UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Concurrencia de flujos según el mix de aplicaciones de red

Máster Universitario en Ingeniería de Telecomunicación

Autor: GARCÍA, GARCÍA, Juan Luis

Tutor: GARCÍA DORADO, José Luis
Departamento de Tecnología Electrónica de Telecomunicaciones

FECHA: Septiembre, 2018

Concurrencia de flujos según el mix de aplicaciones de red

AUTOR: Juan Luis García García
TUTOR: José Luis García Dorado

Dpto. Tecnología Electrónica de Telecomunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Septiembre de 2018

Resumen

La creciente complejidad de las redes de comunicaciones en Internet lleva los gestores de red a buscar nuevas alternativas para identificar de manera eficaz los problemas y anomalías que surgen en las redes. Dos de los parámetros que caracterizan a una red de comunicaciones son el ancho de banda y el número de flujos concurrentes. Estos parámetros resultan determinantes para conocer las prestaciones de cualquier red y son buenos medidores de rendimiento de las mismas. Sin embargo, la relación entre estos dos parámetros no ha sido estudiada en profundidad y aun se plantean muchas dudas sobre la misma.

Este trabajo fin de Máster tiene como primer objetivo mejorar el conocimiento de la relación entre el número de flujos concurrentes con el ancho de banda en redes de comunicaciones. Esta relación será analizada sobre conjuntos de trazas de red reales. Las conclusiones extraídas creemos que pueden ser de utilidad para gestores de red que conociendo el ancho de banda típico de sus redes pueden estimar la concurrencia en flujos. En concreto la concurrencia en flujos es una métrica clave en tareas como la identificación de anomalías, de intrusiones, o clasificación de tráfico. En este sentido, la relación flujos-ancho de banda será estudiada por aplicación, hecho que supondrá averiguar si las aplicaciones determinan el valor de esta relación, o si por el contrario no son importantes.

Para resolver esta cuestión será necesario proveerse de las herramientas necesarias para extrapolar, con las trazas de datos las cuales disponemos, información sobre los flujos de red que se encuentren en dichas trazas. Esta información incluye la aplicación a la que pertenecen, extraída con técnicas *deep packet inspection* (DPI), y diferentes parámetros temporales vinculados a cada flujo, que permitan calcular la relación flujos-ancho de banda.

Una vez desarrolladas las herramientas que permitan analizar el tráfico de red, se estudiará los resultados obtenidos acerca de la relación flujos-ancho de banda. Se discutirá si se pueden realizar baremos aproximados en base a este parámetro para diferenciar las aplicaciones más populares.

Aprovechando que la herramienta DPI extrae multitud de características por cada flujo. Se analizará si algunas de estadísticas son útiles para diferenciar entre aplicaciones, comprobando, si los valores obtenidos dependen de la aplicación usada en cada caso.

Después de encontrar que parámetros pueden ser más útiles, se utilizará una herramienta de aprendizaje automático que muestre que algoritmos son más efectivos para intentar diferenciar los flujos por aplicaciones como una alternativa prometedora a DPI.

Palabras clave

Flujos, ancho de banda, clasificación de tráfico, estimación de probabilidad, aplicaciones, aprendizaje automático, algoritmo

Abstract

The complexity of the communications on Internet are growing continuously, so network managers search for new ways to identify the problems and anomalies that appear in networks every day. The bandwidth and the number of concurrent flows are two parameters that define a network. These parameters are determinant to know the features of networks and both are adequate to measure the performance. However, the relation between bandwidth and concurrent flows has not been studied in detail and the effectivity of this relation is still unknown.

The Master Thesis aims at improving the knowledge about the relationship between the bandwidth and concurrent flows in networks. This relation or ratio is analyzed over a set of network traces captured in a real environment. We believe that the conclusions may result useful for network managers, aware of the typical used bandwidth in their networks, can indirectly estimate the load in concurrent flows. We note that concurrent flows, in particular, is a key metric for tasks such anomalies identification, intrusion detection, and traffic classification.
Besides, the relation is studied for each application separately. This aims at assessing if applications determine the value of this relation, or, otherwise, is not significant.

To this end, it is necessary to provide the tools to extract, with the set of captures of network traffic, information about the network flows that we find in the traffic captures. The information of the flow includes the application it belongs to, and other temporal parameters that help us to calculate the relation between flows and bandwidth. In particular, the technique implemented to label the flows is deep packet inspection (DPI).

Once the tools to analyze the network traffic have been developed, we study the results obtained about the relation between flows and bandwidth. The question of stablish a scale, with this ratio, in the identified applications are discussed.

DPI tool extract diverse characteristics in every flow. We must search what characteristics are useful to identify the application of the flow. It must be verified that the values obtained depend on the application used.

After adequate parameters are identify the flows, they are included in a *machine learning* tool. Machine learning implements automatically algorithms to detect the application in a dataset that include some parameters. Finally, the results prove that this technique is a promising alternative to DPI.

Key words

Flows, Bandwidth, traffic classification, probability estimation, applications, machine learning, algorithm.

Agradecimientos

A mis amigos, los viejos y los nuevos.

A mi tutor.

A todo el que haya puesto su granito de arena para acabar este TFM.

Y a mis padres, que me han aguantado durante siete años de Grado y Máster…

¡Muchas gracias!

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

1.1 Motivación

En la actualidad, los gestores de red encuentran serias dificultades a la hora de realizar medidas sobre una métrica de red tan importante como es la concurrencia de flujos. Esta medida tiene una gran relevancia ya que multitud de sistemas en red funcionan a nivel de flujo, es decir, que realizan cierta tarea por cada flujo que puedan recibir. Sistemas que están basados en flujos son las aplicaciones de clasificación de tráfico, herramientas de detección de ataques por denegación de servicio u otro tipo de ataques o anomalías, herramientas de evaluación de prestaciones y routers software. Estos son buenos ejemplos para dar una idea de su importancia en Internet.

Ante este problema, el presente trabajo tiene como motivación principal contribuir en la mejora del conocimiento sobre la concurrencia de los flujos. Para ello, se plantea estudiar la relación existente entre el número de flujos concurrente con el ancho de banda en redes de comunicaciones. El principal aliciente de conocer esta relación es la mayor facilidad que existe en la estimación del ancho de banda de la red. Por ello, conocer la relación de flujos entre el ancho de banda de la red permitiría a los gestores calcular de manera sencilla el número de flujos concurrentes, multiplicando la relación flujos-ancho de banda por el ancho de banda. Por tanto, este trabajo pretende estudiar si tal relación, o ratio, es general en Internet, o por el contrario es heterogénea. Igualmente, pretende extraer si este ratio varia de forma significativa o no. La manera en que se ha planteado dicho estudio es separando los flujos según la aplicación, una forma novedosa a partir de la cual se pretende comprobar si este ratio, depende, en parte, de la aplicación que se esté usando.
En vez de estudiarse el tráfico como un todo se pretende estudiar el ratio flujos-ancho de banda independientemente por aplicación. De esta manera un gestor de red, con conocimiento del ancho de banda típico de su red, e indicios de la popularidad de las aplicaciones en su red, puede extrapolar que concurrencia va a haber en su red utilizando los resultados del ratio que este trabajo facilita.

El tráfico analizado viene proporcionado de manera conjunta, en consecuencia, será necesario dividir el tráfico por aplicaciones, lo cual representa un desafío. Por ello, se deberá proveer de técnicas de clasificación de tráfico, entre las cuales se profundizará en las técnicas DPI, tecnología para clasificar tráfico que se ha elegido en el presente trabajo.
Ante las dificultades existentes en la clasificación de tráfico, este trabajo buscará alternativas en la clasificación del tráfico de red, como son las técnicas de *machine learning*, las cuales se presentan como método alternativo a DPI cuando este no es viable, debido a que, en muchos casos, las trazas de red carecen de carga útil y DPI no es efectivo.

1.2 Objetivos

El presente trabajo tiene como objetivo contribuir en el mayor conocimiento de la relación entre el número de flujos concurrentes y el ancho de banda en redes de comunicaciones Para llegar a conclusiones sobre dicha relación, se han establecido diferentes objetivos que permitan mejorar el conocimiento sobre esto.
Vista la necesidad de que los flujos de red sean clasificadas por aplicación, se realizará un estudio sobre los métodos de clasificación de tráfico actuales, se apostará por las técnicas

DPI. De esta manera, en primer lugar, se deberá modificar la aplicación DPI de manera que permita analizar el conjunto de trazas de muestra. Por otro lado, la aplicación deberá extrapolar los datos temporales de cada flujo, además de la aplicación a la que pertenece. Esta necesidad se explica por la dificultad para calcular los flujos concurrentes. Con estos datos, se podrán extrapolar de manera que se pueda obtener los cálculos de flujos concurrentes.

Una vez obtenidas las métricas por aplicación de flujos concurrentes, se podrá, junto con el cálculo del ancho de banda, estudiar el ratio flujos-ancho de banda y extraer conclusiones sobre cada aplicación. Se estudiará si las aplicaciones pudieran tener relevancia sobre dicho ratio, determinando en parte el valor que pueda tener.

Una vez valoradas las aplicaciones según esta relación, y ante los inconvenientes para clasificar tráfico en red por DPI, por falta de trazas de datos con la información necesaria. Se van a buscar parámetros proporcionados por las trazas más sencillos de cuantificar que la relación flujos-ancho de banda. Con la información de dichas estadísticas se van a comparar las aplicaciones más populares y determinar si alguno de estos parámetros es útil para distinguir entre las aplicaciones.

Una vez analizado que parámetros son útiles para distinguir entre flujos de varias aplicaciones, se va a usar una herramienta de aprendizaje automático, con el objetivo de buscar aquellos algoritmos que sirvan para construir clasificadores de tráfico sin la necesidad de utilizar técnicas DPI.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Introducción,** donde se explica la motivación de cada capítulo del trabajo, se explican los objetivos que tiene y la organización de la memoria.
- **Estado del arte,** se hace un breve resumen sobre los métodos de clasificación de tráfico y se hace un repaso histórico sobre la tecnología DPI.**…**
- **Diseño,** se explica cómo funcionan las distintas aplicaciones usadas en el presente trabajo.
- **Desarrollo de los análisis,** donde se analizan todos los resultados obtenidos a partir de la aplicación Matlab y los diferentes programas desarrollados en el mismo.
- **Uso de Machine Learning para descubrir aplicaciones,** se explica todo el proceso de selección de muestras y algoritmos usados y se muestran los resultados obtenidos. Además, incluye un caso práctico.
- **Conclusiones y trabajo futuro,** donde se comentan los resultados finales y se sugieren futuras líneas de investigación.
- **Referencias,** que incluye la bibliografía.
- **Glosario,** para especificar términos que puedan ser poco conocidos.
- **Anexos,** donde se incluyen el resto de gráficas y tablas obtenidas a partir de los resultados de los capítulos 4 y 5.

2 Estado del arte

2.1 Introducción

En esta sección se muestra cómo ha evolucionado los métodos de clasificación e identificación de servicios en Internet. El cual ha adquirido una gran importancia en la última década debido al aumento de servicios y aplicaciones en tiempo real que requieren de una alta calidad de servicio (QoS). Reconocer bien el *mix* de aplicaciones de una red puede ayudar a realizar una correcta diferenciación de servicios y un buen dimensionado de las redes, sobre todo en redes de altas prestaciones.

Entre los distintos métodos para clasificar tráficos en este trabajo prestaremos especial atención a la técnica *Deep Packet Inspection (DPI)*, o Análisis de la Carga Útil de los paquetes. Se detallará como funciona esta técnica y se dará un breve repaso a la evolución desde sus primeros desarrollos hasta la actualidad.

Asimismo, y, en primer lugar, también se explicará el origen del conjunto de datos utilizado en este TFM, el cual ha sido proporcionado por MAWI Working Group que ha capturado diversas trazas de la red académica *WIDE*, y las ha hecho públicas, con carga útil incluida, para la comunidad científica.

2.1.1 Dataset de estudio. WIDE Project

Las trazas de red utilizadas en el presente TFM han sido proporcionadas por *MAWI Working Group* [13], una entidad dedicada a la investigación de Internet, con origen en Japón. La particularidad de este conjunto de trazas reside en que los paquetes almacenados cuentan con toda la carga útil *(*o *payload*). Es decir, en estos paquetes se podrá realizar un análisis DPI, el cual no sería posible realizarlo si los paquetes estuviesen truncados hasta las cabeceras de nivel 4, como ocurre en la mayor parte de trazas que se pueden encontrar de forma libre por Internet.
De hecho, este conjunto de trazas no es público y no se puede encontrar de forma libre por Internet, solo es proporcionado por MAWI para grupos y proyectos de investigación. El principal motivo de que estas trazas sean privadas es proteger la privacidad de los datos que se puedan encontrar en esta carga útil de los paquetes.

Para la recolección de estos conjuntos de trazas, MAWI ha tenido acceso a la red académica e investigadora WIDE, la cual es una red o sistema autónomo propio. creado en 1989, con el propósito establecer conectividad a las instituciones participantes del proyecto con la red de Internet, inicialmente conectando diferentes instituciones dentro de Japón y el extranjero. Desde ese momento a la actualidad dicha red ha evolucionado y crecido exponencialmente, cuenta con conexión a multitud de ISP tanto a nivel nacional como internacional, a través de puntos de interconexión centrales como NSPIXP-3 (ver Figura 2-1). Estos puntos de interconexión son espacios neutrales donde las empresas, entidades u operadoras intercambian información que viaja a través de sus propios sistemas autónomos, es decir, su red propia. Asimismo, WIDE forma parte de otra entidad más grande, APAN, la cual gestiona proyectos como el de WIDE para posibilitar una mejor conectividad de todas las redes del área Asia-Pacífico con el resto del mundo.

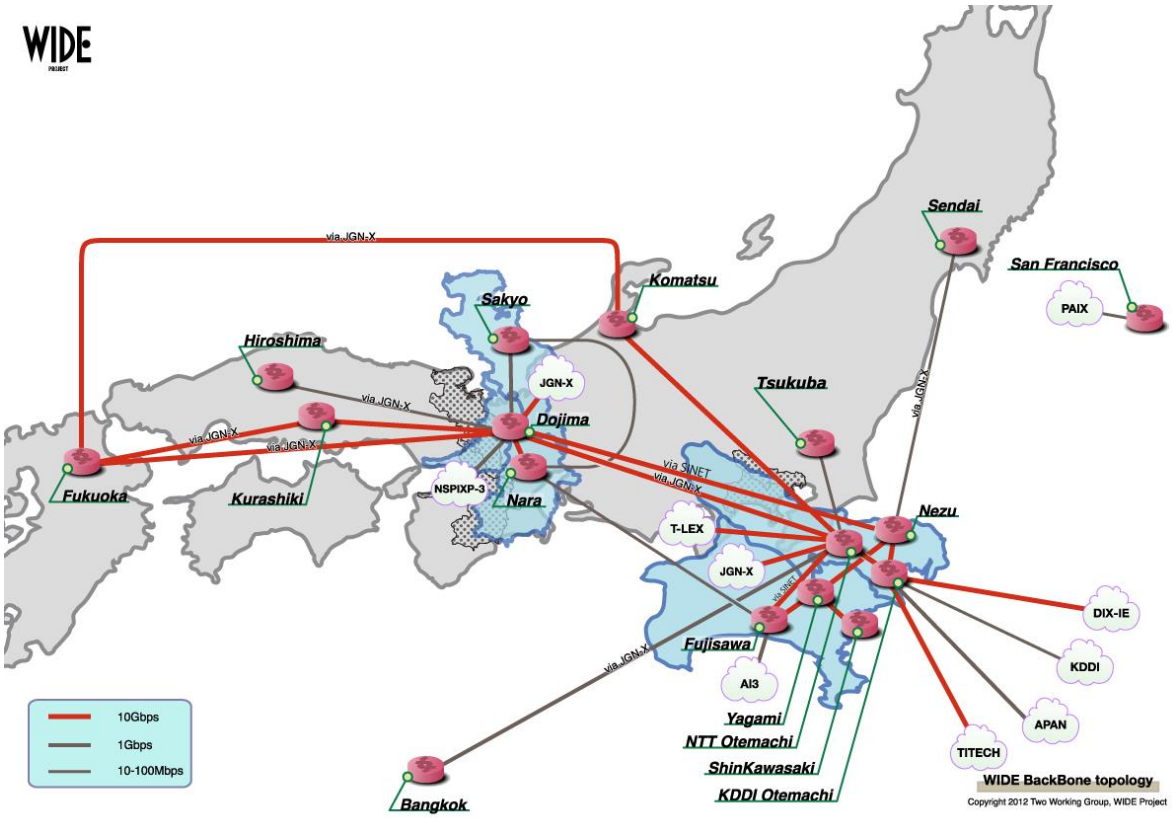


Figura 2-1: Topología de la red troncal de WIDE

Este esfuerzo conjunto y a nivel global en el que colabora MAWI tiene como finalidad principal resolver problemas y estandarizar tecnologías. Asimismo, contribuye al desarrollo de los recursos tecnológicos, con la mejora y mantenimiento de las infraestructuras, y de los recursos humanos, formando a las próximas generaciones
Por ello colabora además con organizaciones a nivel mundial como IETF e ISOC como miembro técnico de dichos consorcios. El proyecto de la red que forma WIDE está formado por más de 100 organizaciones entre empresas y universidades. Cuenta con diferentes grupos dedicados a distintos temas de investigación. WIDE provee una base de datos compartida a los distintos miembros para la investigación a través de la experimentación y demostración. Además, conecta distintos grupos investigadores para compartir y discutir información de manera continua.

La red troncal de WIDE, de donde se han recogido las trazas, tiene un ancho de banda de hasta 10 Gb/s en concreto el punto de conexión donde se han recogido las muestras usadas en el presente trabajo contaba con un ancho de banda de 1 Gb/s (desde 2007).

2.1.2 Métodos de clasificación de tráfico en Internet

Desde mediados de los 2000 de forma paralela al desarrollo de Internet ha ido aumentando la necesidad de clasificar el tráfico que corre por las redes de comunicaciones. El crecimiento exponencial de los datos y la necesidad de una baja latencia en las aplicaciones que han ido surgiendo en los últimos años, unido a un ancho de banda limitado hacen que sea necesario reconocer el *mix* de aplicaciones de la red. Según [15] las ventajas de poder clasificar el tráfico son principalmente las siguientes:

- Diferenciación de servicios: el hecho de poder diferenciar servicios en Internet tal vez sea el principal objetivo de la clasificación de tráfico. Aunque este hecho pone en cuestión la neutralidad de Internet, realmente desde siempre ha sido necesario diferenciar el tráfico para priorizar las comunicaciones de los servicios de emergencias, la gestión de la propia red de las operadoras y el uso de servicios críticos como los de asistencia o incluso servicios para usuarios *premium*.
  Actualmente, esta diferenciación o priorización de tráfico no se limita solo a estos servicios básicos, y las operadoras o ISPs favorecen determinados servicios o aplicaciones que funcionan en Internet, y países como EE. UU. ya han legalizado la no neutralidad o parcialidad de la red.
- Dimensionado de la red, un buen conocimiento de las aplicaciones y servicios que corren a lo largo de nuestras redes, y por tanto conocer las exigencias mínimas de estos pueden ayudar a realizar un diseño de la arquitectura de red de manera adecuada.
  La clasificación de tráfico no solo permite reconocer el *mix* de aplicaciones, sino también extraer características de los flujos y paquetes que corren en cada aplicación. Todas esas características deben tenerse en cuenta para cumplir diseñar y gestionar una red adecuadamente.
- Seguridad, una de claves para proteger a las redes es conocer la naturaleza de los ataques que puedan llegar a sufrir. Por ello, los métodos de identificación y clasificación de tráfico que se han desarrollado han tratado de descubrir datos maliciosos que circulan por Internet, previniendo a los sistemas ante posibles ciberataques. Un ejemplo de uso en la identificación y clasificación de red son los sistemas de protección ante *ataques de denegación de servicio* (*DDoS* por sus siglas en inglés) donde se utilizan herramientas de identificación que permiten bloquear flujos de datos de origen dudoso.
  La seguridad de los sistemas puede aplicarse tanto a gran escala, para grandes ISPs, redes empresariales o académicas hasta la seguridad a nivel doméstico.
- Políticas de red, esto se refiere a las restricciones que pueden presentar ciertas redes empresariales o académicas, las cuales establecen sus propias políticas en sus redes. Estas políticas pueden deberse a motivos de seguridad o simples estrategias para mejorar la eficiencia. Por ejemplo, muchas universidades no permiten el tráfico P2P en sus redes.

Aunque existen diferentes maneras de caracterizar el tráfico que circula en Internet, el presente trabajo se centra en el estudio de los flujos o sesiones que se producen en la red. Un flujo se define como un conjunto de paquetes consecutivos que comparten la misma dirección IP destino y origen, el mismo puerto origen y destino y usando el mismo protocolo [16] (lo que típicamente se llama quíntupla). Como la comunicación puede ser bidireccional, existen dos flujos por sentido, que a su vez pueden juntarse para formar una sesión (esto es, origen y el destino de la IP y el puerto intercambiadas). De esta forma se podrán reconocer paquetes enviados y recibidos en función del origen y destino.
En la Figura 2-2 se puede observar como vienen estructuradas las cabeceras en las capas de red (IP) y capa de transporte (TCP/UDP). Se ha remarcado los campos pertenecientes a la quíntupla que determinan el flujo al que pertenece el paquete.

Otro requisito indispensable para agrupar paquetes dentro de un mismo flujo es definir lo que se entiendo por paquetes consecutivos. Esto es, el tiempo máximo que se espera a un

nuevo paquete que coincida con la quíntupla sin llegar a considerar que el flujo ha acabado. En el presente trabajo, en consonancia con [1] se ha establecido este tiempo, típicamente conocido como *idle timeout*, en 30 segundos. En general, este valor se configura entre 15 [2] y 120 segundos [8].

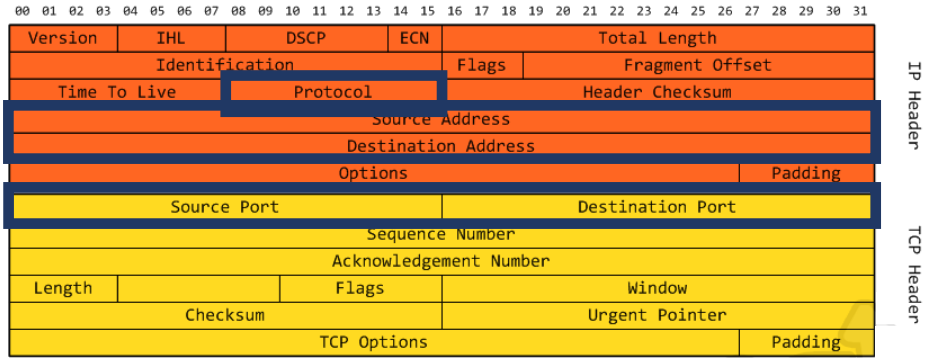


Figura 2-2: Estructura genérica paquete TCP/IP. Recuadrada quíntupla

Los mecanismos existentes para identificar y clasificar tráfico son varios, entre ellos los más importantes son los siguientes:

- Identificación mediante número de puerto, es el método tradicional que ha usado Internet siempre para reconocer los servicios que corren. Actualmente no es un método efectivo pues ya los servicios no usan los puertos asignados ya sea porque funcionan a través de otros protocolos de la capa de aplicación que tienen otro puerto asignado, por el uso de puertos dinámicos, o porque no les interesa ser identificados. Aunque en la actualidad presente gran imprecisión, es el más usado en multitud de sistemas donde la clasificación de tráfico no es crítica.
- DPI, el método usado en el presente trabajo, en el siguiente apartado detallaremos su evolución, se basa en el estudio de la carga útil del paquete. La carga útil se refiere a todos los bytes de información del paquete que no se encuentra en las cabeceras por debajo del nivel de aplicación. De manera genérica, este método busca secuencias de caracteres o ciertos patrones en la carga útil que coincidan con los de ciertos servicios o aplicaciones. No todos estos patrones son del todo fiables y depende de sistema usado existen mayor o menor confiabilidad.
- Identificación mediante características del tráfico. Consiste en estudiar distintos parámetros, que pueden ser de tipo temporal, como la duración del flujo, el tiempo entre paquetes (*interarrival)*, cantidad de datos del flujo, *throughput* que se refiere al cociente entre la cantidad de datos y el tiempo.
  También existen características que no dependen del tiempo, estos parámetros son el tamaño medio del paquete en un flujo, su desviación estándar u otras medidas matemáticas extraídas a través de paquetes agrupados en flujos.
  La sección 4.1, Análisis de los resultados, buscará identificar si aplicaciones detectadas por DPI guardan similitud en algunas de las características nombradas.
- Aunque existen otros mecanismos, como por ejemplo el método de identificación de aplicación por medio de la correlación temporal del flujo [4] o el método que estudia los comportamientos de los sistemas por los que concurre el tráfico [4]. Normalmente solo son útiles en ciertos casos.

Los tres nombrados anteriormente son los más usados, aunque no lleguen a ser del todo fiables como se comentó en la identificación mediante puertos.

El siguiente apartado muestra de manera más detallada como fue evolucionando DPI y como han surgido distintos desarrollos destinados a recoger distinto tipo de tráfico.

2.1.3 Tecnología DPI. Evolución y desarrollo

Las tendencias actuales indican que la tecnología DPI en el mercado está en auge y continuo crecimiento, y así se prevé que continúe al menos durante los próximos años. En las figuras inferiores se muestran distintas estimaciones de los fabricantes. (Fuentes: yahoo e ipfabric).

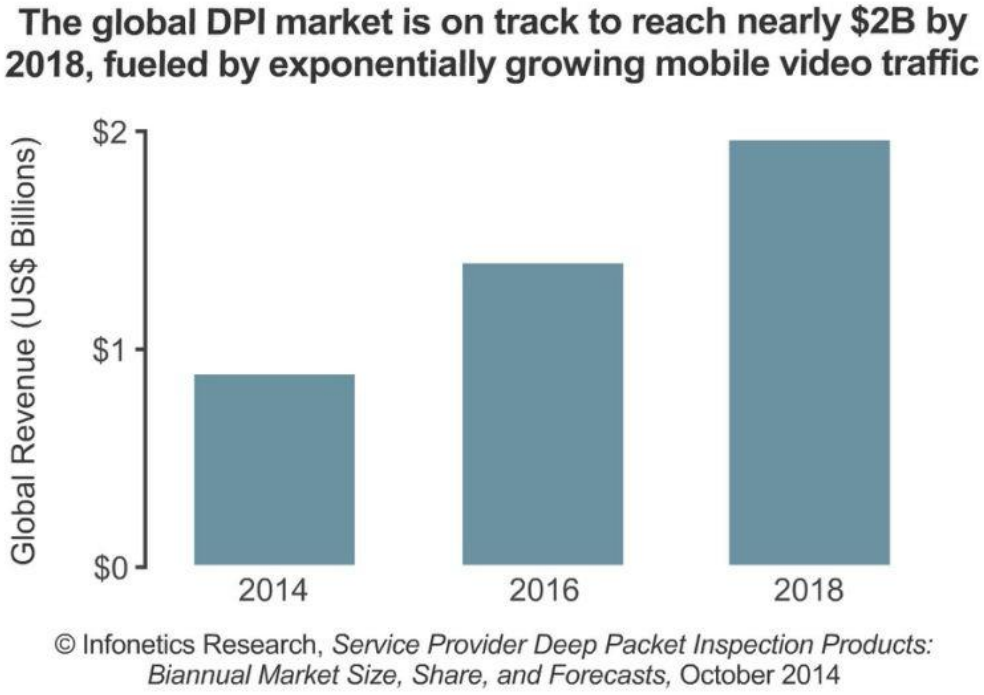


Figura 2-3: Evolución gasto en tecnologías DPI [yahoo.com]

Aunque pueda parecer por las Figura 2-3 y Figura 2-4 que DPI es una tecnología muy nueva, lo cierto es que tiene sus orígenes hace casi 20 años. Se va a hacer un breve repaso a la historia de esta tecnología a continuación.

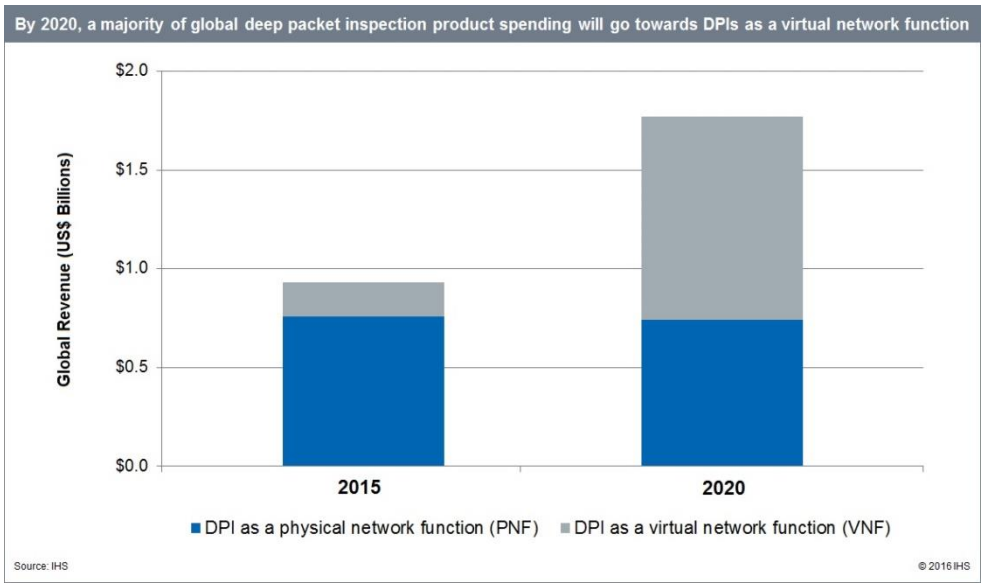


Figura 2-4: Estimación evolución gasto DPI [fuente: ipfabrics.com]

Desde principios de los 2000 los gestores de red de Internet se dieron cuenta de que la tradicional clasificación basada en puertos, que IANA establecía dando a cada servicio de Internet un determinado puerto, no es un método preciso a la hora de identificar tráfico y clasificarlo [10].

Los primeros desarrollos de DPI fueron desarrollados con el objetivo de proteger sistemas y servidores de la red de intrusos, para evitar ataques informáticos al mismo. Estos sistemas denominados IDS por sus siglas en inglés realizan un escaneo de los paquetes de red que entraban a una determinada red. Este sistema realiza un análisis del tráfico y lo compara con patrones, o firmas, de paquetes que anteriormente hayan realizado algún tipo de ataque sobre esa red. Tiene el inconveniente de que para poder establecer patrones primero deben haber ocurrido ataques con dicho patrón, ya sea a la misma red u otras. El uso de estos patrones o firmas como modelo de reconocimiento es en el que se basó DPI para la clasificación de tráfico. El IDS, cuando reconoce paquetes sospechosos que pudiesen comprometer la integridad de la red, levanta alarmas para avisar a los operadores de red que la red puede ser atacada. Estos sistemas tienen el inconveniente frente a los cortafuegos que dejan pasar el tráfico y ya una vez dentro lo evalúan, pero no limita la red como lo hace un cortafuegos, cuando funcionan de manera conjunta se denominan sistemas de prevención de intrusos, IPS, por sus siglas en inglés. El uso de estos patrones en IDS, por tanto, asegura buena fiabilidad ante ataques ya conocidos, pero no genera alarmas ante nuevos intrusos desconocidos. El factor más importante de un buen IDS, como lo será de DPI, será tener una amplia y actualizada base datos sobre el tráfico que se quiere identificar. Otro inconveniente que puede ocurrir en los IDS es la aparición de falsos positivos, esto es, que un determinado flujo de red sea clasificado de una determinada forma cuando no lo es. En DPI también se producirá este hecho, así como el hecho de ver como un flujo es clasificado dos o varias veces, denominado comúnmente como doble coincidencia (o *doble matching*).

Más adelante, aparte de prevenir las redes de ataques, se ha visto como es útil en determinados casos identificar el tráfico que corre por Internet normalmente de determinados servicios. Esta necesidad surgió generalmente en entornos académicos o

empresariales, cuyo objetivo además de aumentar la seguridad era la de favorecer o limitar ciertos servicios. El ejemplo más cercano es el de las redes académicas, que no suelen permitir el tráfico P2P. De hecho, los primeros avances en DPI, surgieron a partir de la necesidad de limitar el tráfico P2P, que tiene la particularidad de realizar conexiones con multitud de flujos activos. Además, este tipo de aplicaciones dejó de usar un determinado puerto para su uso con el objetivo de "esconderse" en la red y no ser detectado. Por ello, algunas de las primeras evoluciones de DPI fueron más encaminadas a detectar servicios basados en P2P.

Uno de los inconvenientes a los que se ha tenido que enfrentar DPI y que en parte han retrasado su desarrollo ha sido la reticencia de los usuarios a que sus datos se vean comprometidos y tanto los operadores como otras entidades encargadas de monitorizar el tráfico puedan observar el tráfico. Es un tema a tener en cuenta, pues como se ha comentado anteriormente en el apartado 2.1.1, no es fácil encontrar datos de tráfico completo de la red para ser analizado. Aunque este hecho es bastante relevante los defensores de DPI replican que cuando se investiga los paquetes de red se investigan únicamente un numero de paquetes iniciales del flujo de red, los correspondientes a las cabeceras de los protocolos de aplicación, buscando la coincidencia de un flujo con ciertos patrones y que no se paran en ningún caso a recoger información sensible de los usuarios. Por ejemplo, en [5] se defendía que los patrones de reconocimiento por DPI solo buscaban en las cabeceras de los distintos protocolos de aplicación que se usan en Internet, cosa que se hace de manera necesario por las aplicaciones de red que direccionan los flujos para facilitar el tráfico entre un punto y otro. En la Figura 2-5 se observa como el análisis DPI de la capa de aplicación solo se fija en sus cabeceras y los protocolos que se están usando y no se fija en los datos. En sentido contrario, sus detractores demuestran que la capa de aplicación del modelo OSI solo intervienen los dos extremos que realizan la comunicación, o sea, el receptor y el emisor. La realidad es que no es necesario llegar a la capa de aplicación en ningún punto intermedio para comunicar dos terminales en Internet.
En [5] también se argumenta que muchos de los sistemas DPI, aprovechando las técnicas de *machine learning*, una vez se detecta cierto patrón el sistema no para a decodificar la información, simplemente la etiqueta como cierta aplicación/servicio.

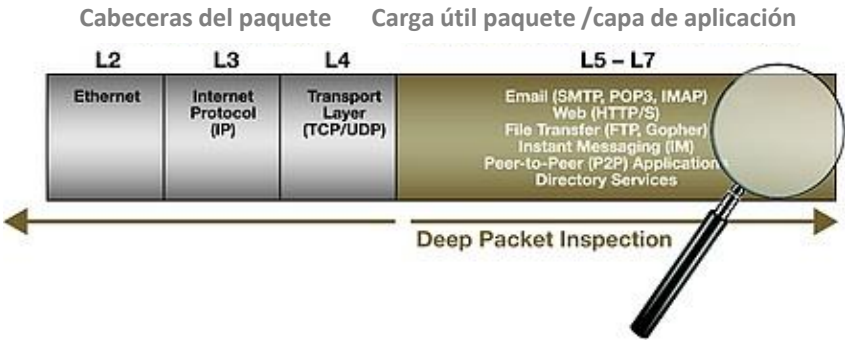


Figura 2-5: Inspección DPI en la capa de aplicación. [Fuente: www.ipfabrics.com]

La tecnología DPI, ha sido un campo en el que el ámbito comercial suele haber ido por delante del SW libre y también del ámbito académico. De esta manera, en la actualidad son empresas privadas, de ciberseguridad, las que ofrecen mejores soluciones incorporando esta tecnología. De todas maneras, a lo largo de los últimos años han ido apareciendo diversas aplicaciones de SW libre que son capaces de realizar DPI, de manera relativamente fiable al tráfico de redes. En [4] se realiza un estudio de fiabilidad sobre los sistemas DPI de SW libre más significativos en los últimos años, que son *Hippie, L7-*

Filter, IPP2P, nDPI, libprotoident y OpenDPI, este último el más relevante. Sin embargo, en el presente trabajo se ha partido del sistema *L7-Filter* [19], el cual se ha modificado en ciertas partes el código para poder realizar pruebas.

