UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

PROYECTO FIN DE CARRERA

Desarrollo de un sistema de medición, monitorización y gestión de servicios OTT

María Lucena Cabello

Marzo 2015

Desarrollo de un sistema de medición, monitorización y gestión de servicios OTT

AUTOR: María Lucena Cabello
TUTOR: Jorge E. López de Vergara Méndez

Computación y Redes de Altas Prestaciones
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Marzo de 2015

Agradecimientos

Me gustaría agradecer en primer lugar a mi tutor Jorge E. López de Vergara Méndez, por darme la oportunidad de realizar con él este proyecto fin de carrera. Por su entendimiento, rapidez y eficacia en todo lo que hace. Por su apoyo y consejos que me han guiado de forma excepcional para la realización de este proyecto.

Así también, me gustaría agradecérselo a todas las personas que me han acompañado en la carrera. En especial a mis amigos, Sara García-Mina, por ser la mejor compañera de prácticas que se puede tener, Fátima García porque juntas hemos compartido momentos increíbles responsables de grandes carcajadas que no se olvidan nunca, Sandra Uceda por haber demostrado saber estar siempre ahí para todo, mi compañero de café Rodrigo López y a todos los toligos. Gracias chicos por todas las experiencias vividas fuera y dentro de la escuela.

Me gustaría dar las gracias de manera especial a Carlos, por todo su apoyo y ánimo, por entenderme y darme tiempo, ayudarme y recargarme, y llenar todos y cada uno de los días. Porque formamos el mejor equipo, gracias por ser como eres.

Y por último a mi familia, mi padre, mi madre y mi hermano, a quienes les dedico este proyecto. Por ser el mejor modelo a seguir, gracias por apoyarme toda mi vida, por ser mis pilares y las personas responsables de ser quien soy.

Resumen

Tradicionalmente los servicios de entrega de contenidos han sido explotados por compañías de telecomunicaciones que controlaban el entorno sobre el que se distribuía el contenido.

En la actualidad, debido a la proliferación de proveedores de contenidos se han desarrollado los servicios Over-The-Top, convirtiéndose en una preocupación para los operadores de telecomunicaciones cuando han empezado a competir directamente con los servicios ofrecidos tradicionalmente. Los servicios OTT se prestan sobre las redes de los operadores, creando la necesidad de conocer cómo perciben los usuarios la prestación de estos servicios, tanto desde el punto de vista del operador de red como del prestador del servicio, de forma que se pueda mejorar la calidad que está ofreciendo. En los servicios OTT no hay, no existe ancho de banda reservado, por tanto, no hay garantía en la calidad de servicio, lo cual plantea una serie de problemas frente a las técnicas de monitorización tradicionales de los servicios.

En este Proyecto Final de Carrera se plantea el desarrollo de una herramienta para la recogida y tratamiento de datos que permitan monitorizar, la calidad de servicio (QoS) de los servicios OTT. Este sistema de monitorización puede ser útil tanto para los operadores, ya que permite verificar si los usuarios están teniendo problemas con los servicios que tienen sobre su red, como para los proveedores de contenidos al controlar si los usuarios están recibiendo un buen servicio, y negociar con el operador mejoras en su tráfico si fuera necesario.

Palabras clave

Sistema de medición, monitorización, calidad de servicio, servicios OTT

Abstract

Traditionally, content delivery services have been exploited by telecommunication companies which controlled the environment where the content was distributed.

Currently, due to the proliferation of content providers, Over-The-Top services have been developed. The more they started to compete directly with the services traditionally offered, the more Telecom operators begun to concern about them. The OTT services are provided on the operators' networks, creating the need to know how users perceive the provision of these services, both from the point of view of the network operator and the service provider, so the offered QoS can be improved. On OTT services, which do not have a reserved bandwidth, there is no guarantee on the service quality, which poses a number of problems compared to traditional monitoring techniques of services.

In this Final Project we have developed a tool for collecting and processing data to monitor the quality of service (QoS) of OTT services. This monitoring system can be useful for operators, so they can verify if users are having problems with the services deployed over the network, and for content providers to control whether users are getting a good service, and negotiate with the operator improvements in their traffic if necessary.

Keywords

Measurement System , Monitoring , Quality of Service , OTT Services

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

1.1 Motivación

En la actualidad se consideran servicios 'over-the-top' (OTT) a aquellos servicios multimedia (incluyendo audio, vídeo y texto) que se ofrecen por terceros sin la intervención de un operador tradicional de telecomunicaciones [1].

Los servicios OTT se han convertido en una preocupación para los operadores de telecomunicaciones cuando han empezado a competir directamente con los servicios ofrecidos tradicionalmente por éstas, como por ejemplo: voz, mensajería y TV (pay-per-view).

Algunos ejemplos de servicios OTT pueden ser:

- Amazon vídeo o Youtube en servicios de TV y Video.
- WhatsApp, Facebook, Blackberry Messenger en servicios de mensajería.
- Skype y Viber en servicios de Voz.

Por ejemplo, Skype en la actualidad cuenta con 250 millones de usuarios activos por mes, quienes hablan 100 minutos en promedio, evitando así el uso de la telefonía tradicional. Tal como Skype se convirtió en un competidor de telecomunicaciones en voz, muchos de los nuevos proveedores de servicios OTT han entrado al mercado como sustitutos de servicios tradicionales en telecomunicaciones.

El proyecto se va a centrar en los servicios OTT relativos a TV, como puede ser por ejemplo Yomvi, Wuaki, Youzee, etc. Según ComScore [2], el consumo de video en línea en sitios como YouTube, Vevo y Globo.com representa de 11 a 13 horas al mes por usuario, dependiendo del país. Ésta es solo una fracción del consumo mensual de televisión, que oscila entre 100 y 165 horas. Sin embargo, la tasa de crecimiento del consumo de video en línea es superior al 70% anual, mientras que el consumo de televisión normal está casi estancado.

La aparición de estos servicios que se prestan sobre las redes de los operadores, plantea la necesidad de conocer cómo perciben los usuarios la prestación de estos servicios, tanto desde el punto de vista del operador de red como del prestador del servicio, de forma que se pueda mejorar la calidad que está ofreciendo.

1.2 Objetivos

En los servicios OTT el contenido llega al usuario final sin intervención del operador de Internet, que solo es responsable de la entrega de los paquetes IP. No hay, por tanto, garantía en la calidad de servicio al no haber ancho de banda reservado. Esto plantea una serie de problemas frente a las técnicas de monitorización tradicionales de servicios de vídeo. En los servicios tradicionales de vídeo (p.e. Imagenio) el flujo de datos se transporta usando *multicast* en RTP (Protocolo de Transporte en tiempo real) sobre UDP; en cambio el flujo de datos de los servicios OTT es independiente para cada usuario y sobre TCP, lo

que hace que el tipo de problemas que puedan surgir sean distintos, debido a que las pérdidas de paquetes se corregirán, al utilizar TCP, pero a costa de que se vacíe el buffer del receptor, lo que supondrá problemas en el servicio. Además, la forma de detectar el tipo de tráfico es también distinta, teniendo que comprobar cuando sea posible, por ejemplo, que el mime-type de HTTP es de tipo vídeo, para no estar midiendo otro tipo de descargas.

En suma, el objetivo general de este Proyecto Final de Carrera es el desarrollo de un sistema para la recogida y tratamiento de datos que permitan monitorizar la calidad de servicio (QoS) de los servicios OTT.

Este sistema de monitorización puede ser útil tanto para los operadores, ya que permite verificar si los usuarios están teniendo problemas con los servicios que tienen sobre su red, como para los proveedores de contenidos al controlar si los usuarios están recibiendo un buen servicio, y negociar con el operador mejoras en su tráfico si fuera necesario.

Para alcanzar este objetivo final se han planteado los siguientes sub-objetivos:

- Estudiar el estado actual de los servicios OTT.
- Aprender a analizar capturas de tráfico de datos.
- Medir parámetros de calidad a partir de las capturas de tráfico recogidas.
- Aprender a analizar los resultados obtenidos.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- *Estado del Arte*. Donde se va a realizar una documentación previa con el fin de recabar información de los conocimientos necesarios para el desarrollo de la herramienta.
- *Análisis de requisitos*. En este capítulo se va a realizar un análisis de una serie de parámetros tomados entre los dos extremos de una comunicación, para posteriormente poder analizarlos para medir la calidad ofrecida por los servicios OTT a los clientes.
- *Implementación de la herramienta.* En este capítulo se explica detalladamente cada uno de los pasos seguidos para la implementación de la herramienta.
- *Integración, pruebas y resultados.* En este capítulo se van a exponer en forma de gráficos los resultados obtenidos mediante la herramienta. También se realizará un análisis de los mismos.
- *Conclusiones y trabajo futuro.* Se resumirán las conclusiones del proyecto y se hará un planteamiento de las líneas de trabajo futuro.

2 Estado del arte

En este capítulo se procederá a realizar una documentación previa con el fin de recabar información acerca de los servicios OTT que existen en la actualidad, así como de su funcionamiento. También se ampliarán los conocimientos tanto de calidad de servicio (QoS) como de calidad de experiencia (QoE). Igualmente se estudiarán el funcionamiento de los protocolos IP, TCP y HTTP, con el fin de conocerlos y sentar las bases para poder medir los parámetros necesarios de calidad del servicio. Además, también se analizará en profundidad la biblioteca de programación libpcap, herramienta que será utilizada en la fase de desarrollo.

2.1 Servicios OTT

Tradicionalmente los servicios de entrega de contenidos de vídeo han sido explotados por compañías de telecomunicaciones que controlaban el entorno sobre el que se distribuía el contenido. Estas compañías controlaban el medio incorporando equipamiento que permitía controlar la calidad del servicio ofrecido. Ejemplos de estos servicios fueron los desarrollados por Telefónica tales como Imagenio u otros equivalentes.

Con la proliferación de proveedores de contenidos se han desarrollado los servicios OTT (Over The Top). Una aplicación o servicio OTT se puede definir como cualquier aplicación o servicio que entrega contenidos utilizando Internet, sin hacer uso de los mecanismos tradicionales, que como se ha comentado anteriormente son controlados por compañías específicas.

La clave de las aplicaciones / servicios OTT es que no provienen de las empresas de telecomunicaciones tradicionales o los proveedores de servicios de Internet. Ejemplos de servicios OTT son los siguientes:

- Distribución de vídeo: Youtube, Netflix,
- Llamadas entre usuarios: Skype o Facetime
- Intercambio de mensajes: Whatsapp

El concepto OTT se empezó utilizando en la distribución de vídeo con la aparición de Netflix o Hulu, que fueron desbancando a proveedores de contenidos en Norte América como Comcast o ATT. Posteriormente, el concepto se utilizó también en el mundo del intercambio de mensajes entre particulares.

La compañía de estudio de medios de Internet, OVUM, emitió un estudio en febrero de 2012 que valoró estos servicios OTT como una pérdida de beneficios a los operadores tradicionales de 8.700 millones de dólares en 2010 y de 13.900 millones en 2011. En la actualidad, la aparición de servicios como Whatsapp han hecho casi desaparecer los servicios de entrega de mensajes SMS y MMS de las operadoras.[3]

Algunas de las empresas de telecomunicaciones y proveedores de Internet se han dado cuenta de esta situación y están tratando de convertirse también en proveedores de servicios OTT.

Figura 2-1: Ejemplos de Servicios OTT [4]

De todo lo anterior se deduce que la mayoría de los proveedores de servicios OTT no controlan el medio sobre el que trasmiten los contenidos. Además, no tienen equipos específicos que permitan asegurar una determinada calidad de servicio. Es por todo esto que las medidas de la calidad de servicio (QoS) que se está prestando al cliente final de servicios OTT (QoE) se convierte en algo esencial y crítico.

2.2 Calidad de Servicio (QoS) y Calidad de Experiencia (QoE)

En los últimos tiempos, el concepto "Quality of Experience (QoE)" o Calidad de la Experiencia de usuario ha conseguido implantarse tanto en el campo académico como industrial. Este concepto está muy unido a la percepción subjetiva de la calidad percibida por un usuario, y supone un avance significativo sobre el concepto tradicional, mucho más centrado en los elementos técnicos de una conexión, de "Quality of Service (QoS)" o Calidad de Servicio.

Para entender el concepto de QoE es interesante repasar cómo se ha llegado al mismo a lo largo de los años. El concepto de Calidad de Servicio ha sido definido y tratado por diferentes organismos internacionales: ITU-T, ISO, ETSI, IETF [5]. Sirva como ejemplo la definición que se recoge en la Recomendación UIT-T E-800 [6] aprobada en septiembre de 2008 y que en la actualidad aún sigue en vigor. En ella se define toda una serie de términos utilizados en el estudio y la gestión de QoS.

Fue Moorse quien introdujo el concepto de QoE en el contexto de servicios basados en el protocolo HTTP. En su presentación, Moorse distingue entre QoE y QoS, mientras que QoE puede tener métricas subjetivas, relacionadas con la percepción del usuario, el QoS no. [7]

QoE lo podríamos definir como el resultado de una valoración hecha por un usuario y por lo tanto tiene mucho que ver con las expectativas que éste tiene. Es por tanto un valor subjetivo y dependiente no sólo del entorno técnico.

En este mismo estudio [5] da un enfoque interesante buscando algo que es el punto fundamental de este proyecto y es cómo conseguir correlacionar las medidas cuantitativas (fácilmente medibles) del QoS de red y aplicaciones con la medida más subjetiva del QoE.

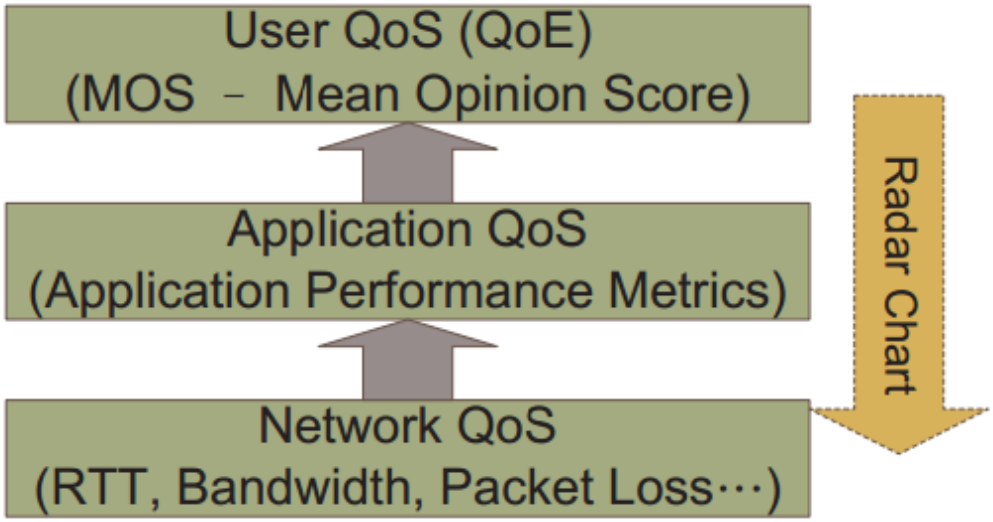


Figura 2-2: Los tres niveles de QoS [8]

La calidad de servicio QoS depende de ciertos parámetros de la red de comunicaciones, ejemplos de estos parámetros podrían ser: retardo de la red, pérdida de paquetes, *throughput* y disponibilidad. Existe una recomendación (ITU-T Recommendation Y.1540) que define cinco clases de nivel de servicio en función de cuatro parámetros de la red:

- ✓ Retardo en la trasferencia de paquetes
- ✓ Variación del retardo de paquetes entre dos puntos de la red
- ✓ Ratio de pérdida de paquetes
- ✓ Ratio de paquetes erróneos

La calidad de la experiencia percibida por el cliente QoE depende de la percepción recibida por el usuario. Uno de los métodos de medida de QoE es el MOS (*Mean Opinion Score*, Puntuación Media de Opinión) que se basa en realizar encuestas al cliente sobre diferentes puntos y valorar éstos mediante puntuaciones del 1 al 5, siendo el 5 el nivel máximo. Los puntos sobre los que se podrían preguntar dependen del servicio concreto (video, mensajería, juego, …)

2.3 Protocolos IP, TCP y HTTP

Los servicios OTT se transmiten utilizando el tradicional protocolo de internet HTML, el mismo que ha sido usado durante muchos años para transmitir información de páginas web, mediante el protocolo TCP.

El estudio de estos protocolos se hace necesario para poder conocer sus características, funciones y el formato de cada uno de ellos.

2.3.1 Protocolo IP

IP es un protocolo de comunicación clasificado funcionalmente en la Capa de Red según el modelo internacional OSI.

Su función principal es el uso bidireccional en origen o destino de comunicación para transmitir datos mediante un protocolo no orientado a conexión, es decir, no hay garantías de que un datagrama IP consiga con éxito llegar a su destino.

El protocolo IP proporciona un servicio de entrega de datagramas sin conexión "*best effort*", no provee ningún mecanismo para determinar si un paquete alcanza o no su destino y únicamente proporciona integridad del paquete mediante checksums o sumas de comprobación de sus cabeceras y no de los datos transmitidos. Cuando algo va mal, IP tiene un algoritmo simple que consiste en tirar un datagrama, normalmente el último en llegar. Si se necesita fiabilidad, ésta es proporcionada por los protocolos de la Capa de Transporte, como TCP.

El término "connectionless" quiere decir que IP no mantiene información que relacione unos datagramas con otros; esto también significa que cada datagrama se maneja de manera independiente de los otros, y por lo tanto, pueden ser entregados fuera de orden, duplicados en tránsito o tener sus datos alterados.

2.3.1.1 ***Cabecera IP***

La cabecera IP contiene las direcciones de las máquinas de origen y destino (direcciones IP), direcciones que serán usadas por los enrutadores (*routers*) para decidir el tramo de red por el que reenviarán los paquetes. La versión más popular de este protocolo de red es IPv4. IPv6 es el sucesor propuesto de IPv4; poco a poco Internet está agotando las direcciones disponibles, por lo que IPv6 utiliza direcciones de fuente y destino de 128 bits, lo cual posibilitará direccionar más máquinas que IPv4 con 32 bits.

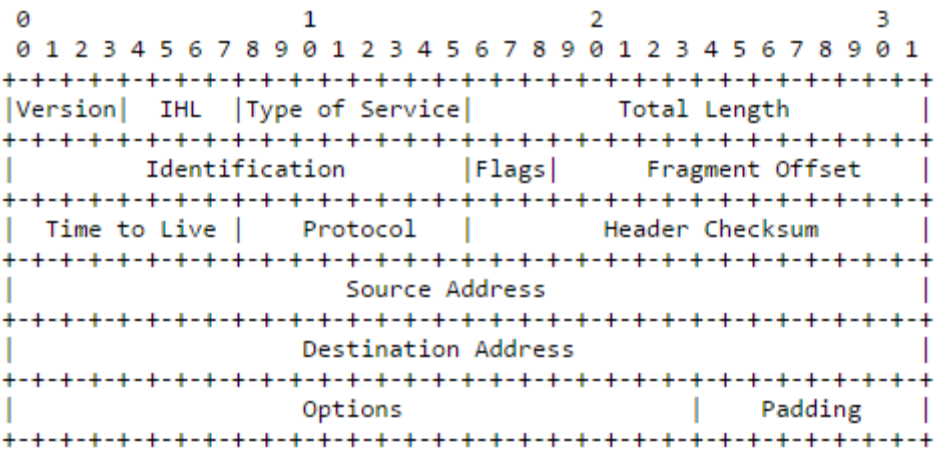


Figura 2-3: Datagrama IP [9]

A continuación se expone una descripción de cada campo de la cabecera:

Tabla 2-1: Descripción cabecera IPv4

Campo	Bits	Descripción
2.3.1.2 *Version*	4 bits	Versión IP (IPv4 o IPv6)
IHL	4 bits	Longitud de la cabecera en palabras de 4 octetos.
TOS	8 bits	DSField+ ECN. Indicador de la calidad de servicio solicitado por el datagrama IP.
Total Length	16 bits	Tamaño total del paquete, incluyendo cabeceras y datos.
Identification	16 bits	Identificador único del datagrama. Debe asegurar un valor único para la pareja origen-destino.
Flags	3 bits	Especifica valores relativos a la fragmentación de paquetes.
Fragment Offset	13 bits	En paquetes fragmentados indica la posición que ocupa el paquete actual dentro del datagrama original.
Time-to-Live (TTL)	8 bits	Indica el máximo número de enrutadores que un paquete puede atravesar.

Protocol	8 bits	Protocolo de las capas superiores al que debe entregarse el paquete.
Header Checksum	16 bits	Suma de Control de cabecera. Se recalcula cada vez que algún router cambia alguno de sus campos.
Source Address	32 bits	Dirección de origen IP.
Destination Address	32 bits	Dirección de destino IP.
Options	(variable, 40 bytes max)	Opciones
Padding	(variable, 65,515 bytes max)	Datos

2.3.2 Protocolo TCP

Como se ha descrito anteriormente, existe un problema de la comunicación en entornos en los que el medio de comunicación puede perder o alterar el mensaje que se entrega. El protocolo TCP usa como base para corregir errores la solicitud de repetición automática (ARQ), la cual consiste en "enviar de nuevo" hasta que la información se reciba correctamente.

TCP está clasificado funcionalmente en la capa de Transporte del modelo internacional OSI. Proporciona un servicio fiable orientado a conexión. El término orientado a conexión significa que las dos aplicaciones que utilizan TCP deben establecer una conexión poniéndose en contacto entre sí antes de que puedan intercambiar datos.

TCP no interpreta el contenido de los bytes de datos en el flujo, serán niveles superiores (como por ejemplo HTTP) los que se encargarán de interpretarlo.

2.3.2.1 ***Diagrama de estados***

El protocolo TCP, para intercambiar información así como para apertura y cierre de conexiones, se comporta siguiendo el siguiente diagrama de estados:

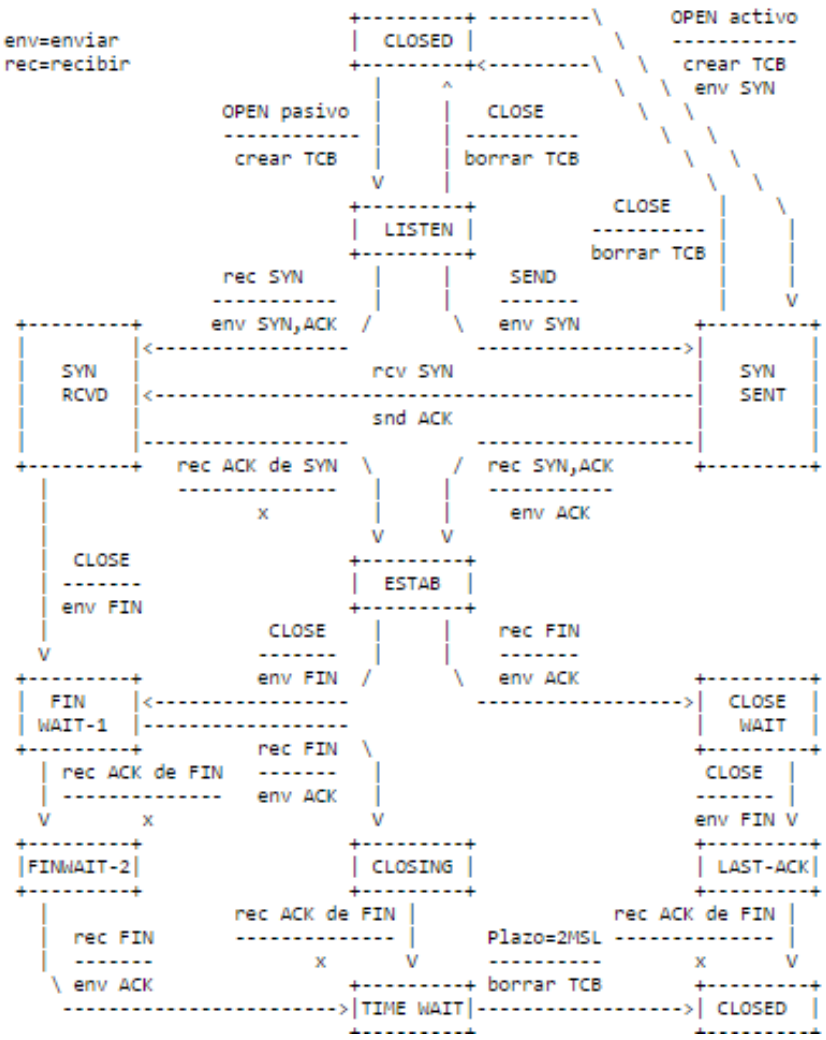


Figura 2-4: Diagrama de estados TCP [10]

Una conexión progresa de acuerdo con una serie de estados durante su tiempo de vida. Los estados son [9]:

Tabla 2-2 : Descripción estados TCP

Estado	Definición
LISTEN	Representa la espera de una solicitud de conexión proveniente de cualquier TCP y puerto remotos.
SYN-SENT	Representa la espera de una solicitud de conexión concordante tras haber enviado previamente una solicitud de conexión.
SYN-RECEIVED	Representa la espera del acuse de recibo confirmando la solicitud de conexión tras haber recibido una solicitud de conexión.

