UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Redes de Computadores I

Protocolo RTMP

Fecha 13/06/2019

Integrantes Valentina Espinoza
Gastón Quevedo
Alejandro Romero
Pablo Troncoso

Asignatura Redes de Computadores I (ELO-322)
Profesor Agustín González

Resumen

El uso de aplicaciones de *streaming* ha aumentado considerablemente en los últimos años. Todos estos servicios, como por ejemplo *Twitch*, ocupan modos de operar bastante parecidos compartiendo ciertos comportamientos o formas de comunicar que son comunes a todos ellos. Por ello, para poder entender cómo funcionan estas aplicaciones, es necesario conocer el uso de los protocolos correspondientes.

Los protocolos usados son varios, pero en este trabajo nos enfocaremos en el protocolo RTMP, que es uno de los que existen para que los *streams* puedan ser comunicados fluidamente y de forma expedita. Se buscará explicar sus funciones principales, sus características más importantes y sus diversas aplicaciones a través de la Web, lo que nos otorgará un mejor entendimiento de este y su uso en las aplicaciones. Luego se verá cómo este protocolo funciona actualmente en la red, y que tan usado es.

Introducción

Los servicios de *streaming* nos permiten ver u oír transmisiones en vivo y en directo a través de reproductores web dentro de una página o un dispositivo móvil.

Esta tecnología se ha masificado en los últimos años al permitirnos acceder a distintos tipos de contenidos de audio/vídeo de forma más segura a través del *streaming*. El uso de ciertos protocolos se han hecho sumamente importantes y vitales para el funcionamiento de las aplicaciones que nos brindan estos servicios, y para el efecto de este trabajo nos enfocaremos en el desarrollo del protocolo RTMP (*Real-Time Messaging Protocol*).

El protocolo RTMP se usa para los servicios *streaming* el cuál trabaja sobre TCP (*Transmission Control Protocol*) y usa por defecto el puerto 1935. Permite al cliente controlar la calidad de distribución de la transmisión y seguridad.

Este trabajo busca hacer un análisis general del protocolo RTMP, cómo funciona, sus características y aplicaciones, explicando así el funcionamiento de múltiples servicios que son utilizados a través de la web.

Real Time Messaging Protocol

Funcionamiento

RTMP es un protocolo basado en TCP (igualmente puede funcionar con otros protocolos) que se utiliza para transmitir datos multimedia entre Flash Media Server y Flash Player. La mayor utilidad de RTMP es la transferencia a través de conexiones persistentes y baja latencia. RTMP usa un puerto exclusivo (1935) para transmisión de vídeo que permite al protocolo la transmisión de baja latencia del contenido. Para las transmisiones la información se divide en fragmentos y su tamaño es negociado dinámicamente entre cliente y servidor, estos fragmentos son llamados *chunks*. El tamaño máximo de cada fragmento de audio es de 64 bytes y 128 bytes para vídeo y la mayoría de otros tipos de información.

RTMP puede enviar y recibir los paquetes por canales virtuales específicos que son independientes el uno del otro. Por ejemplo, hay un canal para datos de flujo de audio, uno para vídeo, etc. Estos canales en una transmisión normal pueden estar activados de forma simultánea. [ref001]

RTMP Chunk Stream

Es el sistema por el cual se multiplexa y empaquetan los paquetes para servicios de streaming. Este puede manejar cualquier protocolo que envía un *stream* de mensajes. Cuando es usado sobre un protocolo de transporte confiable (como en este caso con TCP) provee tiempos de entregas de todos los mensajes a través de todos los canales.

Distintas implementaciones pueden asignar distintas prioridades a distintos tipos de mensajes, lo que puede afectar el como estos mensajes son enviados cuando la capacidad de transporte está limitada.

Estructura del Paquete

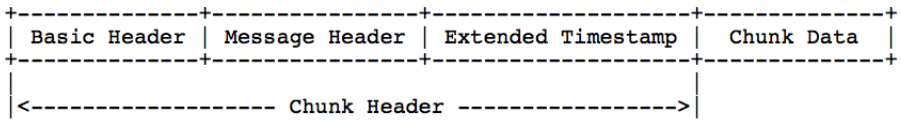

Figura 1: Estructura del paquete

1. *Basic Header* (1 a 3 bytes): Este campo codifica el *chunk stream ID* y el *chunk type*. El *chunk type* determina el formato del *message header* codificado. El largo depende totalmente del *chunk stream ID*, que es un campo de largo variable.