Uno de los factores que obstaculizan el uso de DPI en ciertos sistemas es la aparición de servicios que requieren el trasvase de datos en tiempo real, como son las aplicaciones de *streaming*, de VoIP, o aplicaciones empresariales como las que tienen los bancos para sus operaciones en bolsa. Todo este tipo de aplicaciones tienen unos requerimientos de latencia y retardos mínimos para asegurar una calidad del servicio, suficientes para proveer correctamente el servicio. Estos requerimientos chocan con el gasto de tiempo computacional que demandan muchos sistemas DPI. Aunque se han ido desarrollando sistemas cada vez más rápidos en muchos casos los sistemas DPI sacrifican precisión a cambio de aumentar la velocidad de análisis. En [4] se nos presentan multitud de alternativas, como por ejemplo el uso de lo que ellos llaman *DPI light*, un sistema DPI que solo analiza ciertas partes de paquetes. A consecuencia de esto, los operadores de red han de encontrar un equilibrio entre estos sistemas y el resto de los elementos de red que conviven en sus sistemas autónomos, pues un mal dimensionado puede llegar a ocasionar auténticos cuellos de botella.

A raíz de la evolución de manera heterogénea de las técnicas basadas en DPI, algunos fabricantes de esta tecnología se dividen en dos grupos bien diferenciados las técnicas DPI actuales [18]:

- La clasificación extendida, que implica procesar cada paquete que pasa por el sistema. Se utiliza para clasificar todo el tráfico en general. Tiene menor rendimiento, pero es capaz de analizar conjuntos de tráfico de muchas clases.
- La clasificación específica, que agrega un componente de reglas para poder diferenciar entre paquetes relevantes o no. Las reglas de paquetes relevantes son configuradas en el clasificador, que en función de esa configuración escoge qué flujos de tráfico son importantes. Su objetivo es detectar cierto tipo de aplicaciones en red, por lo que su propósito es mucho más específico que el grupo anterior. Tienen mayor rendimiento y eficiencia, ya que está optimizado para ciertos casos.

Las últimas evoluciones en servicios basados en técnicas DPI han ido en dirección a ir más allá de la simple detección de alas aplicaciones a la que pertenecen un flujo. En la actualidad, sobre todo en sistemas comerciales, el contenido y la información se procesa de una manera más amplia. En lugar de detenerse en el punto en el que se pueden extraer el tipo de aplicación, se extrae además información extra, por ello se diferencia entre transacciones de cliente a servidor y de servidor a cliente dentro de la aplicación. Esto ayuda a, por ejemplo, identificar un envío de correo de un envío en borrador en un servidor de correo. O la capacidad de detectar interacciones fallidas dentro de una sesión por cualquier razón. Otra innovación en los actuales sistemas es la capacidad de identificar grupos de flujos, que son parte de la misma sesión por parte de un usuario. Esto se debe a que en la actualidad muchas aplicaciones de Internet no usan un puerto e IP fija cuando establecen una conexión y usan de manera dinámica tanto los puertos como las direcciones IP durante una sola sesión completa de dicha aplicación. Las técnicas de DPI más avanzadas son capaces de agrupar estos flujos porque suelen contener algún tipo de identificador que asocia dichos flujos en una misma sesión realizada por un usuario.

Figura 2-6: Casos de uso DPI.

En la Figura 2-6 se pueden observar multitud de aplicaciones donde se usa DPI. Por tanto, en la actualidad se puede concluir que esta tecnología está más que integrada en la mayoría de los sistemas que requieren de cierto nivel de seguridad. Los siguientes dispositivos, ya sean físicos o virtuales, utilizan todos algún tipo de técnica DPI:

- Filtros anti-spam de correo electrónico
- Filtros antivirus para contenido web o de correo electrónico.
- Sistema de detección de intrusos (IDS)
- Cortafuegos

Bien es cierto que cada uno de los sistemas antes vistas no usan técnicas DPI de la misma manera, así mientras un filtro de correo anti-spam buscará algún tipo de cadena de caracteres en el contenido del correo sospechosa, que puede dar lugar a falsos positivos. Mientras que en entornos empresariales cuentan con redes securizadas que tienen cortafuegos que además de realizar técnicas de DPI incorporan mecanismos de *machine learning* que ayudan a clasificar el tráfico a partir de las características del mismo. Dependerá de los operadores y gestores de red que tipo de servicio es más adecuado en función de las necesidades y requerimientos de funcionamiento de la red.

Aunque se supone que gran parte de las comunicaciones en la actualidad se producen de manera encriptada el ejemplo más claro ha sido la progresiva sustitución de HTTP por HTTPS o el uso de P2P “ofuscado”, en [5] se duda de la seguridad del segundo, y en el caso del primero, no es problema para un sistema DPI que estos se encuentren encriptados, usando el ejemplo de HTTPS, este funciona inspeccionando campos de los certificados del protocolo TLS donde la web visitada aparece en claro En este caso no es necesario en muchos casos conocer esta información pues requiere únicamente de la información de los patrones que tienen los flujos de red de esta aplicación, u otras encriptadas, a partir de las cuales conocen o prevén que tipo de aplicación lleva ese flujo. La mejor manera de obtener esta información, en todo caso, es mediante entrenamiento previo del clasificador para conocer dichos patrones.

En conclusión, los sistemas DPI se erigen como parte básica en la seguridad en la red. Asimismo, si bien DPI presenta algunas limitaciones cuando se trata de analizar tráfico en tiempo real, es una herramienta muy útil para los gestores de red a la hora de realizar análisis *a posteriori* de los datos que han pasado por su red. La correcta identificación y clasificación de estos datos, divididos por flujos de red, puede ayudar a reconocer, de manera adicional, patrones de red según el tipo de aplicación basados en características temporales y de volumen de los mismos, como pueden ser el ancho de banda, la duración de los flujos o el tiempo entre llegadas de los paquetes. El uso de *machine learning* en estos sistemas se prevé fundamental para seguir mejorando las prestaciones, sobre todo en aplicaciones en tiempo real donde en ocasiones no es posible analizar la carga útil del paquete debido a la encriptación.

Como se ha explicado, todas las herramientas antes descritas han ido evolucionando según aumentaba la complejidad de Internet, y DPI utiliza actualmente técnicas muy avanzadas donde no solo detectan el tipo de aplicación, sino que extrae información sobre el contenido y el comportamiento de cada flujo en particular. Estas técnicas quedan fuera del alcance del programa usado *l7-Filter*.

3 Diseño

3.1 Sistema DPI L7-Filter

Este apartado se va a centrar en el sistema DPI usado en el presente trabajo, se ha usado una aplicación desarrollada en C, basada en una de las técnicas DPI nombradas en apartados anteriores, *L7-filter*. Este clasificador es capaz de categorizar paquetes IP analizando la parte de la capa de aplicación. La mayor ventaja de esta herramienta es que hace posible la identificación de multitud de protocolos, más de 100, entre ellos los más populares de P2P, como eDonkey, Bittorrent o Gnutella.

L7-Filter se basa en expresiones regulares para identificar el protocolo de red que se está usando a nivel de aplicación. Por ello, *L7-Filter* incorpora una serie de firmas que incorporan estas expresiones que por defecto son usadas para detectar ciertas aplicaciones de tráfico. Esta técnica, usada juntos a sistemas basados en Linux, permite la identificación del tráfico independientemente del puerto usado por los flujos de red.

Se ha optado por una solución de SW libre y basada en Linux porque daba la oportunidad de poder realizar modificaciones del programa a gusto del programador. Además, este sistema está en lenguaje C, en el cual tenemos un mayor conocimiento que otros lenguajes de programación. Las principales ventajas que se han encontrado al usar C han sido:

- Programación más aproximada al hardware sin necesitar del uso de lenguaje ensamblador.
- Posibilidad de mayor optimización con el código C comparado con otros lenguajes de programación de alto nivel.
- Alta velocidad de compilación.
- Facilidad para manipular el código.
- Gestión de memoria dinámica y la posibilidad de usar punteros y otras herramientas para ahorrar en el consumo de recursos.

Estos factores enumerados nos han ayudado a realizar simulaciones con bastante necesidad de procesamiento en un ordenador convencional. De todas maneras, el uso de gran multitud de firmas para detectar multitud de aplicaciones ha supuesto a la larga una carga computacional muy alta.

El programa SW de *L7-Filter,* no está concebido en cualquier caso para realizar DPI sobre sistemas en tiempo real, aunque la base del funcionamiento y las técnicas DPI usadas serán las mismas en dicho caso, cambiando únicamente el modo de entradas de los paquetes, las simulaciones realizadas sobre el sistema han sido sobre trazas de red que ya se habían guardado previamente. Estas trazas, como se comentó previamente, han sido proporcionadas por el organismo MAWI. Estas trazas no son de uso público y como cuentan con toda la carga útil donde se encuentran las cabeceras de los protocolos de aplicación para poder ser analizadas con mayor eficacia por nuestro sistema DPI.

En los siguientes apartados se describirá la técnica DPI implementada en el programa, así como las modificaciones realizadas para alterar el funcionamiento del programa para nuestro beneficio.

3.1.1 Funcionamiento programa

El código implementado en C funciona de manera similar al resto de técnicas DPI, el programa recibe una captura o traza de paquetes en cada ejecución del mismo, el formato d la captura debe ser *pcap,* pues es la librería que usa el programa. De igual manera en la ejecución se establecen los siguientes parámetros que determinan la profundidad y análisis.

- **Numero de paquetes máximo en cola** para análisis por flujo, como el programa realiza el análisis por flujo debe guardar en memoria la carga útil del número de paquetes que se especifique, más adelante explicaremos como encola los paquetes y analiza.
- **Carga útil máxima**, al igual que sucede con los paquetes, un flujo tendrá una carga útil acumulada por cada paquete nuevo que deberá ir siendo descartado para no llenar la memoria, se introduce en KB.
- **Tiempo entre limpiezas**, se refiere al tiempo en segundos que espera el sistema entre borrado y borrado de los flujos que se guardan en las tablas hash de flujos y reglas activas.
- **Tiempo máximo entre paquetes**, para que un grupo de paquetes pertenezca a un mismo flujo, además de la coincidencia en la quíntupla ya comentada, se debe establece con este parámetro el *timeout,* o tiempo en segundos máximo que puede haber entre un paquete y uno con la misma quíntupla.

El programa lee el fichero y lo primero que hace es crear una tabla hash donde se situarán la quíntupla que representa a cada flujo concurrente, las direcciones IP destino y origen, los puertos origen y destino y el protocolo, que será UDP o TCP, así como una tabla hash dedicada a cada aplicación o firma que será comparada con el flujo. Estas tablas hash son una forma que tendrá el programa de conocer cuáles son los flujos activos durante toda la ejecución del programa, ya que va insertando/eliminando en función de si aparece/caduca un determinado flujo.
El proceso principal del programa abre la traza de datos, que debe ser en formato *pcap* necesariamente, este formato es de SW libre y es usado por la mayor parte de fabricantes en todo el mundo. El programa llamará de forma recursiva a una función por cada paquete leído en la traza. Cada paquete será analizado y se obtendrá los datos de las distintas cabeceras, tanto el nivel de enlace (direcciones MAC), como los niveles de red y transporte, de los cuales se obtendrán los valores de la quíntupla que proporcione los datos necesarios para saber si pertenece a un flujo o hay que crear un nuevo flujo. Después de esto se extraerá la carga útil del paquete, que incluirá las cabeceras de la capa de aplicación. El paquete no será aceptado ni analizado en el caso de que no sea paquete UDP o TCP, por ejemplo, los paquetes del protocolo ICMP y los paquetes ARP (protocolo a nivel de enlace) serán siempre descartados, así como los paquetes malformados que no son capaces de ser analizados. La cantidad de estos paquetes descartados en la traza suponen un porcentaje nada desdeñable dentro del conjunto de datos. En la Figura 3-1 se puede observar el curso que siguen los paquetes desde que entran en la aplicación hasta que son procesados o descartados.
El programa comprobará si la quíntupla se encuentra en la tabla hash dedicada para los flujos y en caso negativo se insertará dentro de la tabla. Con cada nuevo paquete, se irá anotando las estadísticas agregadas del flujo al que pertenezcan, estas estadísticas se refieren a la duración y al tamaño total del flujo, sumando en cada iteración lo que corresponda según las características del paquete. Si el tiempo asignado a un paquete fuese mayor a la suma del anterior paquete del mismo flujo con el Tiempo máximo entre

paquetes, el programa exporta todos los paquetes anteriores como un flujo, creando un nuevo flujo para este paquete.

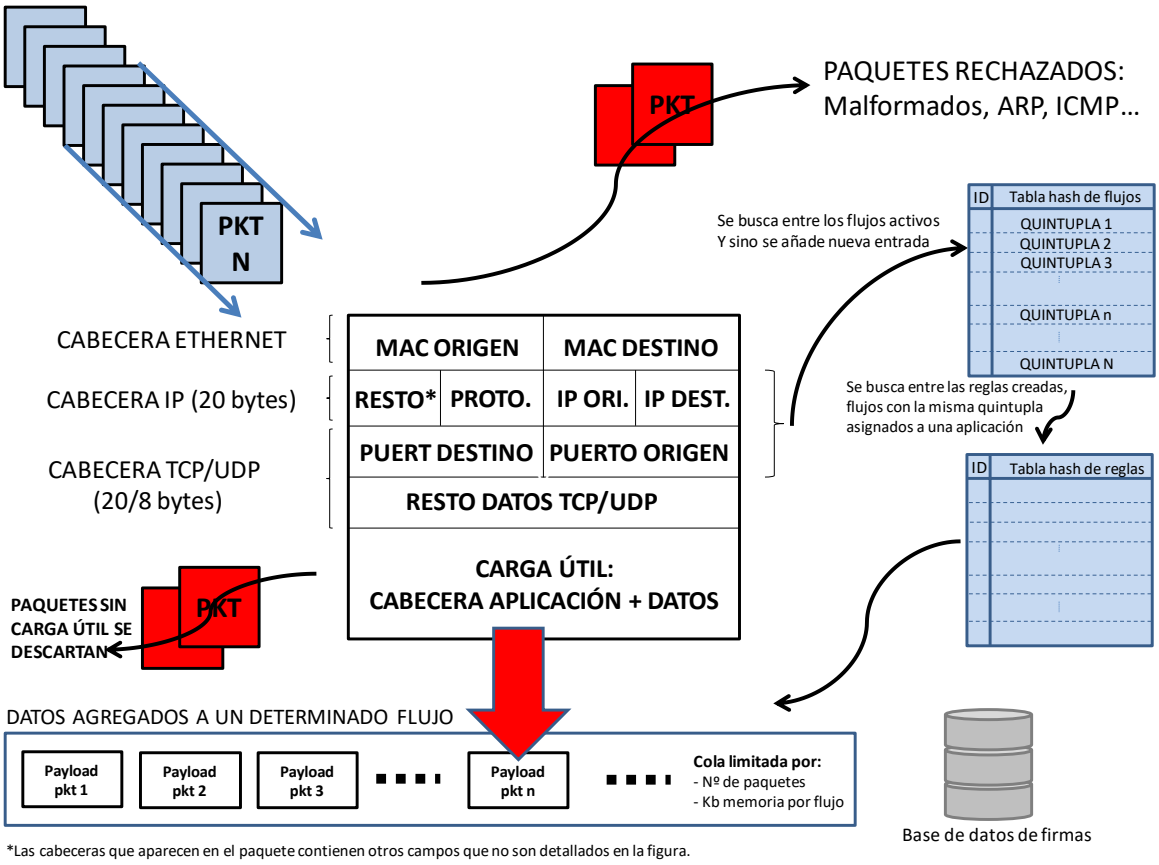


Figura 3-1: Funcionamiento aplicación DPI *L7-Filter*

Cuando un flujo termina porque se ha cumplido el *timeout*, la carga útil agregada de todos los paquetes que había sido guardada en memoria es liberada y todas sus estadísticas son volcadas a un archivo de salida, que será utilizado posteriormente para calcular características del flujo.

El programa, con cada nuevo flujo que se crea, guarda un espacio de memoria de forma dinámica del parámetro introducido como Carga útil máxima al ejecutar el programa. Por ello se deberá establecer un parámetro asequible para no saturar el sistema en cuanto a memoria. Asimismo, con cada nuevo flujo se asignan a través de una estructura de datos todas las características que se extraerán del flujo, tales como duración, numero de paquetes, direcciones IP (en decimal) y muchos más. Algunos de estos nos servirán en nuestros análisis en el presente trabajo. Además, se guarda la quíntupla en la tabla hash antes comentada.

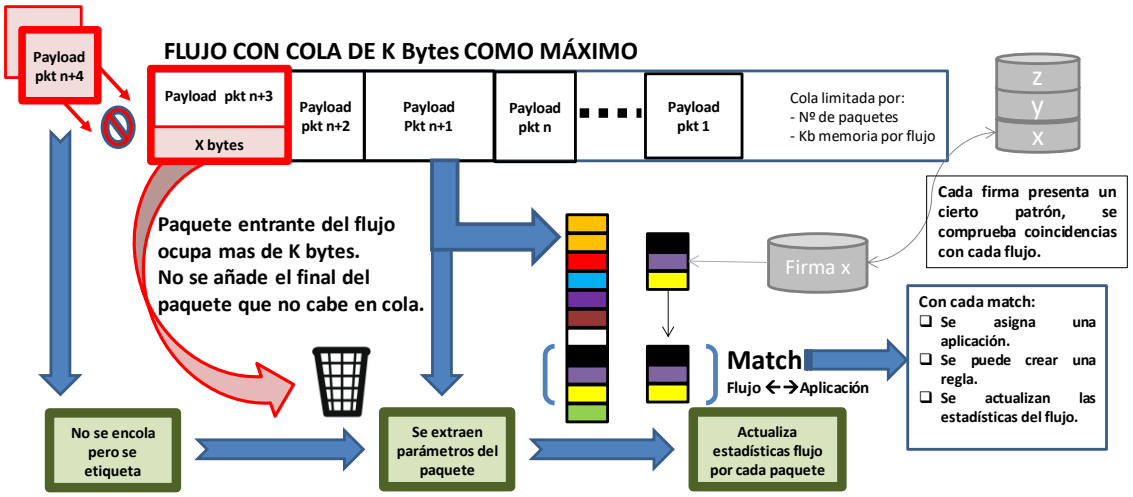


Figura 3-2: Desborde de cola por cantidad de datos

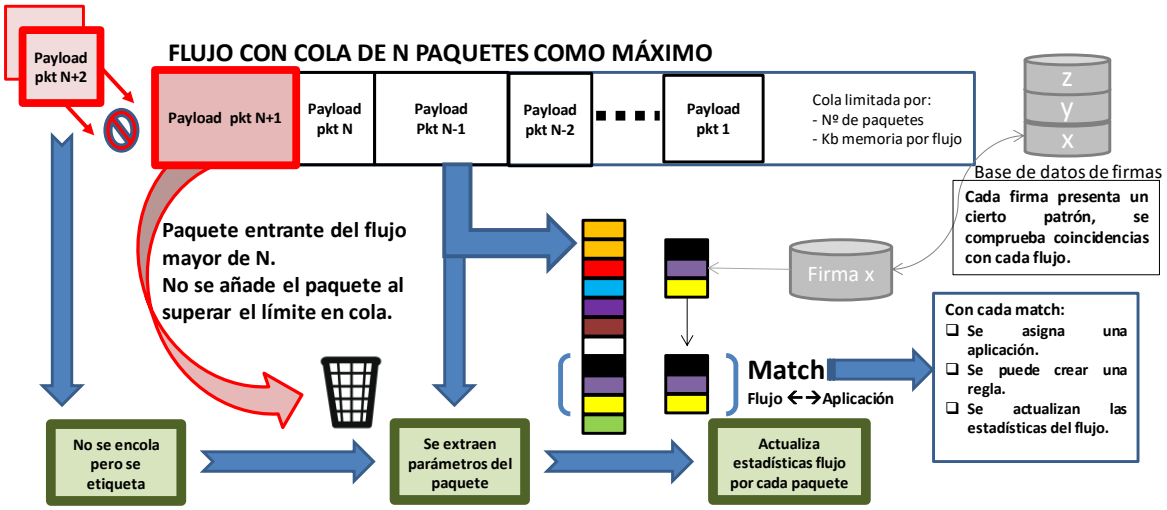


Figura 3-3: Desborde de cola por número de paquetes

Cuando el paquete que se analiza ya pertenece a un cierto flujo, el flujo al que afecta se actualiza agregando al flujo los bytes de carga útil del paquete, y recalculando las estadísticas con la agregación de este nuevo paquete. Asimismo, una vez ha agregado los bytes al flujo el programa pasa las firmas que tenga en la base de datos para tratar de identificar dicho flujo con determinada aplicación.

Tras realizar todas estas operaciones, el programa ya realiza las funciones para identificar cierta aplicación en cada flujo mediante las firmas. Con cada nuevo paquete agregado a un flujo este será, analizado en su conjunto, es decir con la suma ordenada de todos los bytes de todas las cargas útiles de los paquetes. Como el programa asigna una carga en bytes y una cantidad de paquetes máxima, la misma para todos los flujos, habrá algunos flujos más largos que dicha carga en bytes (Figura 3-2) o paquetes (Figura 3-3). En estos casos, cuando se llega al tope de carga los siguientes paquetes que pertenezcan al flujo serán descartados y no serán analizados por la aplicación. Las estadísticas de dichos paquetes descartados sí que serán tenidas en cuenta cuando se incluyan las estadísticas finales del flujo, que es la agregada de todos los paquetes.

En la función de identificación se contrastará cada firma de la base de datos con la carga útil del agregado. Será aceptado el *multiple matching,* o identificación múltiple, que se refiere a que un flujo sea identificado por dos o más aplicaciones. Este supuesto que se da en algunos casos se debe a que la carga útil incluye de manera accidental o a propósito un patrón típico de una aplicación que no es la que realmente pertenece el flujo. Por ello, en ciertos casos tendremos varios protocolos detectados en un mismo flujo.
Cuando una aplicación es detectada en el flujo la firma correspondiente a la aplicación no se vuelve a comprobar dicha firma por el flujo, pero la memoria reservada para el flujo si se mantiene, ya que este seguirá analizándose hasta que bien se pasen todas las firmas sobre el flujo o bien se alcance el número máximo de identificaciones, que es de cinco aplicaciones. Este método hace que el análisis sea mucho más lento, pero lo dota de una mayor seguridad.

Además, el programa incluye, de manera opcional, la identificación tradicional mediante puertos. Siempre que se indique previamente al ejecutar el programa, además de realizar DPI sobre los flujos comprobará si el puerto destino u origen, da igual al suponerse que la comunicación es bidireccional, pertenece a un puerto que está directamente relacionado con una aplicación en la base de datos del programa. En caso positivo, se asignará de la misma forma que se hacía con DPI un determinado servicio o aplicación al flujo estudiado.
En los siguientes apartados se verá como en la práctica en la mayoría de flujos no se encuentra más de una aplicación y solo un pequeño porcentaje tiene *doble matching* y en algún caso excepcional se puede encontrar *triple matching.*
Con cada identificación, ya sea DPI o puertos, si esta estuviera activada, el programa puede crear reglas a partir de los flujos y firmas que se usan en el análisis.

La tabla hash de reglas que se crea permite trabajar con reglas dinámicas. Está pensado para identificar tráfico mediante protocolo. Esto es, tras la detección de una aplicación o cualquier suceso se crea una regla por la cual todo el tráfico entre dos IPs determinadas y puertos se marca como de una cierta aplicación. Para ser más flexible se puede sustituir cada uno de estos puertos o IPs por un número indefinido lo que indicara que vale cualquiera (protocolos con sesión de control y transferencia). Las reglas pueden tener un tiempo de validez determinado.
Las reglas permitirían, por ejemplo:

- Una vez sabemos que hay tráfico de una aplicación en una IP y puerto todo el tráfico es de esa aplicación en esa IP-Puerto.
- Si detectamos un canal de señalización (como ocurre en protocolos P2P) podríamos detectar el de datos.

De nuevo, esta función, al igual que la habilitación de los puertos, es opcional al momento de ejecutar el programa, por lo que cuando no se habilite esa opción el programa no creará reglas. Cuando se habilite se creará una tabla en la cual se le pasará cada flujo recién identificado por firmas y su protocolo junto con la hora actual. Debe tener una parada por cada una de las aplicaciones que generan reglas.

Todo el proceso descrito se realizará con cada paquete, por ello, será necesario realizar un estudio que encuentre un término medio entre la profundidad de análisis del programa con la velocidad con la que se vaya a realizar. Una vez acabados todos los flujos el programa limpia de la memoria todas las tablas. Esta última opción podría modificarse en el programa para mantener las reglas para futuras ejecuciones.

Si cuando un paquete llega a un determinado flujo, y el flujo ya contiene ocupada la **carga útil máxima** o el **número máximo de paquetes en cola** el paquete no será añadido para su análisis en DPI, pero si se añadirán sus características para poder actualizar después las estadísticas completas del flujo.

3.1.2 Modificaciones realizadas sobre el programa

El programa, aparte de todas las funciones descritas, ha sido modificado para poder realizar ciertas funciones adicionales:

- Fuente de datos externa, se refiere a que el archivo o archivos de trazas este en una ubicación diferente a la carpeta donde se ejecuta el programa.
- Simulación de un conjunto de archivos, el programa solo aceptaba como entrada un archivo con formato leíble por la librería de lenguaje C *libpcap/tcpdump*. Con las modificaciones realizadas se puede dar como entrada un fichero de texto plano que contenga las rutas a los diferentes archivos de trazas que serán analizados. Esto permite realizar análisis sobre conjuntos de datos muy grandes, como ocurre en el presente trabajo.
- Durante la ejecución del programa, por último, con cada paquete ya analizado se crea una función que calcula una serie de estadísticas adicionales a las ya creadas y las termina volcando sobre un fichero de datos limitado por comas, que será utilizado para extraer y obtener todos los datos de interés de los flujos estudiados. En el apartado 3.2.1 se explicarán en detalle todas las estadísticas que se vuelcan a dicho fichero.

Estas implementaciones no forman parte de la base del programa de *L7-Filter* y han sido añadidas al programa para obtener información extra más allá de la identificación de aplicaciones. En la Figura 3-4 se puede observar como se incluyen dichas mejoras.

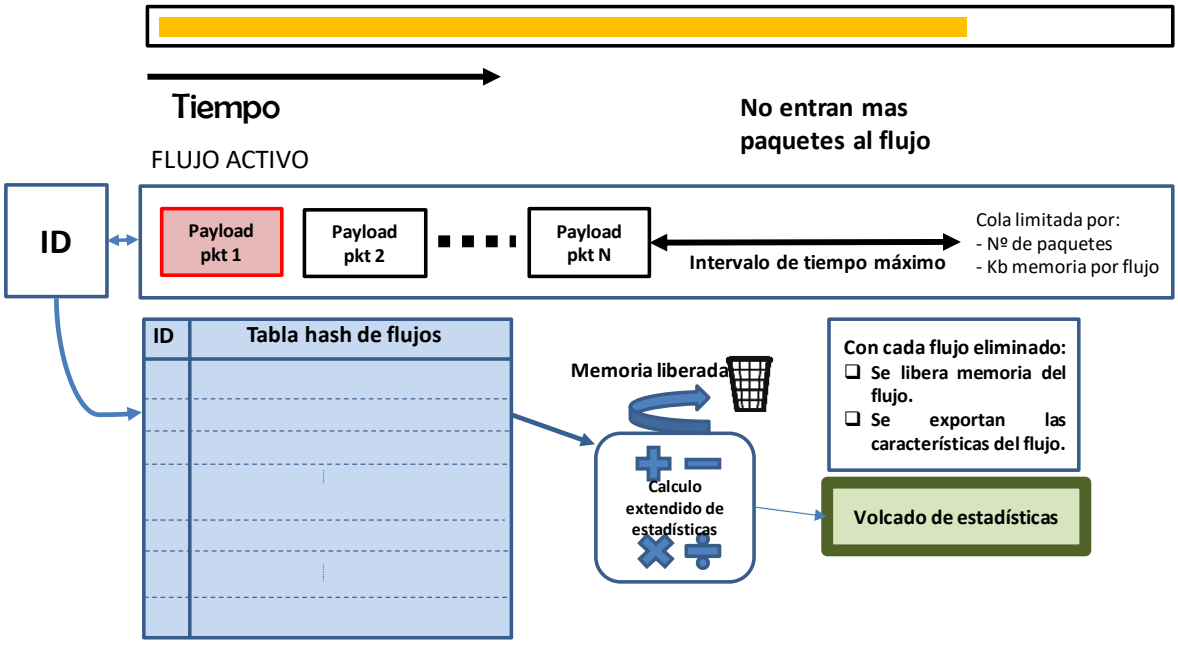


Figura 3-4: Mejoras realizadas sobre aplicación DPI *L7-Filter*

3.2 Diseño de programa para representar datos

Como se ha comentado, el programa en DPI además de realizar las tareas de identificación de firmas para reconocer ciertas aplicaciones incluye en el desarrollo final usado en el presente trabajo un volcado de los datos sobre un fichero de valores separados por comas. Este fichero, que en crudo es imposible de analizar los datos debido a la gran cantidad de datos volcados sobre el mismo, ya que los ficheros generados contienen de media unos 10 GB.

El objetivo de este apartado es explicar que datos se vuelcan sobre estos ficheros, y como a partir del SW de Matlab se han conseguido interpretar para obtener datos más concretos y gráficos que ayuden a entender mejor todas las estadísticas obtenidas.

3.2.1 Datos extraídos en el análisis

En el fichero extraído se pueden encontrar una serie de valores separados por comas los cuales serán utilizados para sacar datos. Estos son los datos de mayor relevancia:

- Dirección IP destino e IP origen, en este caso se muestra el valor, en decimal, de las IP del flujo. Al ser bidireccional el flujo los paquetes en una dirección tendrá una IP origen y un destino y los paquetes en dirección contraria tendrán intercambiadas dichas IPs.
- Puerto origen y puerto destino, ocurre lo mismo que con las direcciones IP. Se toma como emisor la dirección IP y puerto que primero envió un mensaje hacia otra IP y puerto, y receptor la IP y puerto destino que recibe el primer paquete. Por lo que podrá haber flujos con solo paquetes/datos enviados, pero no podrá haber flujos con solo paquetes/datos recibidos.
- Protocolo, UDP o TCP.
- Inicio y fin del flujo, con el valor en *timestamp (timepoch).*
- Duración del flujo, en segundos, que indica la duración entre el primer paquete y el último, si solo hay un paquete se le da el valor -1.
- Tamaño, en bytes, cantidad de datos que se han enviado o recibido a través del flujo. Se entenderán como enviados los paquetes que transmite el emisor hacia el receptor y como recibidos los paquetes que transmite el receptor hacia el emisor.
- Tamaño enviado, datos en bytes que se han enviado desde emisor al receptor.
- Tamaño recibido, datos en bytes que se han enviado desde el receptor al emisor.
- Número de paquetes totales en el flujo.
- Número de paquetes enviados, del emisor a receptor.
- Número de paquetes recibidos, del receptor al emisor.
- Throughput, cociente entre la cantidad de datos y la duración del flujo, si el flujo es de un paquete se establece un Throughput nulo, de cero.
- Throughput de enviados y de recibidos, la misma relación que el anterior pero solo con datos de envío o de recepción.
- Ratio sentido, cociente entre paquetes enviados y recibidos, sino se reciben paquetes el valor será directamente el número de paquetes enviados.
- Tamaño medio paquete, promedio del tamaño en bytes de los paquetes de un mismo flujo.
- Tamaño medio paquetes enviados y recibidos, misma relación que el anterior, pero con solo paquetes enviados y recibidos, calculados cada uno de forma separada.

- Desviación estándar de los paquetes, cálculo de este parámetro con todos los paquetes de un mismo flujo. También se realiza dicho cálculo para los paquetes enviados y recibidos, de forma separada.
- Media entre llegadas, promedio de llegadas entre paquetes del mismo flujo, en segundos. Si el flujo es de un paquete se pone de valor -1. También se realiza dicho cálculo para los paquetes enviados y recibidos, de forma separada. En la Figura 3-5, donde cada fila es un flujo, se puede ver como se calcula el tiempo llegada a llegada, y como a partir de ahí se calcular la media.
- Desviación estándar de los tiempos entre paquetes, con el cálculo de tiempos entre paquete y paquete de un mismo flujo se obtiene este cálculo tanto del conjunto total como los paquetes enviados y los recibidos de forma separada. Si solo hay un paquete se le da el valor -1. Si no tiene paquetes recibidos también se le da valor -1 en el campo correspondiente a recibidos.
- Aplicaciones, se añade las aplicaciones encontradas en hasta 5 posibles opciones, acorde a las posibilidades que daba el programa. Si no se encuentra ninguna aplicación al campo se le da el nombre de desconocido (o *Unknown*).

Todos estos campos nos ayudarán a buscar relaciones entre estas estadísticas y la aplicación identificada. Los datos de mayor relevancia para nuestros cálculos serán principalmente el tamaño medio de paquetes, el tiempo entre llegadas y también la triple relación entre los flujos concurrentes de una misma aplicación, el ancho de banda de dichos flujos concurrentes y la relación entre ellos dos, o sea, el cociente entre flujos concurrentes y ancho de banda en Mbps de todos los flujos concurrentes de una aplicación.

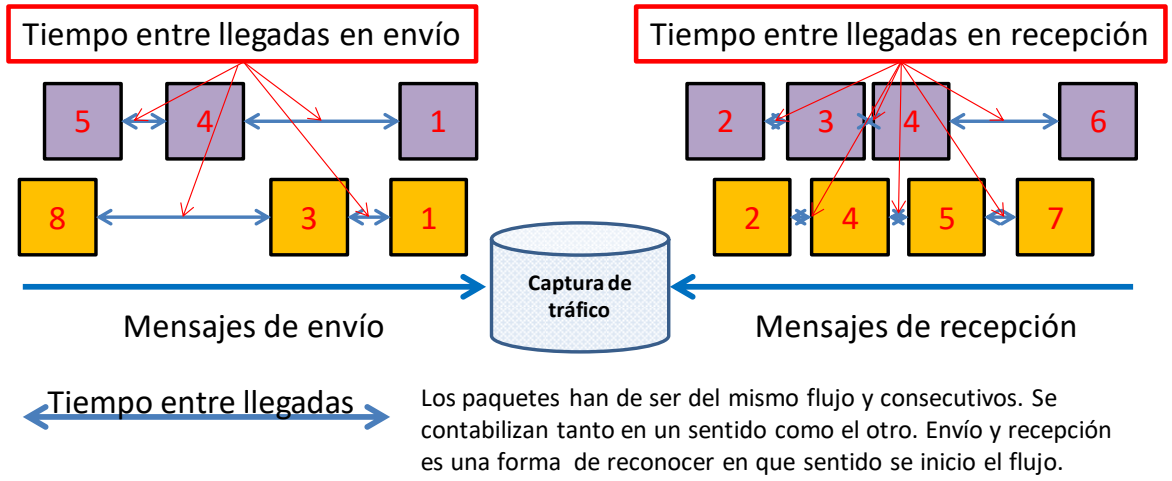


Figura 3-5: Captura de tráfico y medida del tiempo entre llegadas

3.2.2 Implementación del programa

El programa escogido para la representación de datos ha sido Matlab, que es una herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M).

Se ha desarrollado una aplicación que lee los datos del archivo de valores delimitado por comas y extrae cada campo como un vector de valores independiente, de forma que se

puede operar cada uno de forma separada. Con estos vectores de datos se han calculado gráficas por aplicación de estas tres estadísticas:

- Flujos concurrentes por aplicación: estudiando flujo a flujo ya identificado en cada aplicación se ha ido calculando el acumulado de flujos de una misma aplicación a lo largo del tiempo de la traza estudiada. El acumulado de flujos concurrentes por aplicación se ha realizado segundo a segundo. En la Figura 3-6 se puede ver cómo funciona el cálculo de flujos concurrentes.
- Ancho de banda medio por flujo, de nuevo con las estadísticas de bytes transmitidos en cada flujo, se ha podido calcular el ancho de banda medio de cada flujo identificado como parte de una misma aplicación. Después de esto se han sumado de forma agregada para ver el ancho de banda total que ocupa una determinada aplicación en la traza. Como se puede ver en la Figura 3-7, esta forma de calcular el ancho de banda como promedio no tiene en cuenta los picos de Mbps que se producen en cada flujo y por tanto hará que en las gráficas mostradas se visualice el ancho de banda de manera más suavizada.
- En la tercera gráfica se va a representar la relación entre los flujos y el ancho de banda en Mb/s, mostrando la cantidad de flujos concurrentes por cada Mb/s. Esta gráfica es realmente la más interesante de las tres, pues se va a buscar las particularidades que representa cada aplicación en sus flujos. Esta relación ya mostrada de manera genérica para todos los flujos sin importar la aplicación es mostrada en [2]. En apartados posteriores se analizará dicho comportamiento tanto de forma genérica como por aplicación.