ESTABLISHED	Representa una conexión abierta, los datos recibidos pueden ser entregados al usuario. El estado normal para la fase de transferencia de una conexión.
FIN-WAIT-1	Representa la espera de una solicitud de finalización de la conexión proveniente del TCP remoto, o del acuse de recibo de la solicitud de finalización previamente enviada.
FIN-WAIT-2	Representa la espera de una solicitud de finalización del TCP remoto.
CLOSE-WAIT	Representa la espera de una solicitud de finalización de la conexión proveniente del usuario local.
CLOSING	Representa la espera del paquete, proveniente del TCP remoto, con el acuse de recibo de la solicitud de finalización.
LAST-ACK	Representa la espera del acuse de recibo de la solicitud de finalización de la conexión previamente enviada al TCP remoto (lo que incluye el haber enviado el acuse de recibo de la solicitud)

2.3.2.2 ***Cabecera TCP***

La cabecera de un segmento TCP contiene información con tres finalidades diferentes:

1. Identificar los puertos origen y destino que, junto con las direcciones origen y destino de la cabecera IP, identifica cada conexión.
2. Comunicar al otro extremo información, mediante una bandera, de cómo progresa la conexión a través del diagrama de estados expuesto anteriormente.
3. Y en caso de que la conexión se encuentre en el estado 'Established', pasar información sobre las secuencias de mensajes enviadas al otro extremo y las recibidas correctamente en el extremo que emite el segmento TCP.

A continuación se expone una descripción de cada campo de la cabecera:

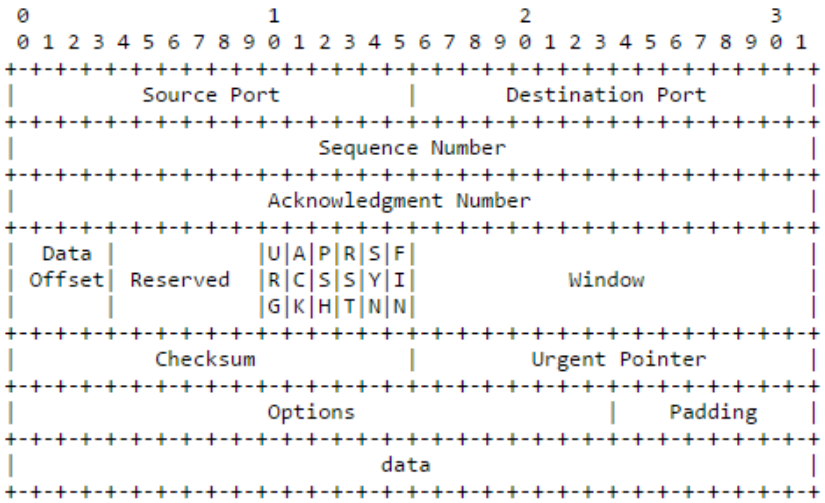


Figura 2-5: Cabecera TCP[11]

Tabla 2-3: Descripción cabecera TCP

Campo	Bits	Descripción
Source Port	16 bits	Puerto Origen.
Destination Port	16 bits	Puerto Destino.
Sequence Number	32 bits	Identifica el primer byte de datos del segmento.
Acknowledgment Number	32 bits	Contiene el siguiente Sequence Number que se espera recibir del otro extremo de la conexión.
Header Legth	4 bits	Longitud de la cabecera en palabras de 4 octetos.
Resv	4 bits	Reservado para uso futuro.
Flags	8 bits	Banderas: CWR, ECE, URG, ACK, PSH, RST, SYN, FIN.
Window Size	16 bits	Sistema de control de flujo, especifica número máximo de bytes pendientes de asentimiento.
TCP Checksum	16 bits	Suma de verificación de datos.
Urgent Pointer	16 bits	Cantidad de bytes desde el número de secuencia que indica el lugar donde acaban los datos urgentes.
Options	Variable (40 bytes max)	Opciones.

2.3.2.3 ***Flujo TCP***

Al conjunto de paquetes que se intercambia entre el cliente y el servidor durante una conexión se le denomina 'flujo' [12]

Se considerará cliente a la estación que inicia la conexión mediante el envío de un primer paquete SYN.

Según las direcciones y puertos los paquetes de un mismo flujo los podemos dividir en dos grupos:

- Paquetes Cliente-Servidor. Son los que salen del cliente hacia el servidor y por lo tanto todos tendrán las mismas direcciones y puertos origen (cliente) y destino (servidor).
- Paquetes Servidor-Cliente. Son los que salen del servidor hacia el cliente, al igual que en el caso anterior todos tendrán la mismas direcciones y puertos origen (servidor) y destino (cliente).

2.3.3 Protocolo HTTP

HTTP es un protocolo de comunicación clasificado funcionalmente en la capa de Aplicación según el modelo internacional OSI.

Este protocolo define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, *proxies*) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

Desde el punto de vista de las comunicaciones, se soporta sobre los servicios de conexión TCP/IP. Articula los intercambios de información entre los clientes Web y los servidores HTTP. El servidor escucha el puerto de comunicaciones TCP (por defecto el 80), y espera las solicitudes de conexión de los clientes Web. Una vez que se establece la conexión, el protocolo TCP se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores.

2.3.3.1 ***Comandos principales de HTTP: GET y POST***

Al protocolo HTTP se le asigna habitualmente el puerto TCP 80. Los métodos de petición más utilizados son GET y POST.

Se usa el método de petición GET para solicitar contenidos de un servidor. El recurso se solicita a través de la URL al servidor Web. Mediante el método de petición POST, los datos a enviar al servidor se incluyen en el cuerpo de la misma petición con las cabeceras HTTP.

2.4 Biblioteca Libpcap

Para la programación del sistema de medición se ha utilizado la biblioteca libpcap. Es una biblioteca para la captura de paquetes, que proporciona una serie de funciones para acceder a los paquetes que circulan por la red, ofreciendo al programador una interfaz para capturar paquetes en la capa de red.

2.4.1 Esquematización de un programa con Libpcap

A continuación se expone el esquema básico que siguen los programas construidos con Libpcap:

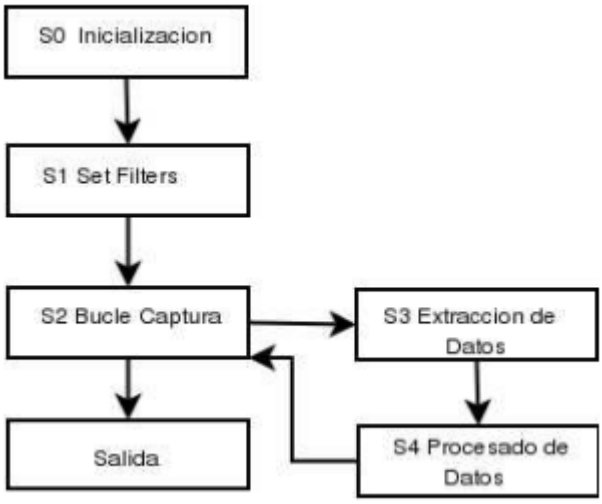


Figura 2-6: Esquematización de un programa con Libpcap [13]

- S0: Fase de inicialización: Engloba las funciones capaces de obtener información del sistema: Las interfaces de red instaladas, la configuración de estas interfaces (Máscara de Red, Dirección de Red, etc).

- S1: Filtrado: No existe un sistema único de filtrado, por el contrario prácticamente cada S.O reescribe su propia solución: NIT para SunOS, Ultrix Packet Filter para DEC Ultrix, BPF para sistemas BSD (originalmente) y más recientemente LSF (Linux Socket Filter) la implementación para Linux.

  BPF es el sistema de filtrado más extendido y se basa en dos componentes principales: El Network Tap (encargado de recopilar los paquetes desde el driver del dispositivo de red y entregárselos a aquellas aplicaciones a las que vaya destinado) y el Packet Filter (encargado de decidir si el paquete debe ser y en caso afirmativo, cuánto de ese paquete debe ser entregado a la aplicación).

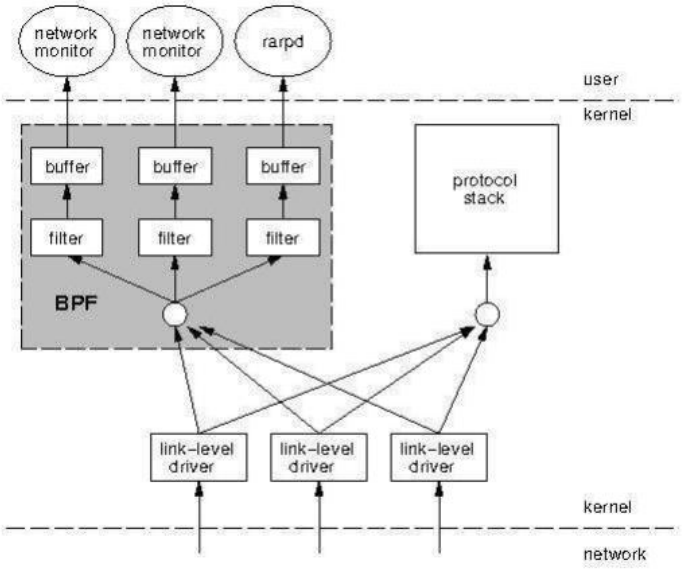


Figura 2-7: Acción BPF [13]

BPF será el encargado de comparar el paquete con cada uno de los filtros establecidos, pasando una copia de dicho paquete a cada uno de los buffers de las aplicaciones cuyo filtro se ajuste a los contenidos del paquete.

- S2: Captura de paquetes: Existen varias funciones para capturar paquetes, las principales diferencias entre ellas son: El número de paquetes que queremos capturar, el modo de captura, normal o promiscuo y la manera en que se definen sus funciones de llamada o *Callbacks* (la función invocada cada vez que se captura un paquete).

- S3: Interpretación de datos: En este paso se extraen e interpretar las cabeceras de los protocolos añadidas por el remitente.

- S4: Volcado de datos: Libpcap nos ofrece también la posibilidad de guardar los datos en un fichero para procesarlos más adelante.

2.5 Conclusión

Para el análisis de los servicios OTT se ha necesitado hacer una pequeña introducción sobre estos así como sobre la calidad de servicio (QoS) y de experiencia (QoE), exponiendo sus principales diferencias y su importancia en la medida de la calidad de los servicios OTT.

El flujo de datos de los servicios OTT se transporta sobre TCP, por lo que en este capítulo se ha expuesto cada uno de los protocolos IP, TCP y HTTP, explicando analizando sus cabeceras y funciones principales que serán útiles en la fase de programación de la herramienta.

Por último, se ha estudiado la biblioteca de programación libpcap usada en la herramienta desarrollada para la captura de paquetes, aportando una serie de funciones que permiten acceder a los paquetes que circulan por la red.

3 Análisis de Requisitos

Para poder medir la calidad de los servicios OTT ofrecida a los clientes, mediante la herramienta desarrollada en este proyecto, se van a analizar una serie de parámetros de la trasmisión entre los dos extremos de una comunicación.

Para poder medir estos parámetros es necesario ir analizando cada uno de los campos paquetes TCP. Mediante este análisis se podrá, posteriormente, determinar la calidad de servicio de la conexión.

En este capítulo se van a estudiar los parámetros que se han considerado necesarios para este análisis:

- ✓ RTT,
- ✓ número de retransmisiones,
- ✓ número de anuncios de ventana cero,
- ✓ *throughput* del servidor,
- ✓ *goodput* del servidor y
- ✓ paradas en la transmisión del servidor.

3.1 RTT

El primer parámetro a analizar es el RTT (*Round-Trip delay Time*). Se denomina RTT al tiempo transcurrido entre que se envía un segmento en una conexión TCP y se recibe el ACK de este segmento [14]. Por lo tanto, el RTT incluye el tiempo que se requiere para que el servidor procese el mensaje que viene desde el cliente y genere una respuesta. Se puede calcular como la diferencia de tiempo entre la emisión de un paquete y la recepción de su asentimiento (ACK).

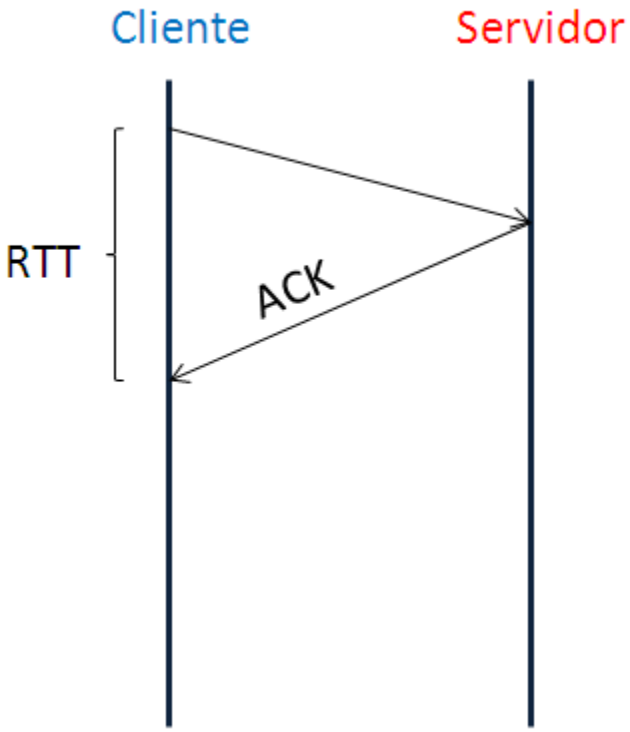


Figura 3-1: Esquema RTT

Este parámetro se puede usar para el análisis del rendimiento del sistema: si aumenta el valor del tiempo RTT, el rendimiento general del sistema disminuye. Además, si los paquetes se pierden o dañan, la situación es aún peor, ya que se daña la información que contiene el paquete y por tanto, el rendimiento sería aún más bajo [15].

Mediante el análisis de este parámetro también se puede estimar el estado del medio de transmisión: líneas, routers, conmutadores. Un valor alto de RTT puede indicar problemas en el medio de trasmisión.

Se ha hecho el análisis de este parámetro, diferenciando entre RTT inicial y RTT promedio. Ambos parámetros van a ser medidos en sentido Cliente-Servidor, teniendo en cuenta que los servicios OTT de vídeo tienen su mayor incidencia en este sentido.

3.1.1.1 ***RTT Inicial***

El RTT inicial se refiere al tiempo que tarda el primer paquete de un flujo en ir desde el cliente al servidor y volver al cliente.

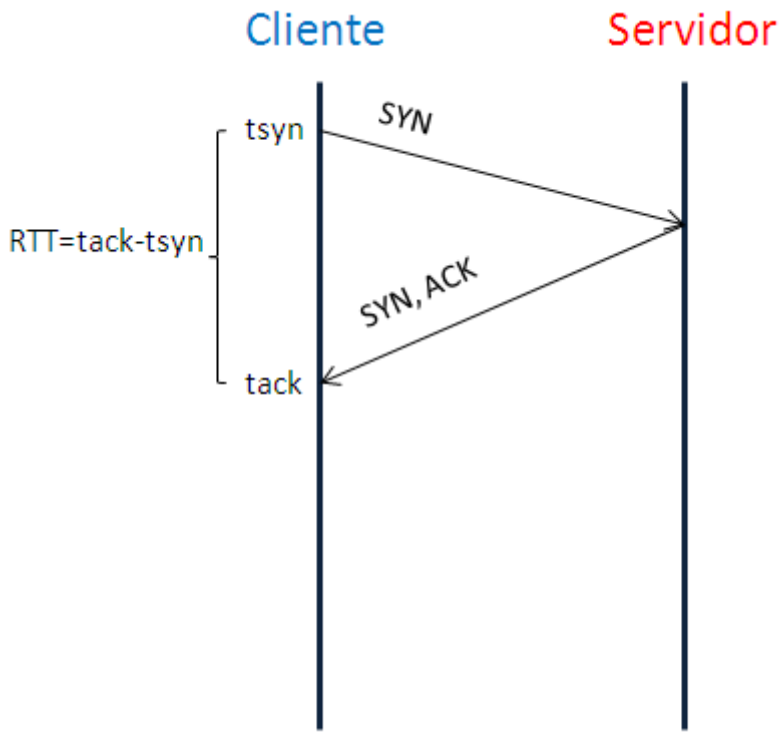


Figura 3-2: Esquema RTT inicial

Para la implementación del análisis en la herramienta, en primer lugar, se procede a la detección del primer SYN de cada flujo sentido Cliente-Servidor, para así poder guardar el tiempo de este paquete al cual llamaremos *tsyn*. A continuación, se detectará el SYN, ACK sentido servidor-cliente generado en respuesta a dicho SYN, y se guardará el tiempo en el que el cliente recibe este ACK al cual llamaremos *tack*. Con lo cual, el RTT inicial será la diferencia entre *tsyn* y *tack*

3.1.1.2 ***RTT Promedio***

Por otro lado, se calculará el RTT promedio de un flujo en el sentido cliente-servidor. El RTT se medirá como la diferencia de los tiempos entre un paquete de datos enviado desde el cliente hasta que se recibe su asentimiento (ACK) confirmando la recepción del mismo.

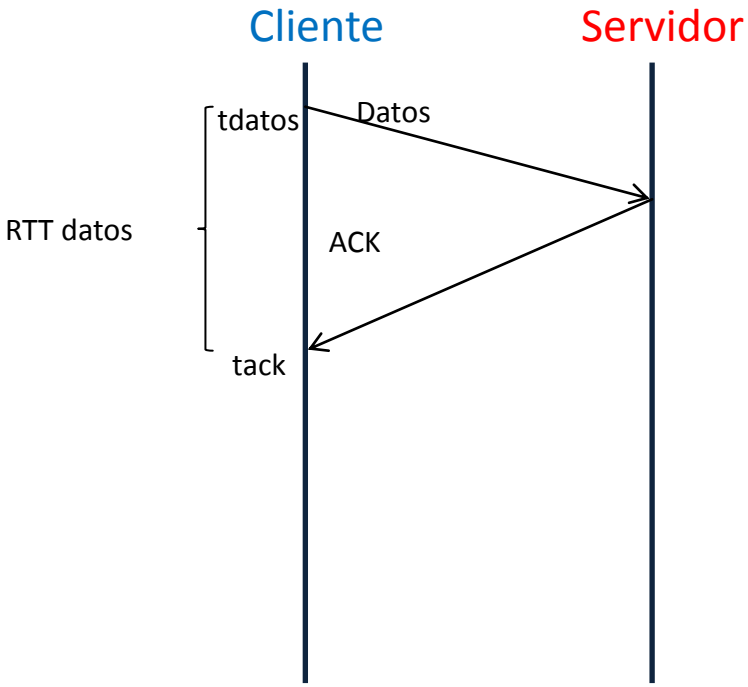


Figura 3-3: Esquema RTT de datos

Para la implementación se toman los paquetes enviados por el cliente que contengan datos, de estos paquetes, se guarda la hora a la que se envía, '*tdatos'*, el *número de secuencia del siguiente paquete esperado*.

El cálculo del *número de secuencia del siguiente paquete esperado* se realiza mediante la suma del número de secuencia del paquete actual más la longitud de datos del mismo:

- El valor del *número de secuencia actual* se toma del paquete que se está analizando, accediendo al campo "Sequence Number" de la cabecera TCP del paquete.

- Para el cálculo de la *longitud de datos* del paquete, en primer lugar se deberá tomar la longitud total del paquete para lo cual se hace uso de la función proporcionada por pcap *struct pcap_pkthdr.*

  Mediante el acceso a esta estructura se obtiene la longitud total del paquete dado en su argumento "len". Ya conocido el tamaño total del paquete, se procede a sustraerle el tamaño de sus cabeceras. En el caso de Ethernet el tamaño de su cabecera entregado por la lipcap es fijo, siendo siempre 14 bytes. Para el tamaño de la cabecera TCP, se accede al campo "Header Length". Y por último, para el tamaño de la cabecera IP se accede al campo "Total Length".

Una vez se han obtenido el tiempo de envío del paquete de datos y el número de secuencia siguiente, se accede a cada uno de los paquetes del flujo sentido servidor-cliente buscando

aquel cuyo número de ACK sea igual al *número de secuencia siguiente esperado*, se toma el tiempo de este paquete que se denomina '*tack'*.

El tiempo RTT de este paquete de datos se calculará como:

RTT datos

Cada vez que se cumplan las condiciones anteriores se irá midiendo los diferentes RTT de datos de todo el flujo y se irán acumulando en:

∑

Igualmente se irá contando el número de paquetes de datos sobre los que se ha medido el promedio de RTT que llamaremos *Número de Paquetes de datos*, por último el *RTT Promedio* será:

3.2 Anuncio de ventanas cero

TCP usa el protocolo de ventana deslizante para el control de flujo entre el cliente y el servidor. Este mecanismo surge como mejora del usado por protocolos con reconocimientos positivos de tipo parada y espera; estos protocolos obligan al servidor a retrasar la emisión de cada nuevo paquete hasta que se recibe el ACK del anterior, desaprovechando así la posible capacidad de comunicación bidireccional de la red. Los protocolos de ventana deslizante (ver Figura 3-4) aprovechan mejor el ancho de banda al permitir transmitir un número determinado de paquetes antes de recibir su reconocimiento ACK correspondiente. El servidor puede transmitir múltiples paquetes antes de comenzar la espera para que el cliente le confirme que ha recibido los paquetes correctamente, es decir, antes de que el cliente envíe un ACK.

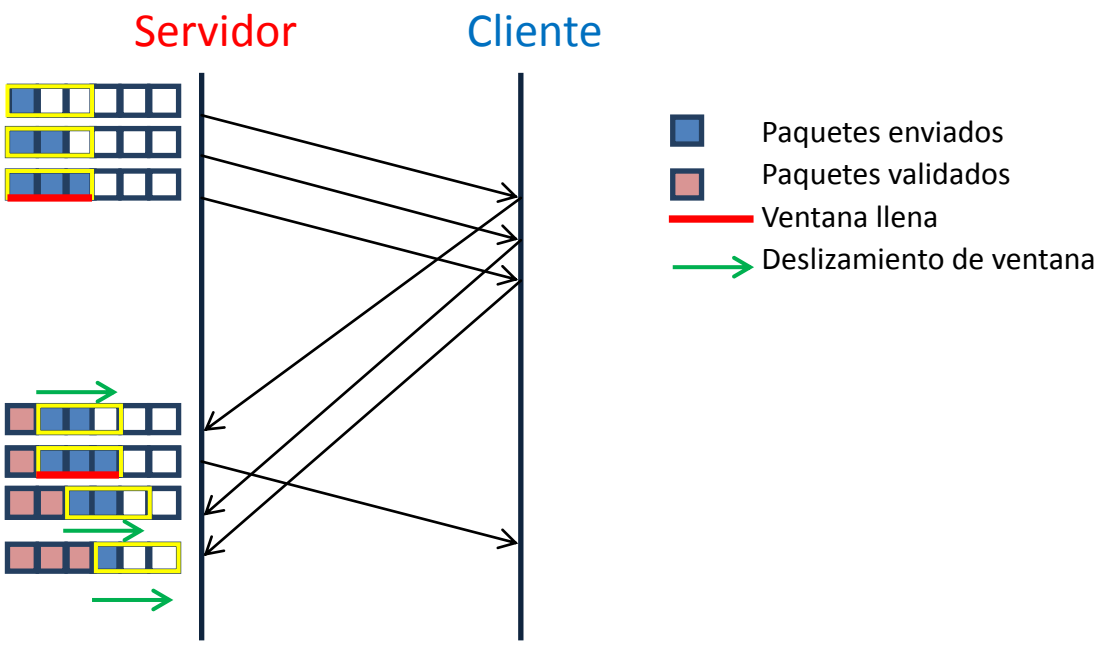


Figura 3-4: Esquema Protocolo ventana deslizante

Para conocer este parámetro se accede al campo *Window Size* de la cabecera del paquete TCP de los paquetes que van en sentido cliente servidor.

En una situación de anuncio de ventanas cero el servidor se encuentra a la espera de recibir una nueva ventana de disponibilidad, es decir, no podrá enviar más paquetes y deberá esperar a que el buffer del cliente se libere, deteniendo así el envío de datos, generando retardos en el envío de paquetes y dando lugar a una peor calidad de servicio.

Un número alto de anuncios de ventanas cero indica problemas de saturación en la parte del cliente.

3.3 Throughput

El término *througthput* hace referencia a la cantidad de datos que se envían a través de un flujo en un período de tiempo dado, es decir, da una idea del rendimiento del sistema midiendo los datos enviados por unidad de tiempo.

En este caso es interesante medir el *throughput* en sentido servidor-cliente, ya que es el servidor quien envía gran volumen de datos al cliente

Para el análisis se ha procedido de la siguiente forma; En primer lugar, se va tomando cada una de las tramas del flujo servidor-cliente y calculando el tamaño total de las tramas que como hemos visto anteriormente viene determinado en el argumento len de la estructura *struct pcap_pkthdr*. Además, se tomará el tiempo de envío de la primera y última trama del flujo, *tinicio y tfin*.

