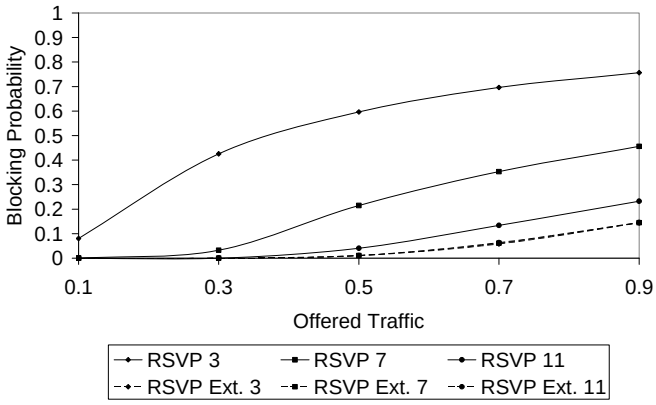

Fig. 9: Blocking probability vs. offered traffic to the server for different numbers of allowed simultaneous ReR connections.

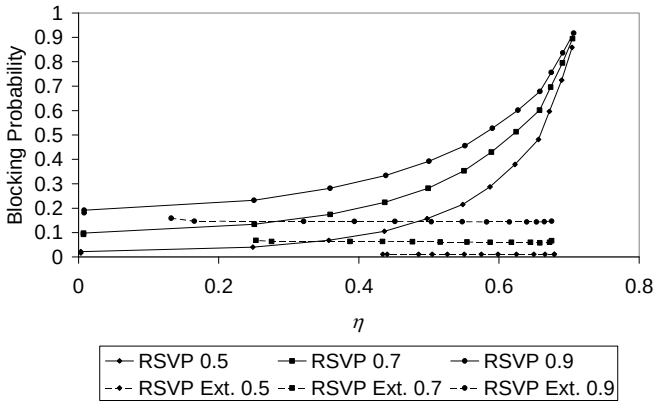

Fig. 10: Blocking probability vs. efficiency for different values of offered traffic.

D. Señalización extra

Medimos las señalización extra necesaria para gestionar el ancho de banda en comparación el RSVP nativo. Son necesarios nuevos paquetes extra cuando se reorganiza el ancho de banda entre sesiones, cuando se acepta una sesión o cuando una sesión finaliza. Además, se necesitan paquetes extra para reorganizar el ancho de banda reservado entre conexiones para no exceder un número de conexiones simultáneas mayor que *Max_resv*.

La Fig. 11 muestra los mensajes de señalización extras por sesión para diferentes valores de *Max_resv*, cuando la carga ofrecida es de 0.5, 0.7 y 0.9. Como se puede ver, para un valor de *Max_resv* igual a 2 la señalización es mínima (aproximadamente 1 mensaje extra por cada 1000 sesiones), lo que clarifica el impacto real de las extensiones. Para *Max_resv* igual a 13, la señalización decrece con el tráfico ofrecido ya que en este caso la señalización extra sólo se utiliza para reorganizar el ancho de banda, cuando el número de sesiones activas varía. La señalización extra se incrementa con *Max_resv* ya que hay un mayor número de conexiones a gestionar.

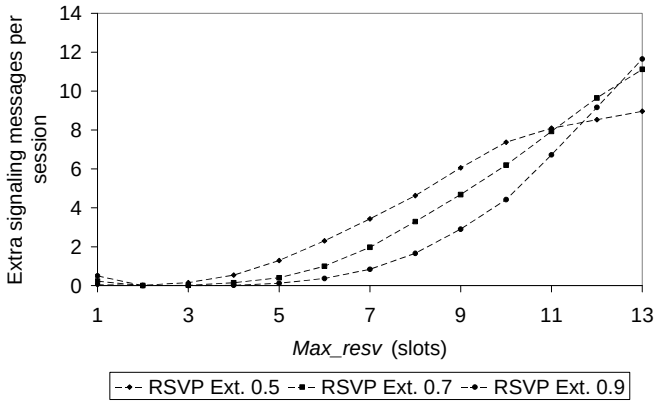

Fig. 11: Extra signaling messages per session vs. the number of allowed simultaneous ReR connections for different values of offered traffic