Además de la representación de estas graficas se ha aprovechado la multitud de opciones de cálculo estadístico que ofrece Matlab y se han analizado todas las estadísticas volcadas que aparecen en el apartado 3.2.1 que se han considerado útiles para diferencias diferentes aplicaciones en cuanto a datos como el tamaño medio del paquete o el tiempo entre llegada y llegada de los paquetes.

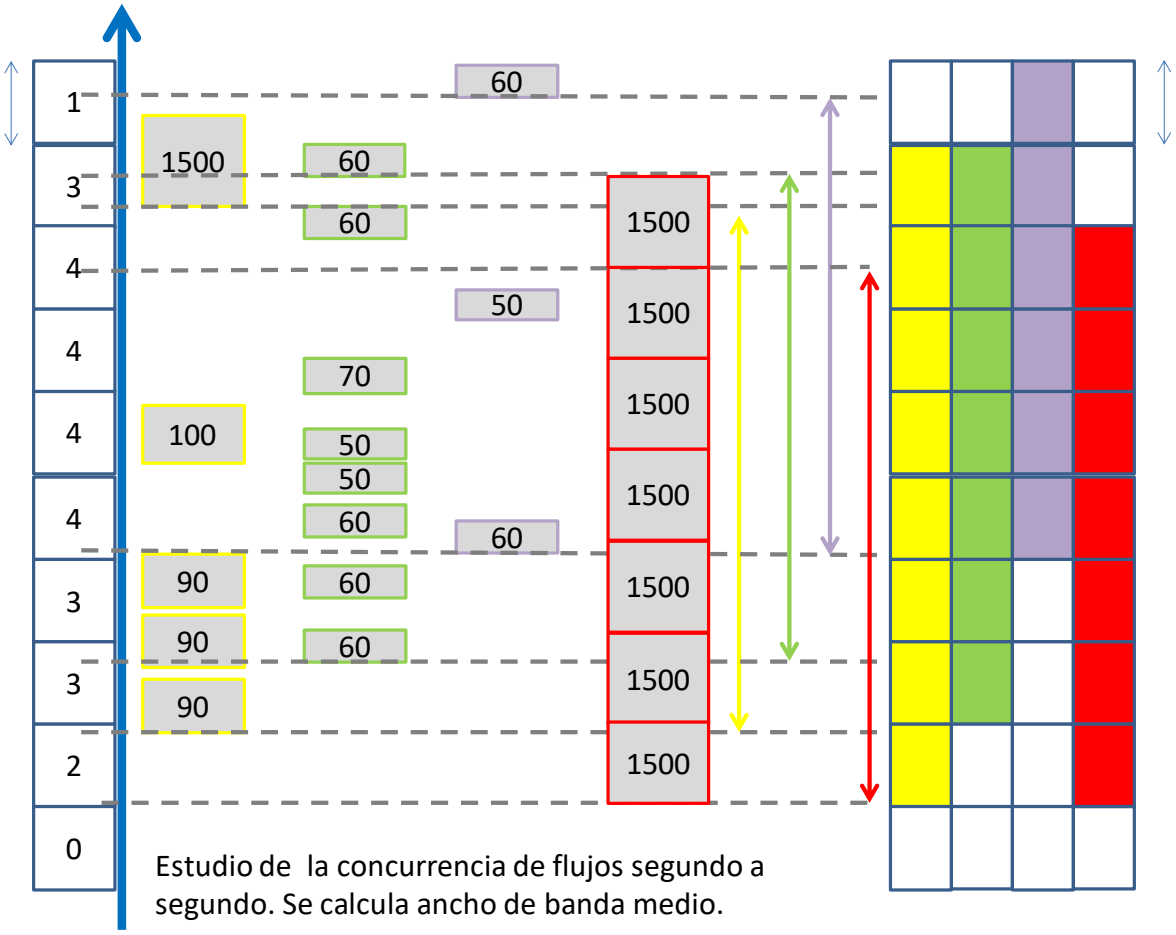


Figura 3-6: Calculo de agregado de flujos y datos.

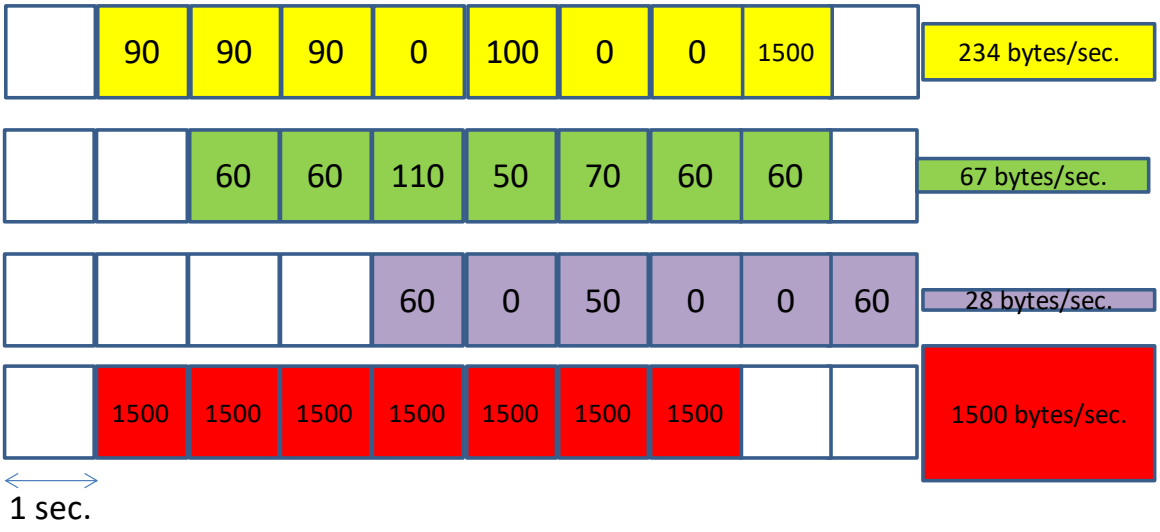


Figura 3-7: Dibujo explicativo sobre el cálculo de ancho de banda en cada flujo

El principal inconveniente que ha aparecido con este cálculo ha sido la dificultad que tiene el SW de Matlab para procesar grandes cantidades de información (~10 GB) y la lentitud para mostrar resultados. Por ello siempre es recomendable utilizar parte de estos archivos y no de manera completa.

3.3 Clasificación y selección de datos por Weka

Por último, además de la utilización de Matlab para facilitar el análisis de los datos exportados por la herramienta DPI se ha usado el programa Weka. Este programa basado en SW abierto implementa técnicas de *machine learning*, ya sea con aprendizaje supervisado o no supervisado y encuentra en función de las estadísticas que contenga un conjunto de datos aquellos campos los cuales son más útiles para discriminar entre distintos tipos en un mismo conjunto de valores. En nuestro caso, el programa ayudará a buscar que campos son más o menos útiles a la hora de poder discriminar entre los flujos para saber qué tipo de clasificación es.

Weka contiene una colección de herramientas de visualización y algoritmos para el análisis de datos y modelado predictivo. Incluye técnicas de preprocesamiento, regresión, modelado, clasificación, selección y visualización de datos. En Weka, un registro de datos estará conformado por un número fijo de atributos. En este trabajo cada registro se corresponderá con un flujo y los atributos serán todas las estadísticas asignadas a cada flujo en particular. El número de atributos son iguales en todos los registros de datos.

La principal ventaja de usar Weka frente a Matlab es que, aparte de ser SW libre, implementa de manera automática multitud de algoritmos usados para el aprendizaje supervisado, no supervisado y semisupervisado. Estas técnicas son principalmente basadas en arboles de decisión, pero también están disponibles algoritmos de agrupamiento (*clustering)* y las máquinas de vectores de soporte.

En el caso del presente trabajo, se usará un aprendizaje supervisado, que significa que los datos están etiquetados previamente antes de su análisis, y en función de la clasificación de estos datos se puede extraer atributos que discriminen mejor o peor estas etiquetas. Para el caso concreto del trabajo, estas etiquetas serás las aplicaciones identificadas por DPI.

Como un porcentaje alto de los flujos analizados por DPI no han podido ser identificados por ninguna aplicación se aplicará los mismos algoritmos que se consideren óptimos dentro del aprendizaje supervisado para ver si estos flujos no etiquetados se pueden agrupar en función de sus estadísticas en grupos diferenciados o no y si guardan similitudes con los flujos etiquetados. Esta parte se considera aprendizaje no supervisado pues no cuenta con las etiquetas que marcan si un flujo pertenece a una determinada aplicación. Por lo que en esta parte no sé podrá descubrir técnicas.
Otra ventaja a la hora de usar Weka es que permite incluir una muestra de entrenamiento y otra de test para realizar estos cálculos. En nuestro caso se usarán trazas de distintos años como muestras de test y entrenamiento para favorecer una mayor independencia de los datos de una muestra a otra. Aunque el hecho de que las trazas hayan sido recogidas en el mismo punto de red ya es un factor limitante a la hora de generalizar la clasificación y modelado de datos al resto de la red.

Una vez obtenidos los resultados se va a evaluar los datos obtenidos por el programa que ha analizado DPI, el objetivo final es descubrir comportamientos de los flujos en la red que

sean discriminantes a la hora de diferenciar determinadas aplicaciones. Con Weka se puede ver más fácilmente la manera de poder discriminar dichos flujos, pero es el usuario el que elige los algoritmos. Será vital usar un algoritmo adecuado si se quieren obtener resultados útiles.

4 Desarrollo de los análisis

4.1 Análisis de los resultados

4.1.1 Resultados obtenidos con Sistema l7-Filter

En este apartado se va a analizar los resultados obtenidos con la técnica de detección obtenida con el sistema DPI usado en el presente trabajo, en primer lugar, se debe explicar cómo se han guardado las trazas de paquetes que se han estudiado.
Las trazas que se han usado corresponden a dos trazas recogidas en dos días completos, durante las 24 horas, una correspondiente a un día de 2008 y otra correspondiente a un día de 2010.
En primer lugar, se muestran los resultados obtenidos el 12 de abril de 2008, que es un sábado, en el cual se ha dividido el día en tramos de 6 horas que podemos llamar madrugada (00:00-06:00), mañana (06:00-12:00), tarde (12:00-18:00) y noche (18:00-24:00).
Para la traza recogida el 15 de abril de 2010, un lunes, se ha dividido en los mismos tramos horarios que las trazas de 2008. El objetivo de estas divisiones es establecer patrones de comportamiento de la red según la hora del día y la situación de los internautas. Así, por ejemplo, sería razonable que en horario de mañana existiese un mayor tráfico dedicado a aplicaciones de correo electrónico mientras que por la tarde las aplicaciones basadas en el protocolo P2P sean las predominantes. Lo normal sería, que habiendo recogido los datos en el mismo punto de red que los porcentajes de uso de cada aplicación fuese extrapolable al resto de días laborables cercanos. Por tanto, estos datos servirían a un gestor de red para dimensionar la red conforme a las necesidades de las aplicaciones que circulan por la misma, por ejemplo. En el caso de 2008, al ser un sábado los cálculos pueden diferir bastante respecto de lo que un día laboral se refiere, por lo que a la hora de obtener los resultados no será extraño ver diferencias grandes.
Además, se pueden extraer conclusiones sobre cómo ha evolucionado el tráfico entre un momento y otro. En [6] se muestra como las tendencias de tráfico pueden cambiar de manera relativamente rápida (menos de 1 año) en cuanto al uso de aplicaciones en la red. Un ejemplo simple es la caída que supuso el cierre de *Megavideo* y *Megaupload,* basados en HTTP, en el tráfico en dicho protocolo, sustituido en gran medida por tráfico P2P, que se utiliza principalmente para compartir contenido multimedia en la red.

Los resultados obtenidos no deben ser generalizados al resto de redes en Internet, en [6] se puede observar cómo según el país analizado (se estudian tres países relativamente cercanos como Italia, Polonia y Hungría), las tendencias de tráfico son diferentes y por tanto las aplicaciones que usan los usuarios también lo son. Que la zona de red analizada solo tenga usuarios con ADSL o solo con FTTH también hace que las estadísticas de tráfico puedan diferir. Por tanto, las estadísticas de tráfico obtenidas en el presente trabajo servirían como buena base para que un gestor de red pueda dimensionar correctamente esa red, pero no sirve como modelo genérico, pues Internet no funciona de manera homogénea.

Mañana (06:00-12:00 12/04/2008)								
Aplicación	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	21326940	100,000%	9973640	46,765%	100,000%	11353300	53,235%	100,000%
No etiquetados	5329808	22,661%	4099987	19,224%	41,108%	1229821	5,767%	10,832%
DNS	9704315	41,259%	4341474	20,357%	43,529%	5362841	25,146%	47,236%
HTTP	2111630	8,978%	517490	2,426%	5,189%	1594140	7,475%	14,041%
BITTORRENT	773594	3,289%	294300	1,380%	2,951%	479294	2,247%	4,222%
EDONKEY	73862	0,314%	36820	0,173%	0,369%	37042	0,174%	0,326%
MSN	261	0,001%	197	0,001%	0,002%	64	0,000%	0,001%
NETBIOS	59070	0,251%	56209	0,264%	0,564%	2861	0,013%	0,025%
NTP	19889	0,085%	9686	0,045%	0,097%	10203	0,048%	0,090%
POP3	6423	0,027%	999	0,005%	0,010%	5424	0,025%	0,048%
SKYPE	1165588	4,956%	438411	2,056%	4,396%	727177	3,410%	6,405%
SMB	5831	0,025%	407	0,002%	0,004%	5424	0,025%	0,048%
SMTP	969184	4,121%	146965	0,689%	1,474%	822219	3,855%	7,242%
SSH	877666	3,732%	604	0,003%	0,006%	877062	4,112%	7,725%
SSL	62513	0,266%	10	0,000%	0,000%	62503	0,293%	0,551%
OTROS	167306	0,711%	30081	0,141%	0,302%	137225	0,643%	1,209%

Tabla 4.1-1. Estadísticas de flujos. Horario Mañana. Año 2008.

Noche (18:00-24:00 12/04/2008)								
Aplicación	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	23063637	100,000%	10575904	45,855%	100,000%	12487733	54,145%	100,000%
No etiquetados	5346764	23,183%	4062285	17,613%	38,411%	1284479	5,569%	10,286%
DNS	10938091	47,426%	4906446	21,274%	46,393%	6031645	26,152%	48,301%
HTTP	2461309	10,672%	473897	2,055%	4,481%	1987412	8,617%	15,915%
BITTORRENT	830446	3,601%	304161	1,319%	2,876%	526285	2,282%	4,214%
EDONKEY	54878	0,238%	26908	0,117%	0,254%	27970	0,121%	0,224%
MSN	258	0,001%	149	0,001%	0,001%	109	0,000%	0,001%
NETBIOS	142766	0,619%	139483	0,605%	1,319%	3283	0,014%	0,026%
NTP	20966	0,091%	8816	0,038%	0,083%	12150	0,053%	0,097%
POP3	4590	0,020%	876	0,004%	0,008%	3714	0,016%	0,030%
SKYPE	1319282	5,720%	480438	2,083%	4,543%	838844	3,637%	6,717%
SMB	1315	0,006%	184	0,001%	0,002%	1131	0,005%	0,009%
SMTP	942942	4,088%	141027	0,611%	1,333%	801915	3,477%	6,422%
SSH	838743	3,637%	544	0,002%	0,005%	838199	3,634%	6,712%
SSL	67526	0,293%	20	0,000%	0,000%	67506	0,293%	0,541%
OTROS	93761	0,407%	30670	0,133%	0,290%	63091	0,274%	0,505%

Tabla 4.1-2. Estadísticas de flujos. Horario Noche. Año 2008.

Mañana (06:00-12:00) ---- 15/04/2010								
Aplicación	flujos totales	TOTAL%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	24079180	100,000%	10555246	43,836%	100,000%	13523934	56,164%	100,000%
No etiquetados	7263527	30,165%	2949322	12,248%	27,942%	4314205	17,917%	31,901%
DNS	9762992	40,545%	5382399	22,353%	50,993%	4380593	18,192%	32,391%
HTTP	3414228	14,179%	435032	1,807%	4,121%	2979196	12,372%	22,029%
BITTORRENT	1319307	5,479%	1111825	4,617%	10,533%	207482	0,862%	1,534%
EDONKEY	134932	0,560%	64476	0,268%	0,611%	70456	0,293%	0,521%
MSN	72813	0,302%	67067	0,279%	0,635%	5746	0,024%	0,042%
NETBIOS	95530	0,397%	14710	0,061%	0,139%	80820	0,336%	0,598%
NTP	69671	0,289%	51020	0,212%	0,483%	18651	0,077%	0,138%
POP3	94639	0,393%	39512	0,164%	0,374%	55127	0,229%	0,408%
SKYPE	512722	2,129%	266052	1,105%	2,521%	246670	1,024%	1,824%
SMB	91471	0,380%	24	0,000%	0,000%	91447	0,380%	0,676%
SMTP	482059	2,002%	122740	0,510%	1,163%	359319	1,492%	2,657%
SSH	459022	1,906%	1434	0,006%	0,014%	457588	1,900%	3,384%
SSL	224848	0,934%	124	0,001%	0,001%	224724	0,933%	1,662%
OTROS	81419	0,338%	49509	0,206%	0,469%	31910	0,133%	0,236%

Tabla 4.1-3. Estadísticas de flujos. Horario Mañana. Año 2010.

noche (18:00-24:00) ---- 15/04/2010								
Aplicación	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	27315922	100,000%	11160712	43,836%	100,000%	16155210	56,164%	100,000%
No etiquetados	8013670	29,337%	3090043	12,833%	29,275%	4923627	20,448%	36,407%
DNS	10705750	39,192%	5823230	24,184%	55,169%	4882520	20,277%	36,103%
HTTP	3593595	13,156%	427797	1,777%	4,053%	3165798	13,147%	23,409%
BITTORRENT	1859315	6,807%	1192683	4,953%	11,299%	666632	2,768%	4,929%
EDONKEY	137080	0,502%	62648	0,260%	0,594%	74432	0,309%	0,550%
MSN	107695	0,394%	100726	0,418%	0,954%	6969	0,029%	0,052%
NETBIOS	100418	0,368%	70116	0,291%	0,664%	30302	0,126%	0,224%
NTP	56748	0,236%	40402	0,168%	0,383%	16346	0,068%	0,121%
POP3	7612	0,032%	120	0,000%	0,001%	7492	0,031%	0,055%
SKYPE	396465	1,647%	170219	0,707%	1,613%	226246	0,940%	1,673%
SMB	101131	0,420%	181	0,001%	0,002%	100950	0,419%	0,746%
SMTP	581035	2,413%	151495	0,629%	1,435%	429540	1,784%	3,176%
SSH	1322145	5,491%	2067	0,009%	0,020%	1320078	5,482%	9,761%
SSL	275675	1,145%	237	0,001%	0,002%	275438	1,144%	2,037%
OTROS	57588	0,239%	28748	0,119%	0,272%	28840	0,120%	0,213%

Tabla 4.1-4. Estadísticas de flujos. Horario Noche. Año 2010.

Los resultados de los porcentajes de uso de cada aplicación en los distintos tramos horarios son similares en todos los casos, en el caso de 2010. Se observa un aumento de tráfico durante la tarde-noche y excepto SSH, que aumenta bastante durante la noche, el resto de aplicaciones no muestran grandes variaciones. Skype también sufre una ligera bajada durante la tarde respecto del día.

En el análisis por aplicaciones se observa como DNS y HTTP copan gran parte del tráfico tanto en 2008 como en 2010, aunque es este último se observa como el tráfico ha aumentado un 20% aproximadamente. Las dos aplicaciones que muestran mayor variación entre un año y otro son *MSN* y *MSB,* pasan de ser prácticamente inexistentes en 2008 a ocupar un pequeño porcentaje, entre el 0,3 y 0,4%, tanto *eDonkey* como la otra aplicación P2P como es *Bittorrent*, aumentan ligeramente. También se observa cómo entre los dos protocolos más usados para correo electrónico, *POP3* y *SMTP*, en 2008 es *SMTP* el más detectado en la traza, pero en 2010 este protocolo baja a la mitad en cuanto a porcentaje aproximadamente y *POP3* aumenta, sobre todo en el tramo de mañana. La capacidad de detección se mantiene estable etiquetando en 2008 a algo más del 75% de los flujos frente al 70 % de los flujos de 2010. Esta bajada tal vez se deba a que dichas firmas puedan haber quedado anticuadas en algunos casos por cambios en los protocolos de aplicación. Otro cambio significativo se produce en que la aplicación SSL es menor en 2008, tal vez porque el uso de HTTPS estaba aún menos extendido.

Las tablas de flujos por aplicación muestran que DNS ocupa un porcentaje bastante grande dentro de todos los flujos concurrentes. Estos flujos se caracterizan por ser de entre uno y tres paquetes y tener un tamaño pequeño. Por lo que, aunque haya gran cantidad de flujos de este protocolo después se observa como en volumen y porcentaje de paquetes no es tan significativo. En la tabla superior se han dividido los flujos entre los de 1 paquete y más de un paquete porque cuando se realicen análisis posteriores no se usarán los flujos de un paquete para considerar que se necesitan flujos de más de un paquete para considerar una sesión real por parte de un usuario en la red para aplicaciones como SMTP o HTTP. En cambio, en las aplicaciones P2P si serán tenidas en cuenta porque estas aplicaciones usan una asignación de puertos dinámica, es decir, pueden usar varios puertos de manera simultánea para realizar la descarga de un archivo. Las aplicaciones más abundantes aparte de DNS son HTTP, SKYPE, SMTP, SMB, SSH y SSL y las aplicaciones P2P, que son *Bittorrent,* y *eDonkey.*

Las aplicaciones basadas en HTTP ocupan gran porcentaje del total, tanto a nivel de flujos como, en mayor proporción, de datos. Los flujos HTTP se caracterizan por funcionar mediante la arquitectura cliente-servidor, donde un solo servidor es el que responde las peticiones a los distintos terminales conectados a Internet que requieren conectarse a dicho servidor.
Las aplicaciones HTTP comprenden multitud de servicios, desde blogs y periódicos electrónicos, servicios de geolocalización, de audio y de video, servicios en *streaming* y multitud de aplicaciones móviles que funcionan bajo protocolo HTTP o HTTPS.
HTTP opera en la capa más alta del modelo OSI, la capa de aplicación; pero el protocolo de seguridad opera en una subcapa más baja, cifrando un mensaje HTTP previo a la transmisión y descifrando un mensaje una vez recibido. Estrictamente hablando, HTTPS no son dos protocolos separados, pero el uso del HTTP ordinario sobre una conexión con Seguridad de la Capa de Transporte (TLS) induce a pensar que los comportamientos en la red de HTTP puedan ser extrapolable a dicho protocolo.
En cuanto a P2P, la característica principal de la arquitectura es que saca el máximo partido de los recursos (ancho de banda, capacidad de almacenamiento, etc.) de los

muchos clientes (*peers*) para ofrecer servicios de aplicación y red, sin tener que confiar en los recursos de uno o más servidores centrales. De este modo se evita que tales servidores se conviertan en un cuello de botella para toda la red. Por tanto, se observa como hay mucho mayor porcentaje de aplicaciones P2P en cuanto a flujos que en cuanto a volumen, por qué al contrario que HTTP, estos flujos tienen utilizan un ancho de banda menor. Sin embargo, un mismo usuario puede usar decenas o hasta cientos de flujos de manera simultánea desde un terminal conectado a Internet.

Mañana (06:00-12:00) ---- 15/04/2010								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0	0	N/A		0%	0	0%	0
HTTP	19.021%	158384	38,173%	126470	N/A		0%	0
BITTORRENT	3,711%	30901	5,991%	19847	2,834%	9853	N/A	
EDONKEY	23,049%	191924	54,795%	181540	1,385%	4817	48,590%	4358
MSN	0,091%	756	0,007%	22	0,189%	658	0,301%	27
NETBIOS	11,056%	92061	0,000%	0	0,047%	165	4,438%	398
NTP	0,000%	0	0,000%	0	0,000%	0	0,000%	0
POP3	0,000%	1	0,000%	1	0,000%	0	0,000%	0
SKYPE	2,432%	20252	0,002%	8	2,690%	9352	10,681%	958
SMB	0,155%	1287	0,000%	0	0,220%	764	5,675%	509
SMTP	0,000%	1	0,000%	0	0,000%	0	0,000%	0
SSH	30,446%	253519	0,000%	1	72,457%	251933	17,672%	1585
SSL	7,808%	65016	0,024%	80	18,527%	64417	3,490%	313
OTROS	2,231%	18576	1,007%	3337	1,651%	5742	9,154%	821
TOTAL	100%	832678	39,788%	331306	41,757%	347701	1,077%	8969

Tabla 4.1-5. Estadísticas de doble match en flujos. Mañana 2008.

Noche (18:00-24:00) ---- 15/04/2010								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0	0	N/A			0	0%	0
HTTP	1,033%	3687	1,637%	3410	N/A		0%	0
BITTORRENT	1,166%	4160	0,790%	1645	12,108%	2215	N/A	
EDONKEY	60,029%	214164	96,713%	201475	39,731%	7268	74,519%	4492
MSN	0,175%	624	0,000%	25	2,531%	463	1,194%	72
NETBIOS	28,360%	101180	0,000%	0	0,000%	0	0,464%	28
NTP	0,001%	2	0,000%	2	0,000%	0	0,000%	0
POP3	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SKYPE	2,941%	10491	0,000%	21	0,159%	29	6,238%	376
SMB	0,111%	395	0,000%	0	1,301%	238	0,017%	1
SMTP	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SSH	0,503%	1793	0,000%	0	9,698%	1774	0,315%	19
SSL	0,225%	801	0,000%	98	2,345%	429	0,100%	6
OTROS	5,458%	19472	0,000%	1647	35,571%	6507	17,153%	1034
TOTAL	100%	356769	58,392%	208323	5,304%	18923	1,690%	6028

Tabla 4.1-6. Estadísticas de doble match en flujos. Noche 2008.

En cuanto a los flujos en los que ha habido un doble match, nos vamos a centrar en la traza de 2010, se pueden observar en la tabla superior las estadísticas. Aproximadamente un 3% de los flujos de red ocurre este fenómeno. Como se puede observar, muchos de los flujos etiquetados como DNS en primer lugar han sido etiquetados también como HTTP. Estudiando la morfología de ambas firmas y como funciona cada protocolo se puede deducir que muchos de los paquetes donde ocurre este doble match son realmente flujos HTTP. Se ha llegado a esta conclusión porque las firmas reconocen patrones de caracteres, y mientras que un paquete DNS no incluye un cuerpo del mensaje, que si tiene HTTP y que puede contener multitud de valores en función de lo que se descargue del contenido web, estos valores podrían coincidir en algunos casos con el patrón DNS. El caso contrario se antoja más complicado, si se observa la forma del mensaje DNS este cuenta con una serie de campos los cuales llevan siempre una forma predefinida y su contenido está más acotado, por lo que pueda coincidir justo con la firma del patrón usado para HTTP es seguramente más difícil. Adicionalmente a este análisis, la discriminación entre uno u otro protocolo se puede realizar a través del puerto usado, en los casos en que se observe que el puerto 53 es usado, el flujo debería relacionarse con DNS y con HTTP en caso del puerto 80. Como en los análisis realizados por DPI a las trazas se mantuvo activo el escaneo de puertos para el caso de DNS, es posible que en algunos casos se hayan asignado a flujos que realmente no lo son, pero sí que usaban ese puerto en concreto. Se realizó una simulación sin incluir el escaneo por puertos para observar en ese caso que cantidad de flujos DNS son encontrados.

En el caso de las aplicaciones P2P, *eDonkey* y *Bittorrent,* se ha detectado como muchos flujos son asignadas a ambas aplicaciones. Esto ocurre porque al incluir el escaneo de puertos para *BitTorrent* (puertos 6881 al 6889) que se suponen los típicos para el uso de esta aplicación. Cuando se han analizado algunos de estos flujos se ha observado como los puertos usados correspondían con los antes mencionados para *Bittorrent,* por lo que se puede suponer que han sido asignados primero a este por el puerto y luego DPI ha encontrado el patrón de *eDonkey*. Otro hecho que nos inclina a pensar que estos flujos pertenecen a *eDonkey* es el estudio realizado en [4] sobre *l7-filter* donde los resultados muestran que un 5% de los flujos de *eDonkey* son etiquetados como *Bittorrent,* mientras que en el caso contrario la probabilidad es nula. En ambas aplicaciones los resultados dan un 40%, aproximadamente, de probabilidad de detección. Por lo que se puede pensar que existen muchos flujos desconocidos sobre esta aplicación.

Para los *dobles match* ocurridos entre SSH y SSL con HTTP se ha observado el puerto usado, como en el escaneo por puertos no fueron incluidos condiciones para estos protocolos todas las asignaciones son producto del análisis DPI. Por tanto, si observamos que el puerto usado es el 22, que es el típico del SSH, podríamos inclinarnos por asignar el flujo a dicha aplicación. Como en el caso del DNS, es posible que ciertos paquetes HTTP contengan cadenas de caracteres que coinciden con los patrones de SSH, aunque en este caso también podría suceder a la inversa ya que SSH no es un protocolo tan definido como el de DNS y esto puede dar lugar a una mayor combinación que estadísticamente aumente el error en cuanto a falsos positivos. Sin embargo, es difícil encontrar una solución única para estos casos. En cuanto a los casos donde se encuentran SSL y HTTP, esto lo más seguro es que se traten de paquetes HTTP, los cuales en algunos casos usan el protocolo SSL en lugar de TLS. Por tanto, este *doble match* está justificado pues realmente se estarían usando ambos protocolos de manera simultánea.

Para NetBIOS se observa también gran cantidad de *doble match*, además teniendo en cuenta que se han detectado pocos flujos como NetBIOS en comparación con el resto, aproximadamente uno de cada dos flujos que se etiquetó como NetBIOS también se ha etiquetado con otra aplicación, lo que es un porcentaje bastante grande en comparación con el resto de los protocolos.

NetBIOS permite a las aplicaciones 'hablar' con la red. Su intención es conseguir aislar los programas de aplicación de cualquier tipo de dependencia del hardware. En una red local con soporte NetBIOS, las computadoras son conocidas e identificadas con un nombre. Cada computador de la red tiene un único nombre.

Cada PC de una red local NetBIOS se comunica con los otros bien sea estableciendo una conexión (sesión), usando datagramas NetBIOS o mediante broadcast. Las sesiones permiten, como en el protocolo TCP, mandar mensajes más largos y gestionar el control y recuperación de errores. La comunicación será punto a punto. Por otro lado, los métodos de datagramas y broadcast permiten a un ordenador comunicarse con otros cuantos al mismo tiempo, pero estando limitados en el tamaño del mensaje. Además, no hay control ni recuperación de errores (al igual que ocurre en UDP). A cambio, se consigue una mayor eficiencia con mensajes cortos, al no tener que establecer una conexión.

Es curioso que por la noche los *doble match* decaen en número de flujos, y además el protocolo *eDonkey* sigue siendo el principal afectado respecto a este fenómeno.

De todas formas, todos los flujos en los que ha ocurrido un *doble match* serán descartados de los estudios estadísticos posteriores que se realicen en cuanto al estudio de las estadísticas agregadas de los flujos.

Respecto a los resultados de doble match de 2008, se pueden ver los resultados de todos los tramos horarios en el anexo B. No se han analizado en profundidad pues la cantidad de doble match es mucho más pequeña que los resultados obtenidos en 2010.

4.1.2 Análisis de la concurrencia de flujos y ancho de banda por aplicación y otras medidas

Una vez revisado el funcionamiento de técnicas de clasificación de tráfico y mostrado las aplicaciones que corren en la red de WIDE, nos proponemos analizar la relación entre el ancho de banda y el número de flujos concurrentes por aplicación. Esto es, si bien estudios de la concurrencia en agregados de tráfico en Internet han sido realizados, poco se sabe de su relación por aplicación.
La utilidad de este estudio es inmediata para gestores de red que conocen o estiman la distribución de aplicaciones de sus redes, basta con que escalen los resultados de nuestro estudio a tal distribución.

En primer lugar, nos hemos centrado en la estimación de las series temporales y funciones de densidad de la medida de flujos concurrentes por Mb/s, especificado por cada aplicación que presente un numero de muestras significativo para su estudio. De esta forma, se han obtenido diferentes gráficas que muestran como las distintas aplicaciones que han sido detectadas por la aplicación DPI tienen patrones en flujos por Mb/s totalmente diferentes. Las funciones de densidad permiten visualizar de manera más suavizada los resultados de las series temporales. Estas funciones han sido estimadas con cálculos que permiten

encontrar más de una moda dentro de estas funciones de densidad, este tipo de función puede ser útil a la hora de encontrar diferentes tendencias dentro de un mismo tipo de aplicación. Un buen ejemplo sería en el caso de las aplicaciones P2P, para encontrar flujos de señalización, los cuales son de muy pocos datos y paquetes, y los flujos de transferencia de archivos compartidos, que, aunque no son muy voluminosos son más largos en media y transmiten mayor cantidad de datos.

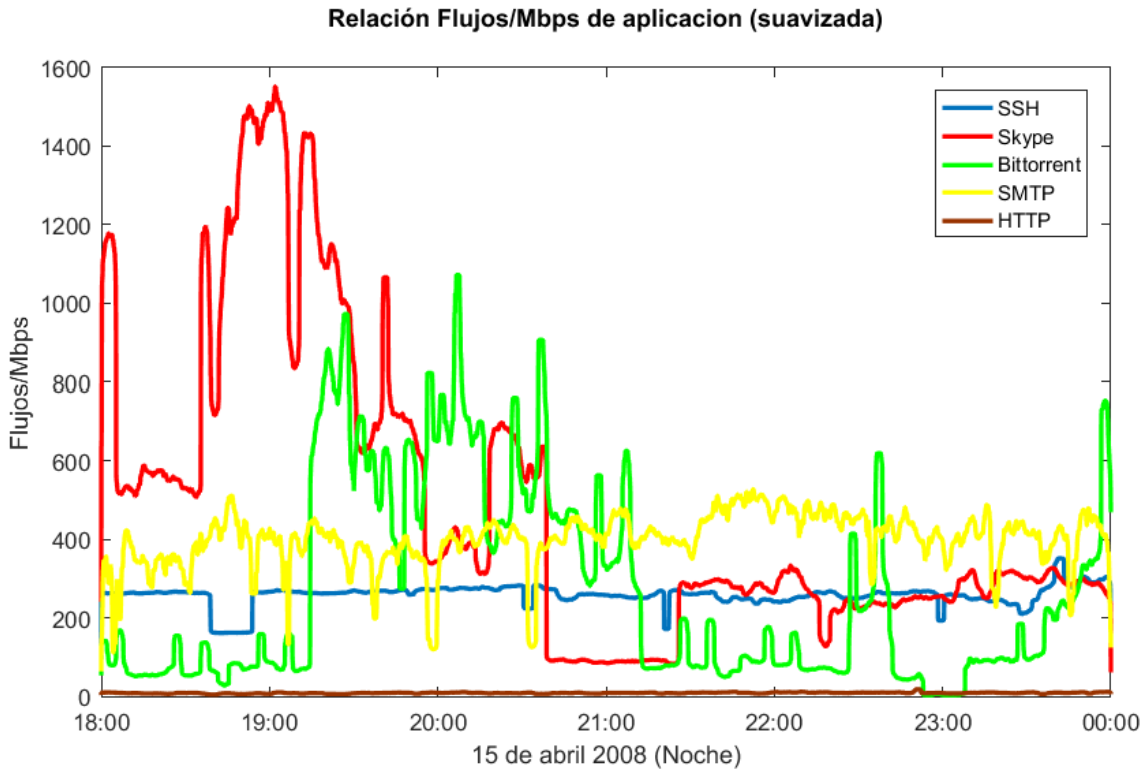


Figura 4-1: Serie temporal de flujos/Megabit. Año 2008

Centrándonos en los resultados de flujos, ancho de banda y la relación de los mismo se ha visto como cada aplicación presenta un numero de flujos concurrentes por Mb/s muy diferente de otras aplicaciones. La principal divergencia se observa con HTTP y el resto de aplicaciones, las cuales presentan un numero de flujos por Megabit mucho más alta que la primera. En la Figura 4-1 se observa como HTTP está en la base de la gráfica prácticamente, teniendo una relación de unos 10-15 flujos por Mb/s siendo, sin embargo, la aplicación que mayor ancha de banda ocupa, en torno a 50 Mb/s en toda la serie temporal. La serie temporal usada ha sido la noche, donde mayor cantidad de tráfico concurrente hay. Que el resto aplicaciones tengan unos resultados mucho más altos es debido a la existencia de mucho tráfico "perdido", el cual representan en aplicaciones como BitTorrent flujos de un paquete los cuales sin ser representativos a nivel de datos aumentan enormemente dicha relación. Si se eliminasen todos aquellos flujos de 1 paquete de la muestra estas diferencias disminuirían. De todas maneras, tiene sentido que HTTP tenga unos valores mucho más bajos que P2P, como ocurre en [4], donde las aplicaciones P2P tienen hasta 50 veces más de flujos en un enlace con la misma bajada que HTTP. La figura 4-1 se ha suavizado con el objetivo de evitar picos que a veces aparecen, porque el marcado de tiempos de la aplicación *L7-filter* no es del todo exacta. La aplicación más estable junto a HTTP es SSH, la cual se puede observar que prácticamente siempre tiene unos 250 flujos por Mb/s, este hecho inclina a pensar que el tamaño de sus paquetes y la frecuencia de envío de estos es constante, lo que provoca que, aunque haya más o menos flujos, al ser muy similares esta

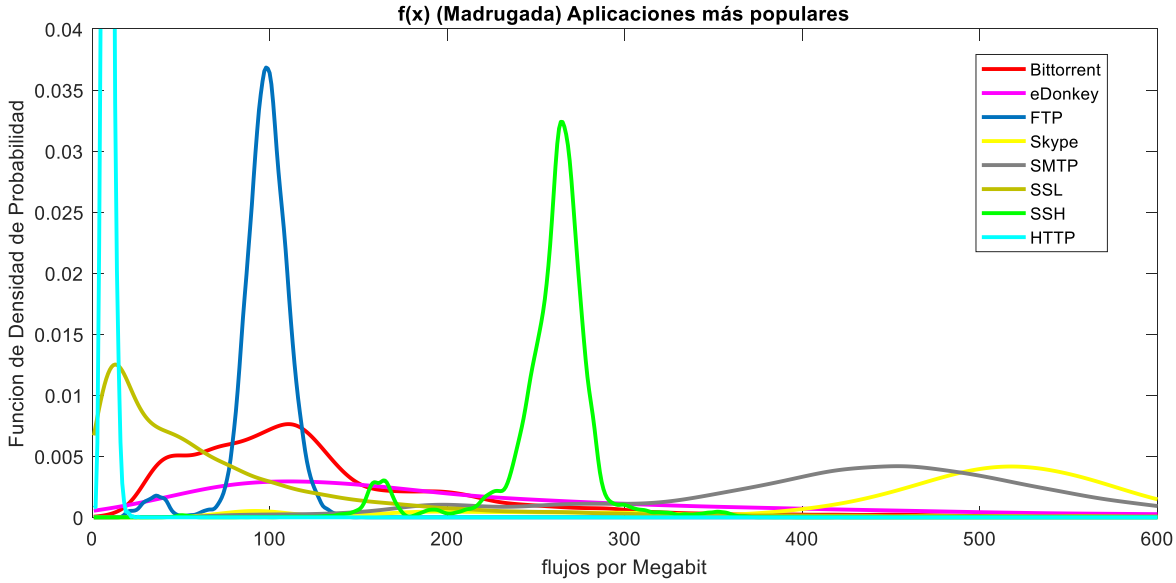
relación apenas cambie. Esta hipótesis se ve refrendada por los resultados obtenidos en las figuras posteriores, donde las funciones de distribución de SSH son muy "verticales", lo que significa que son muy estables, con poca varianza.

