Transmisión video usando RTP y RTSP

Proyecto para Redes de Computadores II

8 de enero de 2016

Autor: Cristibal Badilla – Marco Benzi - Pedro Zepeda Pozo

1. Introducción

Desde que la informática se introdujo en el hogar y aparecieron las primeras GUIs (Interfaces Gráficas de Usuario) los contenidos multimedia han estado presentes tanto en equipos domésticos como en estaciones de trabajo. Y desde hace unos años, también en teléfonos móviles, reproductores portátiles o videoconsolas. ¿Como se representan audio y vídeo en el ordenador? Los archivos de audio se almacenan comprimidos por medio de algoritmos basados en la técnica PNS (norma de percepción del ruido), es decir, aprovechan ciertas frecuencias que el ser humano no reconoce y otras que reconoce mejor.

Hasta aquí se ha hablado de ficheros multimedia para ser almacenados en diferentes soportes; pero desde la creación de Internet, y con el aumento en el número de nodos y de la velocidad de la red, han aparecido tecnologías como la radio por Internet, videoconferencias, vídeo bajo demanda, VoIP (voz sobre IP), televisión por Internet o emisiones en directo. Y todos estos servicios requieren que el envío y la recepción de los datos se produzcan en tiempo real, para así poder ser reproducidos al momento y sin interrupciones. Para que estas transmisiones puedan realizarse en tiempo real, se necesita de nuevos protocolos, formatos, estándares, algoritmos y aplicaciones en los que se profundizará a continuación.

Los temas a tratar en el punto 2 profundizarán en los protocolos RTP (Real Time Protocol) y RTSP (Real Time Streaming Protocol).

La proliferación de equipos, sumada a la disponibilidad de hardware de audio/video económicos y la posibilidad de contar con velocidades de conexión cada vez más rápidas, ha aumentado el interés en el uso de Internet para enviar audio y video, tipos de datos que tradicionalmente se reservaban para redes especializadas. Durante los últimos años, las audioconferencias y las videoconferencias se han convertido en una práctica común. Sin embargo, la misma naturaleza de Internet indica que esta red no está preparada para la transmisión de datos en tiempo real y, por consiguiente, la calidad del audio transmitido por Internet generalmente tiene una calidad mediocre. Esta teoría específicamente trata el análisis y la solución de estos problemas para permitirle a una audioconferencia o aplicación de teléfono por Internet que cambie su funcionamiento para mantener una calidad auditiva aceptable, incluso en los casos en los que la red esté algo congestionada. Estas soluciones, que toman la forma de mecanismos de control, se han implementado y evaluado en la audioconferencia y en el software de teléfono por Internet Free Phone que hemos desarrollado. Un estudio sobre el efecto que estos equipos podrían tener en Internet, desarrollado para integrar la disciplina de servicio Fair Queuing (cola equitativa), ha demostrado que si bien estos mecanismos podrían ser todavía necesarios, funcionarían mucho mejor en este tipo de red.

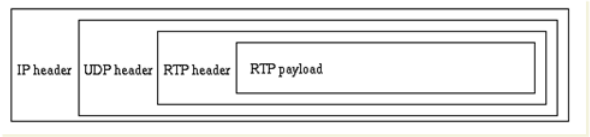
2. Proyecto

2.1. Análisis de los protocolos RTP y RTSP

2.1.1. RTP

Como se sabe del curso de Redes II, RTP, protocolo de transporte en tiempo real, proporciona funciones para redes de extremo a extremo adecuadas para aplicaciones que transmiten datos en tiempo real, como audio, vídeo, o datos provenientes de una simulación, sobre redes unicast o multicast. El transporte de datos es acompañado por un protocolo de control (RTCP) que permite monitorizar el envío de datos de forma escalable en redes multicast.