2. *Message Header* (0, 3, 7 o 11 bytes): Este campo codifica información sobre el mensaje siendo enviado (sea entero o una parte). El largo puede ser determinado usando el *chunk type* especificado en el *chunk header*.

3. *Extended Timestamp* (0 o 4 bytes): Este campo está presente en ciertas circunstancias dependiendo de el *timestamp* codificado o el campo *timestamp delta* en el *Chunk Message header*.

4. *Chunk Data* (tamaño variable): La carga útil de este *chunk*, hasta el tamaño máximo configurado del *chunk*.

Handshake

Una conexión RTMP comienza con un *handshake*. El *handshake* no es como en el resto de los protocolos, consiste en tres *chunks* estáticos en ves de *chunks* de tamaño variable con encabezados.

El cliente y el servidor envían los mismos tres *chunks*. Denominaremos estos tres *chunks* como C0, C1 y C2 cuando son enviados por el cliente, por otro lado, cuando son enviados por el servidor, los definiremos como S0, S1 y S2.

El *handshake* comienza cuando el cliente envía C0 y C1.

El cliente debe esperar hasta que reciba S1 antes de enviar C2. Y el cliente ahora debe esperar hasta que S2 haya sido recibido, para así poder enviar cualquier otra información. [ref002]

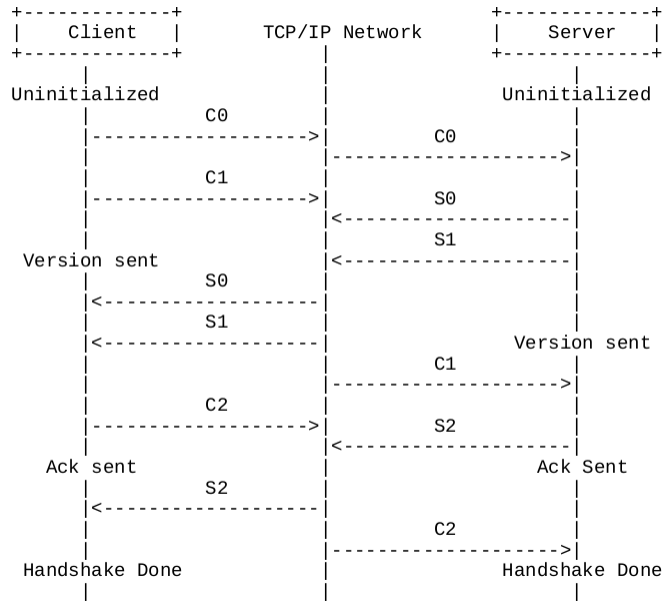

Figura 2: Representación gráfica del *handshake*

Encriptación

Las sesiones RTMP pueden ser encriptadas usando uno de los siguientes métodos:

1. Usando RTMPS, que es el protocolo RTMP sobre una conexión segura TLS/SSL (*Transport Layer Security/Secure Sockets Layer*)

2. Usando RTMPE (*Encrypted RTMP*), que envuelve la sesión de transmisión RTMP en una capa de cifrado ligera.

3. RTMPT, es la encapsulación de los datagramas RTMP en paquetes HTTP (*Hypertext Transfer Protocol*) para evadir *firewalls* es a través del puerto web estándar (puerto 80). ref[003]

Análisis de capturas *Wireshark*

Como se puede ver en la figura 3, logramos identificar características de este protocolo en las capturas de Wireshark, los cuales se vieron teóricamente a lo largo de este informe. Como el *handshake*, que el protocolo trabaja sobre TCP y que utilizael puerto 1935 (Ver figura 4).