De manera adicional a los resultados obtenidos en series temporales, se han hecho estimaciones de probabilidad mediante el método Kernel, que estima una función de densidad que no siga un modelo conocido (Normal, Binomial, Exponencial, etc.) de manera no paramétrica. Tiene una enorme flexibilidad y lo que hace es construir una función de densidad girando en torno a los valores muestrales [22] [23].
En la Figura 4-2 se puede observar como dentro de ocho de las aplicaciones más usadas en Internet, HTTP cuenta con una estimación de flujos por Mb/s muy baja, en torno a 15 flujos, lo que unido a la gran cantidad de tráfico que ocupa hace pensar que cuenta con flujos que transmiten gran cantidad de datos. En el extremo opuesto se encuentran aplicaciones como Skype y SMTP, las cuales se podría estimar una media de flujos por Mb/s de 500 y 450, en el año 2008, respectivamente, pero siempre con una variabilidad bastante alta. Los flujos etiquetados como FTP tienen una caracterización bastante precisa en la gráfica correspondiente a 2008, teniendo por lo general en media unos 100 flujos por megabit y poca variabilidad, sin embargo, en 2010 presenta unos resultados muy variables y no se observan ninguna moda concreta. En SSH, se estimaría la media en unos 280 flujos, además es similar tanto en 2008 como 2010. Una de las hipótesis que se plantean respecto al porque el número de flujos permanece tan estable es debido a que al ser peticiones realizadas sobre un servidor determinado este tiene una limitación en cuanto al manejo de flujos concurrentes, cosa que podría ocurrir también en HTTP, donde su varianza es mínima.
Respecto a las aplicaciones P2P, ambas se estimaría su media en torno a 100 flujos/ Mb/s, sin embargo, su variabilidad es muy grande comparado con el resto de aplicaciones. La explicación de esto podría ser el hecho contrario a lo contado anteriormente para otras aplicaciones con arquitectura cliente/servidor, en este caso al no depender de un único nodo para descargar y comunicarse, la medida de flujos concurrentes puede ser muy variable, dependiendo del estado de la red en cada momento.
A través del estudio de las gráficas que se han estudiado, se puede concluir que el tramo horario no es decisivo y no cambia significativamente, mientras que la caracterización por aplicación detectada si se puede caracterizar de manera más clara, sobre todo en ciertas aplicaciones como HTTP, SSH y SSL.

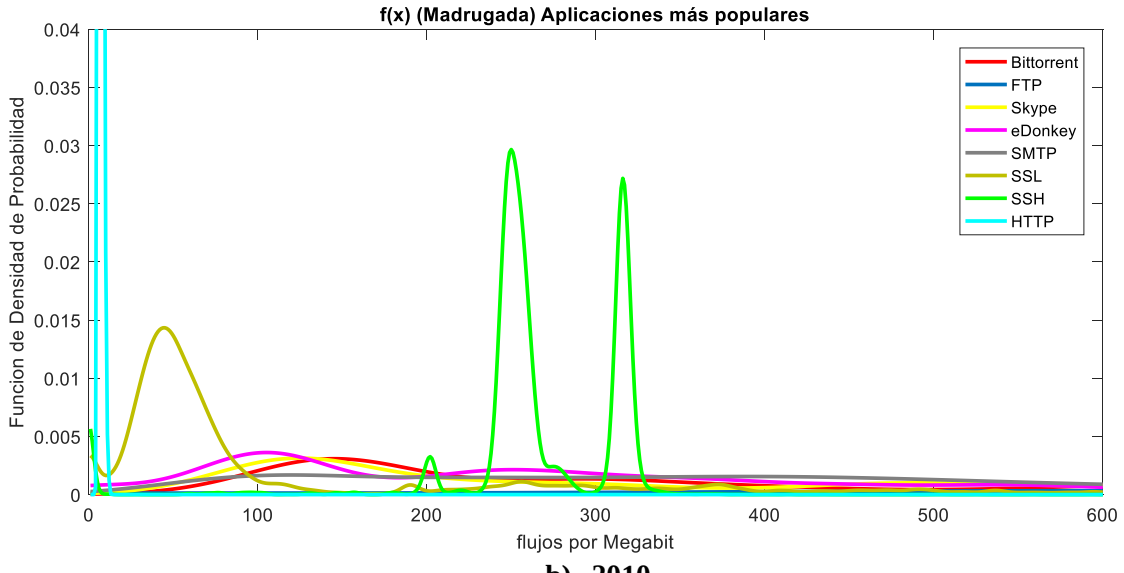
Si comparamos los resultados de 2008 y 2010 se puede observar como en el caso de HTTP, SSL, SSH las estimaciones son más que similares, aunque en SSH detecta dos modas, relativamente cercanas y también en torno a 300 flujos/Mb. En cambio, para el resto de las aplicaciones la estimación de probabilidad no ha encontrado una moda clara, pues tienen gran variabilidad. Las aplicaciones eDonkey y BitTorrent son las que mayor similitud presentan respecto a 2008, sin embargo, aplicaciones como SMTP y Skype tienen unas modas bastante más diferentes, y una variabilidad mayor que el otro año. FTP cambia totalmente su tendencia respecto a la figura de 2008, no se ve al estar solapada con HTTP. Observando los flujos de esta aplicación, se ha observado que habiéndose detectado un numero bastante menor de flujos la cantidad de datos es mucho mayor, lo que ha hecho disminuir mucho la cantidad de flujos/Mb.
Se puede concluir que utilizando estas estimaciones de probabilidad únicamente HTTP, SSL y SSH parecen seguir una estimación similar en ambos años. Se podría llegar a generalizar este comportamiento si realizando dicha estimación sobre trazas de dichas aplicaciones (al final esto ha sido etiquetado por DPI y no es 100% confiable) se obtuviesen resultados parecidos. Llegar a este caso tiene muchos inconvenientes pues

depende de muchos más factores, como la infraestructura y la topología de la red. Respecto al resto de aplicaciones, parece que no es posible, según estos resultados, definir una estimación de flujos/Mb, ya que presentan una gran variabilidad.

Aparte del estudio de las estimaciones de probabilidad por el Kernel de la función, se han obtenido distintas funciones de distribución de probabilidad con las aplicaciones más populares etiquetadas por la aplicación DPI.



a) 2008



b) 2010

Figura 4-2: Función de densidad de Flujos por Megabit de aplicaciones más populares. a) Año 2008 (arriba). b) Año 2010 (abajo)

Otras pruebas que se han realizado de manera independiente a la cantidad de flujos por Mb/s ha sido el estudio del *throughput*, que es el cociente entre la cantidad de datos y la duración del flujo, lo que viene a ser el ancho de banda medio del flujo. Aunque esta medida no muestra las variaciones del ancho de banda de cada flujo, es una buena medida para estudiar los flujos en cada aplicación. Observando la Figura 4-3 se puede ver como en el caso de los flujos HTTP aproximadamente el 90% de los flujos tienen de media mas de

10 kb/s, en comparación con el resto de aplicaciones, las cuales en el caso de BitTorrent y eDonkey, se observan que más de la mitad de sus flujos tienen menos de 10 kb/s. Una de las razones que se han supuesto sobre este bajo ancho de banda en las aplicaciones P2P es que la aplicación DPI usada detecta aquellos flujos que pertenecen al tráfico de señalización, mientras que los flujos usados para transferencia y compartición de datos en muchos casos no han sido detectados.
Es de destacar en dicha gráfica SMTP, la cual tiene una función de distribución más suavizada pero un 85% aproximadamente son flujos de menos de 10 kb/s. En cuanto a grandes anchos de banda medios de más de 10 Mb/s se observa cómo solo HTTP cuenta con flujos de esta cantidad. Se ha elegido las medidas obtenidas durante la madrugada, pero a tenor de los resultados, excepto alguna excepción, se ha observado que el tramo horario elegido no produce grandes cambios en las mediciones respecto a los que el *throughput* se refiere. Más adelante en los apartados referidos respecto a las medidas centradas en HTTP y P2P se estudiará más a fondo las estadísticas por tramos horarios.

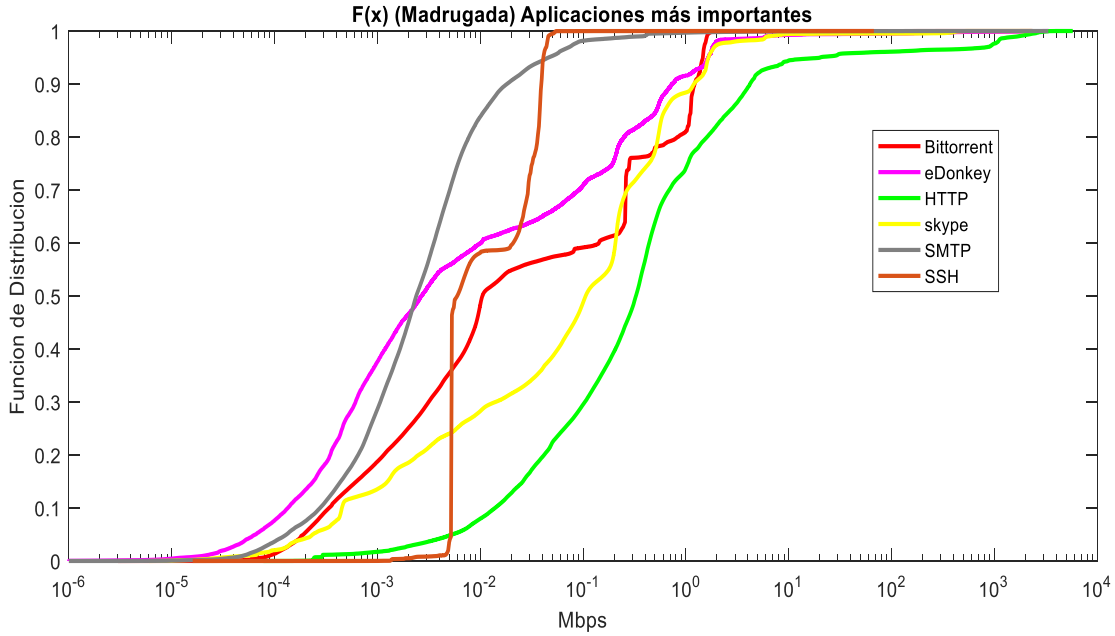


Figura 4-3: Función de distribución del Throughput de aplicaciones más populares. Año 2008

Además de las anteriores estadísticas obtenidas, el programa DPI nos ofrece los resultados como la media y la desviación estándar de tiempo entre paquete y paquete, estas medidas también pueden ser útiles a la hora de conocer cómo se comportan los flujos de red según que aplicación se esté usando. En la Figura 4-4 se han juntado las gráficas obtenidas de las aplicaciones más populares como funciones de probabilidad acumulada. Se ha elegido en este caso el tramo horario de la noche, por ser el que mayor número de flujos, paquetes y datos tiene. Aunque si se compara cada aplicación por separado, los tramos horarios presentan una similitud muy grande.
En este caso se puede observar como las dos aplicaciones P2P, tienen un comportamiento similar, donde en ambos casos aproximadamente el 50% de los flujos, tienen una media entre llegadas de menos de menos de 0.1 segundos y el 80% de menos de 1 segundo, estas medidas, que parecen bastante grande, pueden responder también a lo comentado

anteriormente sobre el tráfico de señalización, cuyos flujos cuentan con pocos paquetes y espaciados en el tiempo con el objetivo de no saturar la red.

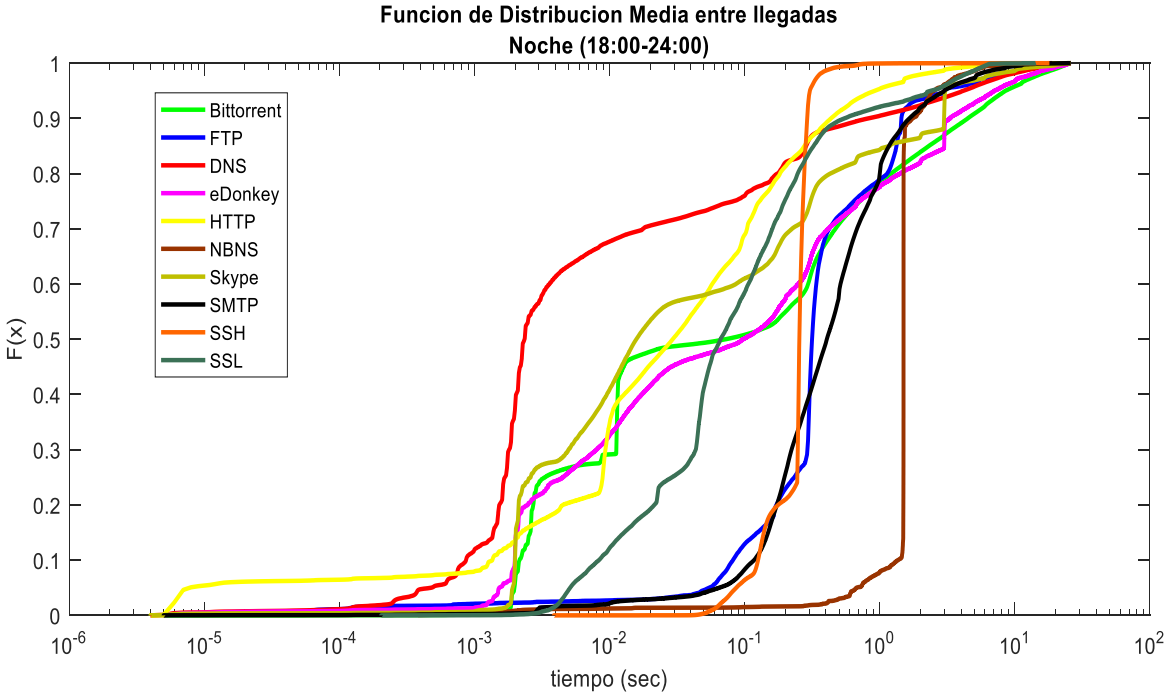


Figura 4-4: Función de distribución de la media temporal de llegadas entre paquetes en un mismo flujo. Se incluyen aplicaciones más populares

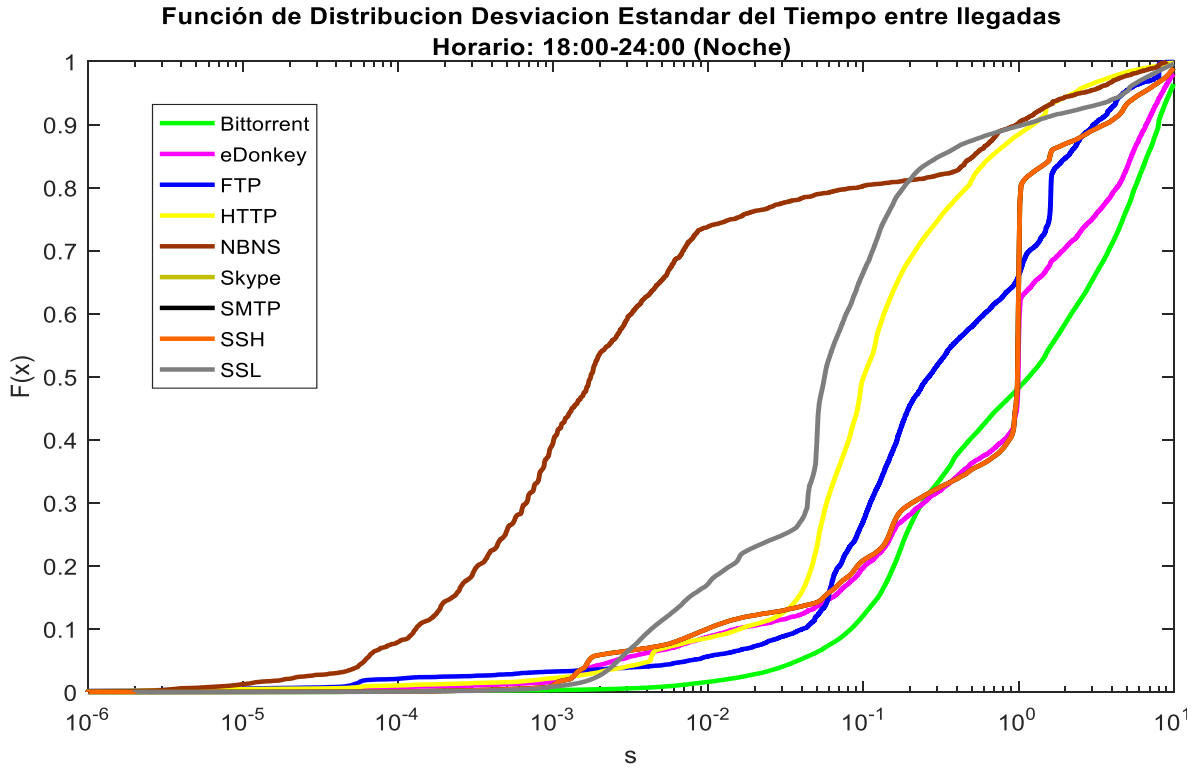


Figura 4-5: Función de distribución de la desviación estándar temporal de llegadas entre paquetes en un mismo flujo. Se incluyen aplicaciones más populares

Los flujos DNS, como se puede observar, presentan una gran cantidad de flujos donde la media es bastante pequeña, más del 60% tiene aproximadamente 1 ms o menos. La

aplicación HTTP es la que presenta una mayor variabilidad, pues se puede observar como su función (color amarillo) tiene una subida suavizada sin saltos abruptos. Esto puede responder perfectamente a la naturaleza de este protocolo, el cual es el predominante en Internet, y a través del cual se pueden encontrar multitud de servicios web, como puede ser el video en *streaming*, periódicos digitales, apps. móviles, blogs web, etc. En el punto contrario se ven aplicaciones cuyas variabilidad es mínima y prácticamente todos sus flujos tienen una media entre llegadas muy similar, es el caso de la aplicación SSH, (200 ms aproximadamente), FTP (entre 80 y 300 ms el 80% de los flujos) o NBNS (90% de flujos en torno a 1 segundo de media), esta aplicación tiene una finalidad similar a DNS, pero de manera mucho más limitada a redes locales, y es la propia definición del protocolo la que determina que esta media entre llegadas sea tan grande. El protocolo SMTP cuenta con un 90% de flujos con más de 100 ms de media de llegadas entre paquete y paquete, un tiempo bastante grande. En cuanto al resto de aplicaciones presentadas, no se pueden obtener muchas conclusiones a partir de la gráfica.
En la gráfica 4-5 se puede observar que el valor de la desviación estándar no es muy útil a la hora de diferenciar aplicaciones, más allá de DNS, que por la propia definición de su protocolo tiene un funcionamiento diferente al del resto de aplicaciones.. HTTP, FTP y BitTorrent presentan las gráficas más suavizadas, que significa que estos valores de desviaciones estándar son más o menos progresivos, frente a otras aplicaciones como SSH, SMTP o Skype que tienen casi siempre el mismo resultado de desviación estándar. Este tipo de medida requiere una mayor carga computacional al tener que realizar cálculos más complejos además que el resultado final se conoce al acabar el flujo.

Por último, también se han incluido en funciones de distribución los datos de tamaño medio de paquete y desviación estándar del tamaño por flujo, incluyendo todos los flujos pertenecientes a las aplicaciones más populares. En la Figura 4-6 se puede observar que los paquetes de tamaños medio más grandes tienen son las aplicaciones HTTP y SSL (la cual suponemos que cuenta entre ellos con flujos HTTPS). Que el volumen de estos flujos sea el más grande de todas las aplicaciones observada concuerda con otros estudios, como [6], donde se demuestra con datos que los flujos de esta aplicación, de naturaleza variable (video en *streaming*, descarga de videos o navegación tradicional en páginas web estáticas) suelen ser flujos de mayor volumen (en cuanto a Mb/s), y asimismo HTTP coincide con lo expuesto en la Figura 4-3, donde se mide el *throughput.* En cuanto al resto de aplicaciones se observa que FTP cuenta con una media de tamaño bastante pequeño, el 90% de los flujos tienen una media de tamaño de menos 100 bytes, que es menor incluso que el de aplicaciones P2P, este hecho induce a pensar que, los flujos reconocidos por la aplicación son los llamados “flujos de control” en dicho protocolo, que cuando realiza una transferencia de archivos abre dos flujos, uno para establecer la conexión y otro para enviar los datos de un servidor a otro. Por tanto, pasaría algo similar a lo explicado ya anteriormente con las aplicaciones P2P, la aplicación *L7-Filter* es buena identificando el tráfico de establecimiento y mantenimiento de la conexión en todos estas aplicaciones, pero no ocurre lo mismo con el tráfico de datos, que es realmente el que mayor banda ancha ocupa. Otro hecho que se puede observar para la aplicación *Bittorrent* es que sus flujos tienen casi siempre la misma media, de 100 bytes y 200 bytes (gráfica casi vertical en estos puntos), los cuales ocupan el 80% de los flujos. Esto también puede ser determinado por la propia definición del protocolo, donde los paquetes ya tienen un tamaño determinado en estos flujos de señalización.

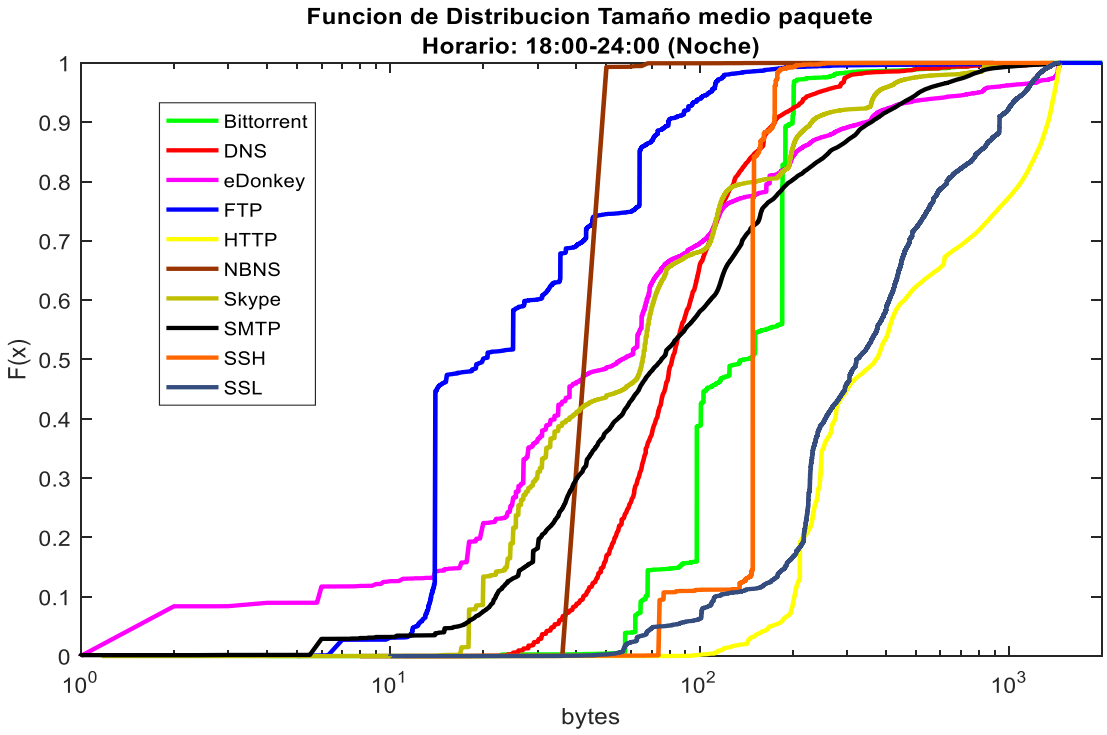


Figura 4-6: Función de distribución de tamaño medio del paquete por flujo. Se incluyen aplicaciones más populares

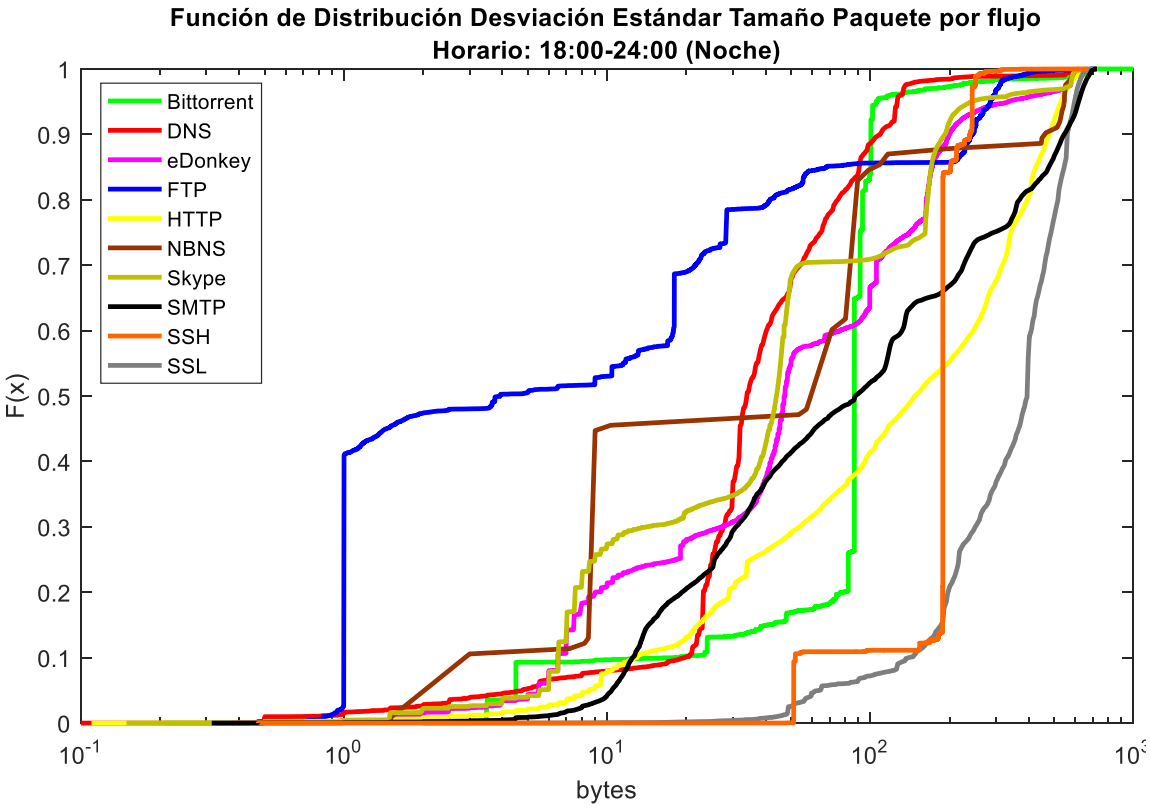


Figura 4-7: Función de distribución de la desviación estándar del paquete por flujo. Se incluyen aplicaciones más populares

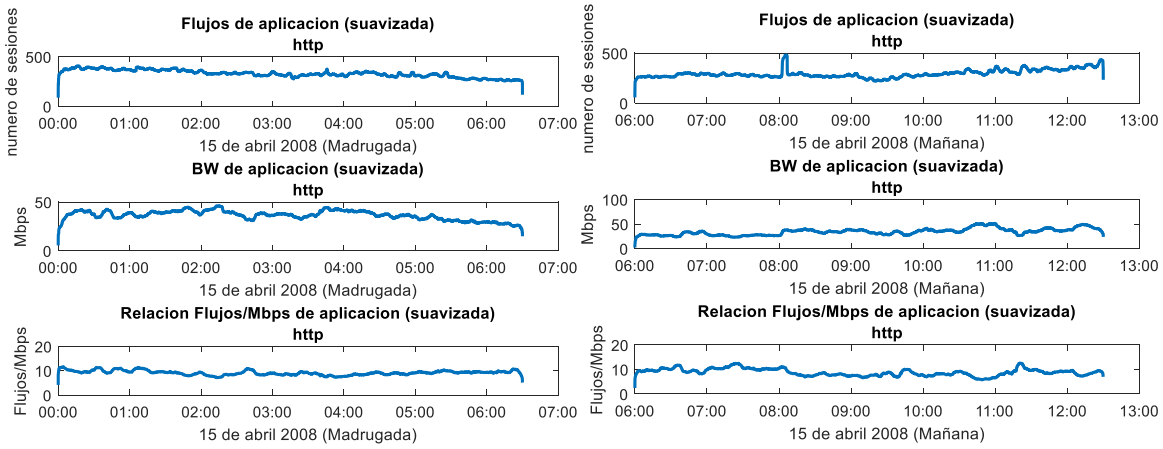
Esta última Figura 4-7 representa la última de las gráficas que se han realizado en torno a las aplicaciones más usadas. Para entender bien esta figura se debe tener en cuenta que

mientras más abrupto sea el salto entre el 0 y el 1 significara que mayor parte de los flujos de una determinada aplicación tiene
Dentro de las aplicaciones P2P, los flujos de eDonkey tienen una menor varianza que las de BitTorrent, lo que significa que los flujos del primero son más parecidos entre ellos que los de eDonkey, y por tanto mientras más se parezca más sencillo será encontrar patrones para detectarlos. Este hecho sucede también en SSH, donde el 90% de los flujos tienen prácticamente la misma desviación estándar, que es bastante grande, lo que nos dice dichos flujos contienen paquetes de tamaños muy diferentes. Estudiando cómo funcionan los flujos SSH en muchos casos cuando se descargan archivos de un servidor a otro, el servidor que descarga datos tiene paquetes muy grandes mientras que el otro solo envía paquetes señalización (los ACK del protocolo TCP) de tamaño muy pequeño. Esto da lugar a una desviación estándar muy grande, como se puede observar en la Figura 4-7.
De manera general, este medidor para diferenciar de manera clara entre aplicaciones no es útil, pues a la vista está que la mayor parte de aplicaciones se muestran más o menos parecidas, aunque se observa que SSL, HTTP y SMTP son las más suavizadas y por tanto con flujos más diferentes.

4.1.3 Estudio estadísticas para protocolo HTTP

En el presente trabajo se ha decidido estudiar el protocolo HTTP de manera separada porque, además de ser el protocolo de red más popular en la web (HTTP y HTTPS) la aplicación DPI usada, *L7-Filter*, al descargar la firma, incluye que los flujos HTTP tienen un 99% de posibilidades de éxito, esta afirmación viene sostenida en [4] donde se realiza un estudio sobre diferentes aplicaciones DPI y su eficacia sobre la detección de diferentes aplicaciones.

Por ello, con una gran cantidad de datos obtenidos en dicha aplicación se ha profundizado más en las características obtenidas de dicho flujo. Al realizar el estudio diario de las trazas dividido por tramos horarios se han buscado si el momento del día tiene influencia en cómo se comportan los flujos de red HTTP.
Observando los datos de las series temporales específicos para este protocolo se puede ver cómo, aunque el tráfico en cuanto a flujos y ancho de banda va aumentando durante las horas de mayor actividad en Internet por parte de los usuarios, la relación se mantiene estable durante todo el día, lo que supone que el horario y que el aumento de tráfico no hace cambiar dicha relación.