Asimismo, se ha implementado un prototipo, Fig. 4, para confirmar las simulaciones. Utiliza Linux, lenguaje C++ y threads POSIX. Las aplicaciones utilizan una RAPI (RSVP-API) que incluye las extensiones que interactúan con el demonio extendido RSVP. En este escenario, el servidor se encarga de la gestión del ancho de banda usando las nuevas extensiones de RSVP. Además, puede ajustar o cancelar reservas para ofrecer el servicio al mayor número de flujos posible. Otros elementos son el cliente que se ejecuta en diferentes máquinas usando diferentes terminales y los routers. Se genera tráfico de fondo CBR usando el generador de tráfico MGEN [33]. Los resultados obtenidos son similares a los de simulación lo que confirma la aplicabilidad de la propuesta como un paso intermedio para mejorar la flexibilidad de RSVP.

V. CONCLUSIONES

En este trabajo, se han propuesto nuevas extensiones para el protocolo de señalización RSVP que alertan al servidor de cambios en el estado de las reservas. Esto es útil para ajustar otras reservas, antes de procesar los mensajes RESV que llegan. Se ha analizado el correcto funcionamiento de las extensiones para el caso de flujos semi-elásticos homogéneos que acceden a un servidor. En ese sentido, cuando llegan nuevas reservas al servidor y el ancho de banda disponible está totalmente ocupado, el servidor ajusta las reservas existentes antes de procesar la nueva reserva para evitar su rechazo sin degradar el nivel de QoS necesario por la aplicación, debido a las características de los flujos semi-elásticos. Los valores de las métricas estudiadas confirman una mejora en la eficiencia, el número de sesiones simultaneas cursadas y la probabilidad de bloqueo, con un pequeño incremento en el número de mensajes extra de señalización.

La ventaja de esta propuesta es la habilidad de aceptar más sesiones semi-elásticas que el RSVP nativo, ya que las reservas se tratan de forma más flexible que con RSVP nativo. Esto se confirma con la mejora en el número de sesiones simultáneas cursadas y una menor probabilidad de bloqueo.

AGRADECIMIENTOS

Este trabajo ha sido subvencionado por los proyectos TSI2005-07293-C02-01, TSI2005-06413, y el grupo consolidado de investigación 2005SGR 00563 financiado por la Generalitat de Catalunya.

REFERENCIAS

[1] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, “Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification”. RFC 2205, Sept 1997.

[2] R. Braden, L. Zhang, “Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules”. RFC 2209, Sept. 1997.

[3] L. Berger, D. Gan, G. Swalow, P. Pan, F. Tommasi, and S. Molendini, “RSVP refresh overhead reduction extensions”, RFC 2961, Apr.2001.

[4] B. Lindell, R. Braden, and L. Zhang, “Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules [REVISION]”. Internet Draft, February 1999.

[5] M. Karsten, “Experimental Extensions to RSVP – Remote Client and One-Pass Signalling”, in Proc. of the 9th International Workshop on Quality of Service (IWQoS’01), Karlsruhe, Germany 2001, pp. 269-274.

[6] X. Fu and C. Kappler, “Towards RSVP Lite: Light-weight RSVP for Generic Signaling”, Proc. of the 17th International Conference on Advanced Information Networking and Applications (AINA’03), Xi'an, China, March 2003, pp. 619-622.

[7] M. A. Malik, S.S. Kanhere, M. Hassan and B. Benatallah, “ On-Board RSVP: An Extension of RSVP to support Real-Time Services in On-Board IP Networks”, IWDC 2004, LCNS 3326, pp. 264-275.

[8] N-F. Huang and W-En Chien, “RSVP Extensions for Real Time Services in Hierarchical Mobile IPv6”, Mobile Networks and Applications 8, Kluwer Academic, pp. 625-634.

[9] B. Moon and H. Aghvami, “RSVP Extensions for Real-Time Services in Wireless Mobile Networks”, IEEE Communications Magazine Dec.2001, pp. 52-59.