Es importante resaltar que en este caso se tomarán todas las tramas del flujo, ya sean de datos o de control.

3.4 Goodput

El término *goodput* hace referencia a la cantidad de datos útiles que se envían a través de un flujo en un período de tiempo dado. Es decir, descartando paquetes de control que no contengan datos y tramas repetidas.

En este caso, al igual que en el *throughput*, también es interesante medir el *goodput* en sentido servidor-cliente, ya que el volumen grande de datos manejados se encuentra en este sentido.

Para el análisis con la herramienta a desarrollar, se deberá tomar cada trama de un flujo servidor-cliente, contrastando que no sea una trama retransmitida y se calcula la longitud de datos de la trama de la misma forma que en el apartado del RTT promedio:

Además, se toma el tiempo de envío de la primera y última trama del flujo, *tinicio y tfin*.

3.5 Número de Retransmisiones

Como se ha explicado anteriormente, el protocolo TCP proporciona un transporte fiable. A fin de conseguir un intercambio libre de errores, el protocolo TCP vuelve a enviar los datos que cree que se han perdido [16]. Para decidir qué datos necesita volver a enviar, TCP depende de un flujo continuo de reconocimientos (ACK) de receptor a emisor. Cuando los segmentos o reconocimientos de datos se pierden, TCP inicia una retransmisión de los datos que no han sido reconocidos. TCP tiene dos mecanismos para llevar a cabo la retransmisión, uno basado en el tiempo y otro basado en la estructura del reconocimiento ACK.

- *Retransmisión basada en tiempo.* TCP asegura el envío de todos los datos de las capas superiores, por tanto debe ser capaz de detectar las pérdidas de datos y retransmitirlos. Una forma de detectarlo es mediante un temporizador de retransmisiones, llamado RTO (*Retransmission Time-Out*). Este temporizador se activa cuando se envía un segmento. Si se cumple el tiempo máximo (RTO) asignado al temporizador, sin que se haya recibido un ACK que reconozca el segmento, se retransmite el primero de los segmentos enviados pendiente de reconocimiento. Algunas de las formas de determinar el valor del parámetro RTO es mediante cálculos utilizando la media y varianza de RTT [17].

  Como se ha visto anteriormente, para los procedimientos de tiempo de espera y de retransmisión de TCP se usa el cálculo de RTO basado en la medición del RTT de una conexión determinada. Alguno de los métodos de cálculo son; el Método

Clásico, el Método Estándar, Método Linux, Método de Estimador por Comportamientos…

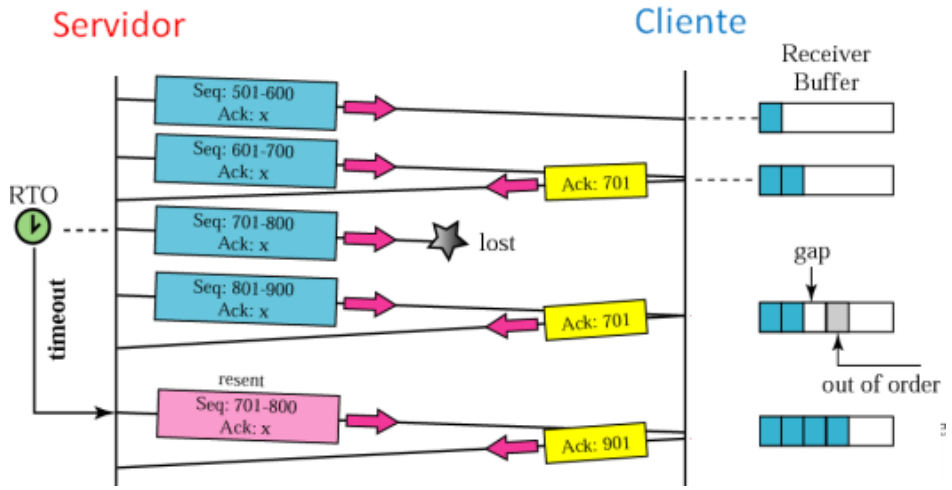


Figura 3-5: Esquema Retransmisión con RTO [16]

- *Retransmisión basada en la estructura del reconocimiento ACK*. Retransmisión por recepción de 3 ACKs duplicados, en el cual la retransmisión es más rápida y no requiere que expire el RTO.

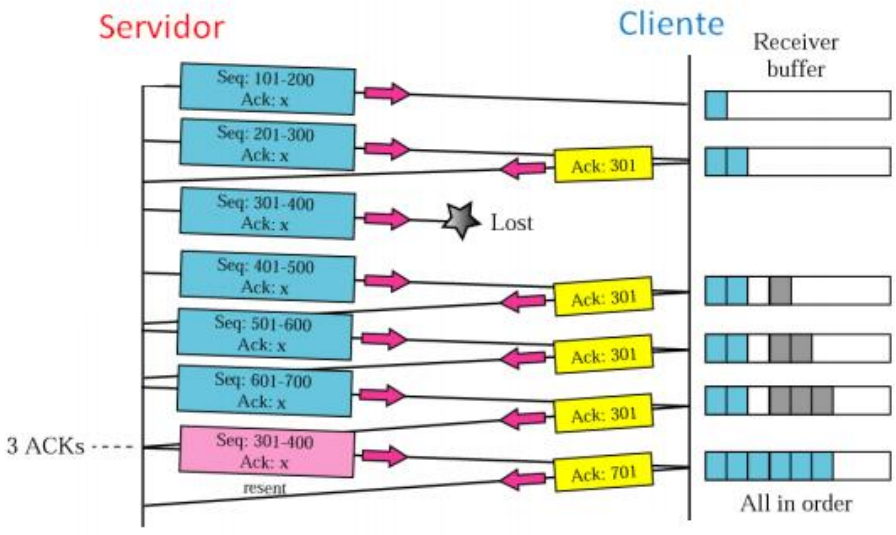


Figura 3-6: Esquema Retransmisión por recepción de [16]

La herramienta a desarrollar medirá el número de retransmisiones en el sentido servidor cliente. Para este análisis, en primer lugar se deberán tomar las tramas que contengan datos en sentido de flujo servidor-cliente. De todas ellas se va comprobando si su *número de secuencia* es mayor que el número de secuencia de la trama anterior. En el caso de ser menor se trataría de una retransmisión o un segmento fuera de secuencia (en cualquiera de los dos casos los datos del paquete no se envían al nivel superior de aplicación y se pueden

considerar equivalentes), y se procedería a sumar uno el contador del número de retransmisiones.

Un elevado número de retrasmisiones de paquetes del servidor indica una mala calidad de servicio y su origen está en la mala calidad del medio de trasmisión.

3.6 Paradas en la transmisión del servidor

Las paradas en el servidor van a ser medidas como el tiempo que el servidor permanece sin enviar datos (*stalled*). Este tiempo se define como el máximo tiempo que hay entre dos paquetes de datos dentro de una respuesta que el servidor le envía al cliente. Se debe tener en cuenta que durante ese plazo no se debe dar la circunstancia de ventana cero por parte del cliente, ya que entonces la parada se debería a la imposibilidad del cliente de recibir datos y por lo tanto, el tiempo máximo no se debe a problemas de envío en el servidor.

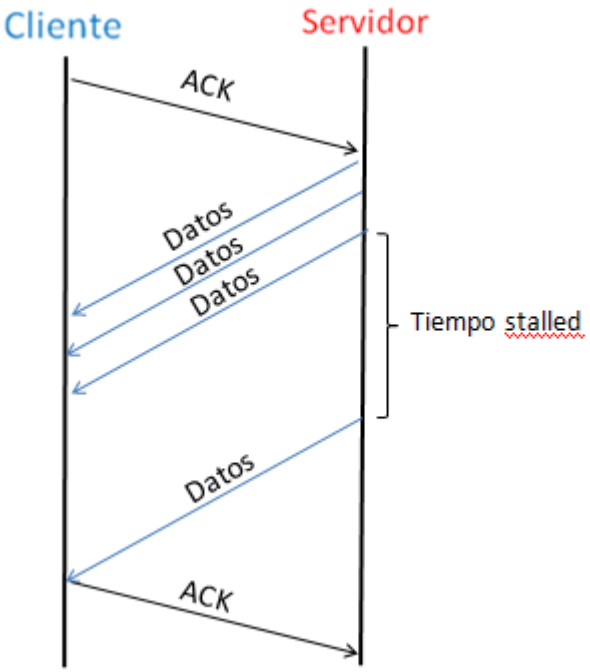


Figura 3-7: Esquema Tiempo *stalled*

Para el análisis con la herramienta se deberá medir el tiempo entre una trama de datos (‘tiempo_trata_datos’) y la siguiente trama de datos (‘tiempo_trama_datos_sig’), al final, cuando se cierre el flujo, se cogerá el máximo de estos tiempos:

)

Es importante tener en cuenta que siempre que se reciba una trama del flujo cliente servidor que contenga datos o con ventana cero, se desactivará el proceso de medida del ‘tiempo *stalled*’, de esta manera nos aseguramos que los dos paquetes de datos enviados del servidor se corresponden a la misma solicitud del cliente y además que el servidor no se ha parado por problemas del cliente (Anuncio de ventana cero).

Un valor elevado de ‘tiempo *stalled*’ podría indicar problemas en el servidor y se puede comparar con una situación de ‘atasco’ en el servidor, el cliente puede recibir datos pero el servidor no puede enviar. No obstante, esto puede deberse también a alguna política de ir sirviendo los vídeos por tramos, no enviando el siguiente tramo hasta que el usuario no esté visualizando el anterior.

3.7 Conclusión

En este capítulo del proyecto se ha procedido al estudio de cada uno de los parámetros: RTT, número de ventanas cero, número de retransmisiones, *throughput*, *goodput* y tiempo *stalled*, así como los problemas que generarían en el tráfico de datos repercutiendo en la calidad de servicio de los servicios OTT.

En capítulos posteriores se explicarán las funciones programadas en la herramienta mediante las cuales se han medido estos parámetros.

4 Implementación de la herramienta

Una vez se conocida la finalidad de la herramienta y los diferentes parámetros elegidos en el proyecto para evaluar la calidad de los servicios OTT, en este capítulo se detalla el desarrollo de la herramienta.

En primer lugar se explicarán las decisiones que se han ido tomando según la evolución del desarrollo. Además, se va a detallar el funcionamiento del programa principal, exponiendo y explicando también las estructuras y funciones utilizadas para la implementación.

4.1 Evolución del desarrollo

La finalidad principal de la herramienta es poder medir, del tráfico de la red, los parámetros requeridos de la forma más precisa posible para poder dar las medidas de calidad de los servicios OTT.

La primera idea para implementar esta herramienta fue un desarrollo que midiese los parámetros del tráfico de manera 'offline'. Con este sistema, para cada flujo, se recorría todo el fichero de datos y analizaba todos los parámetros del mismo. Este procedimiento era fácil de implementar pero muy ineficiente en tiempo de proceso.

Cuando se empezó a probar con archivos de gran tamaño, el hecho de tener que acceder de manera continua al fichero de captura, resultaba muy pesado en tiempo, haciendo que los resultados se demorasen demasiado, ya que por cada flujo que iba a ser analizado la herramienta se recorría el fichero de capturas de tráfico por completo.

Para solventar el problema anterior se decidió replantear radicalmente la forma de tratar los datos. Se pasó a hacer un tratamiento más 'online'; cada trama se trataba de manera independiente y se creó una estructura de nodos con memoria dinámica que permitiera recoger y memorizar los datos de cada flujo.

Con este nuevo planteamiento, se consiguió que los datos de las tramas fuesen analizados al mismo tiempo que la herramienta los fuese cogiendo de la captura, recorriendo así una única vez el fichero de captura de tráfico, y por lo tanto, reduciendo significativamente el tiempo de proceso de la herramienta. Con este último cambio, la herramienta puede analizar las tramas de una forma directa ('online') y con un pequeño cambio se podrían hacer análisis directos en línea.

Haciendo un análisis de la evolución de la implementación se puede resaltar la buena eficiencia en cuanto a tiempo, pudiendo estudiar capturas de tráfico de líneas de 1 Gbps, y por lo tanto, tener la posibilidad de medir el tráfico de manera online.

4.2 Estructura general

A continuación se va a proceder a la explicación funcional de la herramienta. En primer lugar se explicarán las estructuras de datos que se han utilizado, en segundo lugar se

explicará el diseño y funcionamiento general del programa, y en tercer lugar, las funciones desarrolladas.

4.2.1 Estructuras de datos

Para la implementación del programa ha sido fundamental la incorporación de dos estructuras de datos, una para los datos de las tramas y otra para los flujos.

En el desarrollo del programa, siempre que se recibe una trama útil, se crea una estructura de tipo "trama", de tal forma que queden guardados en dicha estructura los datos que se han considerado más importantes de la trama. La estructura será la siguiente:

```
struct trama{
        int hora;                       /* Hora del paquete*/
        time_t tiempo;                  /* Hora en formato GETUS*/
        int num_reg;                    /* Numero de registro en la traza*/
        uint32_t ip_o;                  /* Direccion IP Origen que lanza el SYN de apertura (cliente) */
        uint32_t ip_d;                  /* Dirección IP Destino contra quien se lanza el SYN desde el cliente (Servidor)
*/
        uint16_t p_o;                   /* Puerto origen */
        uint16_t p_d;                   /* Puerto destino */
        uint8_t long_cabeceraIP;        /* Longitud de la cabecera IP*/
        uint8_t long_cabeceraTCP;       /* Longitud de la cabecera TCP*/
        int longitud;                   /* Longitud total del paquete*/
        int longitud_datos;             /* Longitud de datos del paquete*/
        uint8_t banderas;               /* Estado de las banderas del paquete*/
        uint16_t wsize;                 /* Tamaño de la ventana*/
        uint32_t num_sec;               /* Numero de SEC del paquete*/
        uint32_t num_ack;               /* Numero de ACK del paquete*/
        uint32_t num_sec_next;          /* Numero de SEC del siguiente paquete paquete*/
        int sentido_flujo;              /* Sentido del flujo C-S=0 y S-C=1*/
        int trama_retransmitida;        /* Bandera que indica si la trama es retransmitida=1 o no=0*/
};
```

Figura 4-1: Estructura de trama

Para ir recogiendo la información referente a un flujo se ha definido una estructura de tipo "flujo", en esta estructura se va guardando la información sobre las diferentes características y estados por los que va pasando el flujo. Esta estructura estará "viva" mientras esté abierta una conexión. La estructura es la siguiente:

```
struct flujo{
        struct flujo *siguiente;        /* Puntero al siguiente flujo*/
        uint32_t ip_o;                  /* Direccion IP Origen que lanza el SYN de apertura (cliente) */
        uint32_t ip_d;                  /* Dirección IP Destino contra quien se lanza el SYN desde el cliente (Servidor)
*/
        uint16_t p_o;                   /* Puerto origen */
        uint16_t p_d;                   /* Puerto destino */
        int identificador;              /* Identificador del flujo */
        int estado;                     /* Estado en el que se encuentra el flujo*/
        int midiendo_RTT;               /* Bandera indicando que se está midiendo el RTT promedio*/
        uint32_t num_sec_next_RTT;      /* Numero de SEC del siguiente paquete*/
        int contador_RTT;               /* Contador RTTs medidos*/
        int tiempo_RTT_promedio;        /* Tiempo promedio acumulado de todos los RTTs del flujo*/
        time_t t_ultimo_RTT;            /* Hora de envio paquete datos CS*/
        time_t t_SYN;                   /* Hora en que se lanza el SYN desde el cliente */
        time_t t_ultima_trama;          /* Hora en que llega la ultima trama del flujo*/
        int num_SYN;                    /* Registro en que se lanza el SYN desde el cliente*/
        int RTT_inicial;                /* Tiempo RTT inicial del flujo*/
        int num_wsize0;                 /* Contador de ventanas 0*/
        uint32_t num_bytes_total;               /* Numero de bytes totales de todas las tramas*/
        uint32_t num_bytes_datos;               /* Numero de bytes totales de datos de todas las tramas*/
        uint32_t num_sec_ultimo;        /* Numero de secuencia del ultimo paquete de datos*/
        int num_retransmisiones;        /* Contador del numero de retransmisiones totales del flujo*/
        int midiendo_stalled;           /* Bandera indicando que se está midiendo el tiempo stalled*/
        time_t t_stalled_anterior;      /* Guarda la hora del ultimo paquete de datos midiendo stalled*/
        int tiempo_stalled;             /* Tiempo stalled del flujo*/
        int num_registro_stalled;       /* Numero de registro del ultimo trama stalled*/
};
```

Figura 4-2: Estructura de flujo

Para poder mantener en memoria todos los flujos “vivos” en un determinado momento, se ha implementado una estructura de nodos enlazados que se van creando y destruyendo de manera dinámica, haciendo uso de las facilidades de asignación de memoria que nos proporciona el sistema. Esta estructura dinámica de nodos se puede representar de la siguiente forma:

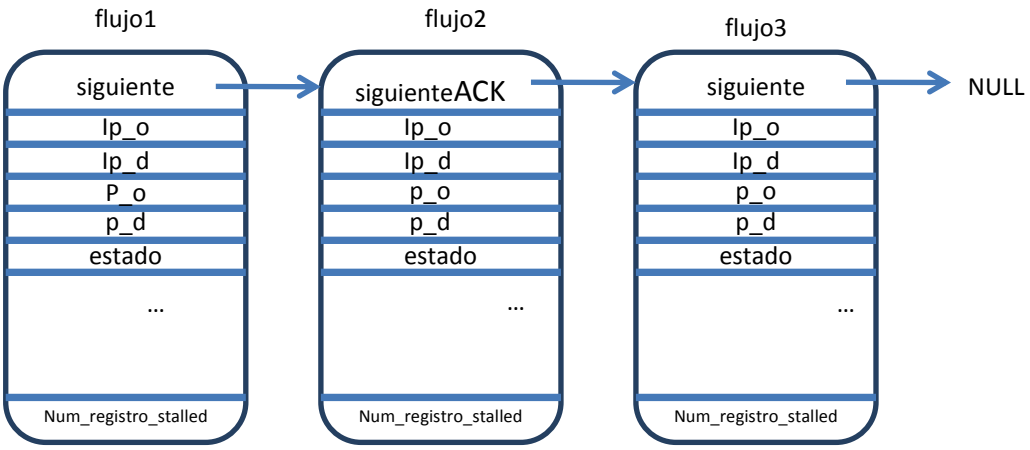


Figura 4-3: Estructura de nodos de flujos

4.2.2 Diseño general

En el siguiente esquema se refleja el proceso que se sigue, en el programa principal, con cada trama que se recibe:

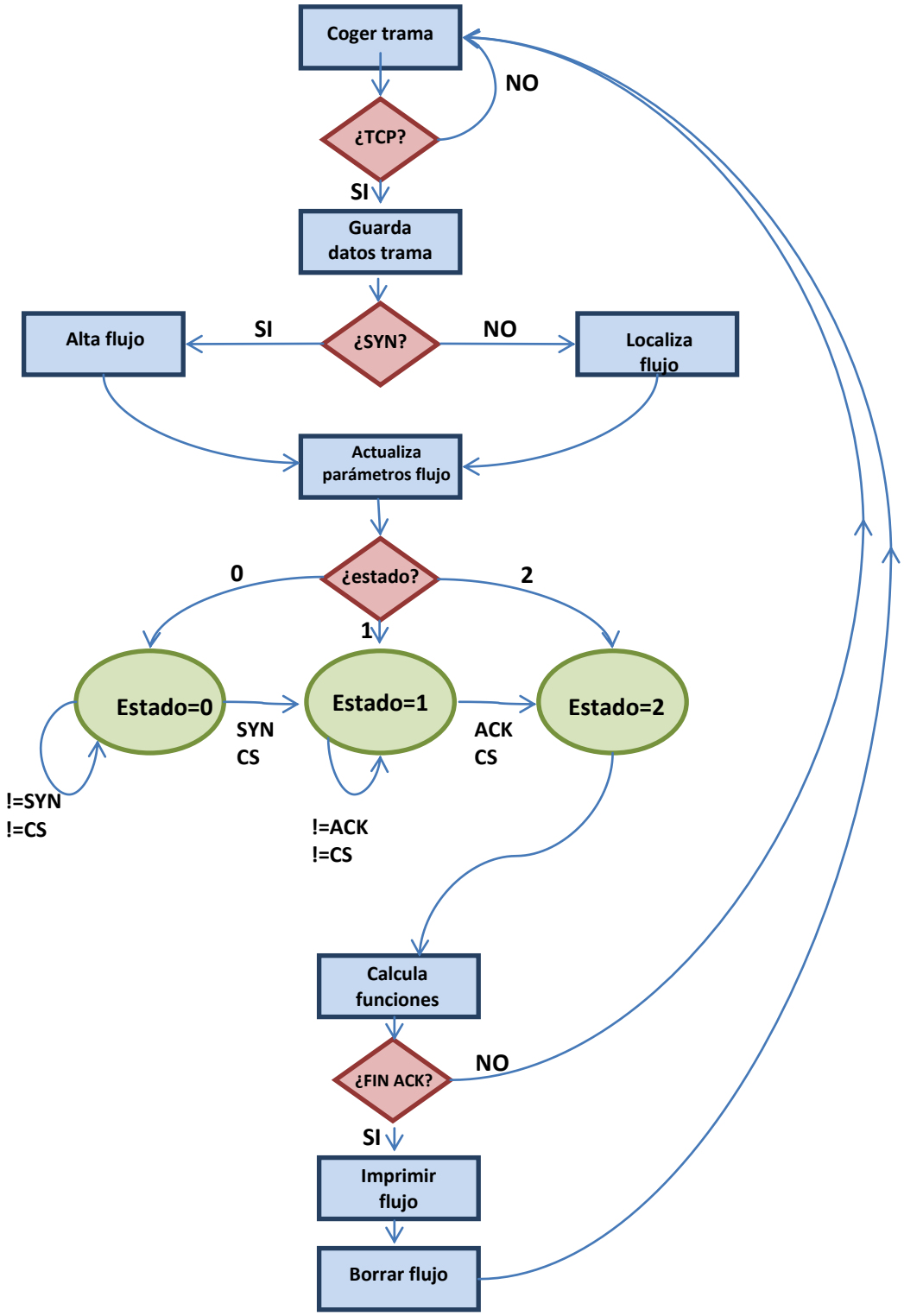


Figura 4-4: Esquema general del procesamiento de cada trama

Cada una de las tramas que va a ser analizada, debe pasar por los filtros de protocolos IP y TCP para ver si es una trama 'útil'. Una vez comprobado que la trama cumple estos filtros, se procede a guardar sus datos en la estructura que se ha explicado anteriormente (struct trama). A continuación se pasará a comprobar, en la estructura trama, el estado de las banderas del paquete, para poder averiguar mediante la bandera SYN si la trama pertenece

a un flujo nuevo, y por lo tanto, proceder a dar de alta dicho flujo. Si la trama pertenece a algún flujo ya creado se pasa a localizarle dentro de las estructuras de nodos de flujos.

Posteriormente, una vez que se ha creado o localizado el flujo, se actualizan los datos de esta estructura de flujo y se pasa a tratar la trama en función del estado en que se encuentre la conexión.

Se ha creado una máquina de estados para poder facilitar el tratamiento de cada trama. La máquina consta de tres estados:

1. Estado 0: Esperando SYN
2. Estado 1: Esperando ACK
3. Estado 3: Transmisión de datos

Si la conexión se encuentra en estado 0, se está esperando recibir un SYN, por lo tanto, se encuentra esperando tramas en sentido de flujo cliente-servidor con la bandera SYN activa. Al recibirla se llamará a la función *trata_syn* que será explicada posteriormente. Cualquier otro trama de este flujo es desechada.

Si la conexión se encuentra en estado 1, ya ha recibido el SYN y se encuentra a la espera de un ACK, por lo tanto trata tramas en sentido de flujo cliente-servidor con la bandera ACK activa. Al recibirlo se llamará a la función *trata_ack* que será explicada posteriormente. Cualquier otro trama es desechada.

Una vez recibido el ACK, se pasa al estado 2 en el cual se tratan los datos. En este estado se van calculando y actualizando los parámetros RTT promedio, número de retransmisiones, tiempo *stalled…* mediante las funciones *calcular_RTT_promedio, calcular_num_retransmisiones* y *calcular_tiempo_stalled*, las cuales serán explicadas más adelante.

En el caso de estar en el estado 2, se debe analizar continuamente las banderas para detectar si están activas de manera simultánea las banderas FIN y ACK, esto indicará que ha finalizado el flujo, y por tanto, se grabarán los datos del flujo y se borrará el nodo de la lista liberando el espacio de memoria utilizado por éste.

4.3 Diseño de funciones

A continuación se describirán cada una de las funciones implementadas para el desarrollo de la herramienta.