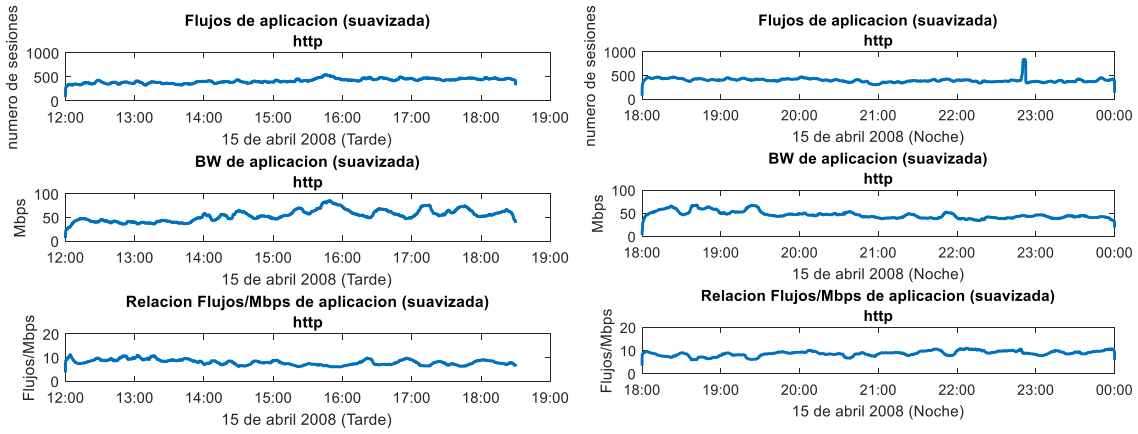


Figura 4-8: Series temporales de flujos, ancho de banda y flujos/Mb para HTTP. Año 2008

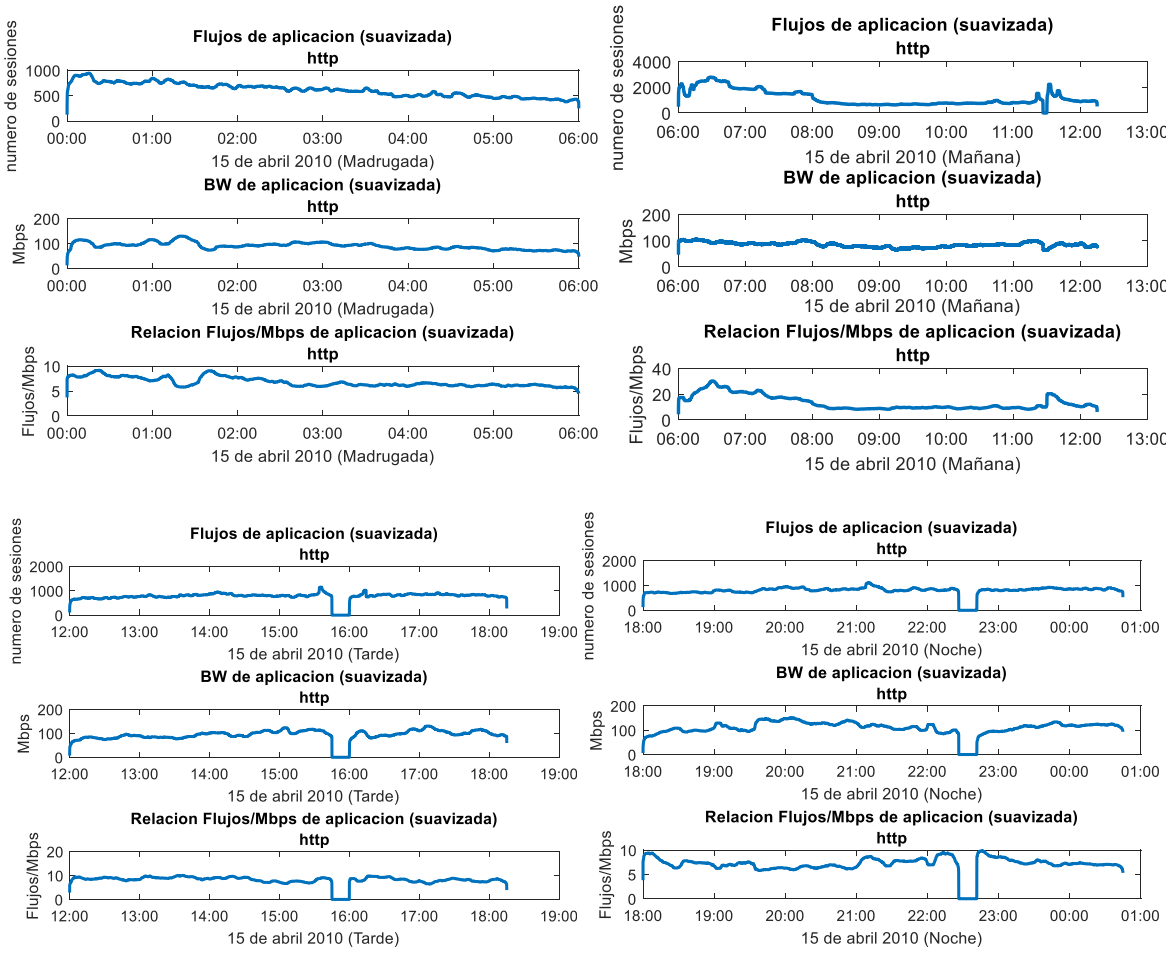


Figura 4-9: Series temporales de flujos, ancho de banda y flujos/Mb para HTTP. Año 2010

Nota: la parte de las figuras donde cae a cero se debe a que la subtraza de ese tramo horario no estaba disponible.

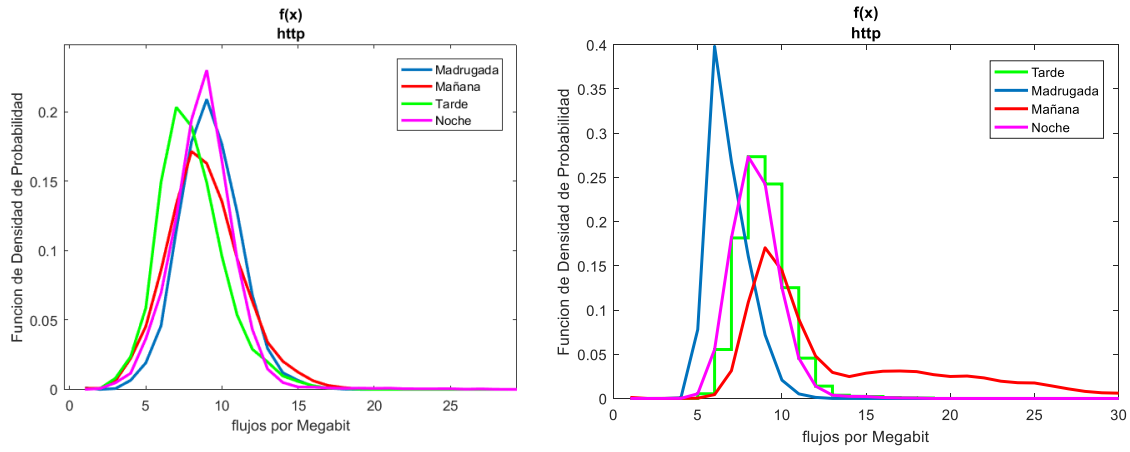
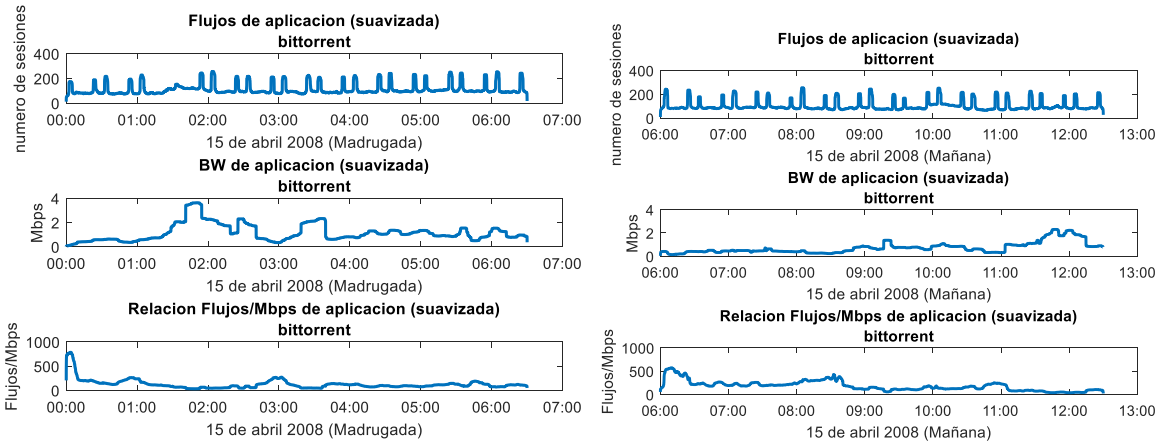


Figura 4-10: Estimación de probabilidad mediante KDE. 2008 (izquierda) 2010 (derecha)

Estos resultados hacen indicar que los flujos HTTP son por lo general más largos y estables que el de otras aplicaciones, además la forma en que funciona dicho protocolo, con una sola sesión por cada vez usuario que utiliza dicha aplicación. Cabe destacar que en la figura se han suavizado las funciones de las gráficas. De todas formas, como se observa en la Figura 4-1, este suavizado no significa que vayan a desaparecer picos de una función. Como se puede ver en la Figura 4-10, HTTP resulta ser la aplicación más estable de todas las analizadas, presenta estimaciones de probabilidad muy similares tanto en 2008 como en 2010, y a pesar de que la cantidad de datos ha aumentado considerablemente la medida de flujos/Mb. En conclusión, HTTP presenta unas características bien definidas. Si se observan los anexos otras aplicaciones presentan una mayor variabilidad al hacer dichas estimaciones. En cuanto al resto de estadísticas que también se han analizado como pueda ser la media entre llegadas o el tamaño medio de paquete, HTTP si presenta una variación más grande y no se puede estimar ningún valor predominante para diferenciar HTTP de otras aplicaciones.

4.1.4 Estudio estadísticas para aplicación BitTorrent

En este apartado se ha analizado de forma separada la aplicación BitTorrent, pues es la aplicación P2P que cuenta con un espacio muestral bastante grande y puede ser la más representativa dentro de este tipo de aplicaciones.



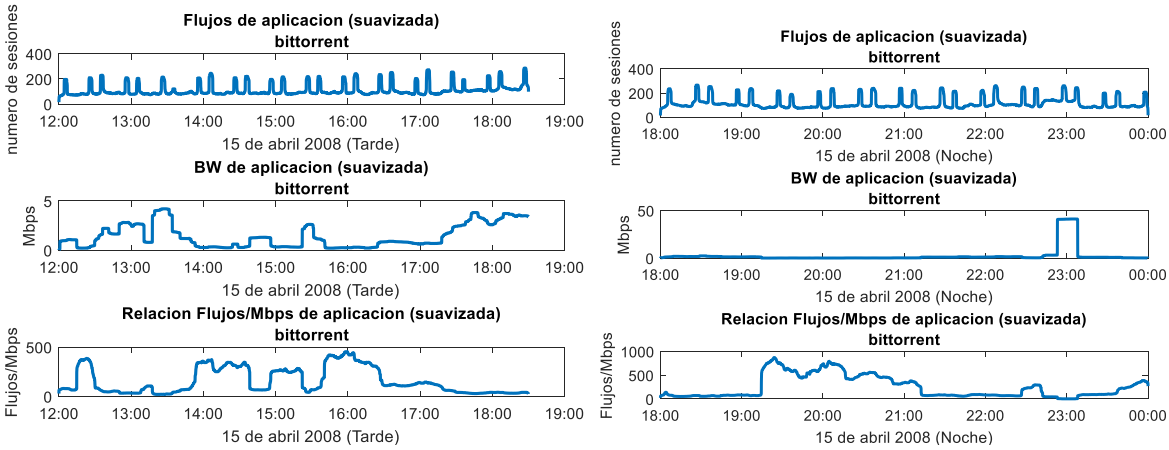


Figura 4-11: Series temporales de flujos, ancho de banda y flujos/Mb para BitTorrent. 2008

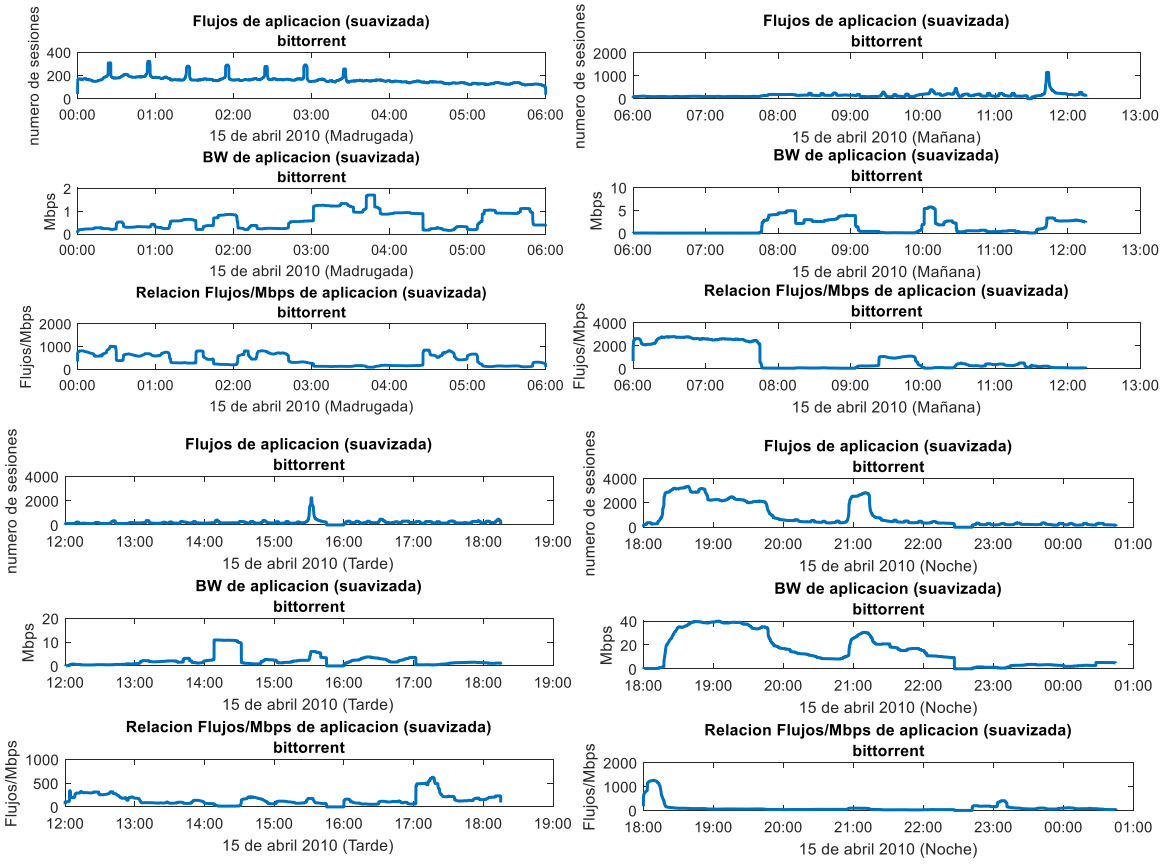


Figura 4-12: Series temporales de flujos, ancho de banda y flujos/Mb para BitTorrent. 2010

Las figuras 4-11 y 4-12 muestran que el tráfico de la aplicación no es constante, con muchos picos, de los cuales no se conocen exactamente su origen. Lo que si se observan es que en 2010 las cotas de Mb/s son bastante más altas que las de 2008. Lo que supone que el aumento de tráfico por esta aplicación es bastante alto durante esos dos años, en lo que la red de WIDE se refiere. Aunque solo se hubiera detectado tráfico de aplicación, cabe pensar que un aumento de este tipo de tráfico conlleve también un aumento del tráfico de archivos compartidos.

Lo más relevante que se puede extraer de analizar las figuras 4-11, 4-12 y 4-13 es que, aunque las series temporales de la aplicación son totalmente irregulares como es el caso, se puede observar que la estimación de probabilidad si son bastante semejantes, tanto en un año como otro. A diferencia de la Figura 4-2, donde comparado con otras aplicaciones no

parece tener una moda muy clara, al escalar la gráfica para esta aplicación si se puede observar como tiene claramente una estimación en torno a los 150 flujos/Mb, unas 10 veces más en proporción a HTTP. Es importante destacar que seguramente la mayor parte del tráfico analizado en estas figuras es de señalización, el cual tiene muy pocos datos en proporción a los flujos, lo que hace que la estimación sea tan alta. Otra cosa que nos revela esto es el bajo ancho de banda que ocupa en ciertas franjas horarias la aplicación, prácticamente nulas, lo que revela que seguramente bastante cantidad de tráfico no esté siendo etiquetado. Observando los diferentes tramos horarios en la Figura 4-13, no se observan diferencias notables, ni tampoco de un año para otro.

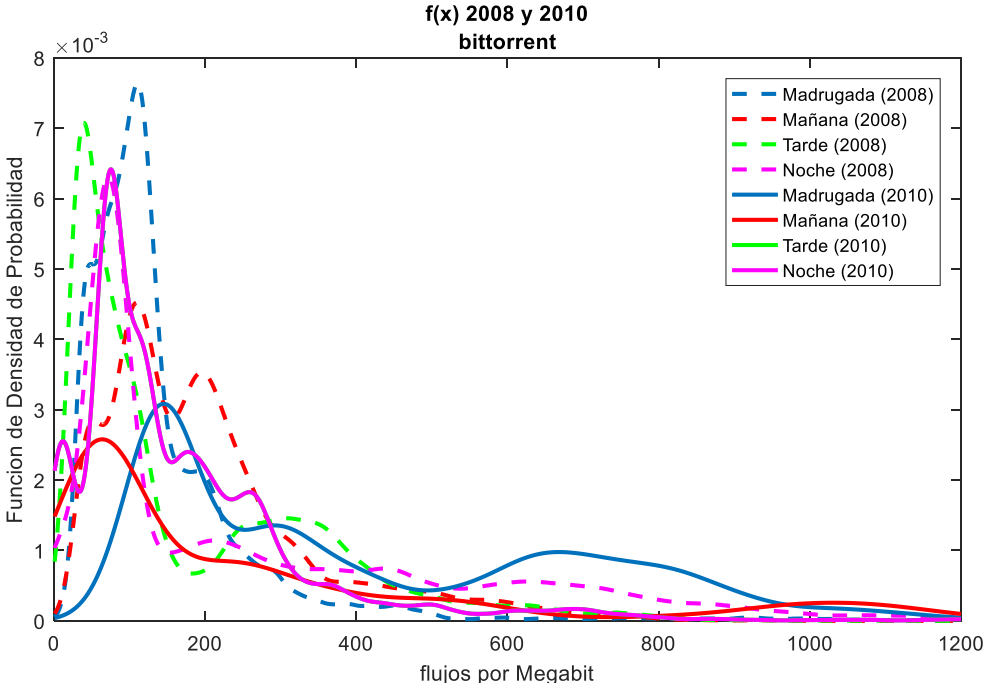


Figura 4-13: Estimación de probabilidad mediante KDE

En conclusión, BitTorrent tiene una estimación de probabilidad mucho menor que HTTP, como se puede ver en el eje y de la Figura 4-13 respecto al Figura 4-10, pero si cuenta con una tendencia clara hacia esos aproximadamente 150 flujos/Mb que se han comentado.

4.1.5 Estudio estadísticas de otras aplicaciones.

Además de haber realizado pruebas con HTTP y BitTorrent, se han obtenido resultados de manera separada de las demás aplicaciones más populares como pueden ser eDonkey, SSH, SMTP o Skype. Observando los resultados de flujos entre ancho de banda se puede observar claras divergencias entre las distintas aplicaciones. Estas divergencias son patentes y similares en todos los tramos horarios vistos y en las muestras de ambos años, lo que se puede extraer de estos hecho es que la relación de flujos entre el ancho de banda puede ser un elemento reseñable a la hora de diferenciar diferentes tipos de servicios o aplicaciones en red.

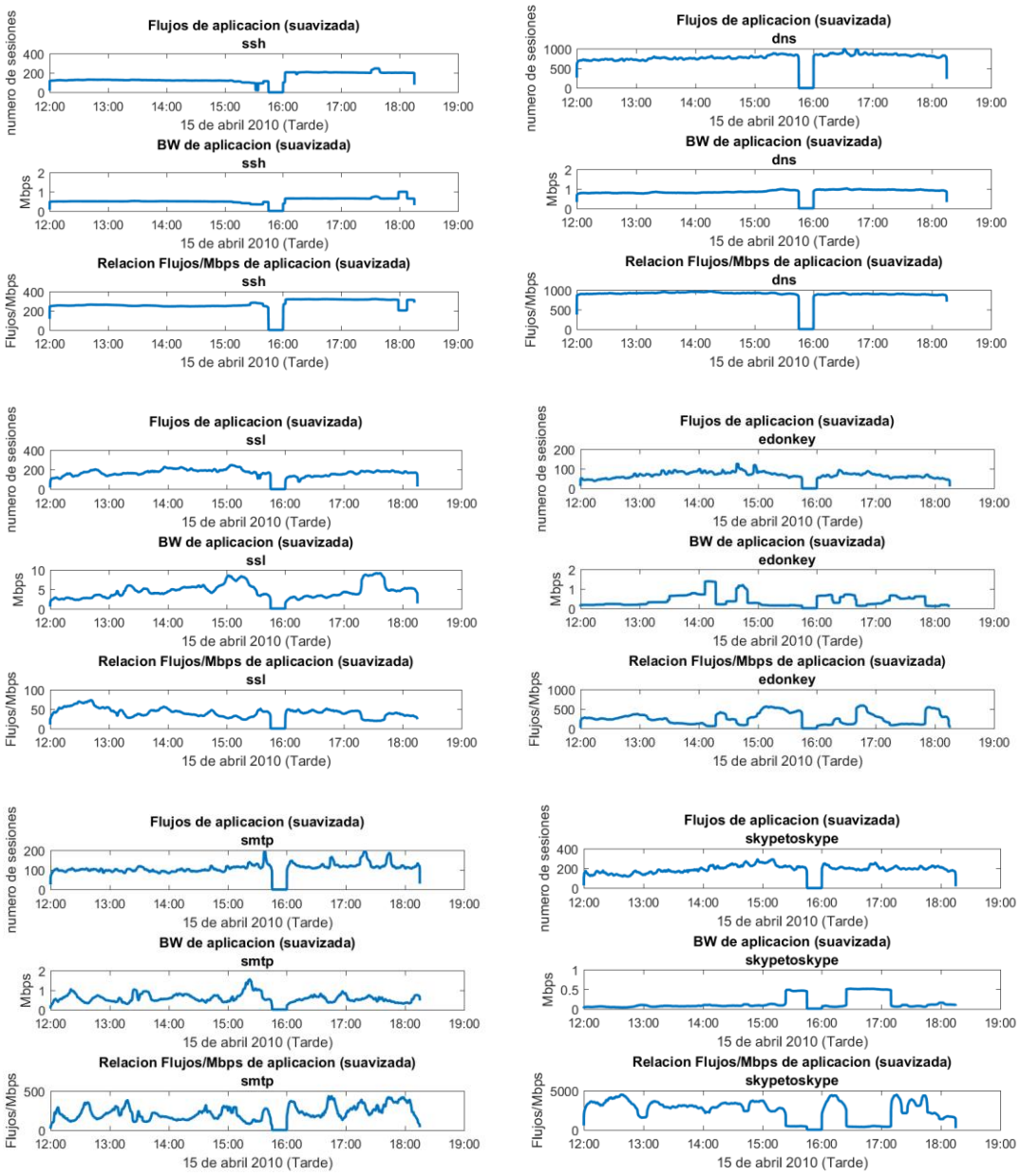


Figura 4-14: Series temporales de flujos, ancho de banda y flujos/Mb. Aplicaciones populares 2010

En la Figura 4-14 se puede observar las series temporales de flujos, ancho de banda y sus cocientes, suavizadas de antemano, de las aplicaciones más populares. Los resultados arrojan que por lo general las aplicaciones populares tienen bastante regularidad en el tiempo, con diferentes picos a lo largo del día. En este caso, se muestra el tramo horario de tarde, el cual tiene un hueco en entre las 15:45 y 16:00 debido a que falta la subtraza de 15 minutos correspondiente a dicho horario. La aplicación P2P eDonkey tiene tramos largos de minutos con ancho de banda más alto y luego vuelve a niveles, más bajos. Es en estos tramos donde el número de flujos por Mb se dispara al bajar el ancho de banda. Las aplicaciones más estables son DNS y SSH, y las aplicaciones que presentan mayores vaivenes son Skype y SMTP. SSL se situaría en un punto intermedio, sin embargo, cabe destacar que esta aplicación cuenta con una mayor correlación entre el ancho de banda y el número de flujos, lo que hace que la relación entre ambas sea más estable a lo largo de la serie temporal.

5 Uso de *Machine Learning* para descubrir aplicaciones

El principal motivo de usar Weka, una aplicación de *machine learning*, para diferenciar el mix de aplicaciones en red se debe a que en muchas ocasiones es normal trabajar con trazas de tráfico las cuales no tienen carga útil en la capa de aplicación. En dichas trazas, no es útil usar métodos DPI como el usado en la sección 3, donde se analizan patrones existentes en los bytes de dicha capa. Por ello, se van a buscar relaciones entre las aplicaciones más populares y las estadísticas obtenidas, gracias a los datos volcados por nuestra aplicación DPI. En Weka se pueden aplicar algoritmos los cuales de manera automática ajustan los parámetros a los valores que mejor discriminan entre las aplicaciones, por ello, se usaran los resultados obtenidos por la aplicación DPI, en la que cada flujo cuenta con una serie de estadísticas las cuales van asociadas a la aplicación detectada. Weka denomina atributos a los datos usados y clase a la aplicación con la cual ha sido etiquetado cada flujo.

5.1 Selección de muestras

Para esta parte del trabajo, tras analizar que las trazas estudiadas contaban con una gran cantidad de flujos, se ha decidido establecer una serie de requisitos a la hora de escoger una serie determinada de flujos, que se consideran válidos para su estudio. Asimismo, se ha decidido, en función de los resultados obtenidos en la sección 4, escoger una serie de parámetros estadísticos que parecen ser más discriminantes a la hora de diferenciar entre aplicaciones. A continuación, se muestran que filtros se han hecho para ver que flujos se usan en el programa:

- En primer lugar, se han eliminado todos los flujos de tamaño un paquete. Se ha decidido eliminarlos porque dichos flujos al ser de un único paquete, no cuenta con varias de las estadísticas que van a ser usadas, como puede ser el tiempo entre llegadas).
- Tambien se han eliminado los flujos que no cuentan con al menos un paquete en cada sentido del flujo, que formarían una sesión como se definió en secciones anteriores.
- Se han eliminado todos los flujos no etiquetados de la muestra, pues en el caso de estudio se va a usar inicialmente un aprendizaje supervisado donde las estadísticas irán asociadas a un cierto tipo, que es la aplicación identificada.
- Igualmente, se eliminarán del estudio aquellos flujos que han sido etiquetadas con aplicaciones poco populares. Esto se ha usado con la intención de reducir los tipos (aplicaciones) en el programa va a intentar discriminar. Los problemas multiclase aumentan su complejidad mientras más clases tengan. [21].
- Se han balanceado las muestras entre las aplicaciones que se van a estudiar. Esto es, se van a coger un numero de muestras similar para cada tipo. [20] [21].
- También se han reducido las estadísticas que se van a usar en el programa, usando solo aquellas que parecen más útiles a la hora de discriminar.

5.2 Explicación algoritmos implementados

Se han usado distintos algoritmos que vienen incluidos en la aplicación para el reconocimiento de patrones. Los que mejor resultado han obtenido han sido los siguientes:

- Métodos predictivos basados en árbol de decisión: el árbol de decisión es un clasificador que trata de encontrar la mejor opción en cada acción o decisión a tomar por el clasificador. Contará con un nodo inicial o raíz desde donde empieza todo el conjunto de muestras y según los valores que tenga en cada atributo (en el caso del presenta trabajo se refiere a las estadísticas del flujo). Es un método de aprendizaje supervisado pues necesita de un entrenamiento previo para optimizar el clasificador [24]. Se han usado los siguientes algoritmos basados en el árbol de decisión:
    - J48 (algoritmo C4.5): este algoritmo estudia la entropía (incertidumbre) de todos los atributos que contiene el grupo de muestras construye el árbol de decisión situando siempre como nodo raíz al atributo con menor incertidumbre o entropía. El árbol de decisión irá descendiendo usando atributos con mayor incertidumbre hasta llegar al final. Con este método además se puede conocer aquellos atributos que resultan más útiles para reconocer la clase (en este caso Aplicación de red) a la que pertenece. Un atributo con menor incertidumbre indica mayor facilidad para reconocer una clase [25].
    - PART: método simplificado del anterior algoritmo C4.5 que se construye mediante reglas. Este método sustituye las ramas de árbol de decisión por reglas, aunque su función es la misma, solo incluye las mejores "hojas", esto es, las ramas del árbol de decisión que son más efectivas para escoger una clase. Facilitaría las tareas de programación del clasificador, ya que Weka muestra las reglas al realizar el algoritmo [26].
    - *RandomForest:* agoritmo que funciona probando de manera aleatoria multitud de modelos, los cuales sean más o menos neutros o imparciales. Una vez se tienen gran cantidad de estos modelos aleatorios se puede reducir la variación y mejorar la calidad del clasificador [26].
- Regresión logística multinomial: este método trata de encontrar la clase a la que pertenece cada elemento de la muestra a través de una función de la muestra. En dicha función cada variable independiente viene representada por el atributo mientras que cada variable es representada por el valor que se le da a dicho atributo. Los resultados de dicha función pueden determinar, si se construye correctamente, a que clase pertenece cada elemento (flujo) según sus atributos (estadísticas).
- SVM (máquinas de vector soporte): este método es una alternativa a la regresión logística, aunque tiene cierto parecido difiere de este en que cuando se trata de un problema multiclase, como en el presente trabajo, el modelo lo convierte en un problema binario, es decir, enfrenta a una clase contra otra y luego combina todos esos modelos.

  El algoritmo que usa Weka, que se llama SMO (Sequential minimal optimization), está basado en este modelo, Esta implementación reemplaza globalmente todos los valores perdidos y transforma los atributos nominales en binarios. También normaliza todos los atributos por defecto. Los coeficientes en la salida se basan en los datos normalizados, no en los datos originales; esto es importante para interpretar el clasificador.

También se han usado algoritmos que incluye Weka que utilizan técnicas de modelos diferentes, es el caso LMT (modelo logístico de árboles) que utiliza funciones de regresión logística en los nodos. Este modelo tiene, por tanto, una carga computacional más grande, y es más difícil de ajustar debido a que la complejidad del clasificador aumenta.

5.3 Resultados obtenidos

Los resultados que se han obtenido en Weka han sido, en ciertos algoritmos, bastantes satisfactorios. En todos los casos se ha usado las muestras de 2008 como muestra de entrenamiento y las muestras de 2010 se han usado de test. Esto quiere decir que la aplicación Weka construye el modelo a partir de las muestras del 2008 y lo comprueba con las del 2010.

Los resultados obtenidos muestran que los métodos de detección basados en arboles de decisión tienen un porcentaje alto de acierto. Por ejemplo, el algoritmo J48 cuenta con una buena cantidad de muestras detectadas correctamente, entre el 84% y el 89%. Son unos buenos resultados sobre todo teniendo en cuenta que se cruzan muestras con dos años de diferencia. Por otro lado, se ha comprobado, que el porcentaje de detección no mejora de manera muy notoria (se sitúa entre el 90% y el 92%) usando una muestra de entrenamiento y de test del mismo año. La poca diferencia entre este porcentaje con los de la tabla puede deberse a dos factores:

- Las muestras, aunque sean de diferentes años, tienen cierta dependencia. Estas dependencias se deben a varios hechos, como puede ser que las trazas hayan sido recogidas en el mismo punto de la red o que el criterio de selección de muestras ha sido el mismo en ambos casos, con lo que se han podido desechar parte de las muestras de uno u otro año que pudieran tener mayores divergencias que las seleccionadas.

- El otro hecho se debe a que se han usado un conjunto de muestras etiquetadas previamente por un clasificador que no es totalmente confiable, esto significa que estas muestras seguramente presenten un sesgo respecto al total de muestras que tendríamos si el etiquetado de dichas muestras fuera 100% confiable y se supiera con seguridad que cada flujo ha sido etiquetado correctamente en cada aplicación.

Estos dos hechos no quitan mérito, de todas formas, a la calidad de clasificador, un factor que puede explicar los buenos resultados ha sido el ajuste de opciones en el clasificador, como pueden ser el factor de confianza o el número de nodos “hoja” máximo. En este caso, se utilizó un factor de confianza de 0.25 y un número máximo de nodos hoja de 3 por nodo.

J48		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	87,665%	86,100%	87,934%	87,468%
	2008-2	88,138%	86,170%	87,933%	87,311%
	2008-3	87,878%	85,370%	85,926%	86,050%
	2008-4	86,898%	84,515%	86,214%	86,429%

Tabla 5.3-1 Porcentaje de aciertos en aplicaciones con el modelo de clasificación J48

Weka, además de dar el porcentaje de aciertos, da otros datos muy útiles a la hora de descubrir que aplicaciones son más fáciles de identificar por el clasificador y en cuales más fallos. Esto se ve claramente en la matriz de confusión que crea tras la clasificación.

a	b	c	d	e	f	g	h	<--	clasificado	como	
9754	1	4	0	226	0	15	0	|	a	=	HTTP
3	7728	2014	143	27	0	29	56	|	b	=	BitTorrent
6	384	8692	533	40	0	72	273	|	c	=	eDonkey
1	35	490	9396	16	1	30	31	|	d	=	Skype
958	137	261	8	7865	0	756	15	|	e	=	SSL
3	63	29	0	17	9706	182	0	|	f	=	SSH
157	26	170	6	698	21	8705	217	|	g	=	SMTP
0	45	247	69	105	0	870	8664	|	h	=	DNS

Tabla 5.3-2 Matriz de confusión. Modelo de clasificación basado en J48

En esta matriz puede observarse que son las aplicaciones HTTP y SSL las que mayor fiabilidad tienen pues cuentan con un 97,5% y un 97,1% de verdaderos positivos, respectivamente. Por otro lado, se observa que BitTorrent es en este caso la aplicación con peor porcentaje de acierto, y es de destacar que la mayoría de sus flujos etiquetados en otra aplicación por el clasificador han ido a parar a la aplicación eDonkey, justamente la otra aplicación P2P, lo que indica que sus estadísticas (o atributos como se denominan en Weka) son similares y dificultan la detección. También pasa de manera inversa, aunque en menor medida. SSL, la otra aplicación con porcentaje más bajo de verdaderos positivos se observa que muchos de los flujos son etiquetados por HTTP o por SSH, las dos aplicaciones con mejor porcentaje de verdaderos positivos (que no de precisión), lo que nos da pistas sobre como el clasificador actúa en muchos casos, teniendo cierta tendencia a escoger estas dos aplicaciones sobre otras en algunos flujos.

Reconocer el comportamiento del clasificador es de suma importancia, los clasificadores pueden ser diseñados de múltiples formas. Al no resultar infalibles en la mayoría de los casos, debe ser el diseñador el que elija que conviene más. Usando el ejemplo con las aplicaciones de red el clasificador podría diseñarse, por ejemplo, para que ciertas aplicaciones sean siempre, o casi siempre, detectadas correctamente (100% de verdaderos

positivos), esto iría en detrimento de la calidad de clasificación de otras aplicaciones, que seguramente se etiquetarían por la aplicación "favorecida" en dicho clasificador. En el caso del presente trabajo, los clasificadores creados con los diferentes son neutros en cuanto a detección por aplicación, pues se ha usado la misma cantidad de flujos en las muestras en cada aplicación. Por tanto, que una aplicación sea detectada con mayor facilidad que otras implica que sus atributos (estadísticas) están más sesgadas respecto del resto de atributos que contienen las otras aplicaciones.

Se puede caer en la equivocada idea de que es mejor ajustar el clasificador a los porcentajes que pueda tener cada clase. En nuestro caso, por ejemplo, significaría dar mayor prioridad a los flujos DNS y HTTP, pues ocupan aproximadamente un 25% y un 20% de los flujos respectivamente. Por lo que si el clasificador detecta con mayor facilidad las aplicaciones más usadas la precisión total será mejor. Esto sí es cierto, pero no siempre puede ser lo adecuado. Como contraejemplo se puede explicar el caso del contenido malicioso en Internet, el cual representa un porcentaje muy pequeño respecto del total. Sin embargo, un cortafuegos debe ser construido para detectar únicamente dicho contenido malicioso. Por tanto, el clasificador se diseñará de manera que prevenga la entrada de dicho contenido, dando lugar a muchos falsos positivos (flujos normales etiquetados como maliciosos y por tanto descartados por el cortafuegos). Como consecuencia, dicho clasificador tendrá de manera general una precisión mala, pero cumplirá el objetivo para el cual se ha diseñado.

Otro conjunto de medidas que ofrece Weka tras la clasificación es la precisión clase por clase. En este caso, lo que la aplicación realiza es tratar como un problema biclase o binario el clasificador, es decir, para cada aplicación las clases serán SI o NO, en función de si pertenece o no a dicha aplicación. Con esta clasificación obtiene las siguientes medidas individuales:

- Verdaderos positivos: muestras clase C que se clasifican correctamente en la clase C.
- Falsos positivos: muestras no pertenecientes a la clase C pero que se clasifican como clase C.
- Precisión: valor entre 0 y 1 que aumenta más cuando hay pocos falsos positivos.
- *Recall:* valor entre 0 y 1 que aumenta cuando hay pocos falsos negativos.

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

tp: verdaderos positivos

fp: falsos positivos

fn: falsos negativos

- F-measure: combinacion de ambos valores anteriores

$$F = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

MCC: medida de calidad de un clasificador binario, varía entre -1 y 1 siendo este el óptimo.

Área ROC: es el bajo la curva ROC, la cual es una gráfica que muestra el rendimiento del clasificador biclase, varía entre 0 y 1 siendo 1 el clasificador óptimo.

Área PRC: mientras que ROC evalúa la calidad del clasificador tanto para las muestras clasificadas como una aplicación como las descartadas, el objetivo del cálculo de esta área es mostrar una curva que evalúe solo el comportamiento de las muestras que realmente pertenecen a la aplicación. Es útil en aquellos casos en los que importa acertar solo una de las dos clases y el resultado de la otra no sea importante.

Como ejemplo se puede poner los datos de J48.

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,975	0,016	0,896	0,975	0,934	0,925	0,988	0,912	HTTP
0,773	0,01	0,918	0,773	0,839	0,822	0,959	0,85	BitTorrent
0,869	0,046	0,73	0,869	0,794	0,765	0,945	0,706	eDonkey
0,94	0,011	0,925	0,94	0,932	0,923	0,981	0,911	Skype
0,787	0,016	0,874	0,787	0,828	0,807	0,922	0,767	SSL
0,971	0	0,998	0,971	0,984	0,982	0,986	0,976	SSH
0,871	0,028	0,817	0,871	0,843	0,82	0,945	0,762	SMTP
0,866	0,008	0,936	0,866	0,9	0,887	0,955	0,897	DNS

Tabla 5.3-3. Calculo de medidores de rendimiento.