La cabecera de un paquete RTP esta formada por los siguiente campos:



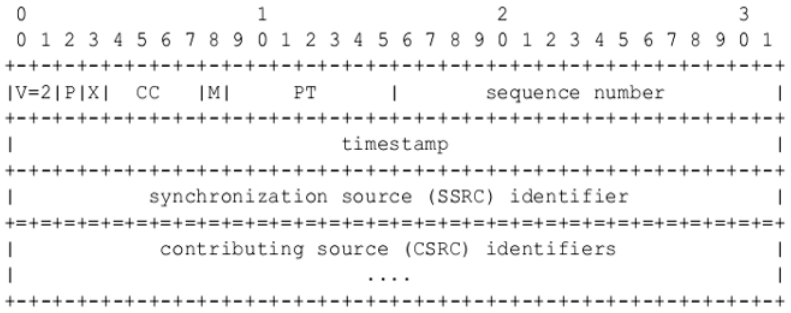


Figura 1: Cabecera de RTP

Dónde se reconocen los siguientes punto:

- Version (V): Identifica la versión del protocolo. La versión 2 corresponde al estándar actual, la versión 1 al primer borrador, y el valor 0 era usado por el protocolo implementado inicialmente en la herramienta “vat”.
- Padding (P): Se usa para algoritmos de cifrado que requieren de un tamaño fijo de bloque. Si P es establecido, el paquete contiene uno o más octetos al final, que no forman parte del payload, el último de estos octetos indica cuantos octetos deben de ser ignorados (incluido él).
- Extension (X): Si el bit de extensión esta activado, la cabecera debe de ir seguida de otra cabecera de extensión (header extension) como se puede comprobar en la figura 3. Se trata de un mecanismo para implementaciones específicas o nuevos formatos de payload que requieran de más información adicional en la cabecera del paquete RTP. Esto permite que una implementación que no soporte dicha extensión pueda trabajar con parte de la información del paquete.
- CSRC Count (CC). Indica el número de fuentes que contribuyen.
- Marker (M). La interpretación exacta de este bit queda establecida por los perfiles (profiles). Se usa para notificar de eventos significativos como un cambio de cámara (frame boundaries). Un perfil puede añadir más bits de marcas o decir que este no es un bit de marca cambiando el tamaño del siguiente campo.
- Payload type (PT): Identifica el formato del payload, o método de codificación del audio/vídeo. Puede cambiar durante la sesión, pero no hay que usarlo para multiplexar varios flujos multimedia. Por último, un receptor debe de ignorar los paquetes con un PT que no entiende.

- Sequence number. Este campo se incrementa en una unidad para cada paquete RTP que es enviado. Y le sirve al receptor para detectar pérdidas de paquetes si encuentra saltos, o para reordenar la secuencia de paquetes. El valor inicial debe de ser aleatorio para dificultar ataques basados en conocimiento de texto plano, incluso si una fuente no cifra, porque más tarde un traductor puede hacerlo.
- Timestamp: Refleja el instante de la muestra del primer octeto en un paquete de datos, esto es, una marca de tiempo. Ese instante debe ser calculado con un reloj que incremente de forma monótona y lineal para permitir sincronización. El valor inicial debe de ser aleatorio, e incrementa en función del tiempo cubierto en el contenido del paquete. A diferencia del número de secuencia, varios paquetes RTP pueden tener el mismo timestamp si son imágenes del mismo frame.
- Syncronization source identifier (SSRC): Identifica a la fuente con la que se sincroniza, también debe ser elegido aleatoriamente con la intención de que no haya dos fuentes en la misma sesión que tengan el mismo identificador SSRC. Sin embargo, todas las implementaciones de RTP deben estar preparadas para detectar y resolver colisiones.
- Contributing source identifiers (CSRC): Identifica a las fuentes que contribuyen al payload. El número de identificadores viene dado por el campo CC y esta comprendido entre 0 y 15. Los identificadores CSRC son insertados por los mezcladores, usando los identificadores SSRC de las diferentes fuentes. Por ejemplo, para paquetes de audio, los SSRC de los emisores que van a ser mezclados son listados en el paquete, permitiendo al receptor identificar quien habla.
- Payload. Hace referencia a los datos transportados por un paquete RTP. Por ejemplo muestras de audio o vídeo comprimido.