Las funciones son:

- filtro_IP_TCP
- saca_parametros_trama
- alta_flujo
- localiza_flujo
- actualizar_parametros_flujo
- borra_flujo
- imprimir_todos_flujos

- imprimir_datos_flujo
- trata_syn
- trata_ack
- calcular_RTT_promedio
- calcular_num_retransmisiones
- alcular_tiempo_stalled

4.3.1.1 ***filtro_IP_TCP***

Parámetros de entrada/salida

Entrada: Puntero al paquete leído del fichero pcap.
Salida: Devuelve 0 en caso de ser una trama TCP/IP o 1 en caso de no serlo.

Descripción

Se accede a la trama para comprobar si se trata de un paquete IP y TCP. Para realizar dicha comprobación se analiza las cabeceras de la trama. En primer lugar se accede a la de Ethernet, para comprobar si el campo "tipo" es 0x0800, lo cual corresponde a paquetes IP. Posteriormente se accede a la cabecera IP para comprobar si el protocolo tiene el valor 6 y por lo tanto se trata de un segmento TCP.

4.3.1.2 ***Saca_parametros_trama***

Parámetros de entrada/salida

Entrada: Puntero al paquete leído del fichero pcap y puntero a la estructura trama.
Salida: Devuelve 0 cuando termina de actualizar los parámetros de la estructura trama con la trama actual.

Descripción

Esta función rellena la estructura trama con los datos del paquete leído del fichero de trazas.

Se accede a la trama del fichero analizado y se van sacando los siguientes parámetros:

- *Tamaño de las cabeceras TCP e IP*, se guardan en las variable *long_cabeceraIP* y *long_cabeceraTCP* de la estructura de la trama.
- Calculo de las banderas actuales del paquete. Serán guardadas en la variable *banderas* de la estructura trama.
- *Longitud total del paquete y longitud de datos del paquete*, para lo cual se restan a la longitud de la trama la longitud de sus cabeceras. Serán guardadas en *longitud* y *longitud_datos* respectivamente. Al realizar estudios de varios casos con distintos ficheros, se ha hecho una excepción al anterior cálculo en el caso de que la cabecera Ethernet contenga *padding*. Se ha comprobado que en el caso de que los paquetes tengan una longitud total de 60, el tamaño de los datos es 0 y estos 6 bytes

pertenecen al campo *padding* de Ethernet, por lo tanto en este caso se guardaría 60 en la longitud total y 0 en la longitud de los datos.

- *Tamaño de ventana*, Contiene el tamaño de la ventana. Se utilizará para analizar el número de 'ventanas 0' de un flujo.
- *Número de secuencia de la trama*, para lo cual se ha tenido que utilizar la función ntohl() para poder transformar el numero de secuencia de formato network a host long. Ser guarda en la variable num_sec.
- *Número de ack* de la trama, para lo cual se ha empleado también la función ntohl(). Se guarda en la variable num_ack.
- *Número de la secuencia siguiente*, sumando el número de la secuencia actual con la longitud de datos de la trama. Se corresponde con el siguiente número de secuencia que se enviará en este sentido del flujo, además también coincidirá con el 'Número de ack' del paquete de datos (en sentido de flujo contrario) que confirme la correcta recepción de esta trama. Se guarda en la variable num_sec_next.
- *El tiempo de llegada del paquete*, tanto en formato time_t para poder operar, como en formato int para posteriormente poder mostrar el dato directamente. Se guardarán en las variables *tiempo* y *hora* respectivamente.
- *Direcciones IP y puertos de la trama*. Se guardarán en las *variable ip_o, ip_d, p_o, p_d* correspondiendo a IP origen, IP destino, puerto origen, puerto destino.

En esta función se inicializa el campo *trama retransmitida* a 0.

4.3.1.3 ***Alta_flujo***

Parametros de entrada/salida

Entrada: Puntero a una estructura trama, y un puntero a una lista de flujos.
Salida: Devuelve un puntero a la nueva estructura flujo que se ha creado.

Descripción

En esta función se da de alta un flujo nuevo, haciendo que éste quede añadido a la lista de flujos. Se inicializan los parámetros de este flujo con los datos de la trama que se está analizando, los parámetros del flujo que no se pueden rellenar con la información de la trama se inicializan a 0.

Para la implementación de esta función, en primer lugar se recorre la lista de flujos hasta encontrar el último, una vez encontrado el último se hace que el campo *siguiente* del flujo apunte a una dirección nueva de memoria que se va a reservar mediante la función malloc:

*lista_flujo_aux->siguiente = (struct flujo *) malloc (sizeof(struct flujo));*

Una vez tengamos el nuevo flujo, se rellenan sus campos con los datos de la trama y se inicializan aquellos de los que aún no se tenga información.

Por último, se hace que el nuevo flujo creado apunte mediante su campo *siguiente* a NULL.

4.3.1.4 ***Localiza_flujo***

Parámetros de entrada/salida

Entrada: Puntero al paquete analizado del fichero y al flujo actual.
Salida: Devuelve NULL en caso de no localizar la trama en la lista de flujos y un puntero al flujo buscado en caso de localizar la trama en la lista de flujos.

Descripción

Esta función trata de localizar, dentro de la lista de flujos, el flujo al que corresponde la trama de entrada. Para lo cual se van comparando las direcciones IP y puertos de la trama con los de cada flujo de la lista de flujos. La comparación se realiza en ambos sentidos, servidor-cliente y cliente-servidor.

Si se encuentra el flujo, se devuelve un puntero a la estructura flujo encontrada, si no, se devuelve un puntero NULL.

4.3.1.5 ***Actualizar_parametros_flujo***

Parámetros de entrada/salida

Entrada: Puntero a una estructura trama, y puntero al flujo actual.
Salida: No tiene parámetros de salida

Descripción

Se van a actualizar algunos parámetros del flujo actual localizado o creado anteriormente con los parámetros de la trama analizada.

En primer lugar se actualiza la variable *sentido_flujo,* para lo cual se comparan las direcciones IP y los puertos de la trama actual con los del flujo. En el caso de que las direcciones IP y los puertos coincidan "origen con origen" y "destino con destino" será tomado como flujo cliente-servidor. En el caso de coincidir direcciones IP y puertos de forma "origen con destino" será tomado como flujo servidor-cliente.

A continuación, se comprueba si el tamaño de ventana de la trama actual es 0 para actualizar el parámetro *num_wsize0* del flujo que contiene el sumatorio de ventanas cero del flujo.

Si es sentido es cliente-servidor se recogen la longitud total de la trama, la longitud de los datos de la trama y el tiempo de llegada de la trama para posteriormente poder hacer los cálculos del throughput y goodput. Para esto se actualizarán las variables *num_bytes_total*, *num_bytes_datos* para las longitudes, *t_ultima_trama* y *t_ultima* para la hora de llegada, guardándolo en formato time_t y en formato int.

4.3.1.6 ***Borra_flujo***

Parámetros de entrada/salida

Entrada: Puntero a la lista de flujos y puntero al flujo que se quiere eliminar.
Salida: Devuelve 0 en el caso de borra el flujo correctamente.

Descripción

Se va a borrar de la lista de flujos un flujo que ha finalizado.

En primer lugar se comprueba que el flujo que se desea borrar no es el primero, es decir, que los punteros de entrada, lista de flujos y flujo a eliminar, no sean iguales. Si fuese el primer flujo de la lista, no se podría borrar el puntero a este flujo ya que lo hemos tomado como referencia, en este caso se pondrían sus direcciones IP y puertos a 0 como indicación de flujo finalizado.

Vamos comparando cada uno de los flujos de la lista con el flujo que queremos borrar. Cuando encontramos el flujo, se hace que el flujo anterior al que se quiere borrar apunte al siguiente de éste. Por último se libera la memoria del flujo que se ha borrado mediante la función free().

4.3.1.7 ***Imprimir_datos_flujo***

Parámetros de entrada/salida

Entrada: Puntero a la estructura flujo que se va a imprimir.
Salida: No tiene parámetros de salida

Descripción

Los datos se grabarán en un fichero de salida. Los parámetros que se han decidido guardar por su relevancia para el estudio de calidad son los siguientes:

- Tiempo de la última trama del flujo.
- Direcciones IP y puertos (origen y destino) del flujo.
- RTT inicial, guardado en la variable *RTT_inicial* de la estructura del flujo.
- RTT promedio, para lo cual se dividen las variables *tiempo_RTT_promedio* y *contador_RTT* guardadas en el flujo.
- Número de ventanas 0, guardado en la variable `num_wsize0` de la estructura del flujo.
- Número de retransmisiones del flujo, guardado en la variable *num_retransmisiones* de la estructura del flujo.
- Tiempo stalled del flujo, guardado en la variable *tiempo_stalled* de la estructura del flujo.
- Número de bytes total, guardado en la variable *num_bytes_total* de la estructura del flujo.

- Throughput del flujo, para lo cual se divide el número de bytes totales del flujo entre la duración, siendo esta la diferencia entre el tiempo de la ultima trama del flujo menos el tiempo de llegada de la primera trama.
- Número de bytes de datos, guardado en la variable *num_bytes_datos* de la estructura del flujo.
- Goodput del flujo, para lo cual se divide el número de bytes de datos totales del flujo entre la duración, siendo ésta la diferencia entre el tiempo de la ultima trama del flujo menos el tiempo de llegada de la primera trama.

4.3.1.8 ***Tratar_syn***

Parámetros de entrada/salida

Entrada: Puntero a la estructura trama, y puntero al flujo actual.
Salida: No tiene parámetros de salida

Descripción

Se guarda el tiempo de llegada del SYN y se cambia del estado 0, esperando SYN, al estado 1, esperando ACK.

4.3.1.9 ***Tratar_ack***

Parámetros de entrada/salida

Entrada: Puntero a la estructura trama, y puntero al flujo actual.
Salida: No tiene parámetros de salida

Descripción

En esta función se calcula el RTT inicial y se cambia del estado 1, esperando ACK, al estado 2, transmisión de datos.

Para el cálculo del RTT inicial se va a restar el tiempo de llegada del ACK menos el tiempo de llegada del SYN del flujo, este cálculo se va a guardar en la variable *RTT_inicial* del flujo.

4.3.1.10 ***Calcular_RTT_promedio***

Parámetros de entrada/salida

Entrada: Puntero a la estructura trama, y puntero al flujo actual
Salida: No tiene parámetros de salida

Descripción

Se va a calcular el RTT promedio de un flujo. Para ello se va a medir como, la media de los tiempos entre que una trama de datos es enviada desde el cliente, hasta que se recibe su asentimiento confirmando la recepción del mismo por parte del servidor.

Para la implementación de la función, se han contemplado los siguientes casos:

1. La trama es del flujo en sentido cliente-servidor y contiene datos. Si esto se cumple se actualizan los siguientes campos de la estructura flujo:

   - *midiendo_RTT*: Se activa la bandera a 1.
   - *num_sec_next_RTT*: Se igualará al campo num_sec_siguiente de la trama, el cual fue rellenado y actualizado anteriormente
   - *t_ultimo_RTT*: Se igualará al campo tiempo de la trama, rellenado anteriormente.

2. La trama es del flujo en sentido servidor-cliente, está activada la bandera *midiendo_RTT* y el número de ack de la trama es igual al número de secuencia siguiente esperado guardado en la variable *num_sec_next_RTT* del flujo. Si esto se cumple se actualizan los siguientes campos de la estructura flujo:
   - *contador_RTT*: Se incrementa en 1 el contador de RTTs
   - *tiempo_RTT_promedio:* Se actualiza el tiempo de RTT promedio guardado en el flujo sumándole la diferencia entre el tiempo de la trama que se está analizando menos el tiempo del que se guardó en la estructura flujo *t_ultimo_RTT*
   - *midiendo_RTT*: Se desactiva la bandera a 0.

3. La trama es del flujo en sentido servidor-cliente, está activada la bandera *midiendo_RTT* y el número de ack de la trama es distinto al número de secuencia siguiente esperado guardado en la variable *num_sec_next_RTT* del flujo. Si esto se cumple se actualizan los siguientes campos de la estructura flujo:
   - *midiendo_RTT*: Se desactiva la bandera a 0.

4.3.1.11 ***Calcular_num_retransmisiones***

Parámetros de entrada/salida

Entrada: Puntero a la estructura trama, y puntero al flujo actual
Salida: No tiene parámetros de salida

Descripción

En esta función se detecta si la trama actual ha sido una retransmisión y se calcula el número de retransmisiones totales en un flujo.

Si la trama analizada es del flujo en sentido servidor-cliente y contiene datos, se comprueba si el número de secuencia de la trama actual es mayor que el numero de secuencia de la trama anterior guardada en la estructura flujo en la variable *num_sec_ultimo*.
En caso de ser mayor, se actualiza la variable *num_sec_ultimo* con el nuevo número de secuencia.

En caso de ser menor, se estaría dando una retransmisión, se incrementaría en 1 el número de retransmisiones guardado en la estructura flujo en la variable *num_retransmisiones* y se indica en la estructura trama que es retransmitida mediante la bandera *trama_retransmitida*. Además se resta la longitud de los datos para no tenerlo en cuenta en el cálculo del goodput.

4.3.1.12 ***Calcular_tiempo_stalled***

Parámetros de entrada/salida

Entrada: Puntero a la estructura trama, y puntero al flujo actual
Salida: No tiene parámetros de salida

Descripción

En esta función se calcula el tiempo stalled de un flujo, siendo éste el tiempo máximo que hay entre dos paquetes de datos en un flujo sentido servidor-cliente, dentro de una respuesta que el servidor le envía al cliente, teniendo en cuenta que durante ese plazo no se debe dar la circunstancia de 'ventana_zero' por parte del cliente.

Para la implementación de la función, se han contemplado los siguientes casos:

1. La trama es del sentido flujo servidor-cliente, contiene datos y no se está midiendo aún el tiempo *stalled*. Si esto se cumple se actualizan los siguientes campos de la estructura flujo:
   - *midiendo_stalled*: Se activa la bandera a 1.
   - *t_stalled_anterior:* Se guarda en la variable del flujo el tiempo de la trama actual.

2. La trama es del sentido flujo servidor cliente, contiene datos, se está midiendo el tiempo *stalled* y no es una trama retransmitida. Si esto se cumple, se actualizan los siguientes campos de la estructura flujo:
   - *tiempo_stalled:* Se actualiza el tiempo stalled de la estructura flujo, sustituyéndolo por el tiempo stalled actual que será la resta del tiempo de la trama actual menos el tiempo guardado en el flujo *t_stalled_anterior*.

     Solo se actualizará en el caso de que el tiempo stalled nuevo calculado sea mayor que el que ya estaba guardado en la variable.
     Además se guarda en la variable *num_registro_stalled* de la estructura flujo el número de registro de esta trama para poder saber a qué trama pertenece el tiempo stalled final.
     Por último se guarda el tiempo de esta trama en la estructura flujo en la variable *t_stalled_anterior.*

     En el caso de que el tiempo stalled nuevo calculado sea menor que el que ya estaba guardado en la variable se actualiza este tiempo en la variable *t_stalled_anterior* de la estructura flujo.

3. La trama es del sentido flujo cliente servidor y contiene datos. Si esto se cumple se actualizan los siguientes campos de la estructura flujo:
   - *midiendo_stalled*: Se desactiva la bandera a 0.

4. La trama es del sentido flujo cliente servidor y su tamaño de ventana es igual a 0. Si esto se cumple se actualizan los sigcuientes campos de la estructura flujo:
    - *midiendo_stalled*: Se desactiva la bandera a 0.

4.4 Programas adicionales

Las pruebas finales de la herramienta se han realizado sobre el tráfico extraído a la salida de los laboratorios de la Escuela Politécnica Superior entre las fechas 29 de Septiembre de 2014 y 29 de Octubre de 2014.

Este tráfico está recogido en 1.290 ficheros .pcap. Para la extracción de los datos de cada uno de los ficheros se ha implementados una serie de procesos divididos en tres fases:

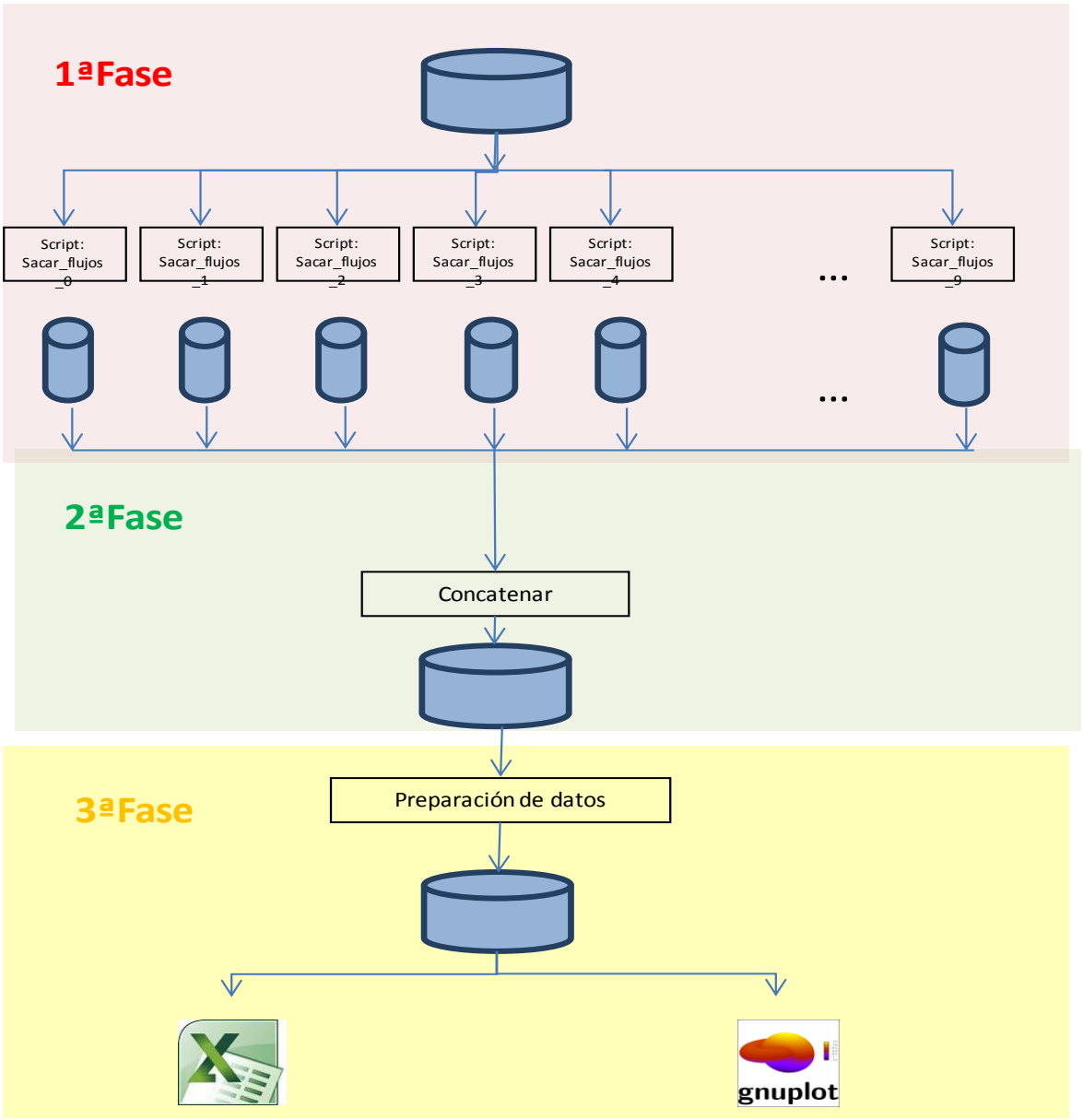


Figura 4-5: Fases de preparación de datos

1ª Fase: Extracción de datos básicos de los ficheros de traza. En esta fase se obtendrá un fichero con los datos de cada uno de los ficheros de trazas que se encuentran en el directorio de trazas a analizar. Para el procesado mediante la herramienta de los 1.290 ficheros se hicieron varias pruebas de ejecución. En los resultados de éstas se pudo comprobar que la forma más eficaz de procesar los ficheros es mediante la ejecución de varios flujos en paralelo, de esta forma se procesarán más rápido los 1290 ficheros. Las pruebas dieron los siguientes resultados:

Tabla 4-1 : Tiempos de ejecución del programa

PROCESO	TIEMPO
1 proceso	20h 30min
2 procesos en paralelo	11h 30 min
5 procesos en paralelo	6h 40min
10 procesos en paralelo	5h 50min

Hay que tener en cuenta que, dentro del análisis de cada fichero, los datos referidos a aquellos flujos que no finalizan se desprecian.

Se ha escogido la opción de 10 procesos en paralelo ya que es el que mostraba mejores resultados en menor tiempo, consiguiendo hacer un análisis del tráfico completo del mes en 5 horas y 50 minutos. La ejecución de más de 10 procesos en paralelo no daba mejoras notables en el tiempo, y además, presentaba problemas de ejecución en algunas ocasiones.

Los ficheros '.pcap' están nombrados según la forma "file_bond0_xxxxxx" donde xxxxxx corresponde a al número de identificación del fichero. Se han creado 10 script para poder procesar todos los ficheros en paralelo. Cada uno de estos script accede al directorio donde están guardados todos los .pcaps y los va tomando un script u otro, dependiendo de la última cifra de su número de identificación. Por ello, se lanzarán de forma paralela 10 script con un proceso independiente en cada uno.

- Script "Sacar_flujos_0" -> Procesará todos los procesos acabados en 0.pcap
- Script "Sacar_flujos_1" -> Procesará todos los procesos acabados en 1.pcap
- Script "Sacar_flujos_2" -> Procesará todos los procesos acabados en 2.pcap
- Script "Sacar_flujos_3" -> Procesará todos los procesos acabados en 3.pcap
- Script "Sacar_flujos_4" -> Procesará todos los procesos acabados en 4.pcap
- Script "Sacar_flujos_5" -> Procesará todos los procesos acabados en 5.pcap
- Script "Sacar_flujos_6" -> Procesará todos los procesos acabados en 6.pcap
- Script "Sacar_flujos_7" -> Procesará todos los procesos acabados en 7.pcap
- Script "Sacar_flujos_8" -> Procesará todos los procesos acabados en 8.pcap
- Script "Sacar_flujos_9" -> Procesará todos los procesos acabados en 9.pcap

La salida de cada uno de los ficheros se va a ir guardando en una carpeta común.

Al ejecutar los script, como se ha visto en el apartado 4.3.1.7 Imprimir_Datos_flujo, para cada uno de los ficheros .pcap se obtiene un fichero .txt de salida con un registro por cada flujo, con los siguientes campos por registro separados por espacios:

- Tiempo de la última trama del flujo en formato *epoch*

- Direcciones IP y puertos (origen y destino) del flujo.
- RTT inicial.
- RTT promedio.
- Numero de ventanas 0.
- Numero de retransmisiones del flujo.
- Tiempo stalled del flujo.
- Numero de bytes total.
- Througthput del flujo.
- Numero de bytes de datos.
- Goodput del flujo.
- Duración

2ª Fase: Concatenación de datos. Una vez han sido procesados todos los ficheros con la herramienta y ha sido guardada su salida en una carpeta común, se van a ir tomando estos ficheros .txt para concatenar todos sus datos, y así tener un fichero único con los datos de todos los ficheros .txt.

La implementación de esta fase se realiza mediante un script que toma cada fichero de la carpeta común y lo va concatenando uno a continuación de otro.

3ª Fase: Preparación de datos. Para posteriormente poder analizar los datos, en esta fase se van a completar cada una de las líneas del fichero único de salida. A cada línea, correspondiente a un flujo, se le concatenan dos columnas adicionales:

- Una primera columna con la hora en formato legible. Como ya se ha explicado, una de las columnas de los flujos posee la hora del último paquete del flujo en formato *epoch*. Mediante esta función se obtendrá la hora en formato: Día de la semana, mes, día, hora, minuto y año. De esta forma, cada uno de los parámetros se podrán graficar en función del tiempo en este formato.
- La segunda columna concatenada corresponde al nombre de la dirección IP del servidor de cada flujo. Una vez se tienen todas las direcciones IP de los servidores en cada una de las líneas de los flujos, mediante esta función se va a comprobar a que nombre de IP corresponden. El motivo principal de calcular este parámetro fuera del programa principal es la disminución del tiempo de ejecución; La función creada va guardando en una tabla cada uno de los nombres correspondientes a las IP de los servidores, de esta forma, solo es necesario comprobar el nombre de los diferentes servidores una vez en todo el mes.

La implementación de esta última fase se realiza mediante un script que coja cada línea del fichero único creado anteriormente y le concatene su hora en formato normal y el nombre de la IP del servidor.

Una vez se tienen los datos en un fichero único y con el formato deseado se va a proceder a extracción de gráficas.

4.5 Conclusión

En este capítulo se ha explicado el desarrollo completo de la herramienta para el procesamiento de los datos.