Los distintos algoritmos basados en arboles de decisión han cosechado buenos resultados, siempre entre un 80% y 90% de acierto, de manera que se puede confirmar que dichos métodos son más efectivos que otros tipos usados como regresión logística o máquinas de vector soporte que en los mejores casos dan valores de 70% de acierto. En el anexo D se puede ver los resultados de los distintos métodos con tablas análogas a la Tabla 5.3-1.

5.4 Caso práctico: trazas recogidas por aplicación

En este apartado se van a probar las trazas recogidas manualmente. Una vez haya pasado por la aplicación DPI *L7-Filter* se extraerán las estadísticas y se aplicaran los clasificadores usados previamente en Weka, para ver si detecta correctamente la aplicación que es. En este caso se tiene la ventaja de que se conoce de manera totalmente confiable la aplicación.

Se han guardado cuatro trazas de red desde un terminal. En las dos primeras trazas se ha usado únicamente BitTorrent por parte del usuario, compartiendo archivos en la red. La tercera, traza ha sido realizada mientras el usuario del terminal navegaba en diferentes

páginas web, pertenecientes al protocolo HTTP y HTTPS. Por último, la cuarta traza fue realizada mientras el usuario realizaba una conexión SSH a un servidor remoto.

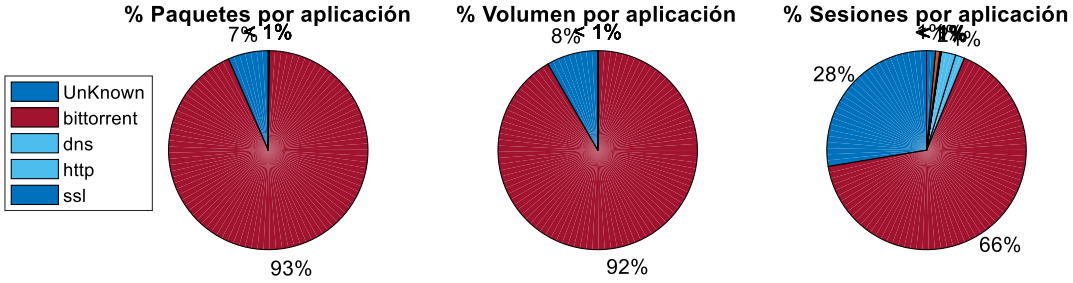


Figura 5-1: Porcentaje de paquetes, bytes y flujos por aplicación. BitTorrent

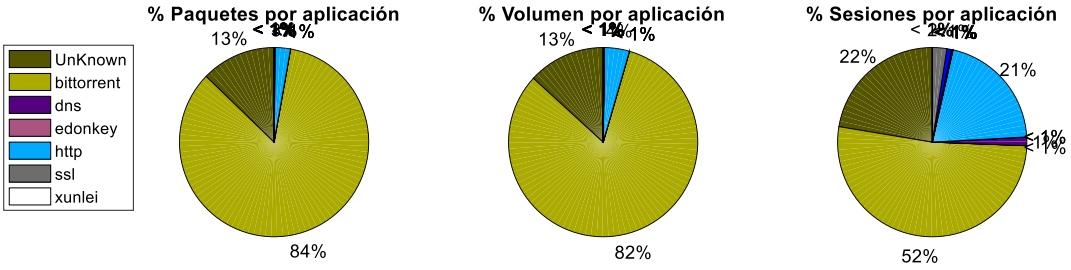


Figura 5-2: Porcentaje de paquetes, bytes y flujos por aplicación. BitTorrent (2ª captura)

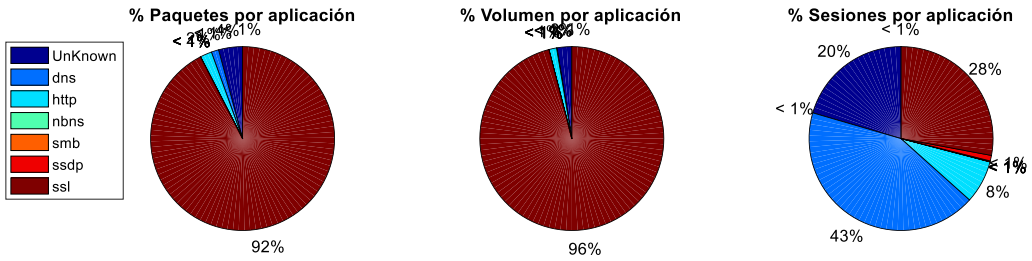


Figura 5-3: Porcentaje de paquetes, bytes y flujos por aplicación. HTTP, SSL y DNS

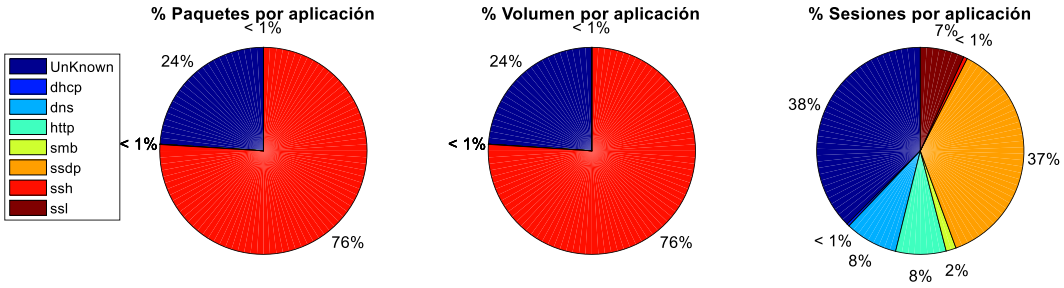


Figura 5-4: Porcentaje de paquetes, bytes y flujos por aplicación. SSH

Analizando los resultados con la Figura 5-1 se observa que en el caso de BitTorrent la detección en la primera traza es bastante buena. Se han podido contrastar múltiples dudas que habían surgido con las trazas de muestra. En primer lugar, se ha visto como el tráfico

de señalización que BitTorrent produce, en muchos casos, flujos de 1 paquete, por lo que el clasificador *L7-Filter* está funcionando correctamente para estos casos. Por otro lado, esta traza se supone que solo contiene tráfico BitTorrent, pues solo se usó dicha aplicación durante la captura. Por ello, es de suponer que el porcentaje que aparece con etiqueta desconocida, un 8% en el caso del volumen de datos y un 28% en el caso de las sesiones, sea también tráfico BitTorrent. Este supuesto también confirma otra de las dudas que surgieron en las muestras de WIDE, pues había tráfico desconocido que se creía podía pertenecer a aplicaciones P2P, y más en concreto a BitTorrent. Estos flujos que no son identificables se deben corresponder con flujos de señalización, los cuales tienen un ancho de banda muy bajo, por ello hay un porcentaje mucho más alto de flujos desconocidos que de bytes de datos desconocidos.
Los resultados obtenidos en la Figura 5-2 son similares a los anteriores excepto por un detalle, hay un porcentaje significativo de flujos que ha sido etiquetado como HTTP. Estos flujos han sido etiquetados de manera incorrecta por la aplicación DPI, se comprobó observando los datos de los flujos que al estar activada la identificación por puertos estos fueron asignados erróneamente a HTTP al usar el puerto 80. Este hecho ejemplifica porque el uso de puertos para identificar aplicaciones ha quedado del todo obsoleto.

En la tercera traza estudiada, en la cual se capturó datos de navegación del usuario a través de diferentes páginas web estáticas y de video en *streaming*. La Figura 5-3 muestra en los resultados como gran parte del tráfico web actual está encriptado con SSL, el cual se utiliza junto con HTTP para formar el protocolo HTTPS. El 96% del volumen de los datos pertenece a HTTPS/SSL mientras que respecto a los flujos solo representa un 28%. Esto se debe a que el video en streaming capturado usaba SSL. En cambio, para encontrar tráfico HTTP se capturó tráfico proveniente de páginas web estáticas con poca cantidad de datos para descargar, por ello esa diferencia entre el porcentaje de datos, 1%, y el porcentaje de flujos, del 8%. A destacar el alto porcentaje que ocupan los flujos DNS, un 43%, cifra similar a lo que ocurre en las muestras analizadas en la Tabla 4.1-1. Como ocurría en el caso de las muestras de WIDE, DNS tiene un ancho de banda bajo comparado con su número de flujos.

La Figura 5-4, realizada a partir de una captura de SSH, muestra de manera bastante exagerada las grandes diferencias que pueden existir entre los flujos de red [3]. Es este caso existe un grupo de varios flujos, 4 o 5, que representan menos de 1% de los flujos pero que sin embargo representan más del 75% de los datos y de los paquetes. Esta desproporción se debe a que durante la captura se descargaron varios archivos grandes de varios GB de memoria. La mayoría de flujos etiquetados pertenecen a aplicaciones como SMB, SSH y HTTP, los cuales han podido ser usados en el terminal de manera secundaria a la aplicación SSH. Destaca un 37% de flujos SSDP, que apenan ocupan ancho de banda (<1%). Este protocolo sirve para la búsqueda de dispositivos o ciertos servicios conectados en una red y utiliza UDP, posible que este protocolo funcione de manera complementaria al programa usado para realizar la conexión SSH, realizando muchas peticiones a la vez para conectar dispositivos. Por ello, la cantidad de flujos que han sido etiquetados por esta aplicación es tan alta respecto al ancho de banda.

Una vez evaluadas la calidad del clasificador *L7 Filter*, con capturas reales, se ha probado a usar los modelos de predicción creados en Weka con las trazas de muestra con estas. Los resultados obtenidos han sido bastante malos respecto a los obtenidos con las muestras de entrenamiento, en torno un 25 % de acierto. Este bajo acierto se debe a que las estadísticas de los flujos son bastante diferentes. Esto se puede deber a los factores que se describieron

en la sección anterior, como el origen de las trazas, la topología de la red y de los equipos que la componen.

5.5 Resultado modelo J48 sobre trazas propias

Modelo: J48, entrenado a partir de las trazas de muestra de 2008.

Muestra: BitTorrent

Instancias correctamente clasificadas	**1505**	**22.17%**
Instancias incorrectamente clasificadas	**5300**	**77.83%**
Número total de instancias	**6805**	**100%**

a	b	c	d	e	f	g	h	<--	clasificado como
36	12	88	53	32	0	1	0	a	HTTP
77	1417	2492	1182	309	1	595	224	b	BitTorrent
0	0	0	0	0	0	0	0	c	eDonkey
0	0	4	4	0	0	0	0	d	Skype
79	0	1	0	48	0	9	0	e	SSL
0	0	0	0	0	0	0	0	f	SSH
0	0	0	0	0	0	0	0	g	SMTP
0	0	132	9	0	0	0	0	h	DNS

Tabla 5.5-1 Matriz de confusión. Traza BitTorrent

En la matriz de confusión se puede observar como la mayoría de trazas de BitTorrent mal etiquetadas son distinguidas como una aplicación eDonkey, que también es P2P. Al ser las dos aplicaciones P2P su comportamiento es similar y mas difícil de distinguir.

Muestra: SSL, HTTP y DNS

Instancias correctamente clasificadas	**2718**	**23.05%**
Instancias incorrectamente clasificadas	**9077**	**76.95%**
Número total de instancias	**11795**	**100%**

a	b	c	d	e	f	g	h	<--	clasificado como
313	115	340	55	99	0	130	83	a	HTTP
0	0	0	0	0	0	0	0	b	BitTorrent
0	0	0	0	0	0	0	0	c	eDonkey
0	0	0	0	0	0	0	0	d	Skype
1143	19	407	3	2401	27	114	91	e	SSL
0	0	0	0	0	0	0	0	f	SSH
0	0	0	0	0	0	0	0	g	SMTP
0	5	5918	528	0	0	0	4	h	DNS

Tabla 5.5-2 Matriz de confusión. Traza SSL, HTTP y DNS

SSL sí tiene un porcentaje de acierto bastante alto, sin embargo, los resultados para HTTP y DNS son bastante malos en este caso. Uno de los motivos es que el clasificador usado aprendió de modelos incompletos que no tienen todos los tipos de flujos posibles etiquetados. Por ejemplo, estas muestras encuentran los flujos de datos de las aplicaciones P2P, con mayor ancho de banda y más paquetes.

Muestra: SSH, HTTP y DNS

Instancias correctamente clasificadas	**28**	**23.33%**
Instancias incorrectamente clasificadas	**92**	**76.67%**
Número total de instancias	**120**	**100%**

a	b	c	d	e	f	g	h	<--	clasificado como
1	11	15	6	7	0	0	0	a	HTTP
0	0	0	0	0	0	0	0	b	BitTorrent
0	0	0	0	0	0	0	0	c	eDonkey
0	0	0	0	0	0	0	0	d	Skype
6	0	0	0	0	0	2	0	e	SSL
0	0	0	0	3	27	0	0	f	SSH
0	0	0	0	0	0	0	0	g	SMTP
0	0	38	4	0	0	0	0	h	DNS

Tabla 5.5-3 Matriz de confusión. Traza SSH.

En este último caso, a pesar de la poca cantidad de flujos existentes, el clasificador consigue etiquetar correctamente los flujos SSH, los que más datos llevan en la captura, pero en el resto obtiene resultados malos al no etiquetar correctamente casi ningún flujo.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

El presente trabajo es capaz de responder a la pregunta inicial con la cual se planteó el trabajo, la cual se cuestiona si el número de flujos concurrentes en una red de comunicaciones está condicionado a las aplicaciones que por ella corren. A partir de los resultados vistos, se puede afirmar que casi todas las aplicaciones presentan resultados muy diferentes en el cálculo del ratio de flujos entre ancho de banda.

A partir del conjunto de trazas estudiado, y con la herramienta DPI usada, ha sido posible etiquetar gran parte de los flujos de dicho conjunto, con el cual se han podido realizar métricas del ratio flujos-ancho de banda. Para explicar los resultados en cada aplicación se usó en primer lugar series temporales en la cual se veía la evolución de dicha métrica durante un día, ante la dificultad de extraer resultados precisos a partir de este tipo de cálculos se realizaron estimaciones de probabilidad mediante el método KDE de dicho ratio en las aplicaciones más populares. Se obtuvieron resultados como la figura 4-2 en la cual se pueden explicar los resultados más fácilmente aplicación a aplicación. Se observa que aplicaciones como HTTP o SSH tendrá un ratio de flujos bastante previsible, entre 15 flujos/Mb y 300 flujos/Mb respectivamente, ya que tienen muy poca varianza. SSL, también tendrá un ratio de media en torno a 50 flujos/Mb, aunque con mayor varianza. BitTorrent o eDonkey serán aplicaciones que tienen de media unos 150 flujos/Mb, pero con mucha mayor varianza. Mientras que habrá aplicaciones que seguirá siendo un reto para los gestores de red cuantificar, pues presentan una varianza muy grande en los valores de este ratio, como son Skype o SMTP.
Otra conclusión importante, que se puede extraer de las gráficas realizadas es la homogeneidad que presentan las aplicaciones a lo largo del día, factor que puede ayudar a los gestores de red. En el anexo C se puede observar la estimación de flujos y como en diferentes tramos del día los resultados son similares.
En conclusión, se puede cuantificar de manera aproximada el número de flujos concurrentes en una red sabiendo que aplicaciones se están usando el ancho de banda que ocupa cada una de ellas. Dicha cuantificación será más aproximada mientras más fiable sea clasificador de tráfico usado, es decir, el grado de confianza en que los flujos sean etiquetados correctamente.

Como se ha explicado, la dificultad de encontrar trazas de red que sean aptas para ser clasificadas por DPI hizo buscar alternativas para clasificar el tráfico, como el *machine learning*, se estudiaron otras estadísticas de los flujos, como puede ser el tamaño medio del paquete o el tiempo entre llegadas de los paquetes de un flujo. En estas otras estadísticas obtenidas, se observan pequeñas diferencias entre aplicaciones, siendo el *Throughput* en la que más diferencias se pueden encontrar. Las estadísticas que más condicionadas al tipo de aplicación han sido la duración y el número de paquetes del flujo, el tamaño medio de paquete, la media del tiempo entre llegadas, el *Throughput* y el cociente entre tráfico saliente y entrante. Dichas estadísticas han sido seleccionadas para construir un clasificador de tráfico con *machine learning.*

En Weka, el programa de aprendizaje automático, se han obtenido buenos resultados al probar las trazas de 2010 como muestra de test frente a las muestras de 2008 como muestra de entrenamiento. Estos resultados indican que son los clasificadores basados en arboles de decisión los más efectivos. Cruzando el modelo de entrenamiento con trazas propias el rendimiento del clasificador ha bajado notablemente, lo que es debido, principalmente, a que la red donde se realizo es muy diferente Por lo que se concluye que el desarrollo de modelos de predicción basados en *machine learning* deben aplicarse a las redes allá donde se hayan entrenado.

6.2 Trabajo futuro

Los resultados obtenidos animan a continuar en la mejora sobre el conocimiento en la relación entre los flujos de red y el ancho de banda. Existen pocos estudios acerca de este ratio y muchos menos si tratamos a las aplicaciones de manera independiente. Este trabajo sería un buen punto de partida para continuar investigando en diferentes aplicaciones el ratio flujos-ancho de banda. Se podría observar cómo funciona esta relación en otras redes de comunicaciones con arquitecturas diferentes a las de WIDE.

Para futuras investigaciones relacionadas con el presente trabajo sería recomendable mejorar la calidad de clasificador DPI usado, ya que a la vista de los resultados se observa que un porcentaje significativo de flujos no son capaces de ser etiquetados en alguna aplicación. Una forma de mejorar la aplicación sería actualizando en algunos casos las firmas usadas para detectar ciertas aplicaciones, las cuales en algunos casos han cambiado estos últimos años. Además, sería útil contar con trazas de red más nuevas que permitiesen conocer tendencias de red más actuales. Los mismos estudios realizados en dicho trabajo podrían ser aplicables a estas nuevas trazas, que deberán contar con la carga útil a nivel de aplicación como se ha comentado a lo largo del trabajo.

En cuanto a la parte aprendizaje automático, existe una cantidad ingente de variables y opciones a la hora de construir clasificadores, por lo que sería una buena línea de investigación ahondar en aquellos algoritmos que son realmente útiles a la hora de construir buenos clasificadores y mejorarlos optimizando los valores ajustables que pueda tener cada algoritmo en cada caso. También se podría estudiar más a fondo que estadísticas son más útiles a la hora de diferenciar entre aplicaciones.

Referencias

[1] Daniela Brauckhoff, Bernhard Tellenbach, Arno Wagner, Martin May, Anukool Lakhina. Impact of Packet Sampling on Anomaly Detection Metrics. ACM SIGCOMM Conference on Internet Measurement. 2006.

[2] Víctor Martín Hernández. Análisis de las concurrencias de flujos en Internet TGF Informatica. 2015

[3] Soule, Salamatian, Taft, Emilion, Papagianaki. Flow Classification by Histograms. 2004

[4] Michael Finsterbusch, Chris Richter, Eduardo Rocha, Jean-Alexander Müller, and Klaus Hänßgen. A Survey of Payload-Based Traffic Classification Approaches. 2014

[5] Ipoque WP Deep Packet Inspection 2009 DPI

[6] J. L. García-Dorado, A. Finamore, M. Mellia, M. Meo, M. Munafò, Characterization of ISP Traffic:Trends, User Habits, and Access Technology Impact

[7] Jesús Díaz-Verdejo, Jawa Khalife and Amjad Hajjar. Performance of OpenDPI in Identifying Sampled Network Traffic. 2013

[8] Myung-Sup Kim, Young J. Won, James W. Hong. Characteristic analysis of Internet traffic from the perspective of flows. Computer Communications. 2006.

[9] Tomasz Bujlow, Valentin Carela, and Pere Barlet-Ros. Extended Independent Comparison of Popular Deep Packet Inspection (DPI) Tools for Traffic Classification. 2014

[10] Alberto Dainotti and Antonio Pescapé, University of Napoli Federico II. Kimberly C. Claffy, University of California San Diego. Issues and Future Directions in Traffic Classification. 2012

[11] Página oficial de Matlab. Disponible en https://es.mathworks.com/

[12] Ricardo Aler. Tutorial Weka. 2009

[13] http://www.wide.ad.jp

[14] https://apan.net/

[15] A Survey on Internet Traffic Identification Published in Journal IEEE Communications Surveys & Tutorials Volume 11 Issue 3, July 2009 Page 37-52.

[16] CISCONETFLOW, CISCO. Netflow services and applications (white paper) (https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html)

[17] https://www.xataka.com/legislacion-y-derechos/golpe-a-la-neutralidad-de-la-red-en-eeuu-que-cambia-con-la-nueva-ley-y-como-nos-afecta-a-nosotros-desde-fuera

[18] https://www.ipfabrics.com/

[19] l7.filter.sourceforge.net

[20] Richard O. Duda, Peter E. Hart, David G. Stork, "Pattern Classification", 2nd ed. John Wiley and Sons, 2001.

[21] Christopher M. Bishop, “Pattern Recognition and Machine Learning”, Springer, 2007.

[22] Ricardo Gutierrez Osuna, “Pattern Analysis”, Texas A&M University (http://research.cs.tamu.edu/prism/lectures/pr/pr_l7.pdf)

[23] Jesus Bescós, “Procesado Avanzado de señales multimedia”, EPS-UAM

[24] Agustín José Calleja Gomez “Minería de datos con WEKA para predicción del precio de automóviles de segunda mano” Proyecto Fin de Carrera. UPV. 2010

[25] S.E. Martínez Galdámez, C.F. Ramírez Echevarría, G. H.R. Rodríguez Hernández, W.O.Salazar González. “Implementación del algoritmo C4.5 de aprendizaje automático para la generación de árboles de decisión”. 2013
[26] http://weka.sourceforge.net/doc.dev/weka/classifiers/

Glosario

ACK	Acknowledgement (Consentimiento)
ADSL	Asymmetric Digital Subscriber Line (Linea de abonado digital simétrica)
APAN	Asia-Pacific Area Network (Área de redes Asia-Pacífico)
API	Application Programming Interface
DNS	Domain Name System (Sistema de Nombres de Dominio)
DPI	Deep Packet Inspection (Inspección profunda de paquete)
FTP	File Transport Protocol (Protocolo de Transporte de Archivos)
FTTH	Fiber To The Home (Fiber to the Home)
FW	Firewall
HTTP	Hipertext-Transfer-Protocol (Protocolo de transferencia de hipertexto)
IANA	Internet Assigned Numbers Authority (Autoridad de Números Asignados en Internet)
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force (Grupo de Trabajo de Ingeniería de Internet)
IP	Internet Protocol
IPS	Intrusion Prevention System
ISOC	Internet Society
ISP	Internet Service Provider (Proveedor de Servicios de Internet)
KB	Kilobyte
KDE	Kernel Density Estimation (Estimación de la densidad del Kernel)
MAWI	Measurement and Analysis on the Wide-Area Internet
MB	Megabyte
MCC	Matthews correlation coefficient (coeficiente de correlación de Matthews)
MSN	MicroSoft Network
NBNS	NetBios Name Service
NTP	Network Time Protocol (Protocolo de Tiempo para Redes)
P2P	Peer-to-Peer
POP3	Post Office Protocol
PRC	Precisión Recall-Curves (Curvas de precisión)
ROC	Receiver Operating Characteristic (Característica Operativa del Receptor)
SMB	Server Message Block (Bloqueo
SMTP	Simple Mail Transfer Protocol (protocolo para transferencia simple de correo)
SSH	Secure SHell
SSL	Secure Socket-Layer
SVM	Support Vector Machine (Máquinas de Vector Soporte)
TCP	Transmission Control Protocol (Protocolo de control de transmisión)
TLS	Transport Layer Security (seguridad de la capa de transporte)
UAM	Universidad Autónoma de Madrid
UDP	User Datagram Protocol (Protocolo de datagramas de usuario)
WIDE	Widely Integrated Distributed Environment

Anexos

A Resto de estadísticas de flujos por aplicación

Madrugada (00:00-06:00 12/04/2008)								
	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	24689783	100,000%	11336492	45,916%	100,000%	13353291	54,084%	100,000%
No etiquetados	5033867	20,388%	4009109	16,238%	35,365%	1024758	4,151%	7,674%
DNS	12609191	51,070%	5752045	23,297%	50,739%	6857146	27,773%	51,352%
HTTP	2187005	8,858%	503291	2,038%	4,440%	1683714	6,819%	12,609%
BITTORRENT	802358	3,250%	308256	1,249%	2,719%	494102	2,001%	3,700%
EDONKEY	62343	0,253%	28703	0,116%	0,253%	33640	0,136%	0,252%
MSN	174	0,001%	152	0,001%	0,001%	22	0,000%	0,000%
NETBIOS	96010	0,389%	93410	0,378%	0,824%	2600	0,011%	0,019%
NTP	17777	0,072%	10067	0,041%	0,089%	7710	0,031%	0,058%
POP3	3992	0,016%	1042	0,004%	0,009%	2950	0,012%	0,022%
SKYPE	1065513	4,316%	392347	1,589%	3,461%	673166	2,726%	5,041%
SMB	3411	0,014%	22	0,000%	0,000%	3389	0,014%	0,025%
SMTP	1314097	5,322%	202959	0,822%	1,790%	1111138	4,500%	8,321%
SSH	1278218	5,177%	235	0,001%	0,002%	1277983	5,176%	9,571%
SSL	41122	0,167%	12	0,000%	0,000%	41110	0,167%	0,308%
OTROS	174705	0,708%	34842	0,141%	0,307%	139863	0,566%	0,930%

Tabla A-1. Estadísticas de flujos. Horario Madrugada. Año 2008

Tarde (12:00-18:00 12/04/2008)								
	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	23804160	100,000%	10641789	44,706%	100,000%	13162371	55,294%	100,000%
No etiquetados	6258831	26,293%	4471240	18,783%	42,016%	1787591	7,510%	13,581%
DNS	9793613	41,142%	4312182	18,115%	40,521%	5481431	23,027%	41,645%
HTTP	2997983	12,594%	531230	2,232%	4,992%	2466753	10,363%	18,741%
BITTORRENT	852352	3,581%	330637	1,389%	3,107%	521715	2,192%	3,964%
EDONKEY	91646	0,385%	45715	0,192%	0,430%	45931	0,193%	0,349%
MSN	607	0,003%	382	0,002%	0,004%	225	0,001%	0,002%
NETBIOS	111601	0,469%	107575	0,452%	1,011%	4026	0,017%	0,031%
NTP	90820	0,382%	9857	0,041%	0,093%	80963	0,340%	0,615%
POP3	9948	0,042%	1054	0,004%	0,010%	8894	0,037%	0,068%
SKYPE	1709069	7,180%	638481	2,682%	6,000%	1070588	4,497%	8,134%
SMB	9025	0,038%	865	0,004%	0,008%	8160	0,034%	0,062%
SMTP	852109	3,580%	138845	0,583%	1,305%	713264	2,996%	5,419%
SSH	859648	3,611%	137	0,001%	0,001%	859511	3,611%	6,530%
SSL	103289	0,434%	2	0,000%	0,000%	103287	0,434%	0,785%
OTROS	28685	0,121%	53587	0,225%	0,504%	10032	0,042%	0,076%

Tabla A-2. Estadísticas de flujos. Horario Tarde. Año 2008

Madrugada (00:00-06:00 15/04/2010)								
	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	25470078	100,000%	10429799	40,949%	100,000%	15040279	59,051%	100,000%
No etiquetados	7118932	27,950%	2355488	9,248%	22,584%	4763444	18,702%	31,671%
DNS	10532483	41,352%	5484602	21,534%	52,586%	5047881	19,819%	33,562%
HTTP	3082300	12,102%	281651	1,106%	2,700%	2800649	10,996%	18,621%
BITTORRENT	1660213	6,518%	1393235	5,470%	13,358%	266978	1,048%	1,775%
EDONKEY	146412	0,575%	62678	0,246%	0,601%	83734	0,329%	0,557%
MSN	40195	0,158%	38015	0,149%	0,364%	2180	0,009%	0,014%
NETBIOS	105993	0,416%	14646	0,058%	0,140%	91347	0,359%	0,607%
NTP	45317	0,178%	31406	0,123%	0,301%	13911	0,055%	0,092%
POP3	66354	0,261%	39512	0,155%	0,379%	26842	0,105%	0,178%
SKYPE	841574	3,304%	549618	2,158%	5,270%	291956	1,146%	1,941%
SMB	122875	0,482%	18	0,000%	0,000%	122857	0,482%	0,817%
SMTP	581580	2,283%	152042	0,597%	1,458%	429538	1,686%	2,856%
SSH	935667	3,674%	2044	0,008%	0,020%	933623	3,666%	6,207%
SSL	159871	0,628%	45	0,000%	0,000%	159826	0,628%	1,063%
OTROS	30312	0,119%	24799	0,097%	0,238%	5513	0,022%	0,037%

Tabla A-3. Estadísticas de flujos. Horario Madrugada. Año 2010

Tarde (12:00-18:00 15/04/2010)								
	flujos totales	%	flujos 1 paquete	TOTAL%	1 PKT %	flujos > 1 paquete	TOTAL%	> 1 PKT %
TOTAL	30511908	100,000%	14210191	46,573%	100,000%	16301717	53,427%	100,000%
No etiquetados	9742686	31,931%	4887329	16,018%	34,393%	4855357	15,913%	29,784%
DNS	11377798	37,290%	6655333	21,812%	46,835%	4722465	15,477%	28,969%
HTTP	4194898	13,748%	435780	1,428%	3,067%	3759118	12,320%	23,060%
BITTORRENT	1762530	5,777%	1191849	3,906%	8,387%	570681	1,870%	3,501%
EDONKEY	175784	0,576%	95479	0,313%	0,672%	80305	0,263%	0,493%
MSN	189976	0,623%	169169	0,554%	1,190%	20807	0,068%	0,128%
NETBIOS	68263	0,224%	25688	0,084%	0,181%	42575	0,140%	0,261%
NTP	90820	0,298%	66943	0,219%	0,471%	23877	0,078%	0,146%
POP3	12782	0,042%	93	0,000%	0,001%	12689	0,042%	0,078%
SKYPE	867567	2,843%	440812	1,445%	3,102%	426755	1,399%	2,618%
SMB	128365	0,421%	83	0,000%	0,001%	128282	0,420%	0,787%
SMTP	500587	1,641%	127246	0,417%	0,895%	373341	1,224%	2,290%
SSH	756826	2,480%	1659	0,005%	0,012%	755167	2,475%	4,632%
SSL	484604	1,588%	201	0,001%	0,001%	484403	1,588%	2,971%
OTROS	158422	0,519%	112527	0,369%	0,792%	45895	0,150%	0,282%

Tabla A-4. Estadísticas de flujos. Horario Tarde. Año 2010

B Estadísticas de "doble matching" en la herramienta DPI

MAdrugada (00:00-06:00) ---- 12/04/2008								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,000%	0	0,000%	0	N/A		0%	0
BITTORRENT	2,219%	5116	0,696%	1232	23,886%	2508	N/A	
EDONKEY	78,850%	181770	97,670%	172959	38,229%	4014	78,202%	2949
MSN	0,318%	732	0,151%	268	2,438%	256	3,050%	115
NETBIOS	1,482%	3416	0,030%	53	1,105%	116	3,792%	143
NTP	0,002%	4	0,002%	4	0,000%	0	0,000%	0
POP3	0,000%	1	0,001%	1	0,000%	0	0,000%	0
SKYPE	9,745%	22464	0,012%	22	0,000%	0	4,800%	181
SMB	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SMTP	0,001%	2	0,001%	1	0,000%	0	0,000%	0
SSH	0,078%	179	0,000%	0	0,000%	0	4,747%	179
SSL	0,052%	121	0,042%	75	0,000%	0	0,000%	0
OTROS	7,253%	16720	1,395%	2470	34,343%	3606	5,410%	204
TOTAL	100%	230525	76,818%	177085	4,555%	10500	1,636%	3771

Tabla B-1. Estadísticas de doble match en flujos. Madrugada 2008.

Mañana (06:00-12:00) ---- 12/04/2008								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,000%	1	0,000%	0	N/A		0%	0
BITTORRENT	2,725%	5606	0,295%	979	31,004%	4060	N/A	
EDONKEY	72,255%	148631	97,863%	140255	30,569%	4003	77,159%	2797
MSN	0,302%	621	0,135%	193	1,840%	241	3,200%	116
NETBIOS	2,911%	5989	0,001%	2	0,420%	55	0,552%	20
NTP	0,000%	1	0,001%	1	0,000%	0	0,000%	0
POP3	0,000%	0	0,001%	1	0,000%	0	0,000%	0
SKYPE	12,125%	24942	0,006%	9	0,031%	4	7,503%	272
SMB	0,005%	10	0,000%	0	0,000%	0	0,000%	0
SMTP	0,002%	4	0,001%	2	0,000%	0	0,000%	0
SSH	0,057%	117	0,000%	0	0,000%	0	3,228%	117
SSL	0,089%	184	0,049%	70	0,000%	0	0,028%	1
OTROS	9,527%	19598	1,260%	1806	36,136%	4732	8,331%	302
TOTAL	100%	205704	69,672%	143318	6,366%	13095	1,762%	3625

Tabla B-2. Estadísticas de doble match en flujos. Mañana 2008.

Tarde (12:00-18:00) ---- 12/04/2008								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,003%	8	0,001%	2	N/A		0%	0
BITTORRENT	0,706%	5875	0,640%	920	22,908%	4670	N/A	
EDONKEY	64,913%	152067	98,045%	140897	34,607%	7055	78,449%	2690
MSN	0,323%	757	0,111%	160	2,247%	458	1,604%	55
NETBIOS	1,945%	9140	0,040%	57	0,245%	50	0,554%	19
NTP	0,001%	3	0,002%	3	0,000%	0	0,000%	0
POP3	0,000%	0	0,001%	1	0,000%	0	0,000%	0
SKYPE	7,853%	36901	0,007%	10	0,029%	6	4,929%	169
SMB	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SMTP	0,000%	1	0,001%	1	0,000%	0	0,000%	0
SSH	0,024%	115	0,000%	0	0,000%	0	3,354%	115
SSL	0,055%	259	0,050%	72	0,000%	0	0,029%	1
OTROS	6,200%	29136	1,102%	1584	39,964%	8147	11,082%	380
TOTAL	100%	234262	61,345%	143707	8,702%	20386	1,464%	3429

Tabla B-3. Estadísticas de doble match en flujos. Tarde 2008.

Noche (18:00-24:00) ---- 12/04/2008								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,000%	1	0,000%	0	N/A		0%	0
BITTORRENT	2,252%	4909	0,582%	896	25,426%	3801	N/A	
EDONKEY	73,454%	160145	97,999%	150984	33,554%	5016	79,976%	2712
MSN	0,302%	659	0,129%	198	2,040%	305	2,359%	80
NETBIOS	0,739%	1611	0,025%	39	0,007%	1	0,088%	3
NTP	0,000%	1	0,001%	1	0,000%	0	0,000%	0
POP3	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SKYPE	12,718%	27728	0,010%	15	0,033%	5	4,984%	169
SMB	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SMTP	0,002%	4	0,003%	4	0,000%	0	0,000%	0
SSH	0,048%	104	0,000%	0	0,000%	0	3,067%	104
SSL	0,095%	207	0,042%	65	0,000%	0	0,029%	1
OTROS	10,390%	22653	1,211%	1865	38,939%	5821	9,496%	322
TOTAL	100%	218022	70,666%	154067	6,857%	14949	1,555%	3391

Tabla B-4. Estadísticas de doble match en flujos. Noche 2008.