Es importante saber lo anterior ya que a raiz de esta información se construye el código para el envío y recepción de los datos por el protocolo RTP

2.1.2. RTSP

El protocolo de streaming en tiempo real (RTSP) fue desarrollado por RealNetworks, Netscape Communications y Columbia University y está publicado en el RFC 2326, es un protocolo a nivel de aplicación para el envío de datos con propiedades de tiempo real que puede trabajar junto a otros protocolos como RTP y RSVP. Proporciona un entorno para el envío de datos de tiempo real bajo demanda, como lo son el audio y el vídeo. Las fuentes de datos pueden incluir tanto datos en directo, como almacenados. Este protocolo puede funcionar sobre UDP, UDP multicast y TCP.
El servidor proporciona el contenido multimedia, los clientes solicitan continuamente datos al servidor; y RTSP se comporta como un mando a distancia entre un cliente y el servidor, que permite:

- Conseguir datos del servidor. El cliente le pide al servidor que configure una sesión para que le envíe los datos solicitados.
- Invitar a un servidor a una conferencia.
- Añadir datos a una presentación existente. El cliente o el servidor pueden notificarle a la otra parte sobre datos que se encuentran disponibles.

RTSP intenta dar los mismos servicios para audio y vídeo que HTTP hace para texto y gráficos; y ha sido diseñado intencionadamente para que tenga una sintaxis y operaciones similares. Cada flujo esta identificado por una URL RTSP. Cada presentación y sus propiedades multimedia quedan recogidas en un fichero de descripción de la presentación, y este fichero puede ser obtenido por los clientes vía HTTP, por email o cualquier otro medio.

Pero RTSP difiere de HTTP en algunos aspectos: mientras que HTTP es un protocolo sin estados, un servidor RTSP tiene que mantener los estados de las sesiones para hacer corresponder pedidos y flujos. Y además, HTTP es un protocolo asimétrico donde el cliente hace peticiones y el servidor responde, mientras que en RTSP ambos pueden hacer peticiones.

Los servicios y operaciones soportados son los siguientes:

- OPTIONS: El cliente o el servidor le comunican a la otra parte que opciones aceptan.
- DESCRIBE: El cliente consigue una descripción de un contenido identificado por una URL RTSP.
- ANNOUNCE: Actualiza la descripción en tiempo real.
- SETUP: El cliente le pregunta al servidor donde conseguir los datos.
- PLAY: El cliente le pide al servidor que comience a mandarle los flujos configurados en SETUP.
- PAUSE: El cliente detiene el envío sin liberar los recursos en el servidor.

- TEARDOWN: El cliente solicita al servidor que detenga el envío del flujo especificado y libere todos los recursos asociados a él.
- GET_PARAMETER: Consigue el valor de un parámetro de una presentación o flujo.
- SET_PARAMETER: Establece el valor de un parámetro de una presentación o flujo.
- REDIRECT: El servidor informa al cliente de que debe conectarse al servidor indicado en la cabecera.
- RECORD: El cliente comienza a grabar datos de la presentación.

Las peticiones RTSP son enviadas normalmente por un canal independiente al canal de los datos. Estos últimos pueden ser transmitidos por otro tipo de conexión.

A continuación se muestra el diagrama de estados que se programó, es lo más basico que se utiliza en RTSP.

Figura 2: Cabecera de RTP

2.2. Pruebas del proyecto

2.2.1. Cliente

En este punto se instaló el cliente en un PC y el servidor en otro PC, configurando el programa del cliente para que hiciera la conexión con la IP del servidor. A continuación se muestra el intercambio de paquetes entre ellos.