Para el proyecto los datos usados han sido un conjunto de 1290 ficheros .pcap. Para su procesamiento completo se han implementado varias fases de proceso.

Una primera fase donde se ejecuta el programa principal, estructuras y funciones que se han expuesto anteriormente. Y, una segunda fase donde se van a estructurar concatenar y preparar los datos obtenidos del programa principal.

En todo momento el objetivo del desarrollo ha sido poder conseguir una herramienta que permita, en el menor tiempo posible, extraer los principales parámetros de los flujos analizados

Una vez desarrollada la herramienta, se procedió a la realización de pruebas con diferentes ficheros de tráfico .pcap para poder hacer un análisis de la calidad de servicio. Los resultados de estas pruebas se muestran en el capítulo siguiente.

5 Integración, pruebas y resultados

Una vez se han procesado y preparado los datos se ha procedido al análisis mediante gráficas de los mismos. Como se ha comentado en el capítulo anterior las pruebas se han realizado sobre el tráfico capturado en la salida de los laboratorios de la Escuela Politécnica Superior entre las fechas 29 de Septiembre y 29 de Octubre de 2014. Este proyecto se centra en en los servicios OTT relativos a TV, por tanto, se han analizado únicamente los flujos con un tamaño de datos superior a 1 MByte, para descartar así el tráfico web, que normalmente está asociado con flujos de un tamaño menor.

Se va a realizar el análisis sobre dos grupos de gráficas: gráficas temporales y gráficas por IP del servidor.

A continuación se exponen los resultados obtenidos

5.1 Gráficas temporales

Todo este grupo de gráficas tiene en común el análisis de diferentes parámetros en el espacio temporal de captura anteriormente reseñado.

Para el estudio temporal se ha realizado el análisis de los parámetros mediante gráficas de promedios, sumatorios, varianzas y dispersiones.

5.1.1 Promedios, sumatorios y varianzas.

Como herramienta para hacer el análisis y sacar las gráficas se ha utilizado el producto Microsoft Excel 2013. Haciendo uso de las facilidades que da esta herramienta para crear tablas dinámicas y filtros por columnas, así como la de columnas calculadas se ha podido realizar un análisis de datos casi en línea consiguiendo profundizar e investigar en ciertos flujos particulares. Como ficheros de datos de entrada de esta herramienta se ha utilizado el fichero de datos obtenido en la 3ª fase del proceso.

El estudio de los promedios se ha realizado para los parámetros que implican tiempo, como son, la duración media de un flujo, el tiempo de RTT, *throughput*, *goodput* y tiempo *stalled*. En las gráficas se representan las medias de cada parámetro por cada hora durante todos los días del mes.

Para el estudio de anuncios de ventana cero y retransmisiones de paquetes se ha usado el sumatorio de cada parámetro por cada hora durante todos los días del mes.

5.1.1.1 ***Promedio de la duración del flujo***

La figura 5-1 recoge el promedio de la duración de los flujos. Analizando esta gráfica se puede observar que durante los días 18, 19 y 20 de Octubre de 2014 aumentó mucho la media de duración de los flujos alcanzando, su máximo el día 19 de Octubre a las 10:00h.

Haciendo un análisis más profundo sobre lo ocurrido esos días se observa que, al ser fin de semana y haber un conjunto de flujos que duraron mucho, estos influyeron en la media dando un valor muy alto. Al hacer un análisis más preciso, filtrando la tabla Excel por día y duración máxima de flujo, se obtiene que, el flujo con mayor duración durante estos días fue contra un servidor que pertenece al dominio una importante compañía telefónica.

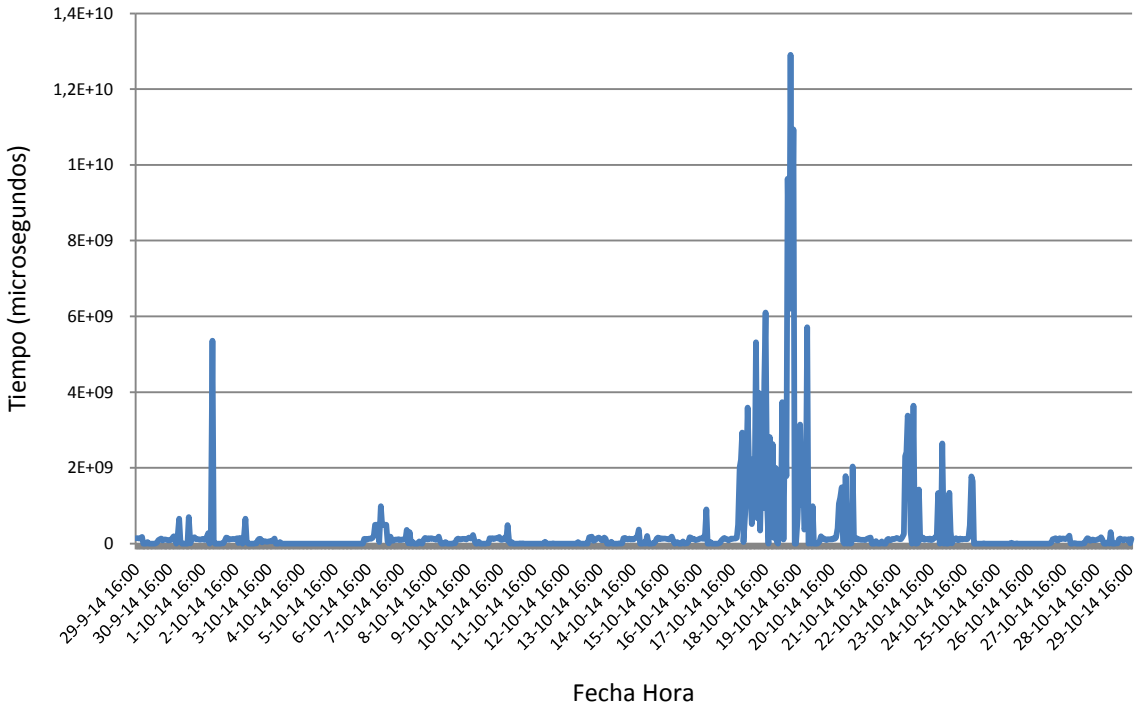


Figura 5-1: Promedio duración flujo

5.1.1.2 *Throughput y Goodput*

En la figura 5-2 se recoge el *throughput* para cada hora del mes analizado. Del análisis de este parámetro se deduce la cantidad media de datos (bits por segundo) que reciben los clientes de los flujos. Hay que tener en cuenta que se han filtrado los flujos que tienen un tráfico inferior a 1 megabyte, para así solo analizar el tráfico con mayor volumen de toda la muestra.

Como era de esperar la gráfica refleja un descenso de tráfico los fines de semana.

Analizando igualmente el perfil de tráfico diario se observa que, todos los días (incluyendo algunos fines de semana), existe una punta de tráfico cercana a 80 Mbps entre servidores de la UAM a las 00, esta se puede apreciar en la gráfica 5-2 como el pico diario.

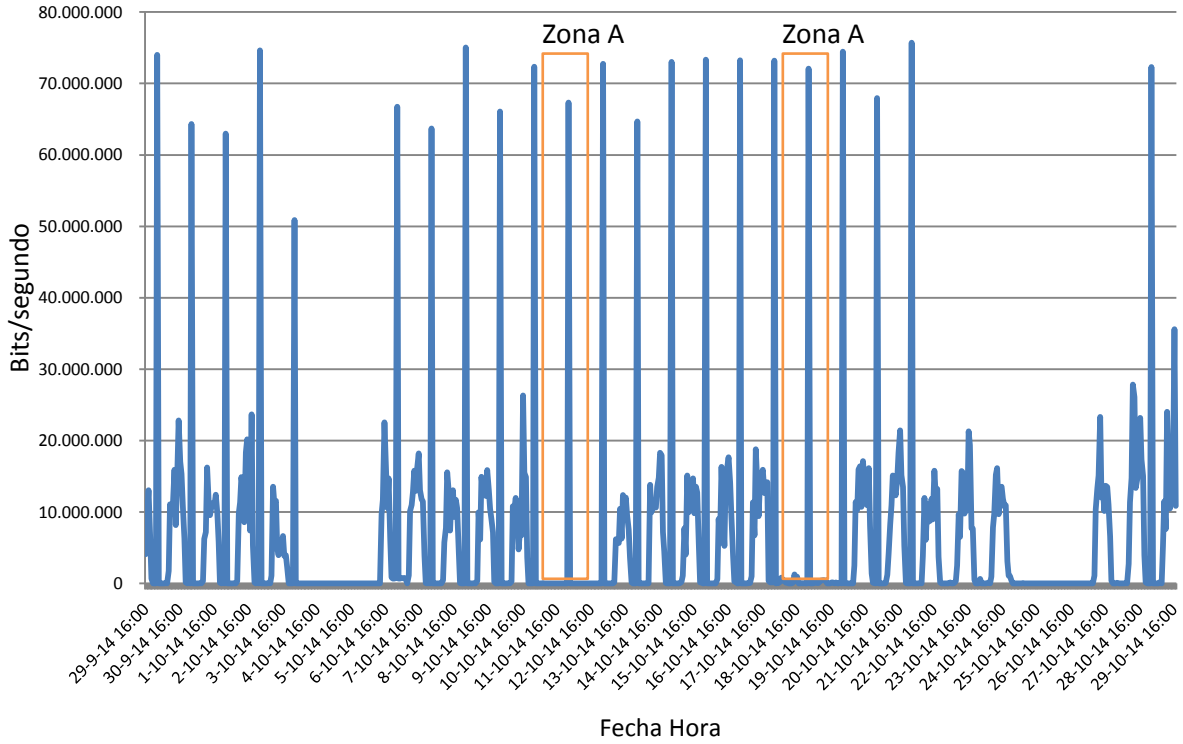


Figura 5-2: Throughput

De la misma forma se analiza el *goodput* (figura 5-3), el cual incluye únicamente los bytes de datos de información útil, despreciando las cabeceras.

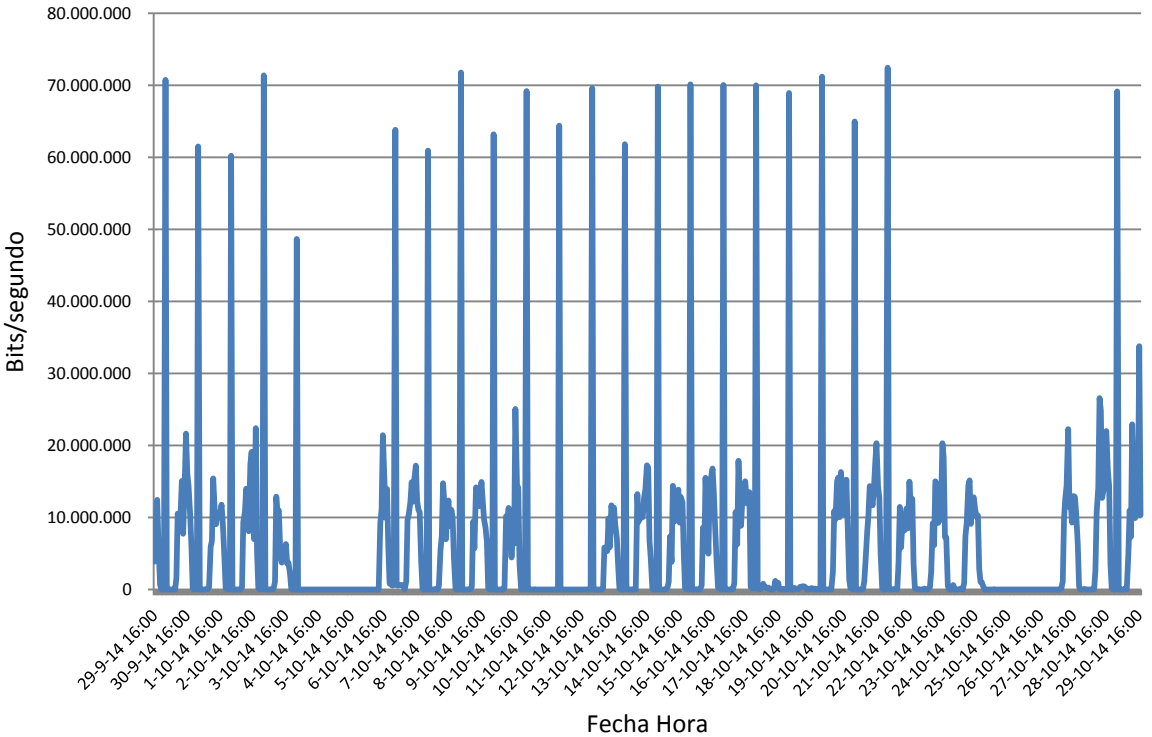


Figura 5-3: Goodput

La gráfica, es similar a la analizada en el *throughput*, pero con un volumen de bytes menor. Se aprecia de nuevo el descenso de volumen de datos los fines de semana, por el mismo motivo que en el caso del *throughput*.

5.1.1.3 ***Promedio de los tiempos de RTT***

A continuación se van a analizar los tiempos de RTT inicial y RTT promedio de los flujos.

Las gráficas de la figuras 5-4 y 5-5 recogen el RTT inicial y promedio del intervalo analizado. En ambas gráficas se observa un aumento del tiempo de RTT en los días del tercer fin de semana y la cuarta semana de Octubre.

El aumento del RTT inicial y promedio del tercer fin de semana de Octubre, vuelve a coincidir con el aumento del promedio de duración de flujo por hora, y al igual que en el caso de la duración de flujos se debe a la influencia, que sobre la media, tuvieron una serie de transmisiones que se dieron en el fin de semana.

Por otro lado, al hacer el análisis con precisión del aumento de tiempo de RTT durante la cuarta semana de Octubre, se obtiene que algunas de las direcciones IP de los servidores a los que pertenecen los tiempos altos de RTT son webs de distribución de vídeo como Justin.tv o Youtube o de almacenamiento de datos como Dropbox.

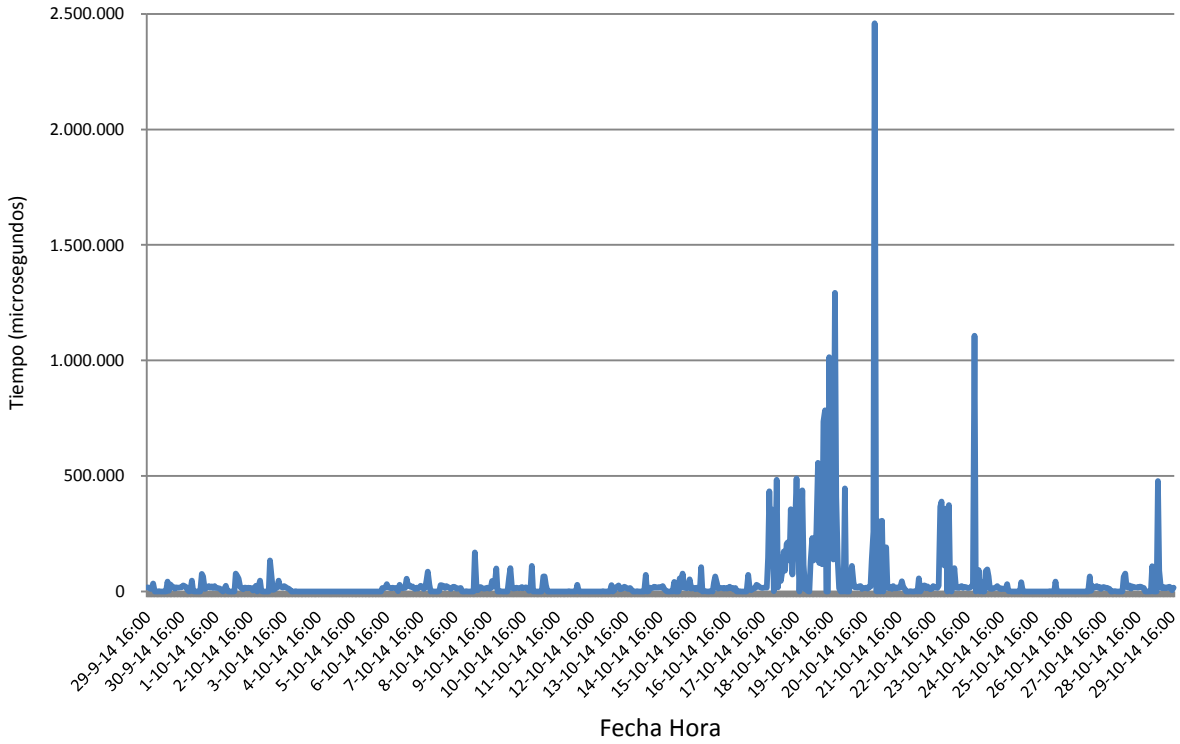


Figura 5-4: Promedio RTT inicial

Del estudio de este parámetro se puede deducir que, tener un valor elevado de tiempo RTT puede deberse a posibles problemas en el envió de datos de los servidores. También puede deberse a que el problema esté en el medio de transmisión y que el servidor esté enviando

los datos correctamente. Incluso puede deberse a que los servidores estén muy lejanos y los datos tarden en viajar. En ambos casos se empeora el rendimiento general del sistema, la velocidad de la entrega de datos y por tanto la calidad de servicio ofrecido.

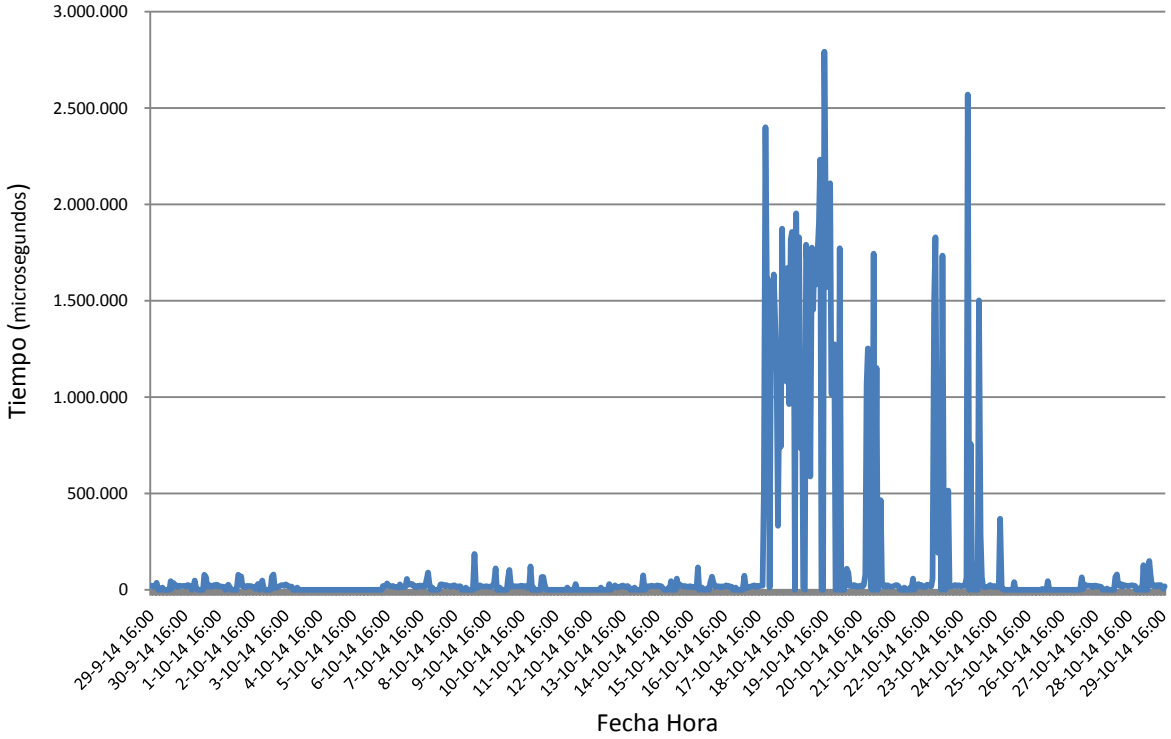


Figura 5-5: Promedio RTT de cada flujo

En la figura 5-6, donde se recoge en la misma gráfica el RTT inicial y promedio, se puede apreciar que en la mayoría de los casos el RTT promedio es superior al RTT inicial. La relación entre el RTT inicial y promedio se analizará más adelante mediante gráficas de dispersión.

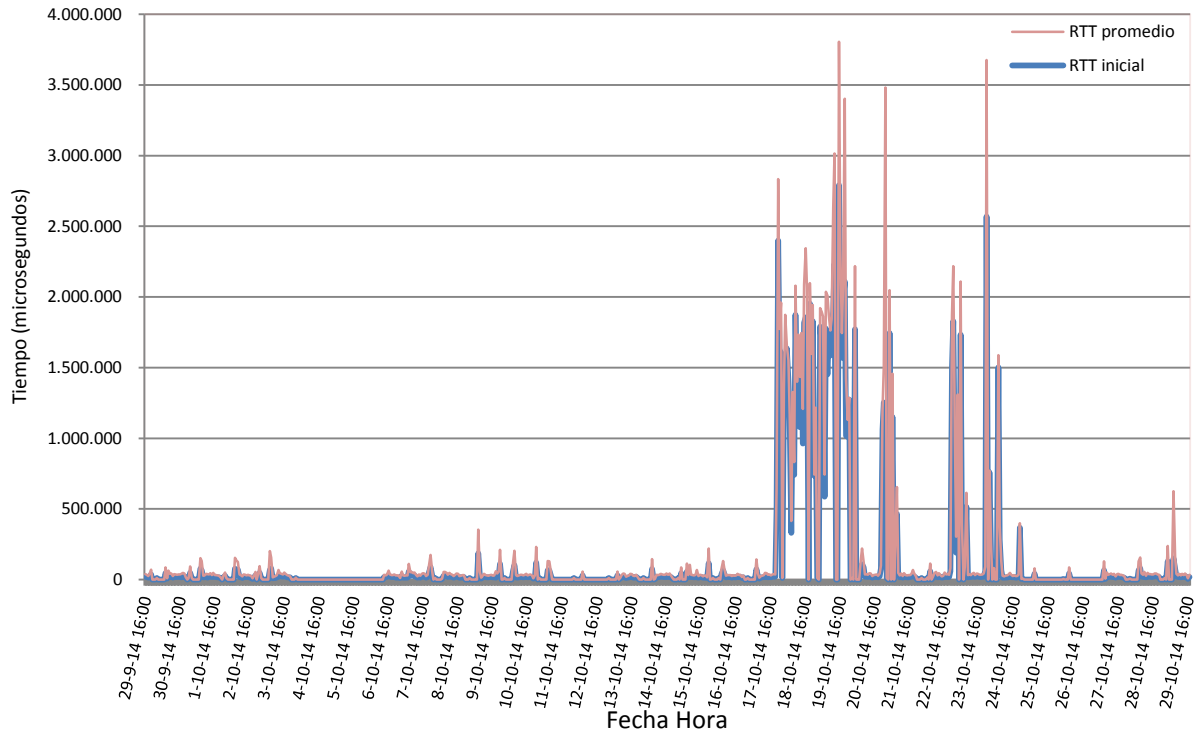


Figura 5-6: Promedio RTT inicial vs RTT promedio

5.1.1.4 *Sumatorio de anuncios de ventana cero*

En la figura 5-7 se representa el número de situaciones de ‘ventanas cero’ por hora durante el periodo de medición. Mediante el estudio de ventanas cero se puede analizar el estado del cliente. En una situación de ‘ventana cero’ el cliente se encuentra congestionado y por tanto hace que se detenga el envío de datos del servidor, generando retardos en el envío de paquetes.

Un valor alto de este parámetro indica problemas de congestión en la parte del cliente, ocasionando peores tiempos de envió de datos y por tanto peor calidad de servicio.

5.1.1.5 *Sumatorio de Retransmisiones*

En la figura 5-8 se representa el número de retransmisiones por hora durante el periodo de medición. Mediante el estudio de retrasmisiones de paquetes por parte del servidor, también es posible detectar problemas que implican una mala calidad de servicio.

El origen de estas retransmisiones puede estar en problemas en los medios de transmisión existentes entre el cliente y el servidor (líneas, routers, firewall,…), Un ejemplo podría ser el descarte de paquetes, por parte de un router, cuando no tenga ancho de banda suficiente de trasmisión.