MAdrugada (00:00-06:00) ---- 15/04/2010								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,000%	0	56,031%	185634	N/A		0%	0
BITTORRENT	0,646%	5379	4,921%	16305	6,429%	22354	N/A	
EDONKEY	25,252%	210265	38,206%	126578	2,457%	8542	56,104%	5032
MSN	0,056%	468	0,020%	65	0,201%	698	0,424%	38
NETBIOS	14,794%	123187	0,026%	87	0,437%	1521	2,286%	205
NTP	0,000%	0	0,000%	0	0,000%	0	0,000%	0
POP3	0,000%	0	0,000%	0	0,000%	0	0,000%	0
SKYPE	2,179%	18144	0,020%	65	7,151%	24863	7,336%	658
SMB	0,002%	18	0,000%	0	3,982%	13845	0,000%	0
SMTP	0,000%	2	0,000%	0	0,000%	0	2,955%	265
SSH	0,000%	0	0,002%	5	60,853%	211587	17,126%	1536
SSL	0,044%	366	0,077%	254	15,775%	54851	13,435%	1205
OTROS	39,643%	330096	0,698%	2313	2,715%	9440	0,334%	30
TOTAL	100%	687925	48,160%	331306	50,543%	347701	1,304%	8969

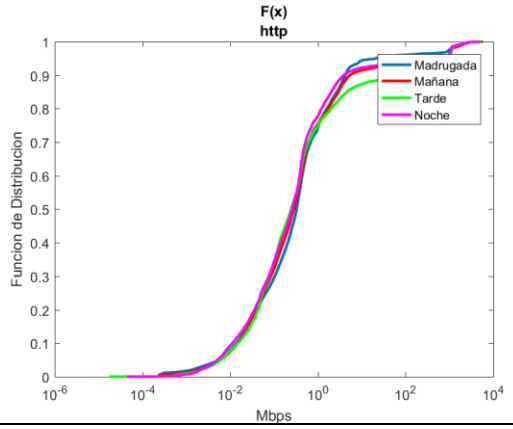
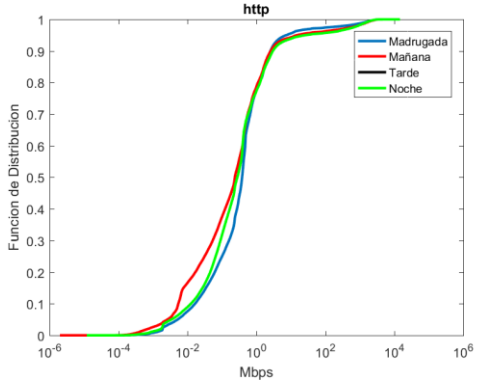
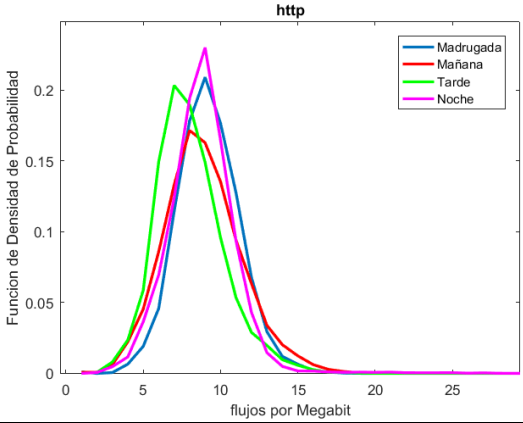
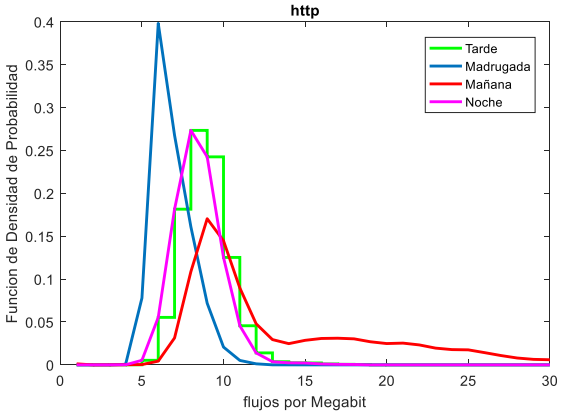
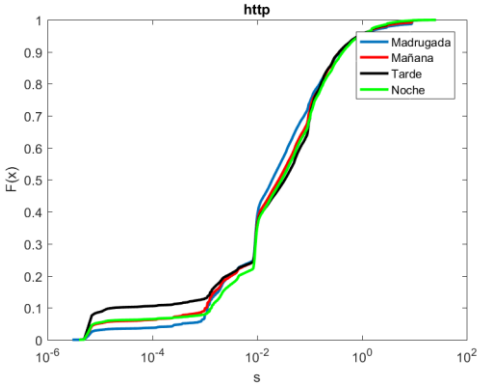
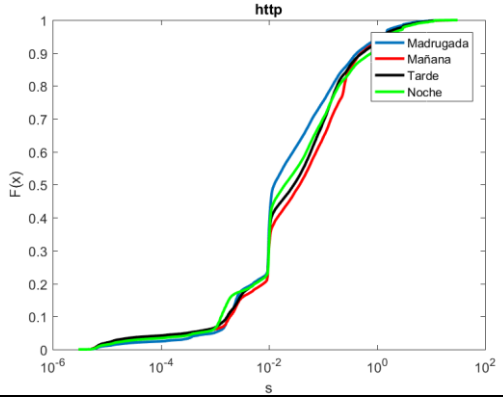
Tabla B-5. Estadísticas de doble match en flujos. Madrugada 2010.

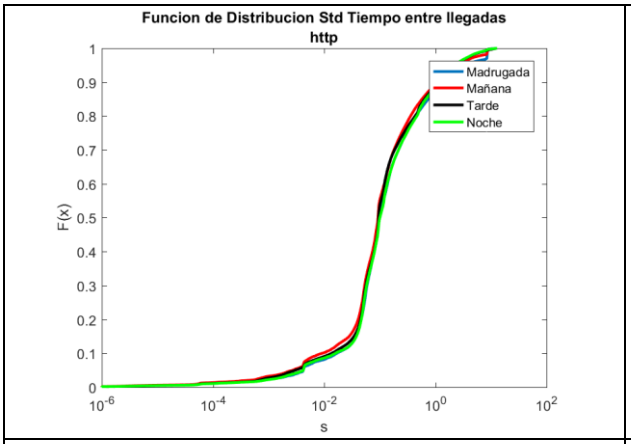
Tarde (12:00-18:00) ---- 15/04/2010								
Nº DE FLUJOS	PRIMER MATCH							
SEGUNDO MATCH	TOTAL		DNS		HTTP		BITTORRENT	
DNS	0,000%	0	N/A		0%	0	0%	0
HTTP	0,768%	3610	1,273%	3163	N/A		0%	0
BITTORRENT	11,736%	55148	11,589%	28796	50,116%	24572	N/A	
EDONKEY	48,609%	228419	0,084%	208	17,732%	8694	47,018%	4217
MSN	0,159%	748	0,010%	26	0,899%	441	2,319%	208
NETBIOS	27,326%	128405	0,000%	0	0,000%	0	0,357%	32
NTP	0,000%	0	0,000%	0	0,000%	0	0,000%	0
POP3	0,000%	1	0,000%	1	0,000%	0	0,000%	0
SKYPE	4,043%	18997	0,009%	22	0,530%	260	14,784%	1326
SMB	0,118%	555	0,000%	0	0,010%	5	5,764%	517
SMTP	0,000%	1	0,000%	1	0,000%	0	0,000%	0
SSH	4,043%	18997	0,000%	1	6,780%	3324	25,945%	2327
SSL	1,010%	4745	0,035%	88	4,108%	2014	25,399%	2278
OTROS	2,188%	10283	86,998%	216169	19,825%	9720	13,647%	1224
TOTAL	100%	469909	52,877%	248475	10,434%	49030	2,581%	12129

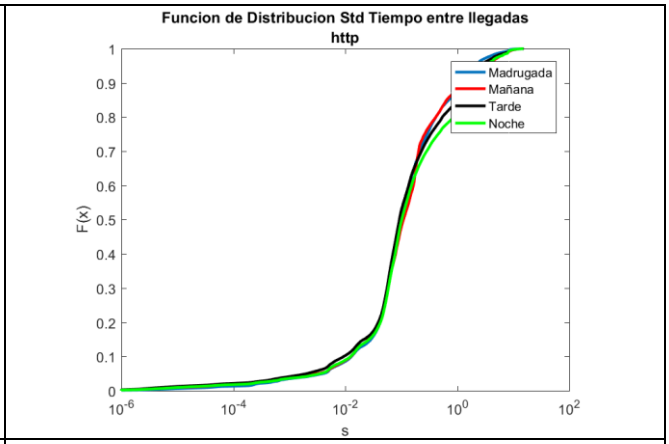
Tabla B-6. Estadísticas de doble match en flujos. Tarde 2010.

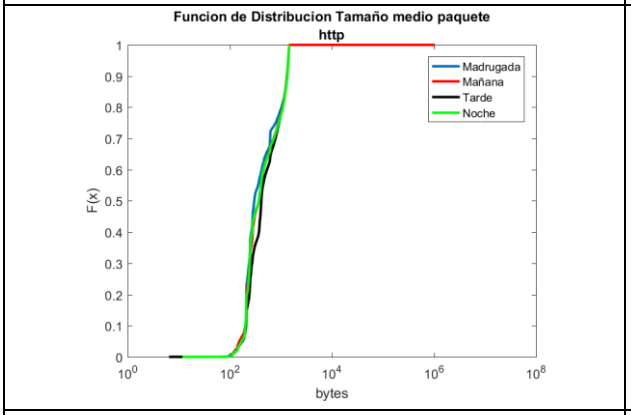
C Resultados análisis por aplicaciones populares

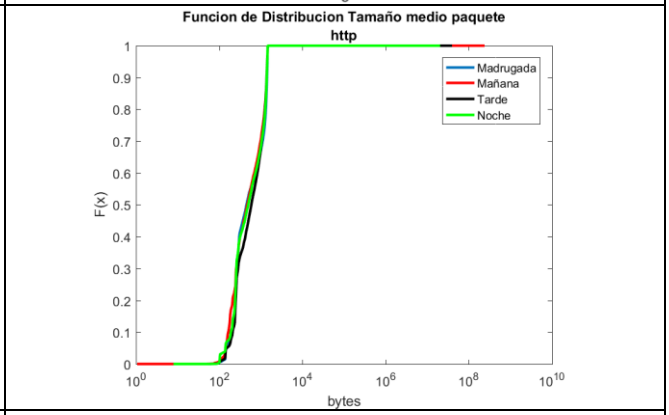
HTTP

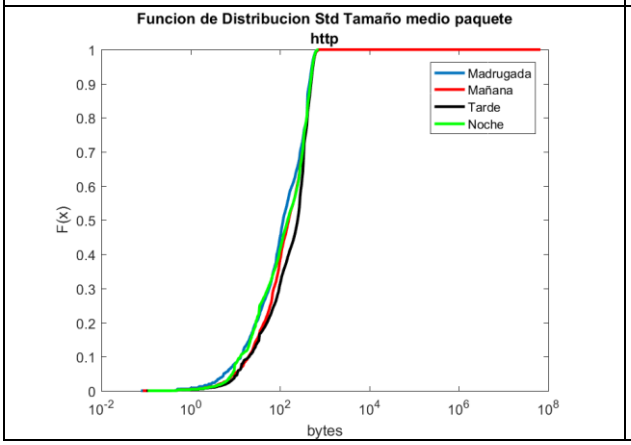
2008	2010
Función de distribución Throughput F(x) http 	**Función de distribución Throughput** F(x) http 
Estimación de flujos por Mb/s f(x) http 	**Estimación de flujos por Mb/s** f(x) http 
Funcion de Distribucion Media entre llegadas http 	**Funcion de Distribucion Media entre llegadas** http 

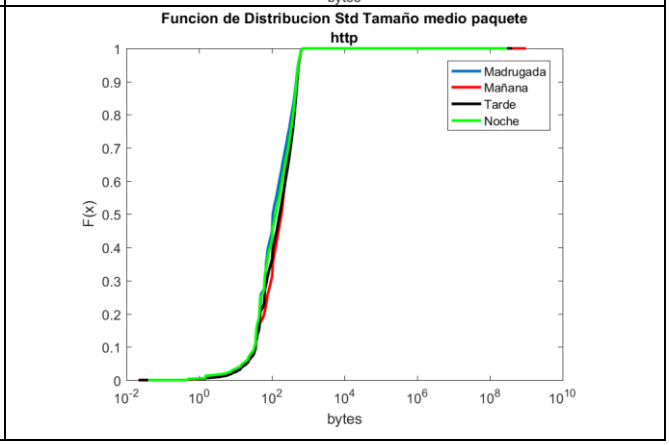

Funcion de Distribucion Std Tiempo entre llegadas
http
Madrugada
Mañana
Tarde
Noche
F(x)
s


Funcion de Distribucion Std Tiempo entre llegadas
http
Madrugada
Mañana
Tarde
Noche
F(x)
s

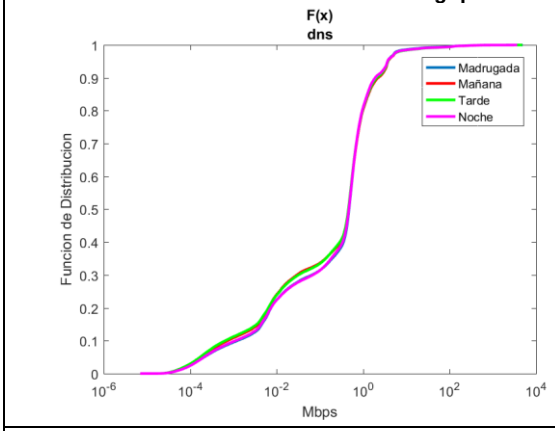
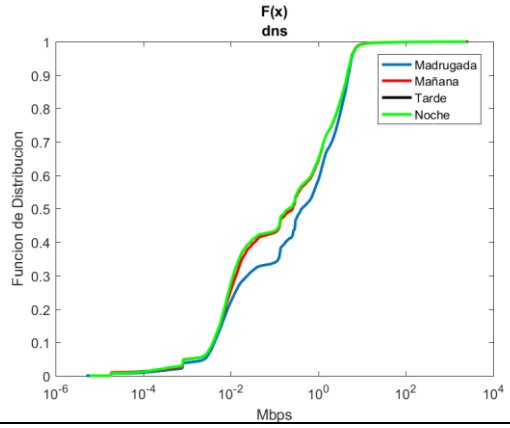
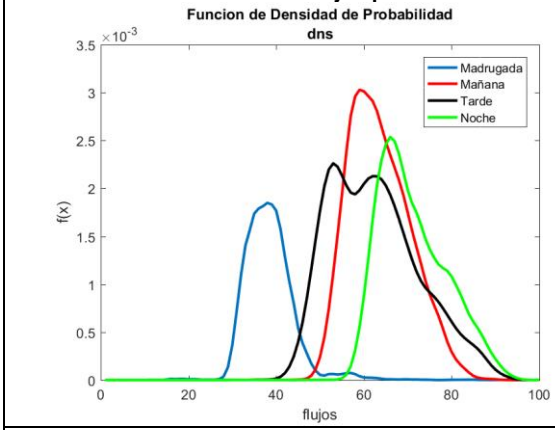
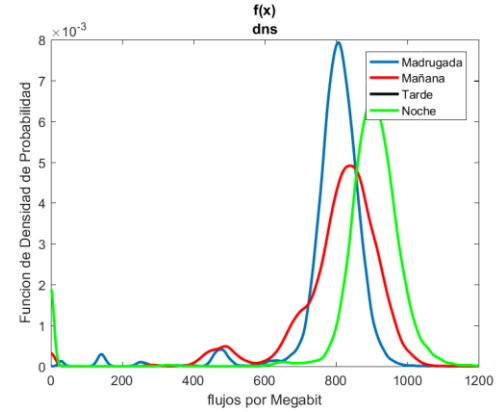
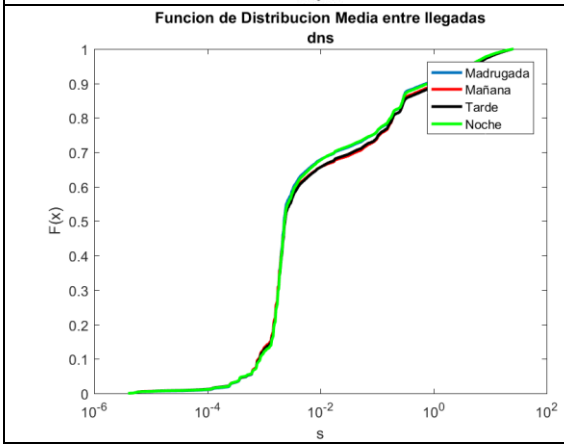
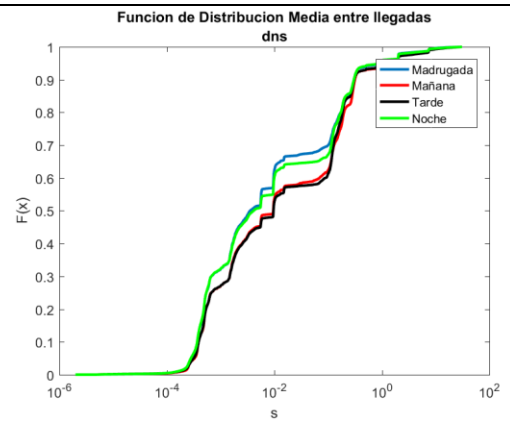

Funcion de Distribucion Tamaño medio paquete
http
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

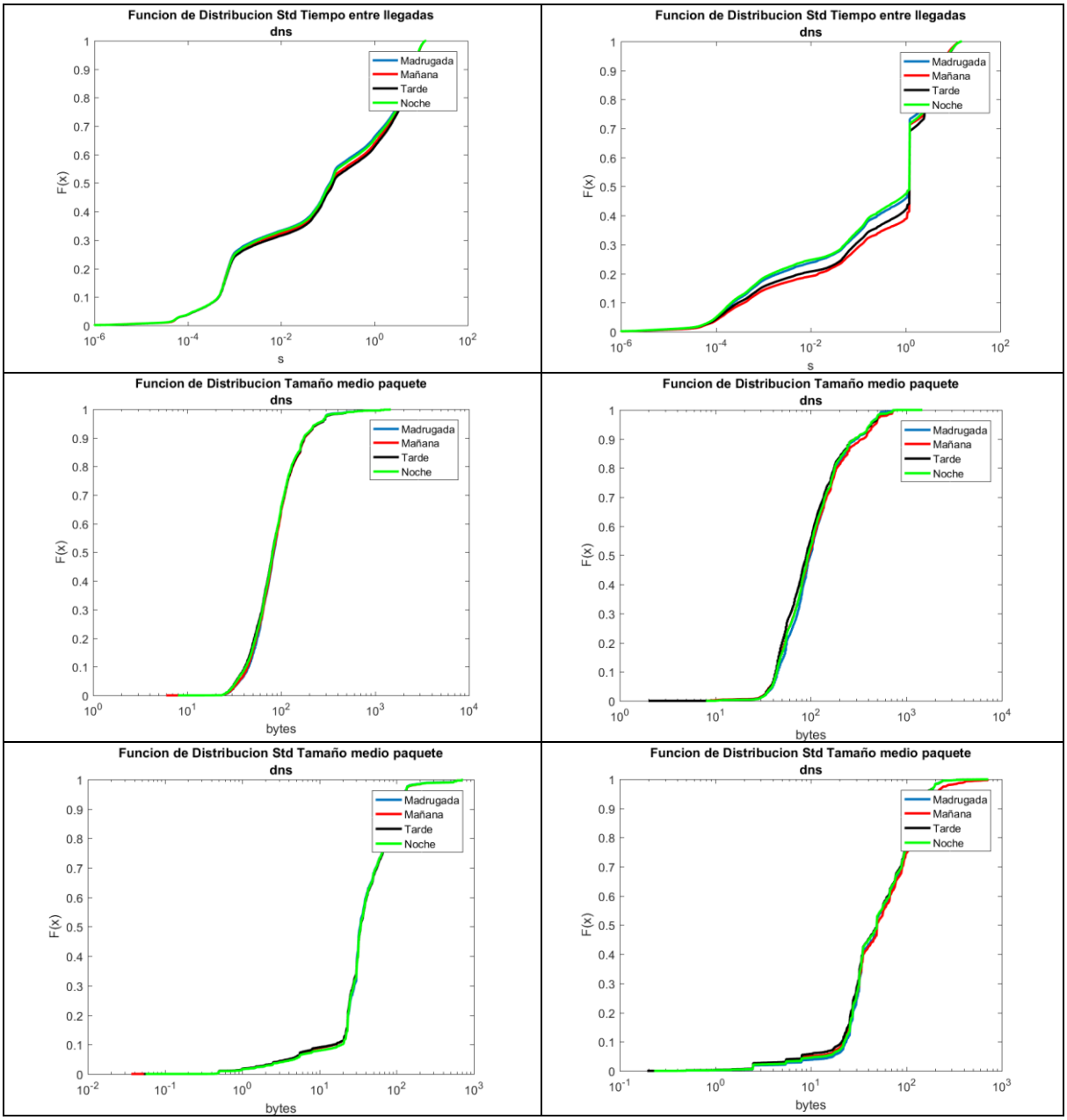

Funcion de Distribucion Tamaño medio paquete
http
Madrugada
Mañana
Tarde
Noche
F(x)
bytes


Funcion de Distribucion Std Tamaño medio paquete
http
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

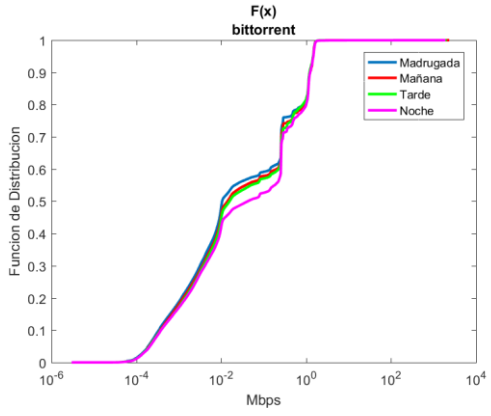
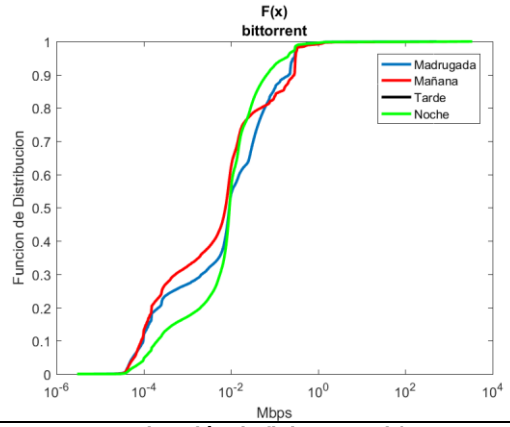
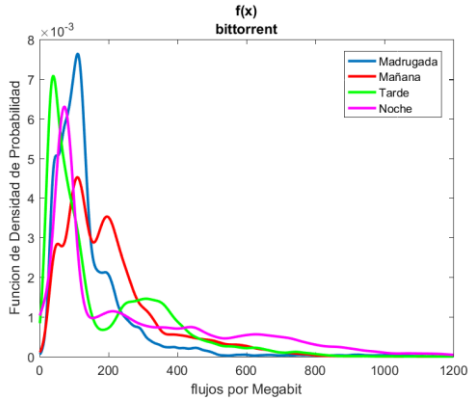
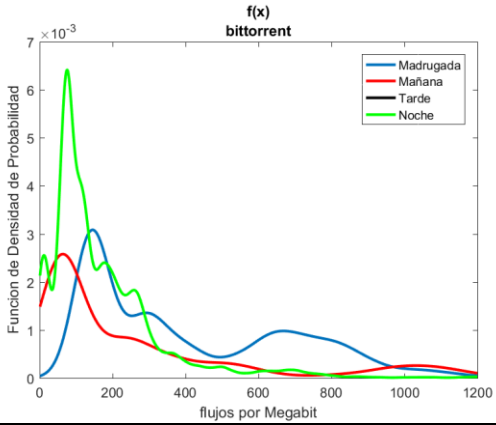
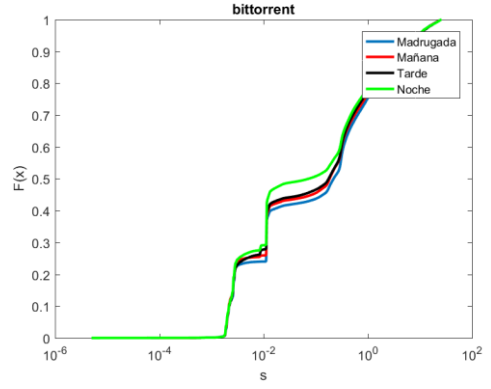
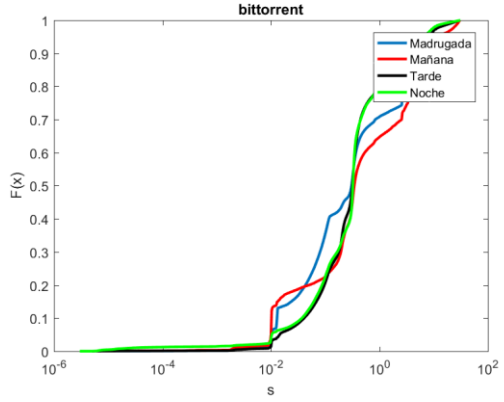

Funcion de Distribucion Std Tamaño medio paquete
http
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

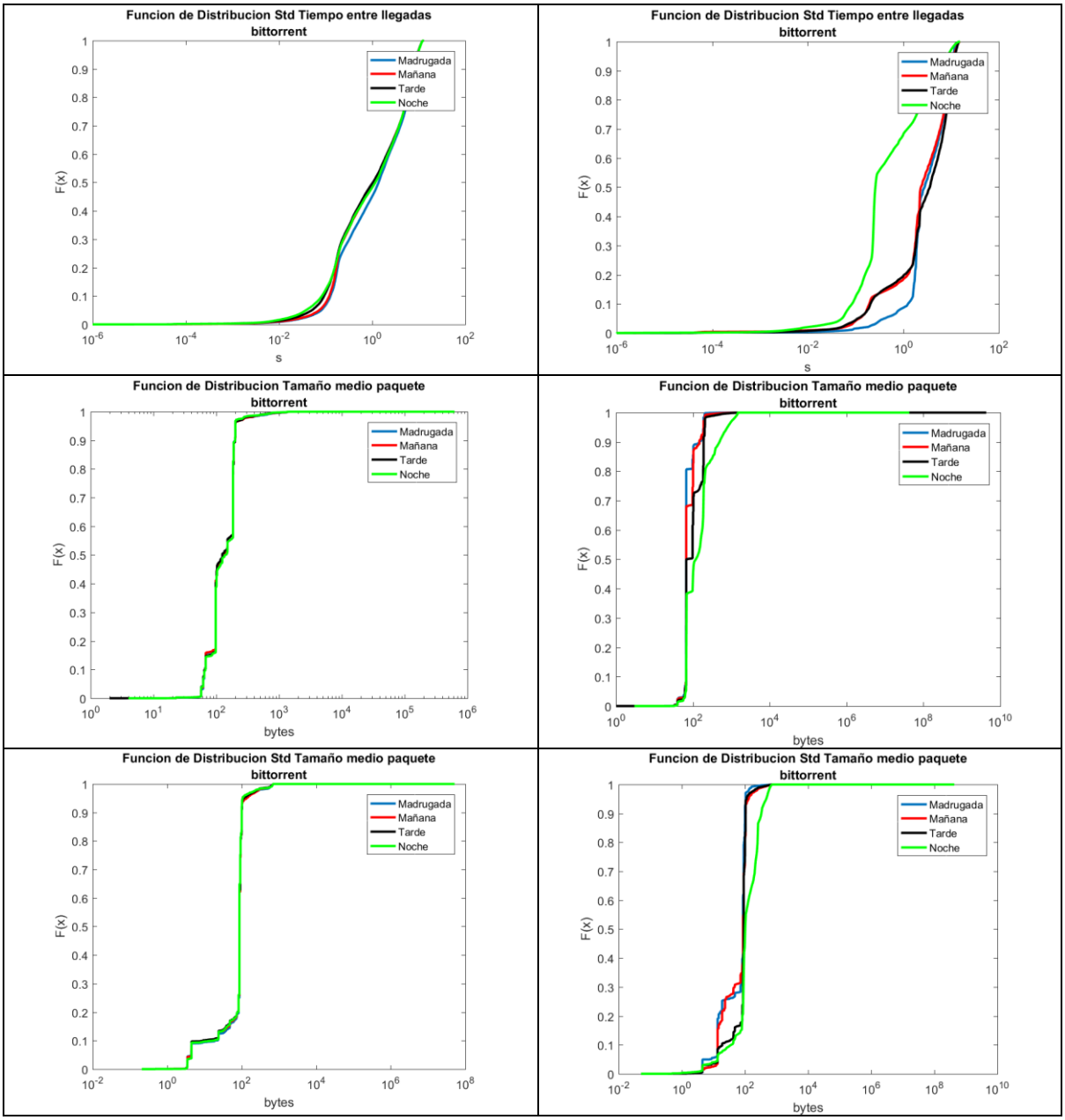
DNS

2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

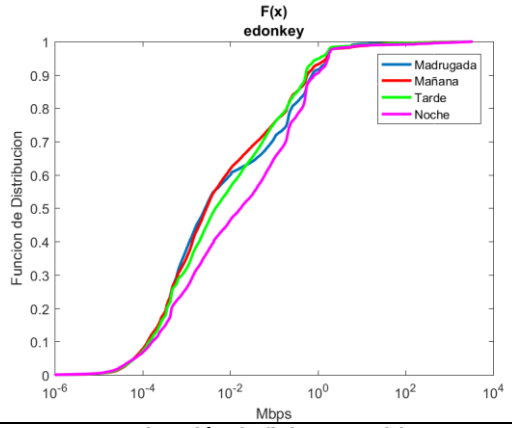
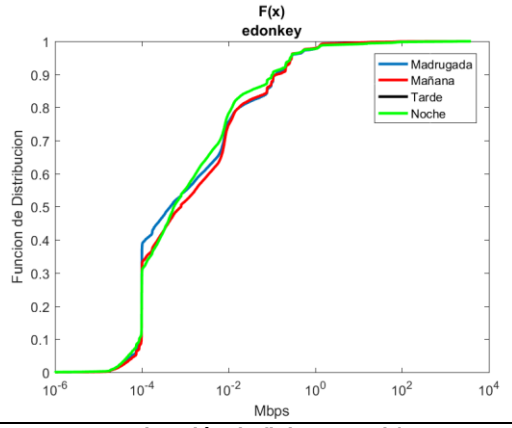
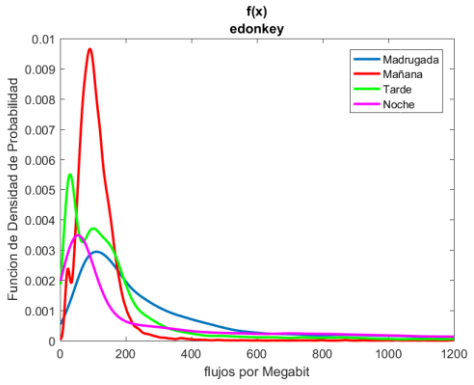
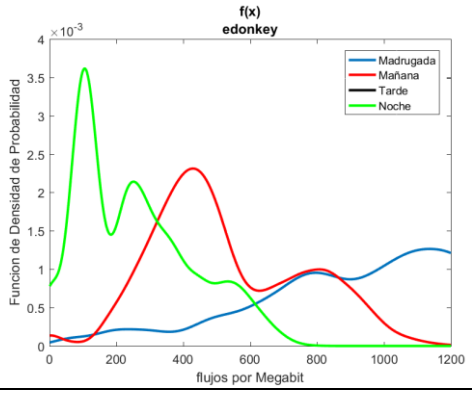
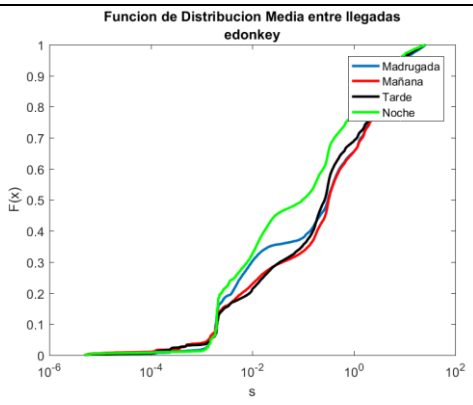
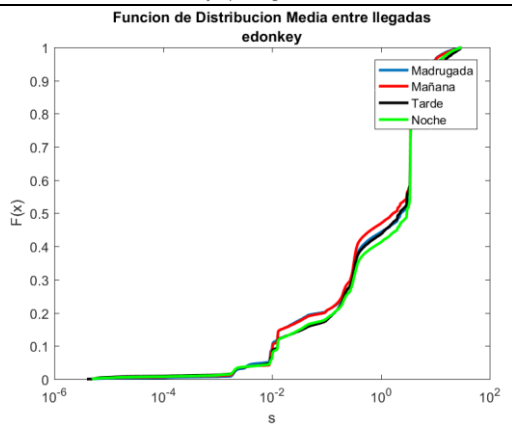

Funcion de Distribucion Std Tiempo entre llegadas
dns
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
dns
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
dns
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
dns
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
dns
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
dns
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

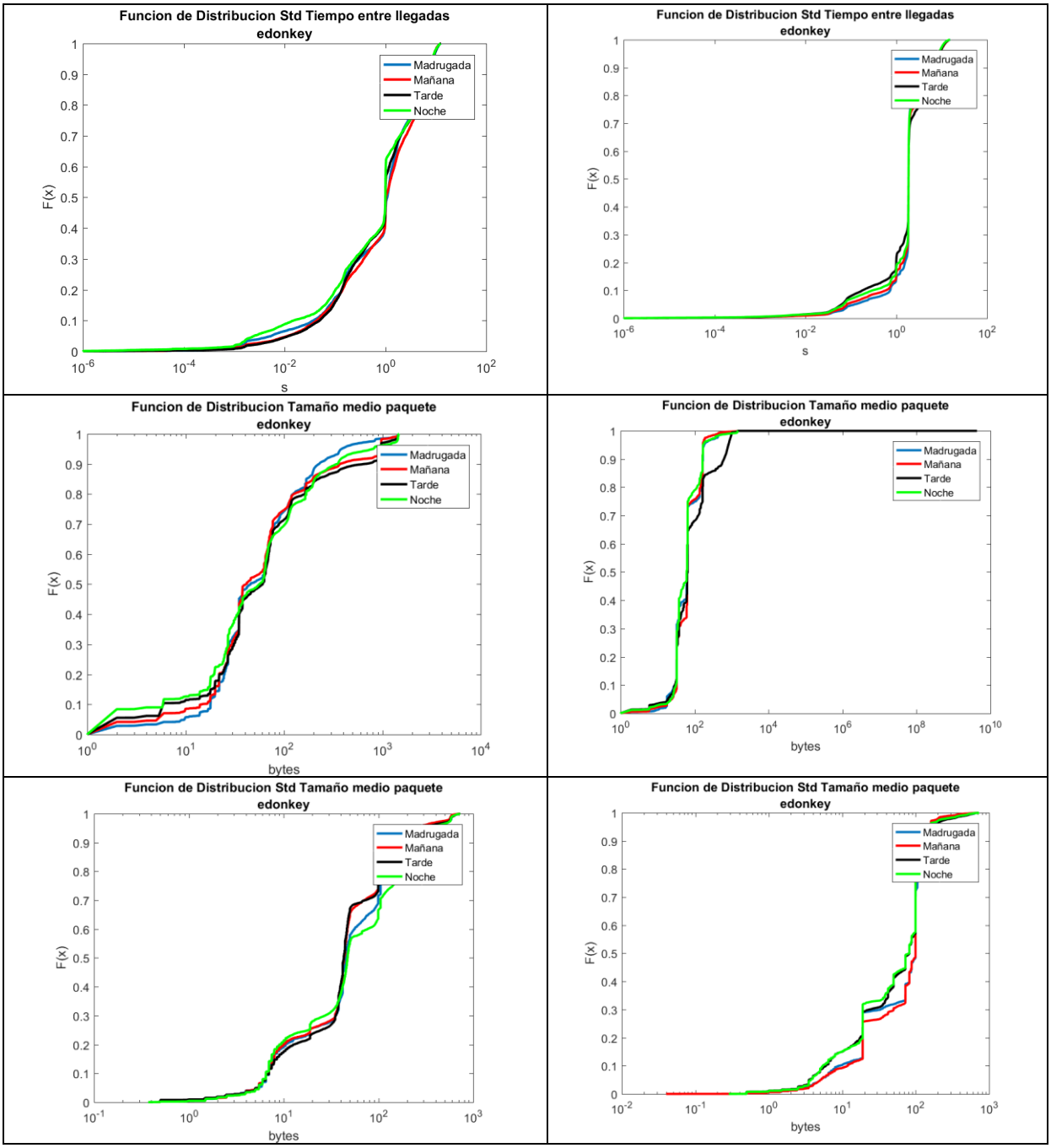
BitTorrent

2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

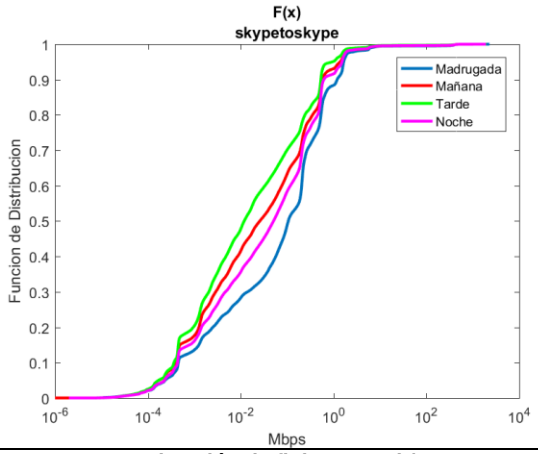
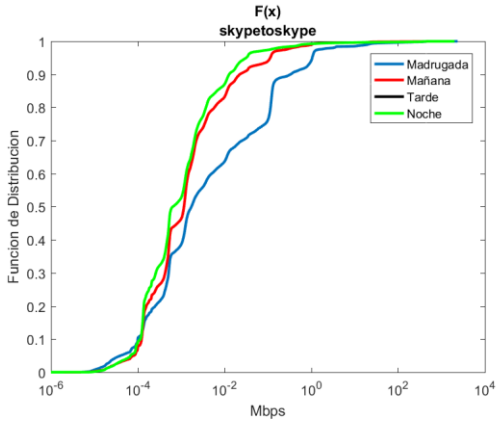
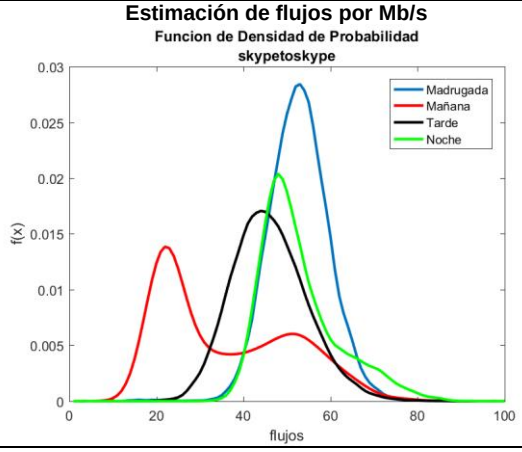
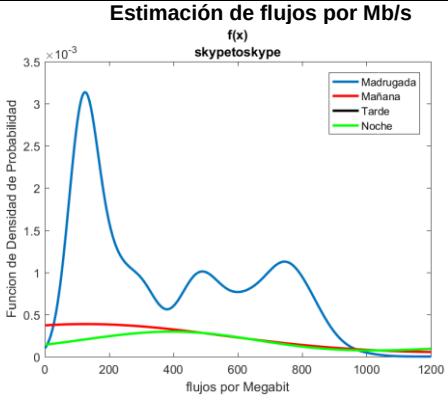
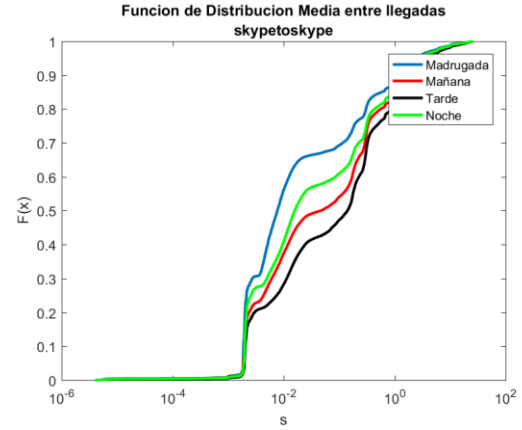
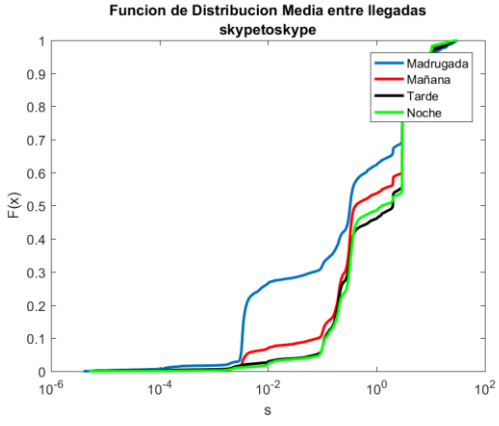