1. Inicia conexión TCP con servidor

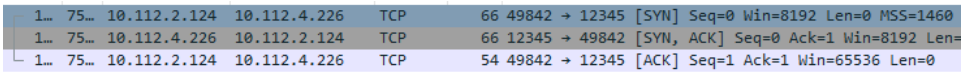


Figura 3: Handshake con el servidor

2. Hace un setup para configurar puertos UDP para transmisión de video

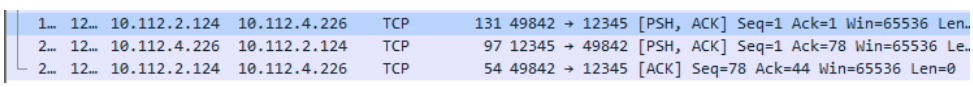
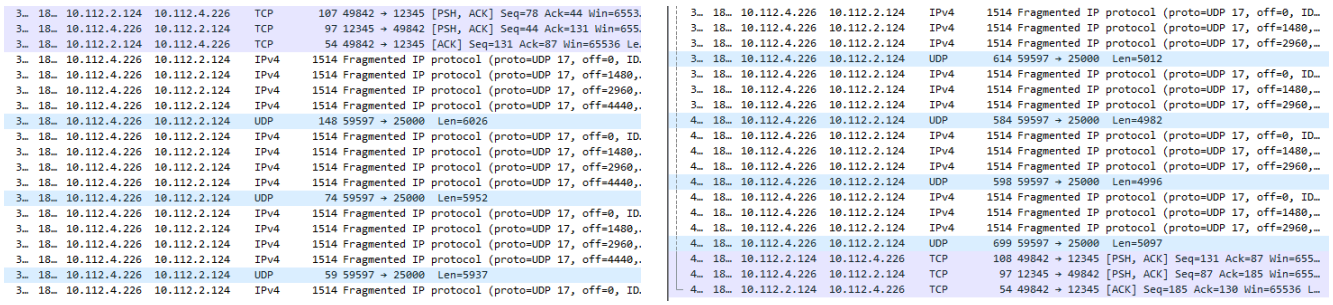


Figura 4: Setup con el servidor

3. Deja al servidor en stand by
4. Control de play y pause usando RTSP



(a) Play (b) Pause

Figura 5: Uso de los comandos de RTSP

5. Teardown (cierra las conexiónes, apaga servidor y cliente)

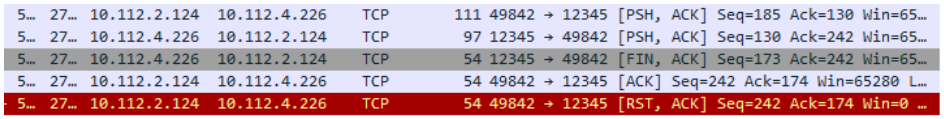


Figura 6: Handshake con el servidor

2.3. Servidor

1. Espera conexión del cliente y se produce el handshake

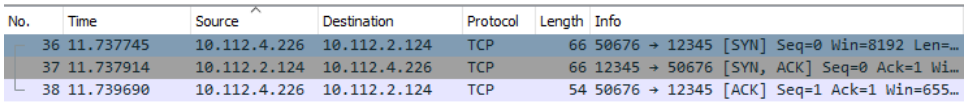


Figura 7: Handshake con el servidor

2. Espera Setup

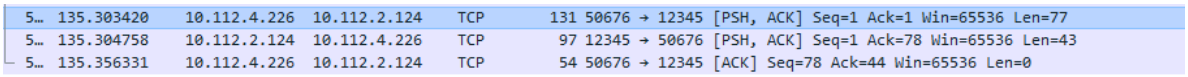


Figura 8: Setup con el servidor

Además en éste punto se pudo observar el intercambio de información entre cliente-servidor y cómo se establecía los formatos a utilizar:

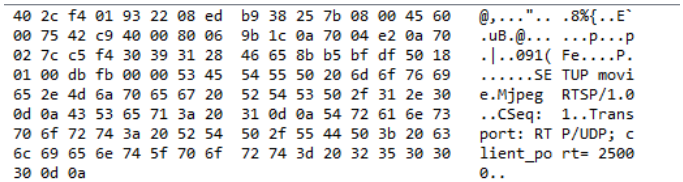
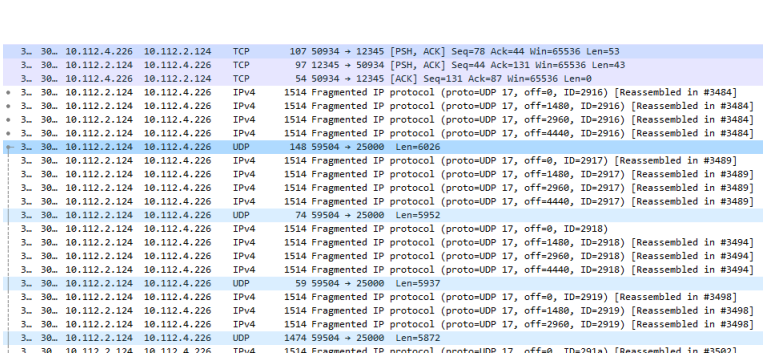
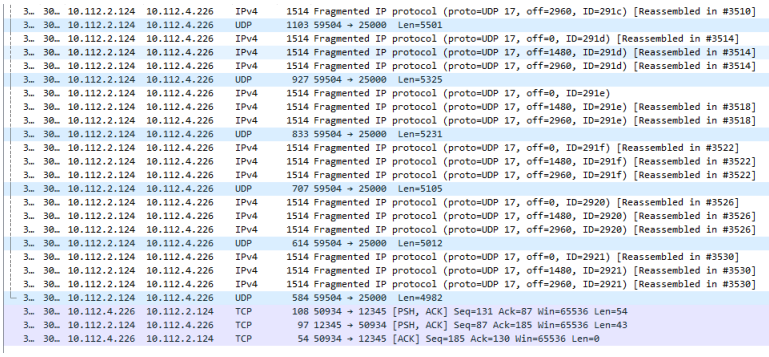