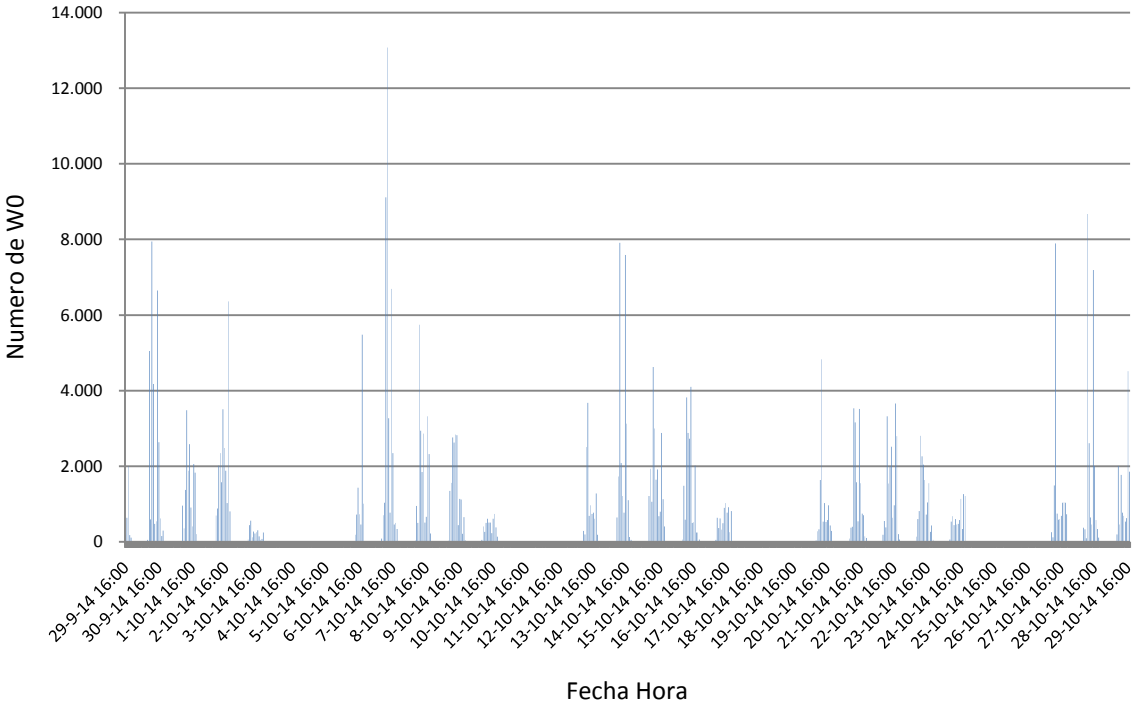


Figura 5-7: Sumatorio Zero Windows

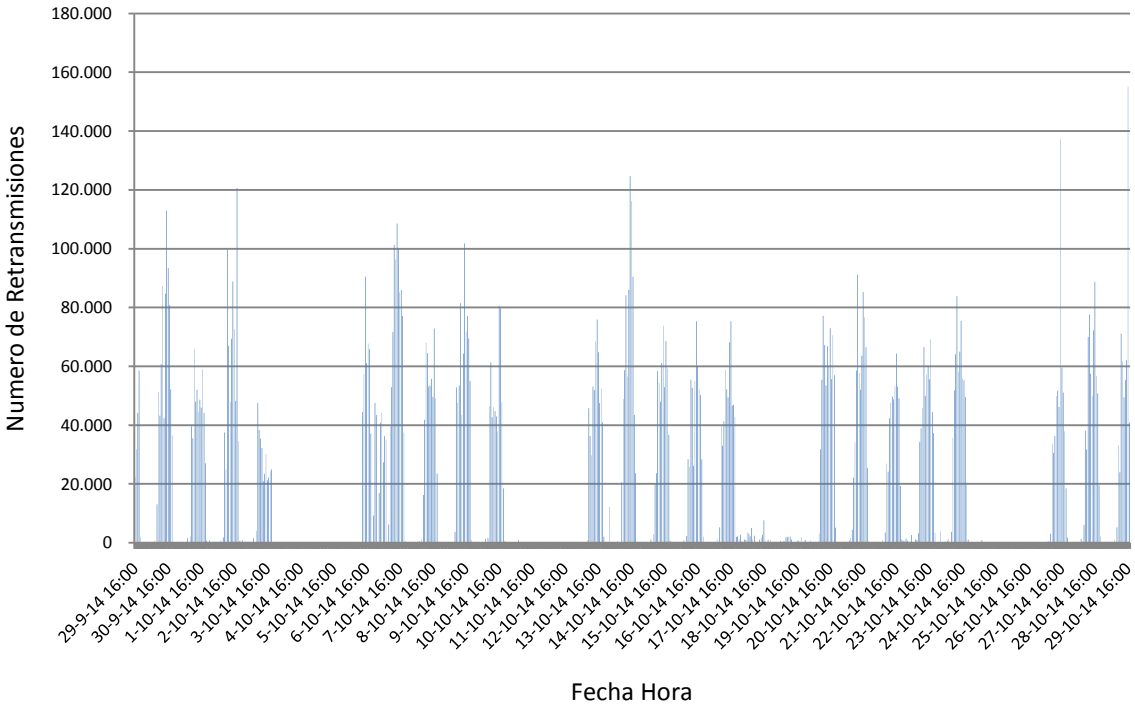


Figura 5-8: Sumatorio Retransmisiones

5.1.1.6 ***Promedio de Tiempo Stalled***

En la figura 5-9 se representa la media del tiempo *Stalled* de todos los flujos de cada hora del intervalo de tiempo medido.

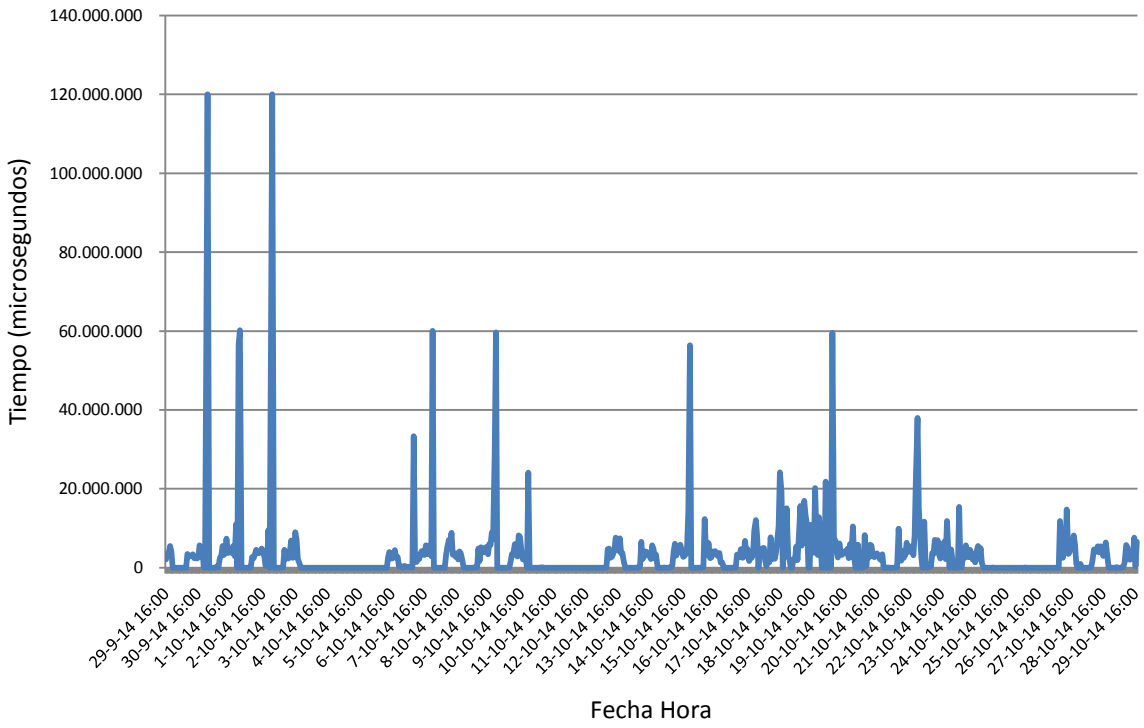


Figura 5-9: Promedio tiempo stalled

Con el estudio del tiempo *stalled* se pueden analizar los problemas de "atasco" en el servidor. En este caso el cliente está a la espera de recibir datos pero el servidor no puede mandarlos.

Un tiempo *stalled* alto implica una peor calidad de servicio debido a peores tiempos de entrega de datos al cliente.

Como se puede apreciar, en el fin de semana del 18 y 19 de Octubre se produjeron valores altos de tiempo *stalled*. La explicación de estos altos tiempos es la misma que se dio en el apartado de duración de flujos (5.1.1.1) y RTT (5.1.1.3), esto es, la media de cada hora se ha visto muy influenciada por los valores muy altos de unos pocos flujos.

5.1.2 Dispersiones

A continuación se va a hacer un estudio de dispersión para cada uno de los parámetros analizados. Este análisis es interesante para poder ver como se distribuye una determinada variable y sacar posibles interrelaciones entre las variables analizadas. Las gráficas representan, para cada parámetro, el comportamiento de cada flujo durante todo el mes.

Estas gráficas se sacaron utilizando como herramienta el producto gnuplot [18] y como ficheros de entrada se utilizaron los que se obtuvieron como salida de la 3º fase del proceso de tratamiento descrito anteriormente.

5.1.2.1 *Dispersión de la duración*

En la figura 5-10 se representa la duración de cada uno de los flujos durante todo el mes. La duración media de los flujos es de 113 segundos, lo cual se puede observar en la gráfica adjunta.

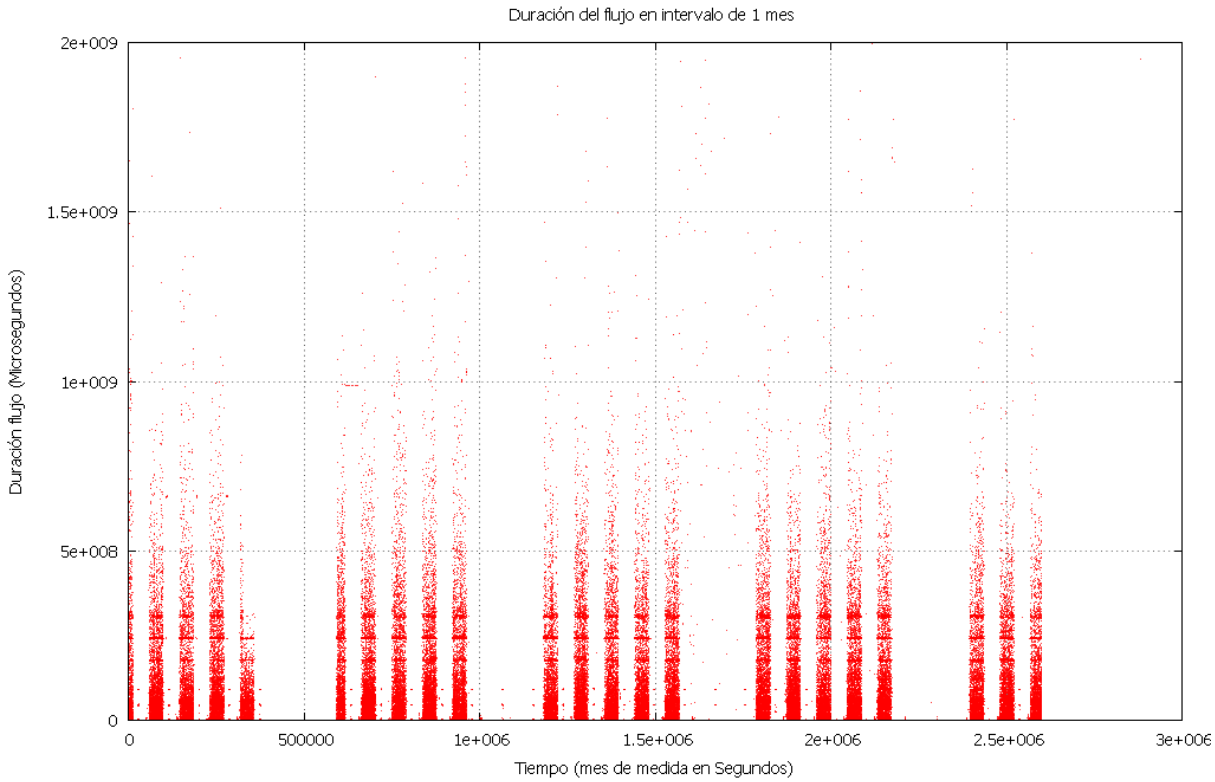


Figura 5-10: Duración de cada flujo

5.1.2.2 *Dispersión del tiempo RTT*

En la figura 5-11 se representa el RTT inicial de cada uno de los flujos.

En la gráfica se observa claramente la ausencia de tomas de RTT durante las noches y los fines de semana lo cual concuerda con los perfiles de tráfico analizados.

Se puede observar, ver zona A de la figura 5-11, que la gran mayoría de los flujos tienen un RTT inicial de 5.000 microsegundos, es decir de 5 milisegundos. En algunos casos, ver zona B de la figura 5-11, este tiempo es algo más elevado, estando en torno a los 30.

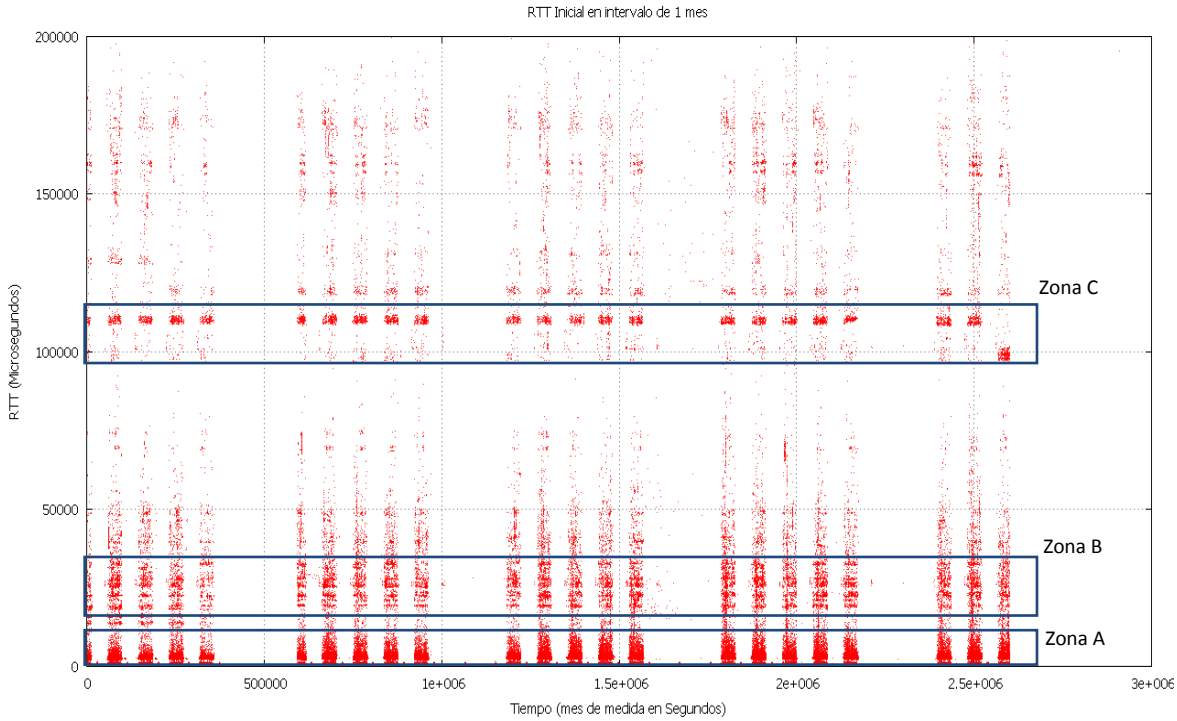


Figura 5-11: RTT inicial de cada flujo

A continuación, en la figura 5-12, se sobrepone el RTT promedio de cada flujo al RTT inicial. En rojo se representan las muestras de RTT promedio y en verde las de RTT inicial.

Se observa que el RTT promedio es superior al RTT inicial en la mayoría de los casos.

En la zona C de la figura 5-11, se encuentran acumulaciones de tiempos elevados de RTT inicial alrededor de los 100 milisegundos. Estas mismas acumulaciones de tiempos ocurren también en los promedios de RTT de los flujos (ver zona C de la figura 5-12). Esto puede significar que hay servidores que trabajan con estos tiempos medios de RTT, y por lo tanto, se podría descartar que los tiempos elevados de RTT para estos servidores se asocien a problemas, pudiendo ser debido a que los servidores estén lejanos (en EEUU, por ejemplo).

Para hacer un análisis más preciso de esto se ha hecho un estudio del ratio entre el RTT promedio y el RTT inicial. Este estudio se muestra en la figura 5-13.

En la gráfica podemos observar que generalmente existe un volumen alto de RTT promedio mayor que RTT inicial. Lo normal, en la muestra tomada, es que RTT promedio sea 3 veces mayor que RTT inicial.

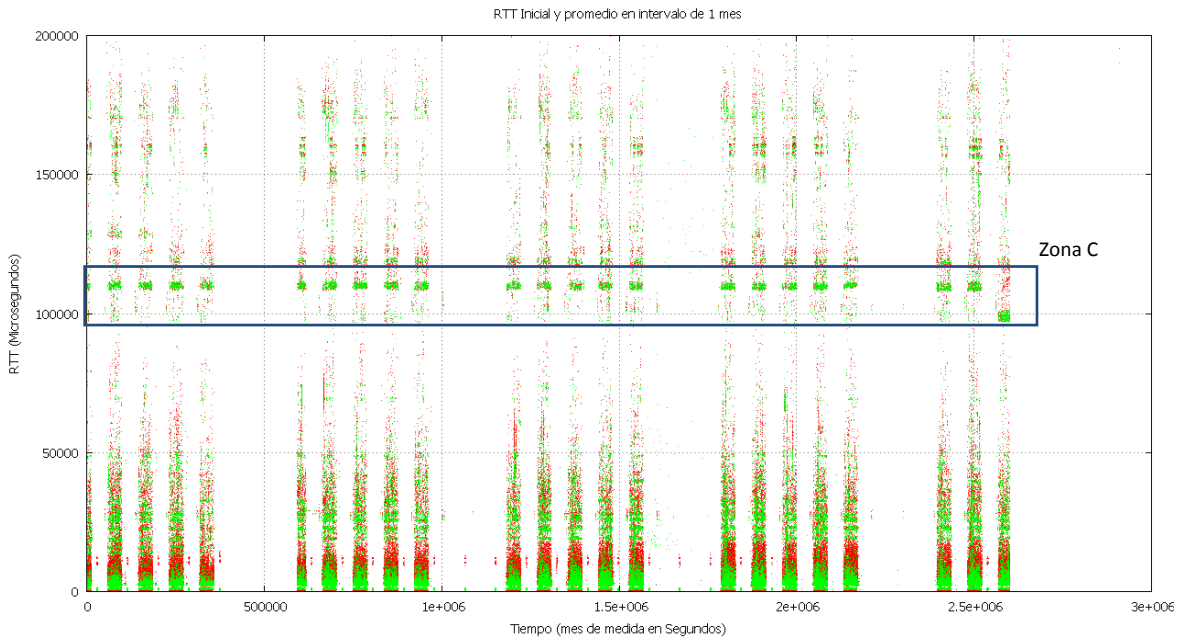


Figura 5-12: RTT promedio de cada flujo

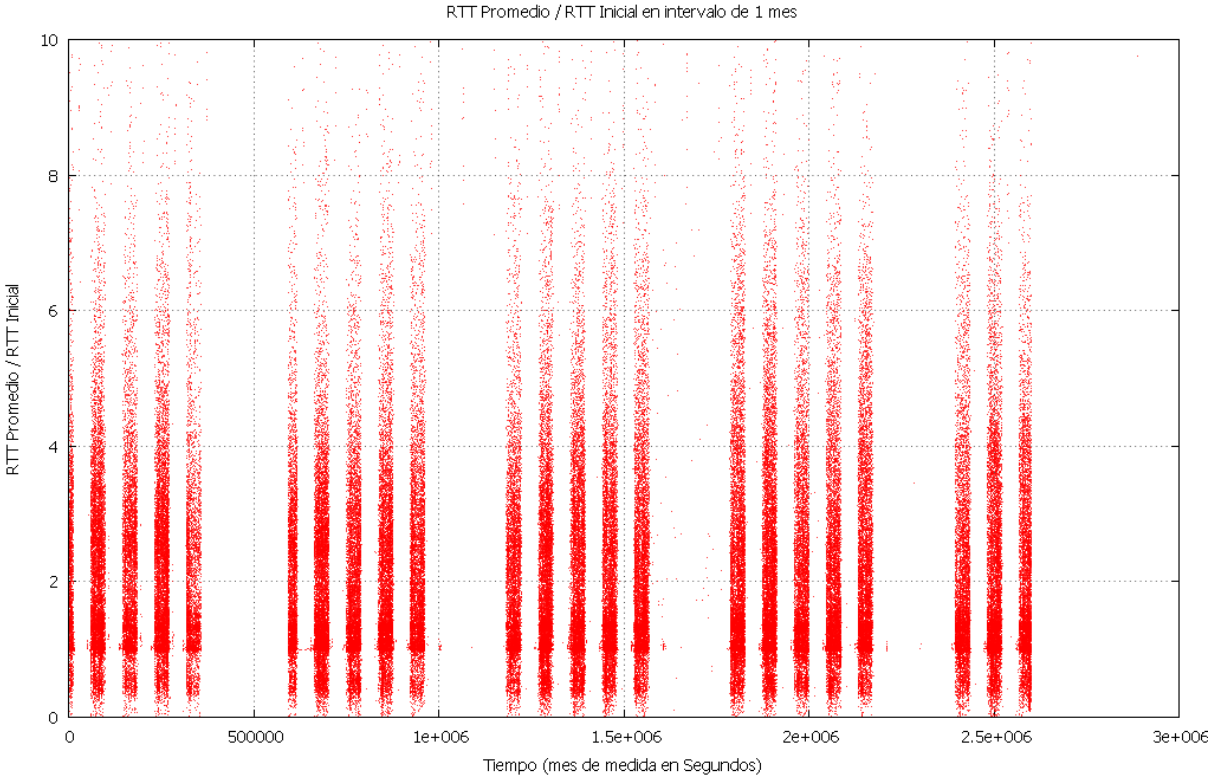


Figura 5-13: RTT promedio/RTT inicial de cada flujo

5.1.2.3 ***Dispersión del tiempo stalled***

En la figura 5-14 se representa la dispersión de la variable tiempo_stalled.

En el estudio del tiempo *stalled* de todos los flujos, resulta interesante comentar que este tiempo va tomando valores escalonados entre 15, 60, 120 y 180 segundos en cada uno de los flujos.

Si enfrentamos el tiempo *stalled* con el RTT promedio (figura 5-15), en la zona A de la figura 5-15, se puede llegar a la conclusión de que para RTT promedio altos existen valores de tiempo *stalled* altos y para valores de RTT promedio bajos los valores de tiempo *stalled* también son bajos. Esto nos podría llevar a la conclusión de que existe relación entre tiempo_stalled y RTT promedio concluyendo que un tiempo *stalled* alto está relacionado con un RTT promedio alto.