Funcion de Distribucion Std Tiempo entre llegadas
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
bittorrent
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

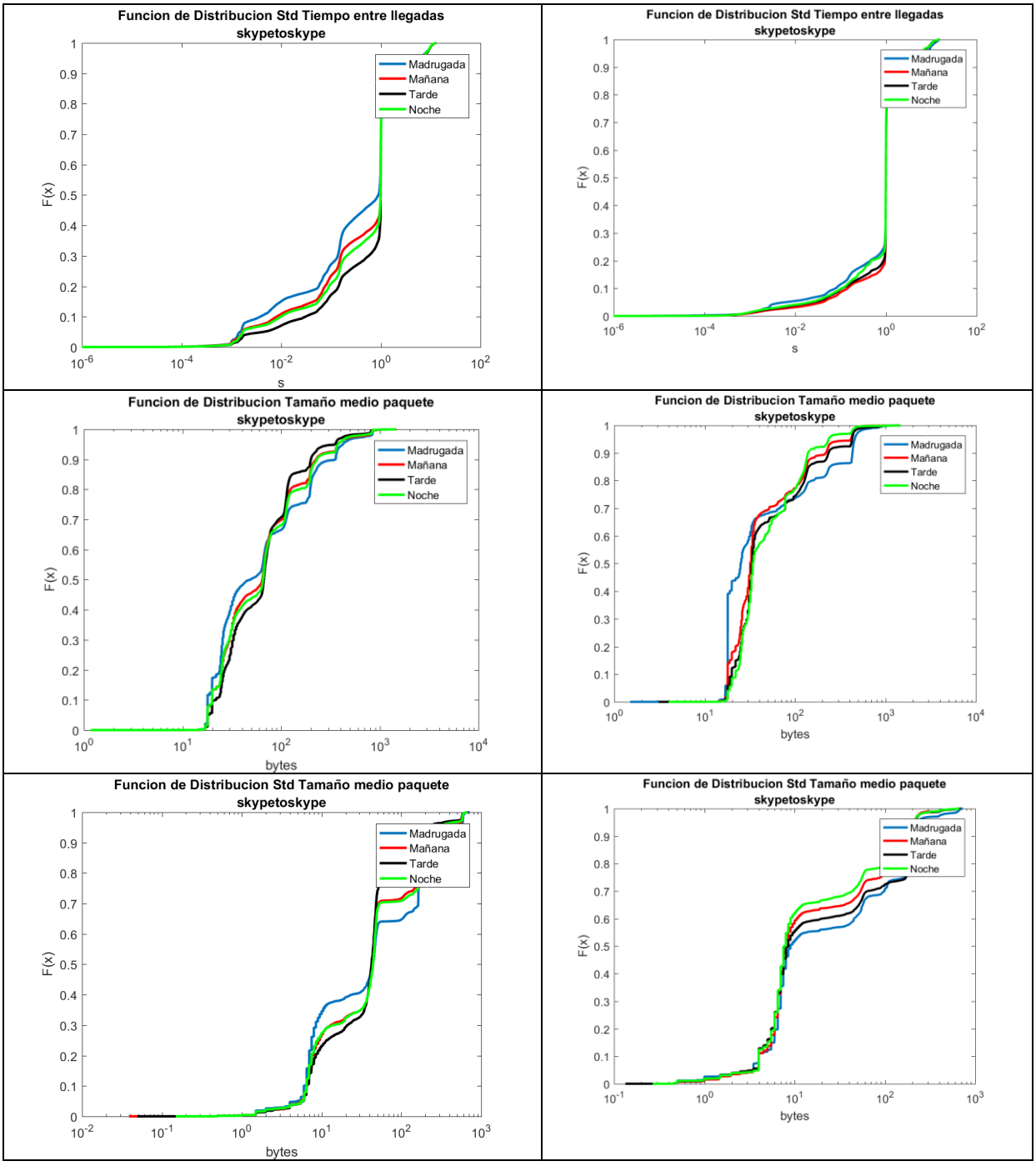
eDonkey

2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

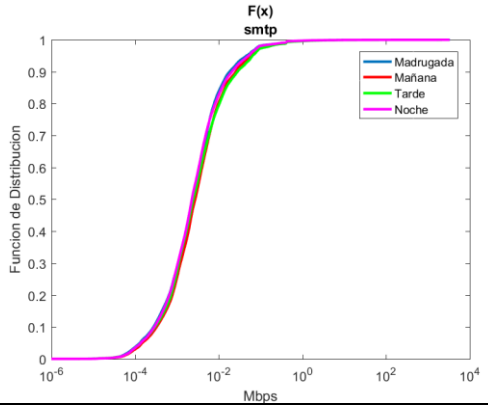
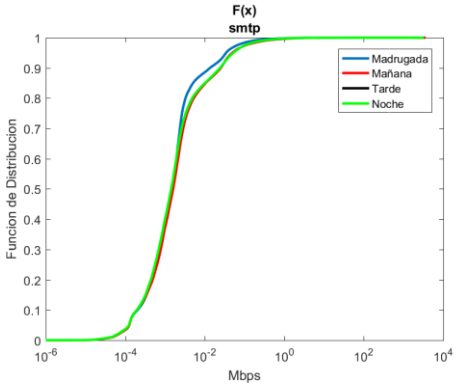
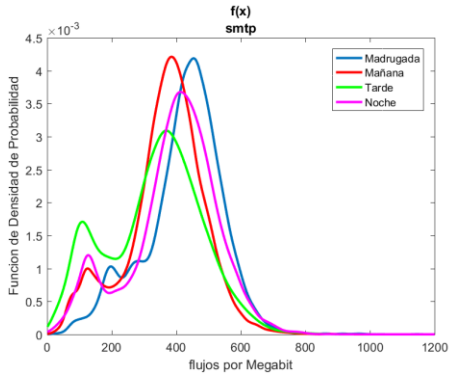
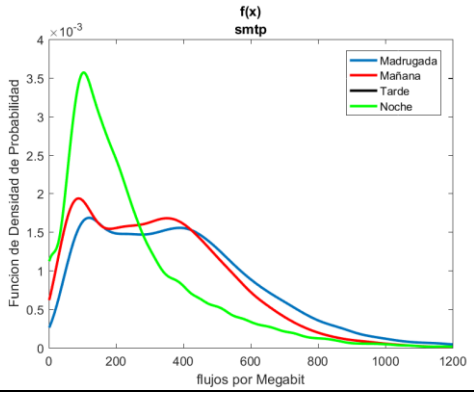
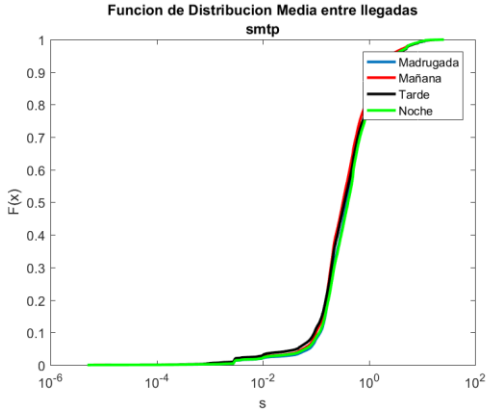
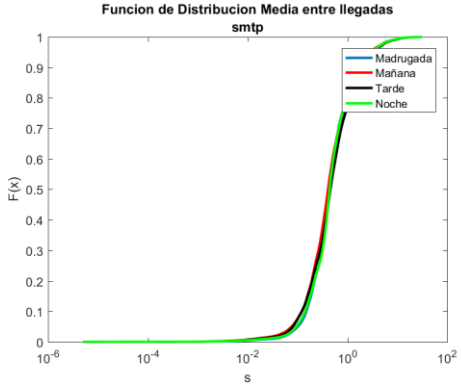

Funcion de Distribucion Std Tiempo entre llegadas
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
edonkey
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

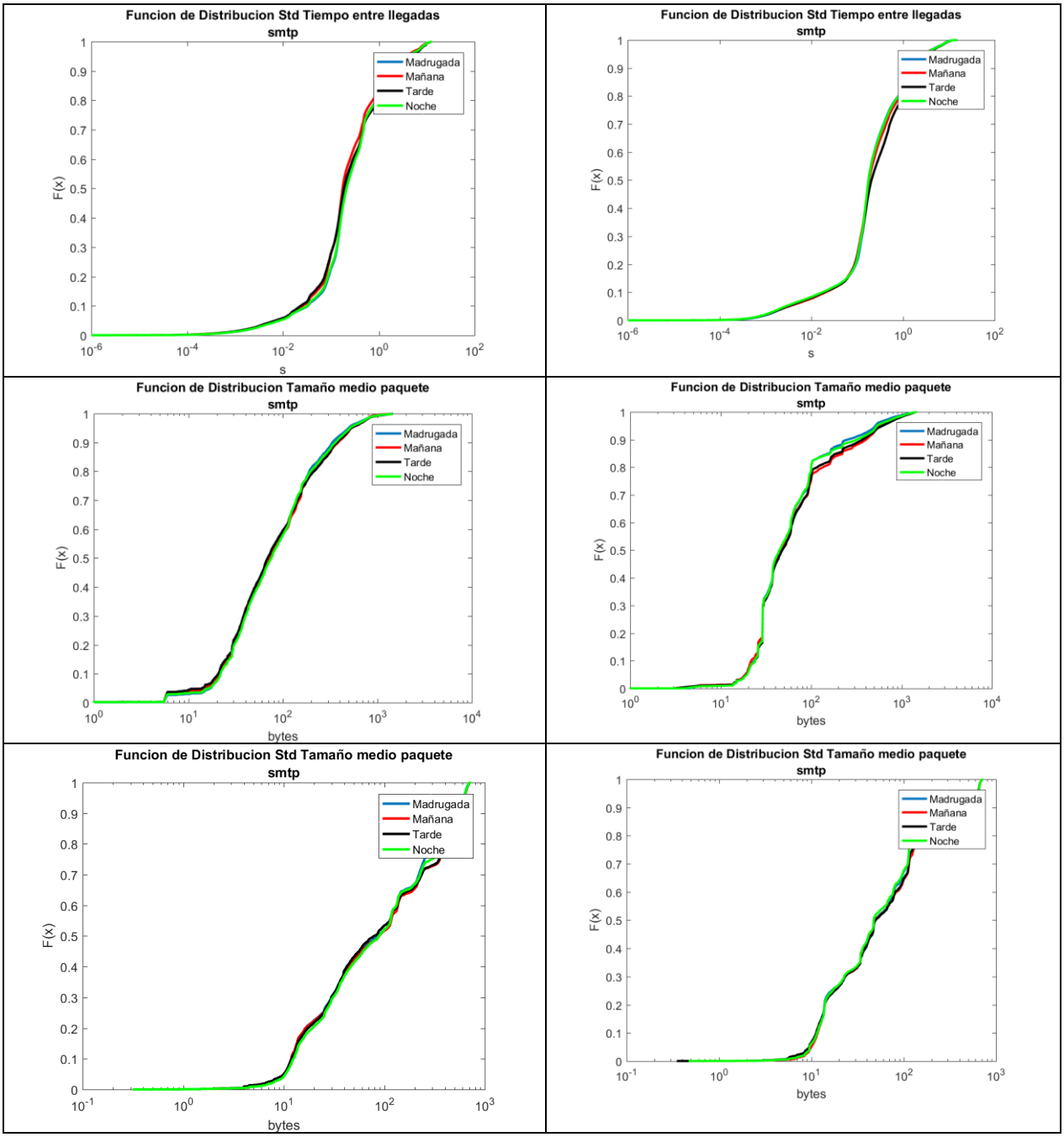
Skype

2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

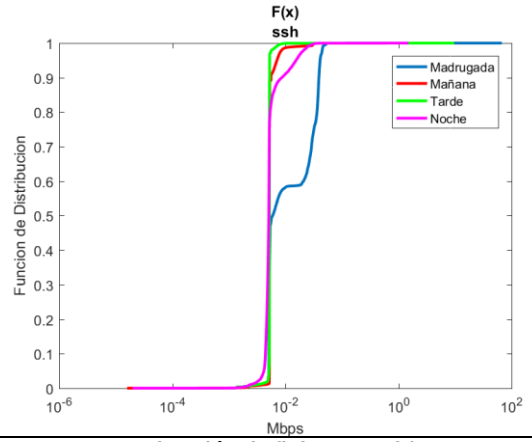
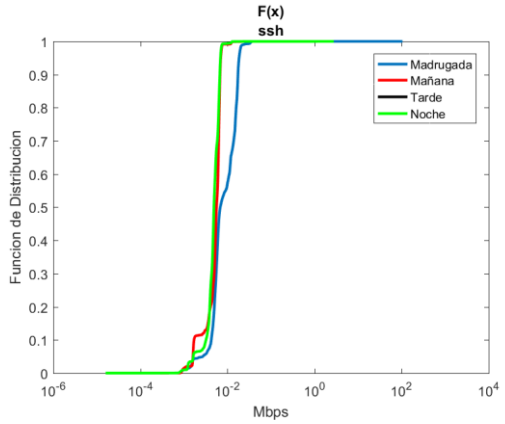
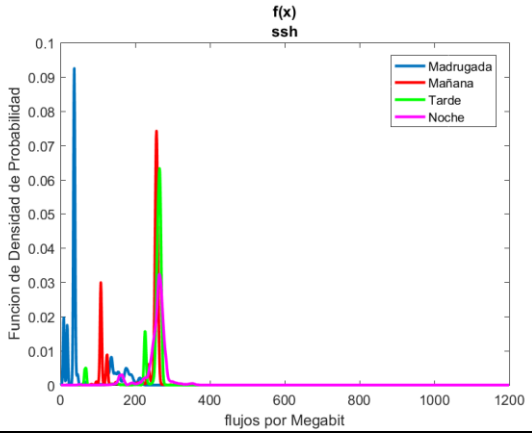
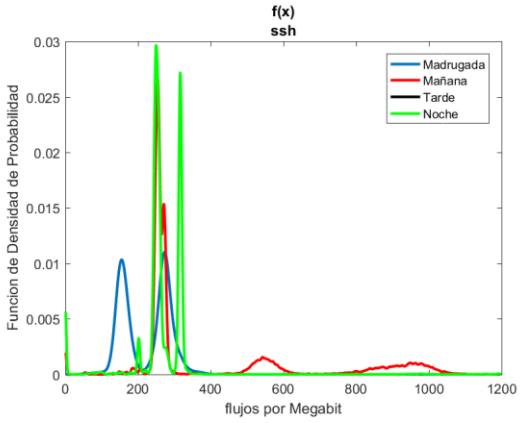
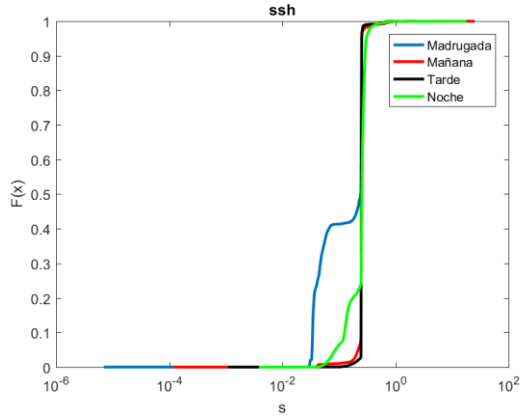
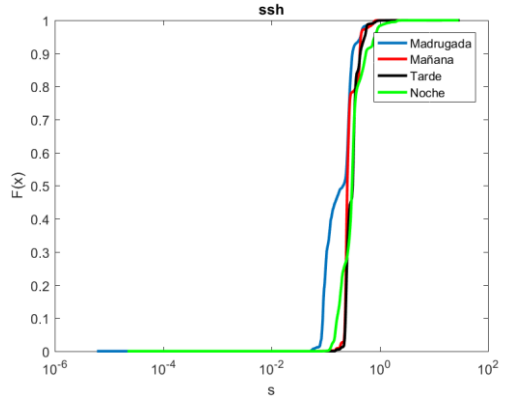

Funcion de Distribucion Std Tiempo entre llegadas
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
skypetoskype
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

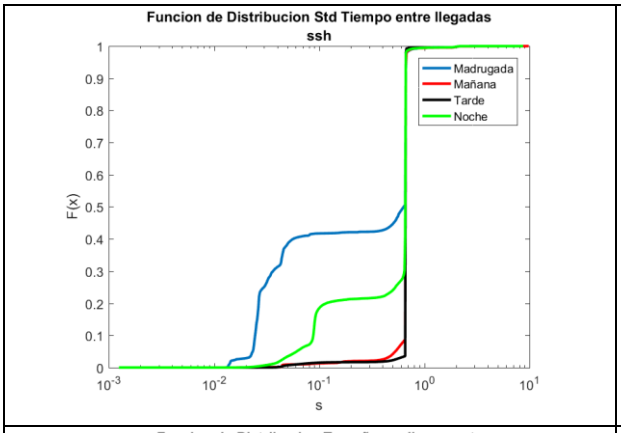
SMTP

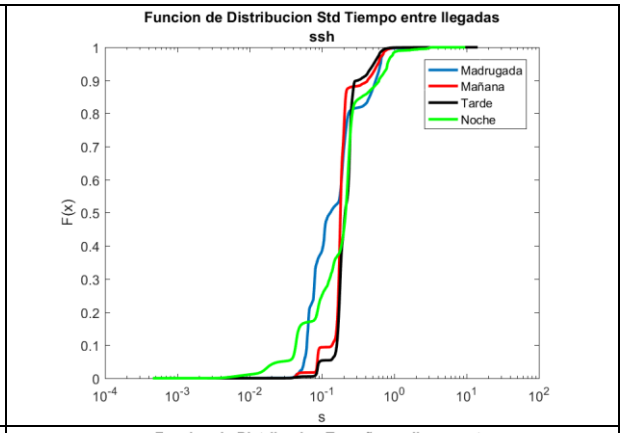
2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

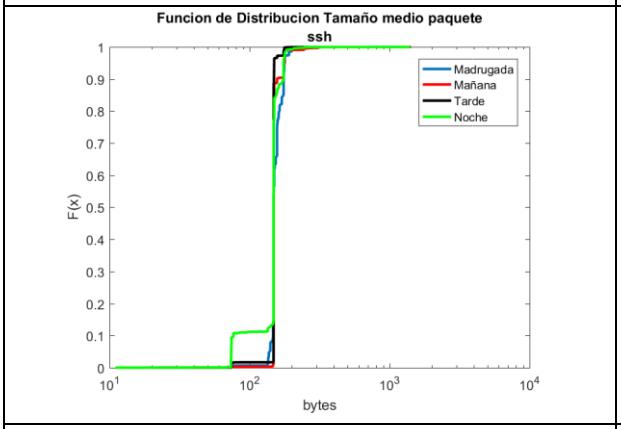

Funcion de Distribucion Std Tiempo entre llegadas
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
smtp
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

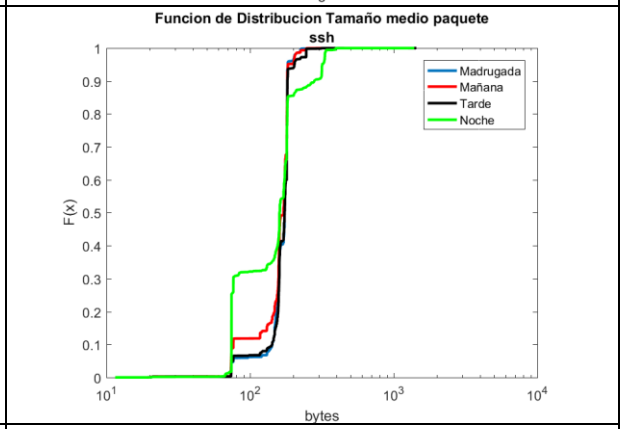
SSH

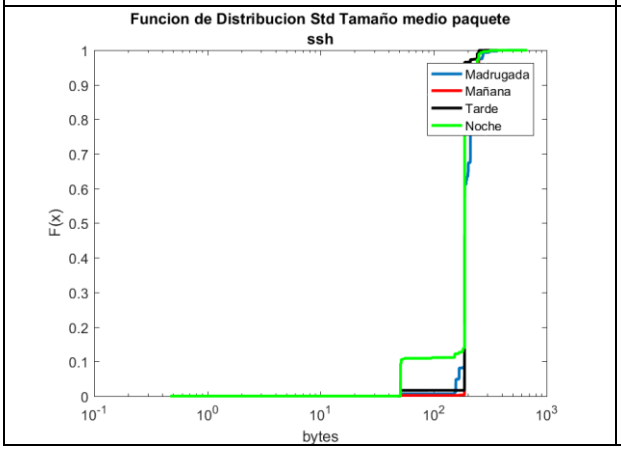
2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 

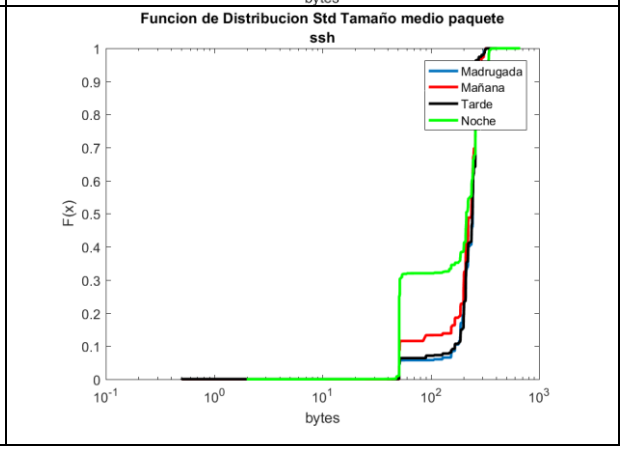

Funcion de Distribucion Std Tiempo entre llegadas
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
s


Funcion de Distribucion Std Tiempo entre llegadas
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
s

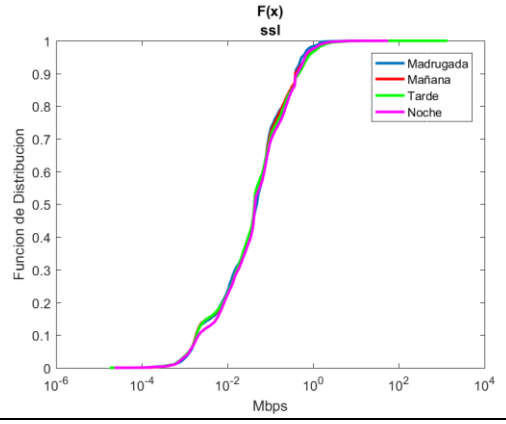
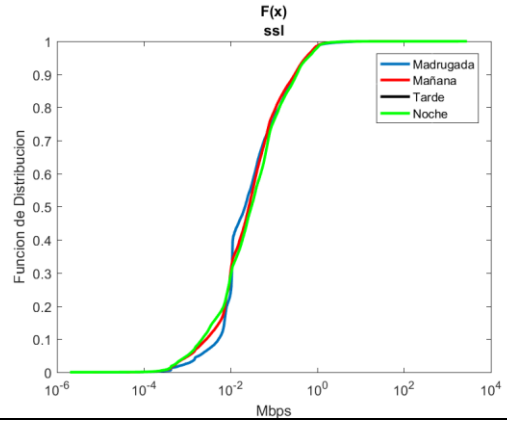
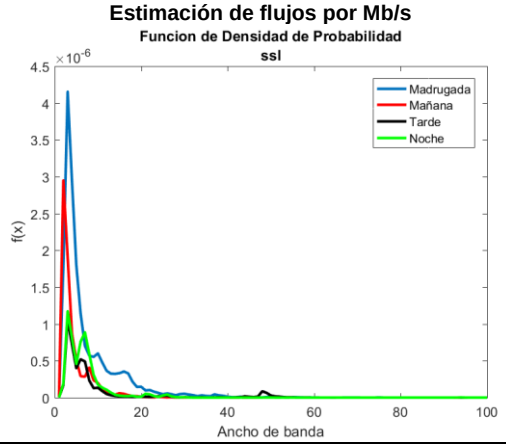
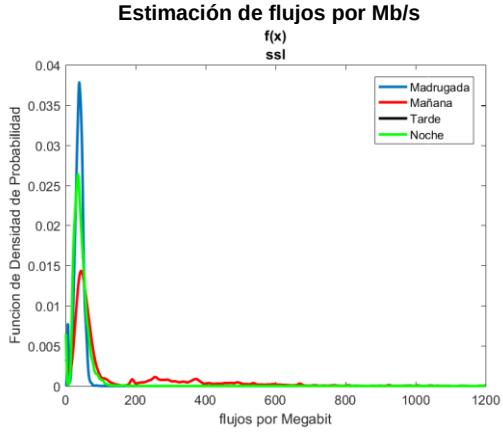
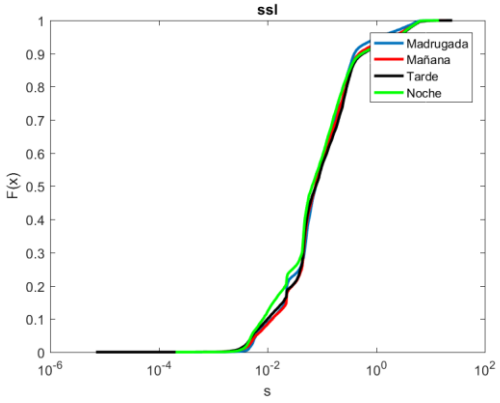
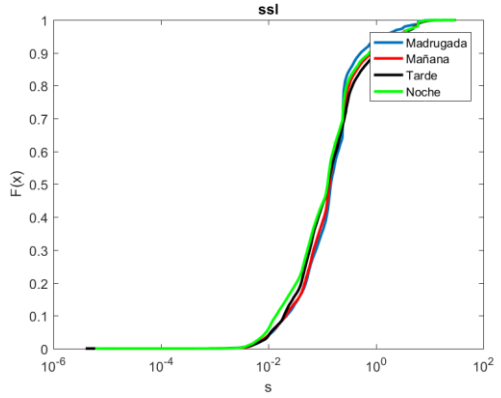

Funcion de Distribucion Tamaño medio paquete
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

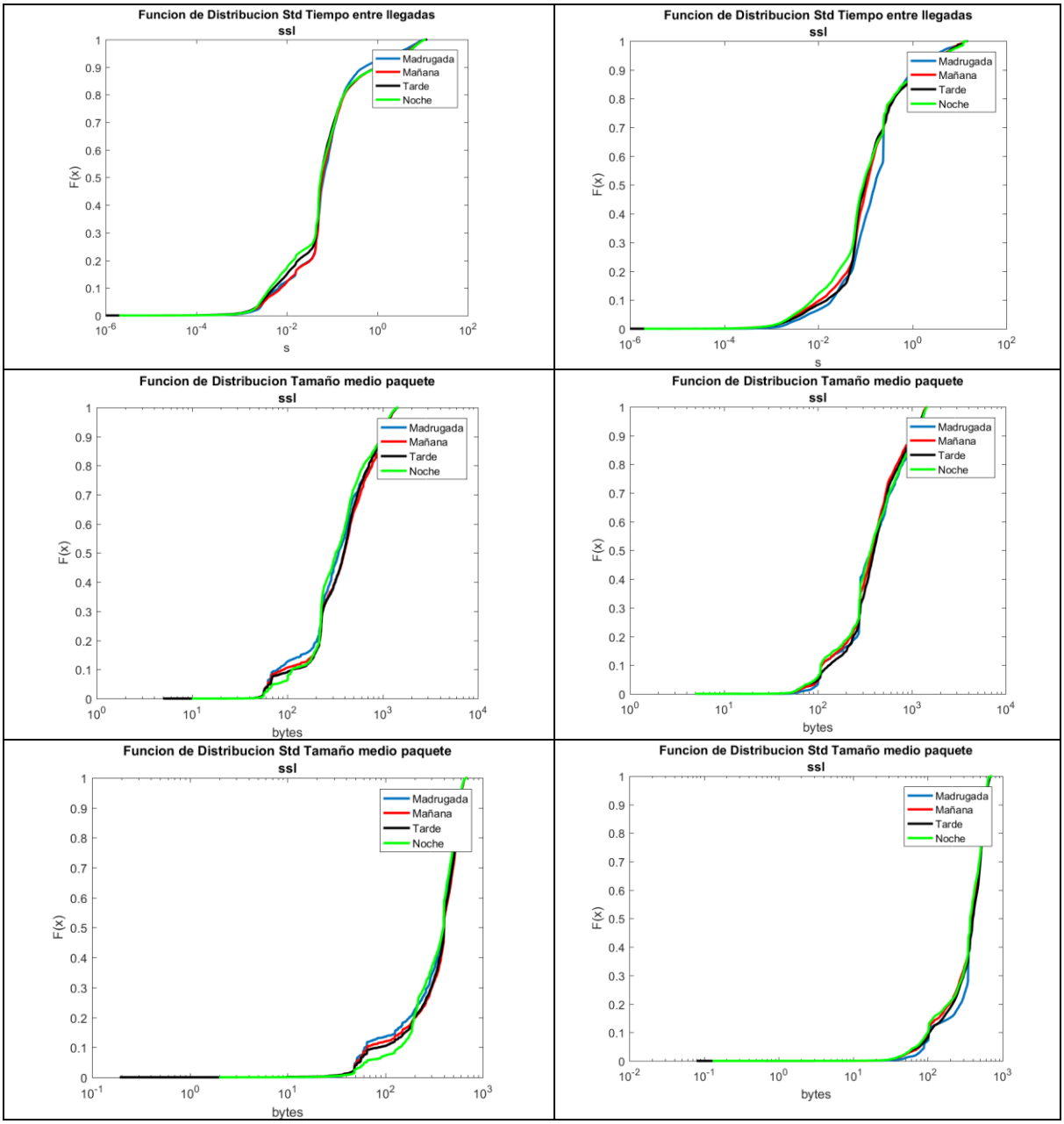

Funcion de Distribucion Tamaño medio paquete
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
bytes


Funcion de Distribucion Std Tamaño medio paquete
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
bytes


Funcion de Distribucion Std Tamaño medio paquete
ssh
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

SSL

2008	2010
Función de distribución Throughput 	**Función de distribución Throughput** 
Estimación de flujos por Mb/s 	**Estimación de flujos por Mb/s** 
Funcion de Distribucion Media entre llegadas 	**Funcion de Distribucion Media entre llegadas** 


Funcion de Distribucion Std Tiempo entre llegadas
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Std Tiempo entre llegadas
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
s
Funcion de Distribucion Tamaño medio paquete
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Tamaño medio paquete
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
bytes
Funcion de Distribucion Std Tamaño medio paquete
ssl
Madrugada
Mañana
Tarde
Noche
F(x)
bytes

D Resultados obtenidos en WEKA

Porcentaje de acierto de los clasificadores: se entrena con las muestras de 2008 y los modelos obtenidos se prueban con las muestras de 2010.

Basados en árboles de decisión

J48		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	87,665%	86,100%	87,934%	87,468%
	2008-2	88,138%	86,170%	87,933%	87,311%
	2008-3	87,878%	85,370%	85,926%	86,050%
	2008-4	86,898%	84,515%	86,214%	86,429%

Tabla D-1 Porcentaje de aciertos en aplicaciones con el modelo de clasificación J48

RandomForest		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	88,251%	86,699%	88,218%	87,750%
	2008-2	89,230%	87,725%	89,960%	88,810%
	2008-3	88,319%	86,530%	88,390%	88,320%
	2008-4	83,190%	82,717%	84,520%	84,751%

Tabla D-2 Porcentaje de aciertos en aplicaciones con el modelo de clasificación RandomForest

PART		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	88,108%	85,953%	87,094%	87,344%
	2008-2	87,420%	84,473%	86,816%	86,080%
	2008-3	87,730%	86,100%	85,376%	84,820%
	2008-4	83,480%	83,270%	85,346%	84,470%

Tabla D-3 Porcentaje de aciertos en aplicaciones con el modelo de clasificación PART

Basados en Regresión logística

Regresión Logística		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	67,031%	66,078%	67,900%	65,760%
	2008-2	68,200%	68,840%	70,440%	68,980%
	2008-3	58,335%	57,848%	59,165%	57,998%
	2008-4	69,176%	70,486%	71,940%	69,493%

Tabla D-4 Porcentaje de aciertos en aplicaciones con el modelo de clasificación Logistic

MultilayerPerceptron		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	74,760%	75,270%	76,650%	75,860%
	2008-2	75,808%	75,479%	77,960%	76,860%
	2008-3	73,380%	75,704%	78,020%	77,270%
	2008-4	72,508%	72,680%	74,303%	72,210%

Tabla D-5 Porcentaje de aciertos en aplicaciones con el modelo de clasificación MultilayerPerceptron

Basados en Máquinas de Vector Soporte

SMO		conjunto de test			
		2010-1	2010-2	2010-3	2010-4
Conjunto de entrenamiento	2008-1	65,109%	66,200%	67,704%	66,380%
	2008-2	69,595%	68,470%	69,168%	68,707%
	2008-3	70,020%	69,085%	69,830%	69,540%
	2008-4	65,570%	66,540%	67,990%	66,535%

Tabla D-6 Porcentaje de aciertos en aplicaciones con el modelo de clasificación SMO