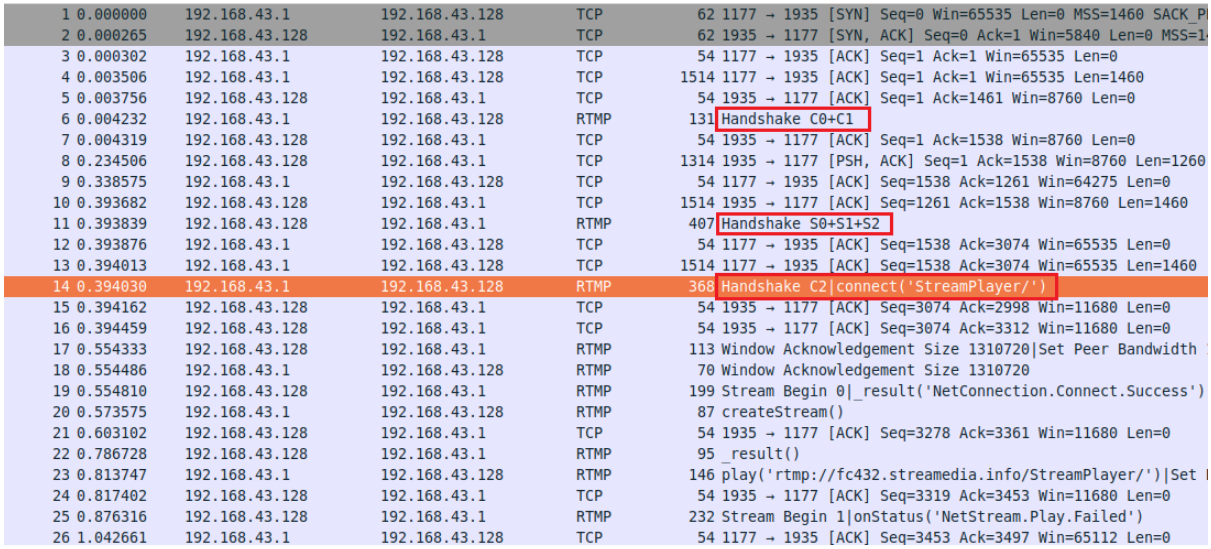

Figura 3: Captura *Wireshark*

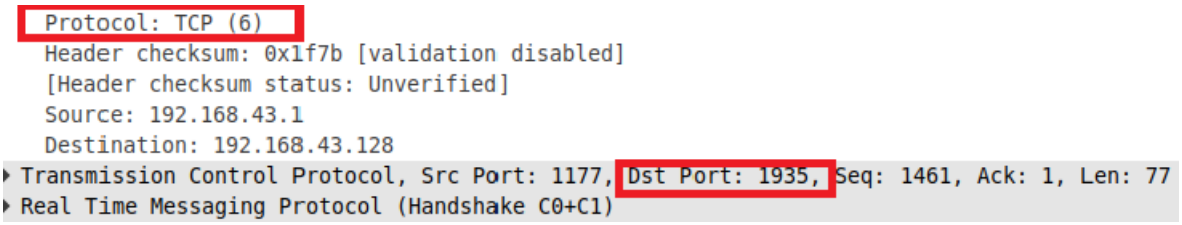

Figura 4: Captura *Wireshark*

ref[004]

Conclusiones

El protocolo RTMP contiene valiosas utilidades en el ámbito de la transmisión de datos en tiempo real. Como se mencionó, su principal uso se encuentra en las aplicaciones *streaming* donde se puede aprovechar su funcionamiento para la entrega persistente y de baja latencia del contenido. Aunque hace algún tiempo este protocolo tuvo bastante importancia, a través de la investigación de RTMP se descubrió que su uso ha decaído notablemente en los últimos años, esto debido a su poca seguridad en la transmisión de información, generando la aparición de protocolos RTMP con modificaciones para paliar sus ineficiencias tales como RTMPE Y RTMPS. También es importante el hecho de que este protocolo fue construido sobre Flash Player, el cual su uso ha decaído notoriamente y en donde se evita en la mayoría de aplicaciones modernas, lo que ha afectado notablemente a que RTMP ya no sea la opción más viable en lo que respecta a las transmisiones en tiempo real, existiendo mejores alternativas como lo puede ser el ampliamente utilizado HTTPS. Aun así, corresponde un recurso para tener en cuenta cuando se habla de protocolos para transmisiones en tiempo real, pues empresas como *Twitch* y *Facebook* todavía usan este protocolo en sus servicios de *streaming*, pero con modificaciones que permiten su funcionamiento en la red actual.

Referencias

[ref001] Wikipedia (2019) https://en.wikipedia.org/wiki/Real-Time_Messaging_Protocol

[ref002] Parmar & Thornburgh, (2012) *Adobe's Real Time Messaging Protocol*, versión 1.0, Adobe. Recuperado de: https://www.adobe.com/devnet/rtmp.html

[ref003] https://www.vdocipher.com/blog/2016/11/rtmp-encrypted-rtmpe-streaming-technology/

[ref004] https://wiki.wireshark.org/SampleCaptures