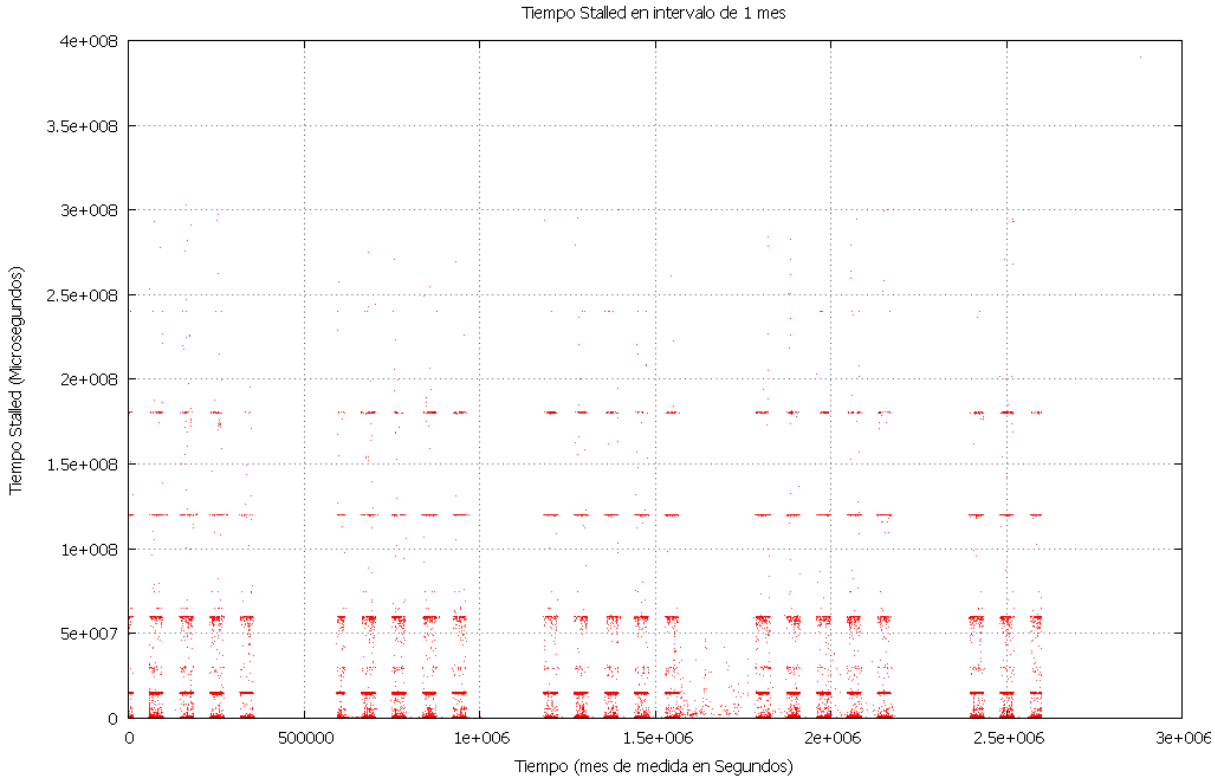


Figura 5-14: Tiempo stalled de cada flujo

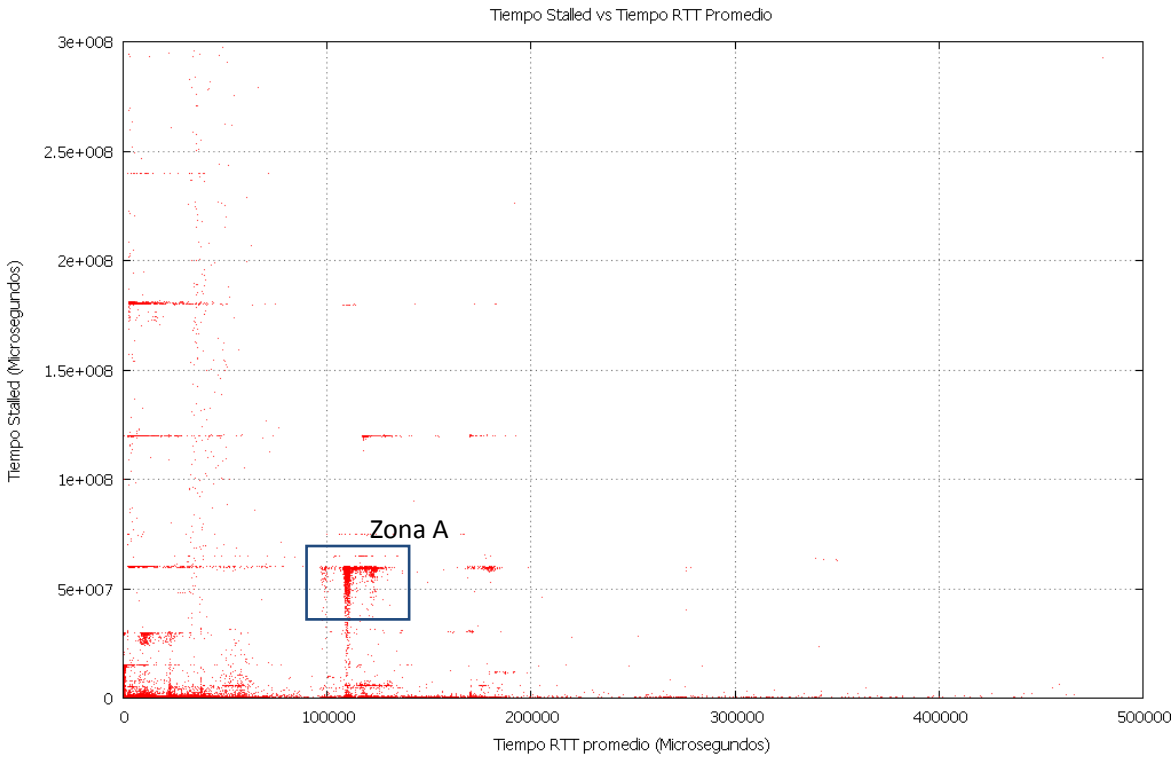


Figura 5-15: Tiempo stalled vs Tiempo RTT de cada flujo

5.2 Gráficas por IP del Servidor

El objetivo de este estudio es poder conocer cuáles son las direcciones IP de los servidores con mayor y menor volumen de datos. En muchos de los casos son las direcciones IP con throughput elevado las que dan problemas de rendimiento en el sistema por su elevado intercambio de volumen de datos.

La figura 5-16 representa los flujos con mayor *throughput* por dirección IP del servidor.

Las direcciones IP de los servidores con las que se intercambia mayor volumen de datos son las que pertenecen a la red de la UAM y a Google, llegando a transmitir 250 MB/seg.

En la tabla 5-1 se relacionan las direcciones IP de los servidores con mayor *throughput* de todo el fichero con sus nombres.

Tabla 5-1 : Tabla relación IP-nombre

IP Servidor	Dominio IP Servidor
150.244.9.132	uam.es
173.194.0.70	Google
93.93.64.184	Furanet.com (Web de almacenamiento contenido)
150.244.9.233	uam.es
93.93.71.231	Furanet.com
74.125.71.91	Google

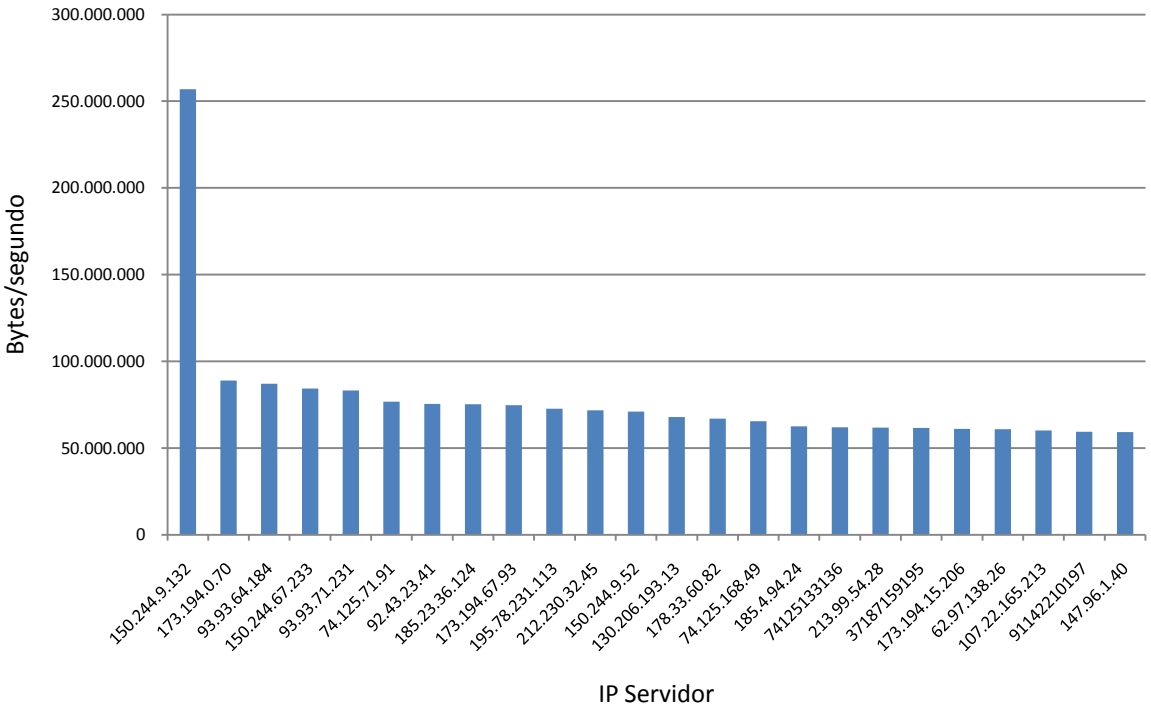


Figura 5-16: Throughput por direcciones IP de Servidor

La figura 5-17 representa los flujos con mayor *throughput* por dirección IP del cliente.

Las IP de los clientes pertenecen a la red de los laboratorios de la EPS de la UAM, y se puede comprobar que el volumen máximo transmitido es de 70 Mbps. También se aprecian unos valores de *througthput* muy homogéneos, lo cual es habitual en aplicaciones web con tráfico altamente simétrico.

En la Figura 5-18, se muestra un histograma del *throughput* para todos los equipos del laboratorio. El eje de abscisas representa el *throughput* en agrupaciones de 1Mbps y, el eje de ordenadas el número de estaciones que pertenecen a un determinado grupo de *throughput.* Es decir, por ejemplo, hay aproximadamente 15 estaciones con un throughput de entre 3Mbps y 4Mbps. Se puede evidenciar el comportamiento heterogéneo de los equipos.

En la misma figura, si se observa la frecuencia acumulativa, representada por la línea roja, el 50% de las estaciones tienen un *throughput* menor de 14 Mbps y el 90% de 32 Mbps. Se pueden apreciar dos grupos de distribución; un primer grupo (Zona A) de hasta 25Mbps donde están el 95% de las estaciones, y un segundo grupo (Zona B) con un throughput superior a 25Mbps que presenta una distribución cuasi uniforme.

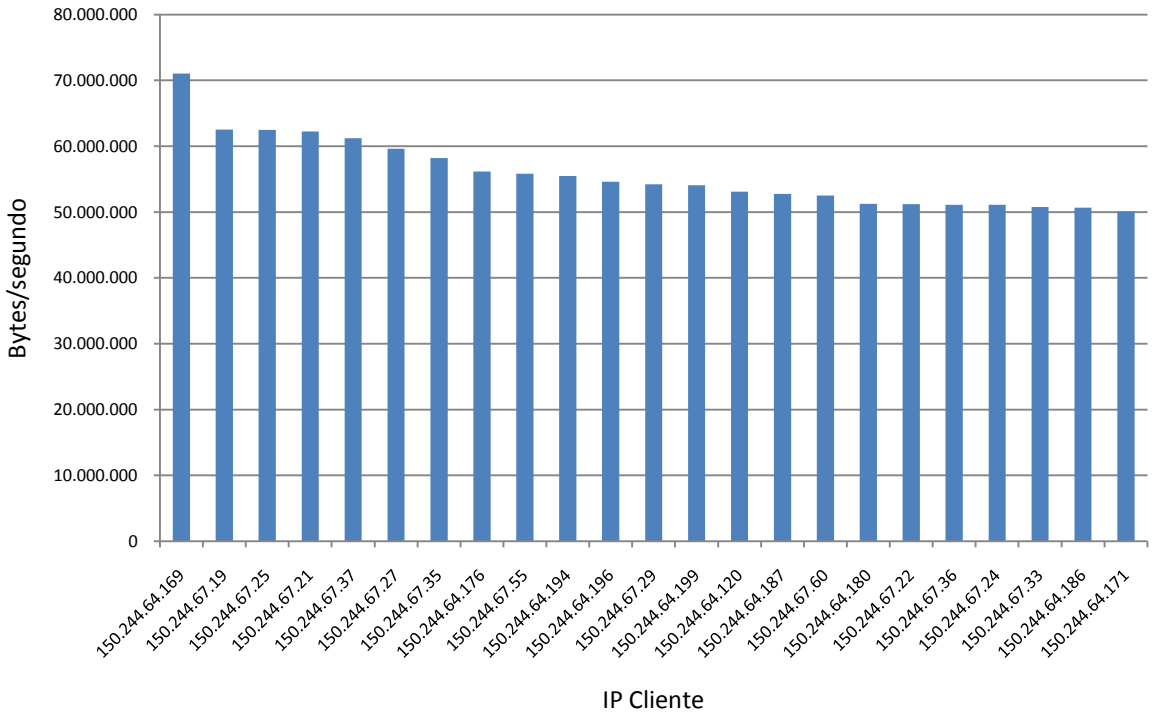


Figura 5-17: Throughput por direcciones IP de Cliente

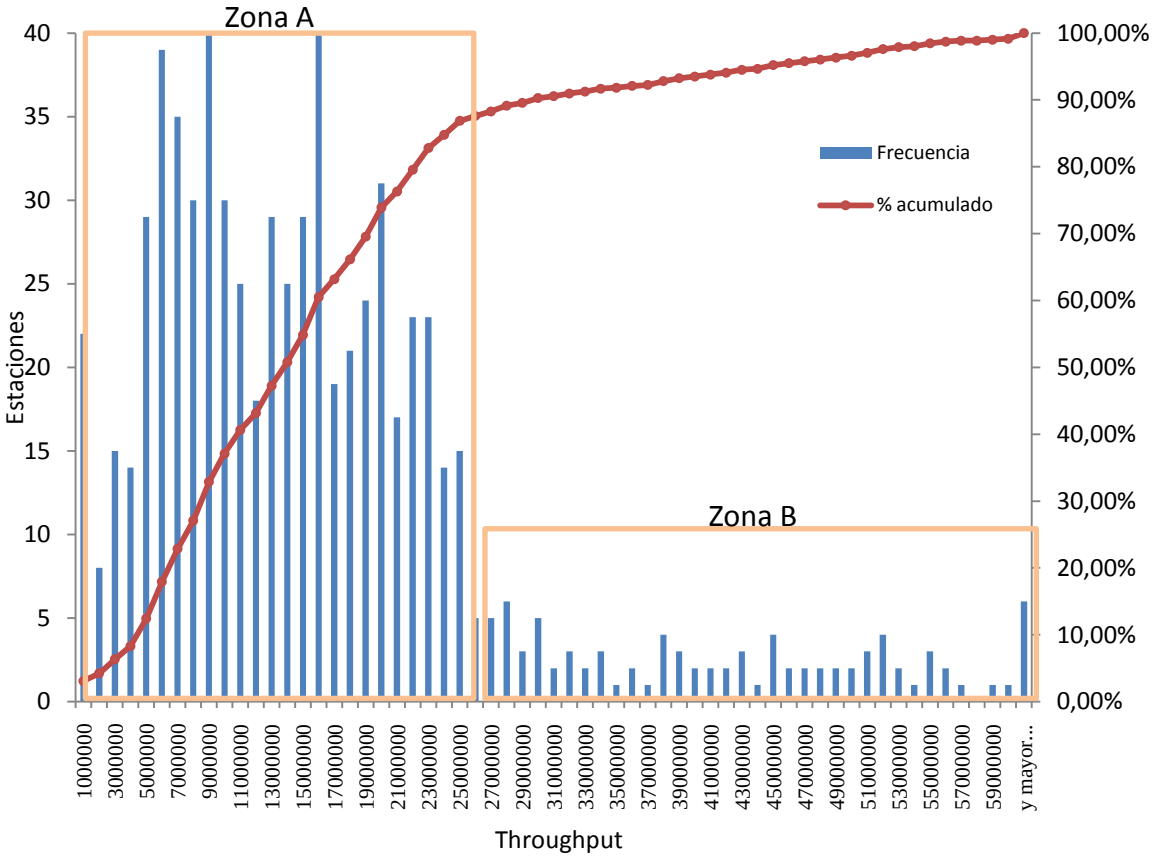


Figura 5-18: Histograma del Throughput de las IP Cliente

5.3 Conclusión

Con los datos analizados podemos llegar a la conclusión de que los parámetros sacados de los flujos mediante la herramienta son coherentes y tienen sentido.

Por otra parte, mediante el estudio de los parámetros elegidos se puede dar una idea aproximada e instantánea de la calidad de servicio.

Del análisis de la muestra de tráfico se puede empezar a señalar que las variables tiempo *stalled* y RTT promedio nos pueden dar una buena indicación o pista de la calidad de una trasmisión.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

El objetivo principal de este PFC ha sido la implementación de una herramienta capaz de medir un conjunto de parámetros que permitan dar una idea de la calidad de servicio ofrecida a un usuario final.

La medida de la calidad de servicio en las telecomunicaciones se asocia generalmente a la satisfacción del cliente, a la percepción que éste tiene del servicio que se le presta. Sin embargo, se suele hablar de cuatro perspectivas de medición de la QoS, dos desde la perspectiva del operador (calidad ofertada y calidad proporcionada) y otras dos del usuario (calidad recibida y calidad percibida).[19] Con los parámetros medidos por la herramienta se puede inferir la calidad de servicio (QoS) y de manera indirecta la calidad percibida por el usuario final (QoE).

Para verificar el buen funcionamiento de los programas y procesos desarrollados en el PFC se han realizado pruebas sobre el tráfico capturado durante un mes en la Escuela Politécnica Superior.

Los resultados de estas pruebas han permitido verificar el funcionamiento de la herramienta mediante la obtención de parámetros coherentes analizados con gráficas. Así también, los programas cumplen con un requerimiento fundamental de partida y es que debe tener un rendimiento óptimo en ejecución, prueba de ello es que se consigue analizar de manera online el tráfico generado por una línea de 1 Gbps. En el caso analizado se ha conseguido tratar el tráfico generado durante un mes por el laboratorio de la EPS en menos de seis horas.

De los datos aportados por la herramienta es posible estimar el funcionamiento del servidor, del cliente y de la red:

- *Funcionamiento del servidor.* Se ha observado que los parámetros que sirven para estimar el funcionamiento del servidor son el tiempo *stalled* y el tiempo RTT. En particular, el estudio del tiempo RTT concluye con que, valores altos de tiempo RTT no siempre suponen problemas en el servicio. Así también se observa que por lo general, los flujos poseen un valor de RTT promedio de flujo 3 veces superior al RTT inicial.
- *Funcionamiento del cliente*. El análisis del funcionamiento del cliente se centra en el estudio de las ventanas cero. Los valores altos indican problemas de congestión en la parte del cliente.
- *Funcionamiento de la red*. El análisis del funcionamiento del cliente se centra en el estudio de las retransmisiones. Los valores altos indican problemas de congestión en la red.

Por otro lado, el estudio de estos parámetros ha servido para sacar conclusiones de cada uno de ellos. Por ejemplo, tanto los valores del tiempo *stalled* como del tiempo RTT se concentran alrededor de unos valores fijos. En el caso del tiempo *stalled* estos valores son

escalonados en tiempos de 15, 60, 120 y 180 segundos. Para el tiempo RTT se encuentran acumulaciones en los tiempos 5, 30 y 100 milisegundos.

6.2 Trabajo futuro

Durante el desarrollo del presente proyecto se han detectado una serie de acciones de mejora que se incluyen en este apartado. Estos trabajos futuros irán encaminados, por un lado, a mejorar la funcionalidad de la herramienta de análisis, y por otro, a buscar nuevos parámetros de medida de la calidad de servicio.

En relación a lo anterior se adjunta la siguiente lista de trabajos futuros:

1. **Análisis de tráfico directamente de la línea.** Como ya se ha comentado, la herramienta de análisis desarrollada en el presente proyecto, parte de una serie de ficheros de tráfico capturado por un producto de análisis de protocolo, como por ejemplo 'Wireshark'. A estas trazas se accede mediante librería 'pcap' [20] y haciendo uso de las funciones *pcap_open_offline* y *pcap_next*. Esta misma librería tiene la posibilidad de analizar tráfico directamente de un dispositivo, haciendo uso de las funciones *pcap_open_live*.

2. **Análisis de parámetros ya señalados en el presente proyecto.** Consistiría en analizar nuevas trazas, utilizando esta misma herramienta, y centrando el estudio en el comportamiento de ciertos parámetros en los que se ha observado una casuística muy especial, por ejemplo, los valores escalonados de los tiempos RTT y *stalled*.

3. **Relacionar QoS y QoE.** Con la herramienta desarrollada podemos sacar parámetros que nos dan una idea de la Calidad de Servicio (QoS) de una conexión. Para poder avanzar, y ver como a partir de estos parámetros podemos intuir o conocer la Calidad de Experiencia de usuario (QoE), habría que disponer de un conjunto de trazas de flujos en los que supiéramos con absoluta certeza que están dando una mala calidad de experiencia de usuario, a continuación, ver que parámetros de calidad de servicio tienen estos flujos e inferir reglas que permitieran relacionar QoS y QoE. Para esto, es fundamental disponer de trazas de flujos que estén dando mala calidad.

4. **Diseñar un cuadro de mando (Dashboard) de calidad de servicio.** Con la mejora del punto 1 y suponiendo que también se ha conseguido tener una relación entre los parámetros de QoS y QoE (punto 3) se podría hacer un análisis en línea, que se reflejara en un cuadro de mando de calidad de servicio. Habría que escoger los indicadores (KPI's) de nivel de servicio y poder trasladar los parámetros de QoS a dichos indicadores generales del estado de todos los flujos.

Este último punto de mejora podría ser una herramienta de suma utilidad para una empresa que esté proveyendo servicios OTT a la comunidad, ya que de manera directa y Online podría establecer alertas ante problemas generalizados en los servicios que están entregando a sus clientes.

Referencias

[1] A. Cuadra, M. Cutanda, A. Aurelius, K. Brunnström, J.E. Lopez de Vergara, M. Varela, J-P Laulajainen, A. Morais, A. Cavalli, A. Mellouk, B. Augustin, I. Perez-Mateos: "Ecosystem for Customer Experience Management", en Quality of Experience Engineering for Customer Added Value Services: From Evaluation to Monitoring, capítulo 3, ISTE-Wiley, Hermes Science Publishing Ltd, en imprenta Junio 2014

[2] Artículo: Guilherme Lopasso, "Como lidiar con servicios OTT" Teledonaktiebolaget LM Ericsson http://www.ericsson.com/res/region_RLAM/press-release/2013/ott-es.pdf

[3] Artículo: Jeremy Kirk, "Operation Face SMS Revenue Dip Due to Social Networking" IDG News Service http://www.pcworld.com/article/250443/operators_face_sms_revenue_dip_due_to_social_networking.html

[4] Artículo:Guilherme Lopasso, "Como lidiar con servicios OTT" Teledonaktiebolaget LM Ericsson http://www.ericsson.com/res/region_RLAM/press-release/2013/ott-es.pdf '

[5] ITU-T Recommendations and other International Standards Relevant to QOS/QOE. Daniel K. Waturu. Manager /Telecoms Compliance Communications Commission of Kenya

[6] Recomendación UIT-T E-800 http://www.itu.int/rec/T-REC-E.800-200809-I/es

[7] Artículo: Heng Cui, Ernst Biersack, "On the Relationship Between QoS and QoE for Web Sessions" Francia. January 4th, 2012 "http://www.eurecom.fr/~cui/techrep/TechRep12263.pdf"

[8] Paper: Aad van Moorsel, "Metrics for the Internet Age: Quality of Experience and Quality of Business" July 19th , 2001, HP Laboratories Palo Alto "http://www.hpl.hp.com/techreports/2001/HPL-2001-179.pdf"

[9] Ricky K. P. Mok, Edmond W. W. Chan, and Rocky K. C. Chang, "Measuring the Quality of Experience of HTTP Video Streaming" Department of Computing The Hong Kong Polytechnic University Hunghom, Kowloon, Hong Kong

[10] Internet Protocol, Protocol Specification RFC 791 "http://tools.ietf.org/html/rfc791" Arlington, Virgina. Sept 1981.