Figura 9: Configuración del setup entre cliente-servidor

3. Control de play y pause usando RTSP



(a) Play



(b) Pause

Figura 10: Uso de los comandos de RTSP

4. Teardown (cierra las conexiónes, apaga servidor y cliente)

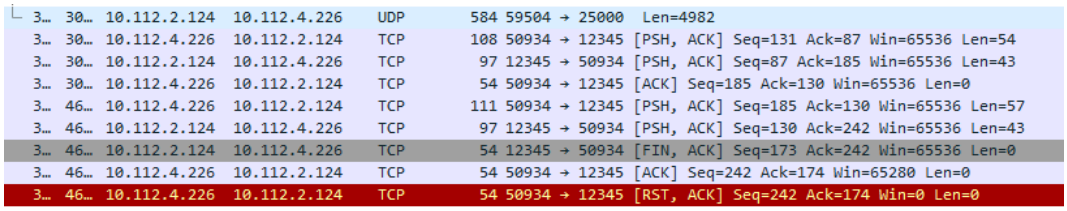


Figura 11: Handshake con el servidor

3. Conclusiones

Hemos visto como ha evolucionado el transporte de contenidos multimedia en Internet, las empresas implicadas, estándares usados y algunas aplicaciones finales. Gracias a estos estándares es posible la compatibilidad entre clientes y servidores de distintas plataformas y sistemas operativos. Hemos pasado de usar la red telefónica para conectarse a Internet, a usar esta red para telefonía IP o televisión por Internet. Y a medio plazo podremos ver todas estas tecnologías funcionar sobre el protocolo IPv6 en las variantes unicast, multicast, y también anycast.

Se logró observar la compatibilidad entre clientes y servidores en distintas plataformas y sistemas operativos que hacen streaming de contenido y se demostró alguno de los conceptos básicos de cómo implementar un servidor basado en “streaming” de contenidos mediante los protocolos RTP/RTSP. Cabe destacar que en este proyecto no se implemento un código completo de funcionamiento de “streaming” (incluyendo buffering entre otras cosas) para todo tipo de formatos tanto para video como audio y mecanismos de transporte.

Como se logró mostrar, en el proyecto hubo una interacción con solo un formato de video por lo cual no fue necesario incluir protocolos que garantizaran calidad de servicio, envíos a tiempo, etc. es por eso que se optó por RTP, que, funcionando con best effort, realiza la tarea que se precisaba, además de ofrecer servicio “multicast” y una cómoda forma de implementación.

4. Bibliografía

- Texto James F. Kurose and Keith W. Ross, "Computer Networking: A top-Down Approach Featuring the Internet", Pearson, 6º Edition 2013.
- RTP: About RTP and the Audio-Video Transport Working Group `http://www.cs.columbia.edu/~hgs/rtp/`
- rtsp.org: Real Time Streaming Protocol Information and Updates `http://www.rtsp.org`
- RFC 3550 " RTP: A Transport Protocol for Real-Time Applications"`http://tools.ietf.org/html/rfc3550`
- RFC 3551 "RTP Profile for Audio and Video Conferences with Minimal Control" `http://tools.ietf.org/html/rfc3551`
- RFC 2326 "Real Time Streaming Protocol (RTSP)" `http://tools.ietf.org/html/rfc2326`
- `http://www.monografias.com/trabajos17/formatos-audio/formatosaudio.shtml`
- Real-time Transport Protocol `http://www.lab.dit.upm.es/~labscom/almacen/sld/rtp.pdf`
- Java Media Framework API (JMF)`http://java.sun.com/products/java-media/jmf/`
- `http://www.csee.umbc.edu/~pmundur/courses/CMSC691C/lab5-kurose-ross.html`