[10] G-S Kuo and Po-Ch. Ko, “Dynamic RSVP protocol”, IEEE Communications Magazine, May 2003, pp.130-135.

[11] Q. Huang and G-S. Kuo “Dynamic RSVP Extension for Wireless Mobile IP Networks”, IEEE 60th Conference on Vehicular technology VTC2004-fall.2004, pp.2683-2687.

[12] S. Yasukawa, J. Nishikido and H. Komura, “Scalable QoS Support mobile Resource Reservation Protocol for Real –time Wireless Internet Traffic”, IEEE GLOBECOM´02, pp. 1475-1479.

[13] A. Mahmoodian and G. Haring, “Mobile RSVP with Dynamic Resource Sharing”, IEEE Wireless Communication and Networking WCNC 2000, pp.896-901.

[14] J. M. Chung, “Analysis of MPLS Traffic Engineering”, Procc. of 43rd IEEE Midwest Symp. On Circuits and Systems 2000, pp.550-553.

[15] Z.H. Xia and Y. Hu, “Extending RSVP for Quality of Security Service”, IEEE Internet Computing, M-April 2006, pp. 51-57.
[16] D. Awduche et al., “RSVP-TE: extensions to RSVP for LSP tunnels”, RFC 3209, Dec 2001.
[17] C. Curescu and S. Nadjm-Tehrani, “Time-Aware Utility-Based Resource Allocation in Wireless Networks”, IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 7, July 2005.
[18] N. Bhatti and R. Friedrich, “Web Server Support for Tiered Services”, IEEE Network, September/October 1999, pp. 64-71.
[19] T. Abdelzaher and N. Bhatti, “Web Server QoS Management by Adaptive Content Delivery”, in Proc. of the 7th International Workshop on QoS, May 1999, London, England, pp. 216-225.
[20] S. A. Azad, M. Murshed and L. S. Dooley, “Bandwidth Borrowing Schemes for Instantaneous Video-on-Demand Systems”, in Proc. of the 2004 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, June 2004, pp. 2011-2014.
[21] G. Su and M. Wu, “Efficient Bandwidth Resource Allocation for Low-Delay Multiuser Video Streaming”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15, No. 9, September 2005.
[22] O. Komolafe and J. Sventek, “An Evaluation of RSVP Control Messsage Delivery Mechanisms”, in Proceedings of IEEE Workshop on High Performance Switching and Routing 2004.
[23] O. Komolafe and J. Sventek, “An Evaluation of RSVP Control Messsage Delivery Mechanisms”, in Proceedings of INFOCOM 2005.
[24] R. Hancock, G. Karagiannis, J. Loughney and S. Van den Bosch, “Next Steps in Signaling (NSIS): Framework” RFC 4080, June 2005.
[25] J. Manner, G. Karagiannis, A. MacDonald and S. Van den Bosch, “NLSP for Quality-of-Service signaling”, Internet draft (draft-ietf-nsis-nltp-07) work in progress, July 2005.
[26] X. Fu, et all., NSIS: A new Extensible IP Signaling Protocol Suite, IEEE Communications Magazine 43 (10), October 2005, pp. 133-141.
[27] Next Steps in Signaling (nsis) Charter, http://www.ietf.org/html.charters/nsis-charter.html.
[28] NSIS Implementation, University of Goettingen, http://user.informatik.uni-goettingen.de/~nsis/news.html
[29] M. Postigo-Boix, J. Garcia-Haro and J. L. Melús-Moreno, “A cost efficient method for streaming stored content in a guaranteed QoS Internet”, Computer Networks nº 51, January 2007, pp 309-335.
[30] ISI RSVP Project. http://www.isi.edu/div7/rsvp.
[31] R. Braden and D. Hoffman, “RAPI -- An RSVP Application Programming Interface”. Internet Draft, August 1998.
[32] Contributed Code – Nsnam, http://nsnam.isi.edu/nsnam/index.php/Contributed_Code.
[33] Naval Research Laboratory, MGEN - The Multi-Generator Toolset, http://mgen.pf.itd.nrl.navy.mil/