[11] Transmission Control Protocol, Protocol Specification RFC: 793 "http://tools.ietf.org/html/rfc793" Arlington, Virginia. Sept 1981

[12] Artículo: Aprendiendo a programar con Lipbcap "http://www.e-ghost.deusto.es/docs/2005/conferencias/pcap.pdf" Alejandro López Monge. Feb 2005

[13] Network QoS (RTT, Bandwidth, Packet Loss) Application QoS (Application Performance Metrics) User QoS (QoE) (MOS – Mean Opinion Score) Radar Chart

[14] Transmission Control Protocol, Protocol Specification RFC: 793 "http://tools.ietf.org/html/rfc793" Arlington, Virginia. Sept 1981

[15] TCP/IP Essentials: A Lab-Based Approach. Shivendra S. Panwar. Cambridge University Press, 18 de nov. de 2004 http://books.google.es/books/about/TCP_IP_Essentials.html?id=NbYKmCfw_PsC&redir_esc=y

[16] Protocolos y aplicaciones Internet. José María Barceló Ordinas. Editorial UOC, 2008

[17] Ampliación de Sistemas Operativos y Redes http://web.fdi.ucm.es/profesor/rubensm/ASOR/Trasparencias/Tema%203-

%20Conceptos%20Avanzados%20del%20Protocolo%20TCP.pdfW. Richard Stevens, TCP/IP Illustrated: The protocols. Addison-Wesley Pub. Co., 10 de ene. de 1994
[18] Manual GNUplot. http://www.gnuplot.info/
[19] Estudio de factibilidad del diseño de la red de acceso corporativo para sede del Instituto de Desarrollo Urbano según factores de tráfico y cobertura. Helman Hernando Mora Arévalo, Jonathan Sánchez Coronado. Octubre 2011
[20] Librería pcap http://www.tcpdump.org/pcap.html

Glosario

OTT	Over The Top
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
RTP	Realtime Transport Protocol
IP	Internet Protocol
HTTP	Hipertext Transfer Protocol
QoS	Quality of Service
QoE	Quality of Experience
RTT	Round Trip Time
ISO	International Standard Organization
ITU-T	International Telecommunication Union- Telecommunication
ETSI	European Telecommunications Standards Institute
IETF	Internet Engineering Task Force
SLA	Service Level Agreement
HTML	Hypertext Markup Language
URG	Urgent Pointer field significant
ACK	Acknowledgment field significant
PSH	Push Function
RST	Reset the Connection
SYN	Synchronize Sequence Numbers
FIN	no more data from sender
OSI	Open Systems Interconnect
LSF	Linux Socket Filter

Anexos

ANEXO A Manual de programador

Ejecución y salida de la herramienta

Como se ha explicado en la memoria de este proyecto, la ejecución de la herramienta consta de varias fases:

- 1ª Fase: Extracción de datos básicos de los ficheros de traza
- 2ª Fase: Concatenación de datos
- 3ª Fase: Preparación de datos

Para invocarlas desde línea de comandos es necesario seguir los siguientes pasos:

Para la 1ª Fase: Se va a llamar a la herramienta donde se ejecuta el programa principal. Para ello, Se van a ejecutar de manera simultánea 10 script con el siguiente formato:

```
#!/bin/bash

rm salida/logerror_0.txt
############################################################
ls /disco_capturas_0/epslab/1mes | grep 0.pcap > lista_0.txt
while read file
do

                ./analisis_trafico /disco_capturas_0/epslab/1mes/$file salida/$file.txt salida/logtotal_0.txt >> salida/logerror_0.txt
done < lista_0.txt
```

Figura A-0-1: Script de ejecución del programa principal

Para ejecutar los script es necesario situarse en la carpeta donde estos han sido guardados y, ejecutar para cada uno de los 10 script, el siguiente comando:

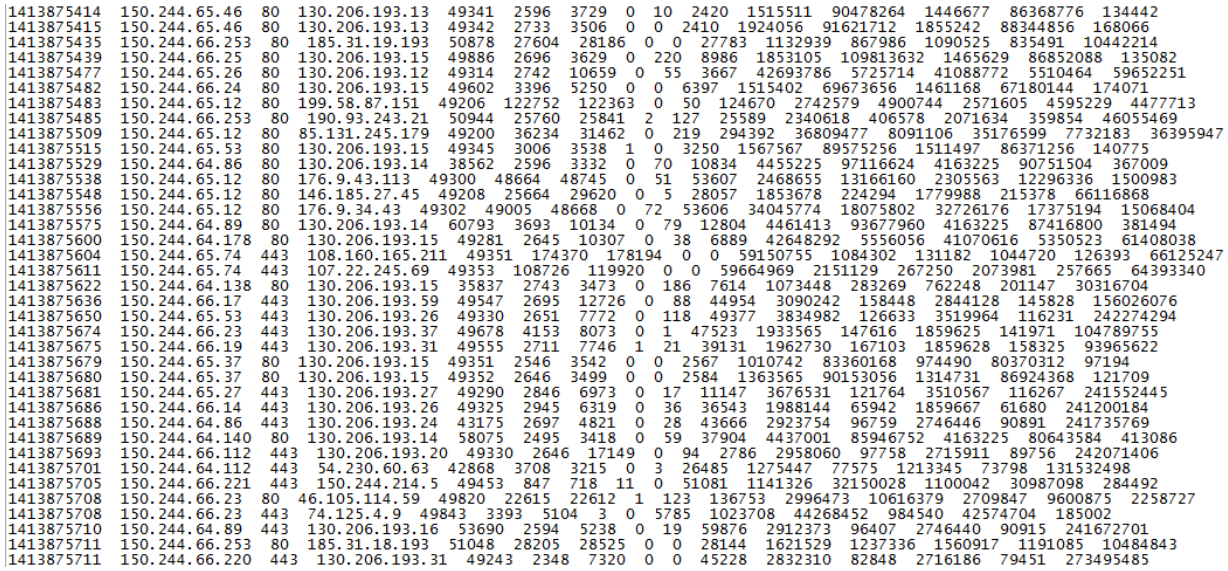
```
/home/mlucena/Proyecto/analisis_de_trafico$ ./sacar_flujos_0
```

De esta forma se van a ir guardando en la carpeta "salida" los siguientes archivos en formato .txt:

- ✓ La salida $file.txt. Por cada fichero *.pcap se crea un fichero *.txt con los datos de cada flujo del fichero (hora epoch, IP Cliente, Puerto Cliente, IP Servidor, Puerto Servidor, RTT inicial, RTT promedio, número de Zero Windows, número de retransmisiones, tiempo stalled, número de bytes totales, Throughput, numero de bytes de datos, Goodput y duración) por columnas. En la figura 39 aparece un ejemplo de salida.

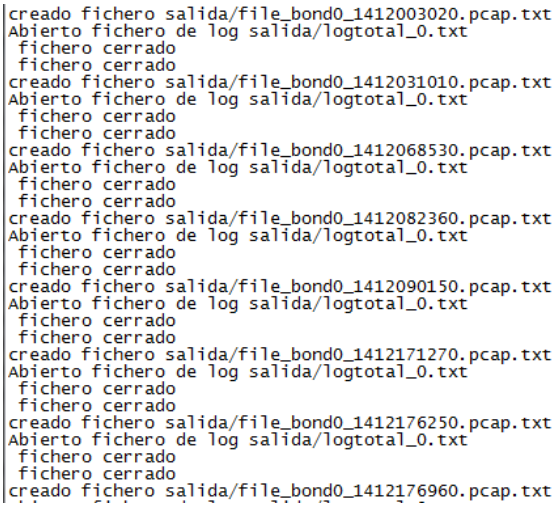
- ✓ Un fichero de logtotal_0.txt. Por cada script de ejecución, se crea un fichero de log donde se refleja la actividad de ese script. Este fichero recogerá posibles errores que haya en el procesamiento de los archivos. En la Figura 40 aparece un ejemplo de salida de este tipo de ficheros.

- ✓ Un fichero de logerror_0 .txt. Por cada script de ejecución se crea un fichero con la salida a pantalla que tuvieran los programas ejecutados baje dio script.



```
1413875414  150.244.65.46  80  130.206.193.13  49341  2596  3729  0  10  2420  1515511  90478264  1446677  86368776  134442
1413875415  150.244.65.46  80  130.206.193.13  49342  2733  3506  0  0  2410  1924056  91621712  1855242  88344856  168066
1413875435  150.244.66.253  80  185.31.19.193  50878  27604  28186  0  0  27783  1132939  867986  1090525  835491  10442214
1413875439  150.244.66.25  80  130.206.193.15  49886  2696  3629  0  220  8986  1853105  109813632  1465629  86852088  135082
1413875477  150.244.65.26  80  130.206.193.12  49314  2742  10659  0  55  3667  42693786  5725714  41088772  5510464  59652251
1413875482  150.244.66.24  80  130.206.193.15  49602  3396  5250  0  0  6397  1515402  69673656  1461168  67180144  174071
1413875483  150.244.65.12  80  199.58.87.151  49206  122752  122363  0  50  124670  2742579  4900744  2571605  4595229  4477713
1413875485  150.244.66.253  80  190.93.243.21  50944  25760  25841  2  127  25589  2340618  406578  2071634  359854  46055469
1413875509  150.244.65.12  80  85.131.245.179  49200  36234  31462  0  219  294392  36809477  8091106  35176599  7732183  36395947
1413875515  150.244.65.53  80  130.206.193.15  49345  3006  3538  1  0  3250  1567567  89575256  1511497  86371256  140775
1413875529  150.244.64.86  80  130.206.193.14  38562  2596  3332  0  70  10834  4455225  97116624  4163225  90751504  367009
1413875538  150.244.65.12  80  176.9.43.113  49300  48664  48745  0  51  53607  2468655  13166160  2305563  12296336  1500983
1413875548  150.244.65.12  80  146.185.27.45  49208  25664  29620  0  5  28057  1853678  224294  1779988  215378  66116868
1413875556  150.244.65.12  80  176.9.34.43  49302  49005  48668  0  72  53606  34045774  18075802  32726176  17375194  15068404
1413875575  150.244.64.89  80  130.206.193.14  60793  3693  10134  0  79  12804  4461413  93677960  4163225  87416800  381494
1413875600  150.244.64.178  80  130.206.193.15  49281  2645  10307  0  38  6889  42648292  5556056  41070616  5350523  61408038
1413875604  150.244.65.74  443  108.160.165.211  49351  174370  178194  0  0  59150755  1084302  131182  1044720  126393  66125247
1413875611  150.244.65.74  443  107.22.245.69  49353  108726  119920  0  0  59664969  2151129  267250  2073981  257665  64393340
1413875622  150.244.64.138  80  130.206.193.15  35837  2743  3473  0  186  7614  1073448  283269  762248  201147  30316704
1413875636  150.244.66.17  443  130.206.193.59  49547  2695  12726  0  88  44954  3090242  158448  2844128  145828  156026076
1413875650  150.244.65.53  443  130.206.193.26  49330  2651  7772  0  118  49377  3834982  126633  3519964  116231  242274294
1413875674  150.244.66.23  443  130.206.193.37  49678  4153  8073  0  1  47523  1933565  147616  1859625  141971  104789755
1413875675  150.244.66.19  443  130.206.193.31  49555  2711  7746  1  21  39131  1962730  167103  1859628  158325  93965622
1413875679  150.244.65.37  80  130.206.193.15  49351  2546  3542  0  0  2567  1010742  83360168  974490  80370312  97194
1413875680  150.244.65.37  80  130.206.193.15  49352  2646  3499  0  0  2584  1363565  90153056  1314731  86924368  121709
1413875681  150.244.65.27  443  130.206.193.27  49290  2846  6973  0  17  11147  3676531  121764  3510567  116267  241552445
1413875686  150.244.66.14  443  130.206.193.26  49325  2945  6319  0  36  36543  1988144  65942  1859667  61680  241200184
1413875688  150.244.64.86  443  130.206.193.24  43175  2697  4821  0  28  43666  2923754  96759  2746446  90891  241735769
1413875689  150.244.64.140  80  130.206.193.14  58075  2495  3418  0  59  37904  4437001  85946752  4163225  80643584  413086
1413875693  150.244.66.112  443  130.206.193.20  49330  2646  17149  0  94  2786  2958060  97758  2715911  89756  242071406
1413875701  150.244.64.112  443  54.230.60.63  42868  3708  3215  0  3  26485  1275447  77575  1213345  73798  131532498
1413875705  150.244.66.221  443  150.244.214.5  49453  847  718  11  0  51081  1141326  32150028  1100042  30987098  284492
1413875708  150.244.66.23  80  46.105.114.59  49820  22615  22612  1  123  136753  2996473  10616379  2709847  9600875  2258727
1413875708  150.244.66.23  443  74.125.4.9  49843  3393  5104  3  0  5785  1023708  44268452  984540  42574704  185002
1413875710  150.244.64.89  443  130.206.193.16  53690  2594  5238  0  19  59876  2912373  96407  2746440  90915  241672701
1413875711  150.244.66.253  80  185.31.18.193  51048  28205  28525  0  0  28144  1621529  1237336  1560917  1191085  10484843
1413875711  150.244.66.220  443  130.206.193.31  49243  2348  7320  0  0  45228  2832310  82848  2716186  79451  273495485
```

Figura A-0-2: Salida del programa



```
creado fichero salida/file_bond0_1412003020.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412031010.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412068530.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412082360.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412090150.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412171270.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412176250.pcap.txt
Abierto fichero de log salida/logtotal_0.txt
 fichero cerrado
 fichero cerrado
creado fichero salida/file_bond0_1412176960.pcap.txt
```

Para la 2ª Fase: Se van a concatenar cada uno de los ficheros $file.txt cuyo contenido es la salida de la ejecución del programa.

Para ello se ha creado el siguiente script:

```
#!/bin/bash
rm fichero_total.txt
ls *.txt | while read file
do
        echo $file
        cat $file >> fichero_total.txt
done
```

Para su ejecución hay que, situarse en la carpeta donde están todos los ficheros .txt que se quieren concatenar y escribir con la siguiente línea de comandos:

```
/home/mlucena/Proyecto/analisis_de_trafico/salida$ ./concatenar
```

Para la 3ª Fase: Para la preparación de datos se va a llamar a otro programa creado a parte "*rellenar_nombre.c*". En este programa se irá cogiendo cada una de las líneas de fichero .txt donde se encuentran todos los flujos y, como se ha explicado en el proyecto, se le irá añadiendo el nombre de la IP Servidor y la hora en formato normal. Este programa tiene dos argumentos; uno de entrada que será el fichero con todos los datos que se quieren preparar y, uno de salida con los datos preparados.

Para su ejecución, hay que situarse en la carpeta donde tengamos el fichero .txt con todos los flujos y, ejecutar la siguiente línea de comandos:

```
/home/mlucena/Proyecto/rellenar_nombre$ ./rellenar_nombre fichero_salida.txt fichero_salida _total.txt
```

La salida final será tendrá el siguiente formato:

```
Mon Oct  6 18:46:01 2014 IP_ERROR:130.206.193.57 1412613961  150.244.64.107  443  130.206.193.57  36143  2495  15388  0  59  3881  2774845  91815  2569421  85018  241776456
Mon Oct  6 18:46:02 2014 moodle.uam.es 1412613962  150.244.66.181  443  150.244.214.5  49284  848  519  24  0  15015198  2221347  876876  2140463  844947  20266385
Mon Oct  6 18:46:04 2014 moodle.uam.es 1412613964  150.244.66.185  443  150.244.214.5  49311  850  641  113  0  15012908  21121505  10458132  20366443  10084271  16157168
Mon Oct  6 18:46:05 2014 par10s10-in-f23.1e100.net 1412613965  150.244.64.191  443  173.194.40.151  59428  22412  26177  0  0  323820  1119679  26299  1050701  24679  340602425
Mon Oct  6 18:46:07 2014 IP_ERROR:130.206.193.13 1412613967  150.244.65.90  443  130.206.193.13  53501  2350  9294  0  223  37990  18513907  1629046  17381251  1529383  90919932
Mon Oct  6 18:46:08 2014 par10s10-in-f20.1e100.net 1412613968  150.244.64.191  443  173.194.40.148  42212  25605  41232  0  1  97815  1909398  45940  1816747  43710  332506558
Mon Oct  6 18:46:10 2014 gamer.ru 1412613970  150.244.65.127  80  91.228.238.66  40355  85562  85519  0  59  726334  2334176  60299  2147144  55467  309680765
Mon Oct  6 18:46:14 2014 moodle.uam.es 1412613974  150.244.66.184  443  150.244.214.5  49256  749  686  172  0  15008206  21122228  8041202  20366464  7753484  21014037
Mon Oct  6 18:46:19 2014 IP_ERROR:130.206.193.14 1412613979  150.244.64.191  80  130.206.193.14  52229  2245  3270  0  0  1741  4354105  446574880  4163225  426997440  78127
Mon Oct  6 18:46:29 2014 moodle.uam.es 1412613989  150.244.64.214  443  150.244.214.5  36473  847  389  0  58  15015827  4295862  1129858  4017131  1056549  30417121
Mon Oct  6 18:46:30 2014 moodle.uam.es 1412613990  150.244.66.188  443  150.244.214.5  49254  648  482  195  0  15003668  29953320  4480090  28878980  4319402  53487514
Mon Oct  6 18:46:30 2014 moodle.uam.es 1412613990  150.244.64.214  443  150.244.214.5  36478  893  394  0  65  13952281  2874517  831536  2652349  767268  27655042
Mon Oct  6 18:46:30 2014 moodle.uam.es 1412613990  150.244.64.214  443  150.244.214.5  36475  15475  410  0  58  14991861  2921759  762712  2712117  707986  30646155
Mon Oct  6 18:46:30 2014 moodle.uam.es 1412613990  150.244.64.214  443  150.244.214.5  36476  1048  425  0  12  15016532  3878581  1010277  3670219  956004  30713347
```

Columna	Dato	Ejemplo
1	Día de la semana	Mon
2	Mes	Oct
3	Numero de día de la semana	6
4	Hora. Formato "hh:mm:ss"	18:46:02
5	Año	2014

6	Nombre IP Servidor	moodle.uam.es
7	Hora. Formato epoch	1412613962
8	IP Cliente	150.244.66.181
9	Puerto Cliente	443
10	IP Servidor	150.244.214.5
11	Puerto Servidor	49284
12	RTT Inicial en microsegundos	848
13	RTT Promedio en microsegundos	519
14	Numero Zero Windows	24
15	Numero Retransmisiones	0
16	Tiempo Stalled en microsegundos	15015198
17	Numero de Bytes totales	2221347
18	Throughput en bps	876876
19	Numero de Bytes de datos	2140463
20	Goodput en bps	844947
21	Duración en microsegundos	20266385

ANEXO B PRESUPUESTO

Con el fin de poder hacer una valoración económica del esfuerzo empleado en la elaboración presente proyecto, se detalla el presupuesto del mismo, que se desglosa en las siguientes partidas:

• Presupuesto de ejecución material.
• Gastos generales y Beneficio industrial.
• Honorarios por la redacción y dirección del proyecto.
• Presupuesto total.

Todas las cantidades aparecen expresadas en euros.

Presupuesto de ejecución material.

Este presupuesto hace referencia a los Costes de recursos materiales y a los Costes de mano de obra.

Desglose por tareas ejecutadas

Para poder comprender de manera clara los costes de mano de obra del proyecto, se va a exponer la división de las tareas del trabajo realizado en este Proyecto fin de Carrera.

Posteriormente, se van a representar cara una de estas tareas junto con su relación de precedencia temporal a través de un diagrama de Gantt.

El proyecto consta de las siguientes tareas:

- **Tarea 1**:

  Objetivo: Estudio del estado del arte de los servicios OTT que existen en la actualidad, así como de calidad de servicio (QoS) y calidad de experiencia (QoE).
  Igualmente se estudiarán el funcionamiento de los protocolos IP, TCP y HTTP.

  Duración: 3 meses

  Esfuerzo: Ingeniero superior, 1 persona-mes.

- **Tarea 2**:

  Objetivo: Preparación del entorno. Instalación de los sistemas operativos. Instalación del entorno de desarrollo. Estudio y búsqueda de las librerías de programación (Libpcap), herramienta que será utilizada en la fase de desarrollo. Estudio de herramientas para tratamiento de datos finales: Excel de Microsoft y 'gnuplot'

  Duración: 1 mes

Esfuerzo: Ingeniero superior, 1 persona-mes.

- **Tarea 3**:

Objetivo: Análisis de Requisitos. Se definirá la estrategia a seguir para el desarrollo de la herramienta. Así también se hará un estudio de los parámetros utilizados en la herramienta.

Duración: 2 meses

Esfuerzo: Ingeniero superior, 1 persona-mes.

- **Tarea 4**:

Objetivo: Implementación de la herramienta. Desarrollo en código C los diseños teóricos.

Duración: 3 meses

Esfuerzo: Ingeniero superior, 3 persona-mes.

- **Tarea 5**:

Objetivo: Pruebas. Realización de experimentos del funcionamiento de la herramienta frente a ficheros de prueba.

Duración: 1 mes

Esfuerzo: Ingeniero superior, 0,5 personas-mes.

- **Tarea 6**:

Objetivo: Corrección. Programación de cambios para la mejora de la herramienta, y reprogramación por los errores encontrados en las pruebas.

Duración: 1 mes

Esfuerzo: Ingeniero superior, 0,5 personas-mes.

- **Tarea 7**:

Objetivo: Pruebas. Validación de la corrección de errores.

Duración: 0,25 meses

Esfuerzo: Ingeniero superior, 0,25 persona-mes.

- **Tarea 8**:

Objetivo: Análisis de datos y obtención de gráficas para el certificado del rendimiento de la herramienta. Revisión de los datos obtenidos, volcado de datos en la herramienta Excel y creación de tablas dinámicas y gráficas. Revisión y análisis de datos mediante la herramienta 'gnuplot'.

Duración: 0,75 meses

Esfuerzo: Ingeniero superior, 0,75 personas-mes.

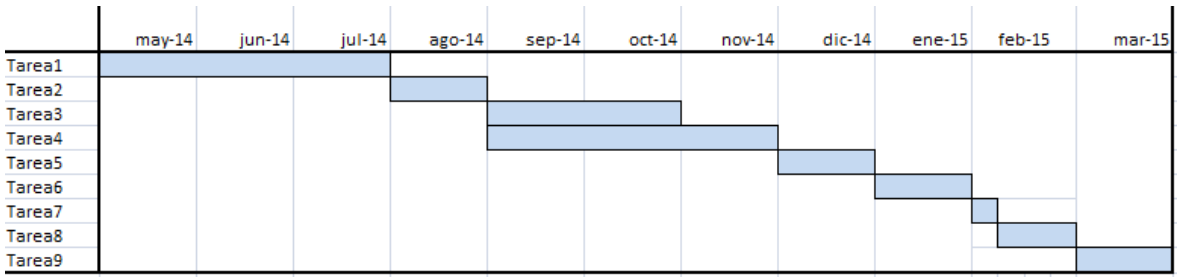
- **Tarea 9**:

Objetivo: Redacción de la documentación asociada al Proyecto Final de Carrera.

Duración: 1 mes

Esfuerzo: Ingeniero superior, 1 persona-mes.

A continuación se exponen las tareas mediante un diagrama de Gantt:



Costes de mano de obra

Para la realización del proyecto se requiere un profesional con el siguiente perfil profesional:

Un Ingeniero Superior de Telecomunicación, encargado tanto del planteamiento, desarrollo e implementación del trabajo técnico, así como de la redacción, presentación y encuadernación del proyecto.

La estimación de los costes se realiza en base a los siguientes datos:

Cotizaciones según el Régimen General de la Seguridad Social. Un ingeniero pertenece al grupo 1.

Jornada laboral de 8 horas/día y 21 días laborales/mes.

El coste de la mano de obra considerado para la ejecución del proyecto asciende a 30.407,22 €. Este coste salarial se descompone en la siguiente tabla:

Costes Salariales	
Concepto	**Coste**
Base Cotizable máxima mensual según bases de 1.1.2015	3.606,00 €
- Contingencias comunes (23.6%)	851,02 €
- Desempleo (5,5%), F.G.S (0,6%) y Formación profesional (0,2 %) (Total 6,3%)	227,18 €
- Accidentes de trabajo y enfermedades profesionales (8,33%)	300,38 €
Coste de la seguridad social (€/mes)	1.378,58 €
Salario bruto mensual (€/mes sobre base 24.000 €/año)	2.000,00 €
Coste salarial mensual (€/mes)	3.378,58 €
- Numero de meses	9 meses
Coste total mano de obra	**30.407,22 €**

Coste de recursos materiales

En la siguiente tabla constan los costes de los recursos materiales empleados, considerando un periodo de amortización para el hardware y el software de 4 años.

En primer lugar se indicarán los costes totales y a continuación se imputarán las cuantías correspondientes a la amortización de los recursos durante su periodo de utilización en el desarrollo del proyecto (12 meses sobre 48 total, suponen un 25%)

Recursos Materiales		
Concepto	**Coste total**	**Coste real**
Equipo de desarrollo	1.500,00 €	375,00 €
Sistema operativo Windows 7	283,62 €	70,90 €
Sistema operativo Ubuntu Desktop 10.04 32 bits	0,00 €	0.00 €
Herramienta de desarrollo C	0,00 €	0.00 €
Microsoft Office 2014	269,00 €	67,25 €
Coste total recursos materiales	**2.050,62 €**	**512,65 €**

Coste Total de los recursos

La suma de los costes por mano de obra y de los costes por recursos materiales es lo que constituye el Presupuesto de ejecución material.

Presupuesto de ejecución Material	
Concepto	**Coste**
Coste mano de obra	30.407,22 €
Coste total recursos materiales	512,65 €
Total	**30.919,87 €**

Gastos generales y beneficio industrial

Bajo Gastos Generales se incluyen todos aquellos gastos derivados de la utilización de instalaciones, cargas fiduciarias, amortizaciones, gastos fiscales, etc. Con esto, el Presupuesto de Ejecución por contrata queda como sigue:

Presupuesto de ejecución por contrata	
Concepto	**Coste**
Presupuesto de ejecución material	30.919,87 €
Gastos generales (16% del PEM)	4.947,17 €
Beneficio industrial (6% del PEM)	1.855,19 €
Total	**37.722,23 €**

Honorarios por la redacción y dirección del proyecto

Los Honorarios que recomienda aplicar el Colegio Oficial de Ingenieros de Telecomunicación, tanto para la redacción como para la dirección del proyecto son los asociados a Trabajos tarifados por tiempo empleado, con un valor de un 5.6%.

Presupuesto total

Finalmente, sumando todas las imputaciones anteriores, y aplicando el 21% de IVA, se obtiene el presupuesto total.

Presupuesto total	
Concepto	**Coste**
Presupuesto de ejecución por contrata	37.722,23 €
Honorarios por dirección	2.112,44 €
Honorarios por redacción	2.112,44 €
Total libre de impuesto del valor añadido	41.947,11 €
IVA (21%)	8.808,89 €
Coste total	**50.756,00 €**

El presupuesto total del proyecto asciende a Cincuenta Mil Setecientos Cincuenta y Seis Euros.

Madrid, Marzo de 2015

La Ingeniera Jefe de Proyecto

Fdo: María Lucena Cabello
Ingeniero de Telecomunicación

ANEXO C PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un de un sistema de medición, monitorización y gestión de servicios OTT desarrollado en este PFC. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.
Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma,

por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.