Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Proyecto Final de Carrera
Ingeniería de Telecomunicación

Análisis de tráfico Ip multimedia en entornos empresariales: Automatización del proceso

Autor: Elisa Santos Tena

Director: Julián Fernández Navajas

Septiembre de 2012

Departamento de Electrónica y Comunicaciones
Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza

El triunfo no esta en vencer siempre,
sino en nunca desanimarse.
Napoleón

A todos los que no me dejaron desanimarme,
especialmente a mis padres.

Análisis de tráfico IP multimedia en entornos empresariales: Automatización del proceso

Abstract

El crecimiento en el uso de las aplicaciones multimedia en los últimos años ya sean Web, video en tiempo real, conversaciones de voz, aplicaciones P2P y un largo *etcétera*, ha hecho que aumente su tráfico en la red. El incremento de ese tráfico ha provocado la aparición de ciertos puntos críticos a lo largo del trayecto, siendo necesaria la identificación y el análisis de los mismos. Este hecho, ha incitado a los expertos a buscar soluciones para controlar y gestionar este tráfico, y por ello se están desarrollando continuamente herramientas que faciliten esta tarea.

Este proyecto se basa en la implementación de una herramienta automatizada enfocada al sector empresarial, para el estudio del comportamiento del tráfico de aplicaciones sobre un entorno de red controlado. La necesidad viene dada porque es necesario conocer de antemano si la red será capaz de soportar el lanzamiento de un servicio y para ello es muy útil contar con una herrameinta que garantice repetibilidad en su análisis.

Con el fin de desarrollar la herramienta, el primer paso fue seleccionar aquellas aplicaciones multimedia que se pueden definir como casos de uso común y que presentan modelos de tráfico significativos en la red, como pueden ser voz, juegos online y videovigiliancia entre otros. Para facilitar esta seleción se acudió al entorno empresarial, de tal forma que se identificaron aplicaciones de uso frecuente. En segundo lugar se realizó la selección de escenarios comunes en los entornos (WiFi y Ethernet) y que se han utilizado en el presente trabajo. Por último se eligió la herramienta adecuada para proceder a la captura del tráfico.

Como resultado de todo el proceso llevado a cabo se obtuvo una herramienta automatizada para el análisis de tráfico, la captura y generación de flujos IP multimedia que permite obtener medidas de calidad en diferentes entornos controlado de laboratorio, así como en entornos reales, de manera que sirva para la evaluación de distintas tecnologías y aplicaciones, así como la caracterización de entornos.

Índice

1. Introducción **1**

1.1. Motivación 1

1.2. Objetivo del proyecto 2

1.3. Contexto 3

1.4. Estructura 3

2. Herramientas utilizadas **5**

2.1. Herramientas para la captura de datos 5

2.1.1. Tcpdump 5

2.1.2. Wireshark 6

2.2. Herramientas para la generación y gestión de tráfico 6

2.2.1. ETG/ETGAnalyzer 6

2.2.2. TC y NETEM 7

2.3. Herramientas desarrolladas 7

2.3.1. *Script* de captura (captura.sh) 8

2.3.2. *Script* para la generación y captura del tráfico (generacion.sh) 8

2.3.3. *Script* para la caracterizacion de entornos (entornos.sh) 10

2.4. Herramientas matemáticas 11

2.4.1. Matlab 11

2.5. Aplicaciones multimedia y servicios 11

2.5.1. Videovigilancia 12
2.5.2. VLC 12
2.5.3. Sistema de Videoconferencia Vidyo 12

3. Metodología de trabajo **15**
3.1. Descripción 15
3.2. FASE 1: Estudio teórico, selección de aplicaciones y escenarios y planificación 16
3.3. FASE 2: Desarrollo de los *scripts* 16
3.4. FASE 3: Montaje y configuración de las aplicaciones y escenarios 17
3.5. FASE 4: Pruebas de análisis y obtención de resultados 17
3.5.1. Modelado de flujos IP 17
3.5.2. Medidas de calidad 18
3.5.3. Caracterización de entornos 20
3.6. FASE 5: Optimización y depurado 20

4. Modelado de flujos *IP* **21**
4.1. Modelado del sistema de videovigilancia 21
4.1.1. Configuración 22
4.1.2. Proceso de captura 23
4.1.3. Resultados 23
4.2. Modelado VLC 25
4.2.1. Configuración 25
4.2.2. Proceso de captura 26
4.2.3. Resultados 27

5. Medidas de calidad **29**
5.1. Configuración 30
5.2. Proceso de generación y captura 31
5.3. Resultados 32

6. Caracterización de entornos **35**

6.1. Configuración 36

6.2. Proceso de generación, captura y análisis 37

6.3. Resultados 38

7. Conclusiones y Líneas futuras **43**

7.1. Conclusiones 43

7.2. Líneas futuras 44

A. Acrónimos y términos **46**

A.1. Acrónimos 46

A.2. Términos 46

B. Desarrollo **47**

C. Configuración de las aplicaciones **49**

C.1. AXIS 2120 49

C.2. VLC 51

D. *Scripts* **55**

D.1. *Script* de captura de tráfico (captura.sh) 55

D.2. *Script* de generación y captura (generacion.sh) 57

D.3. *Script* de caracterización (entornos.sh) 62

Índice de figuras

1.1. Esquema general 1
1.2. Inputs y Outputs herramienta desarrollada 2

3.1. Menú principal de nuestro sistema *software* 17
3.2. Entorno de pruebas 18
3.3. Montaje generación tráfico 19

4.1. Escenario utilizado para el sistema de videovigilancia 22
4.2. Detalle de una cáptura de tráfico de la AXIS 2120 24
4.3. Escenario utilizado para el servicio de *streaming*. 25

5.1. Montaje pruebas medida de calidad 29
5.2. Medida de calidad sobre puntos de acceso 32

6.1. Montaje específico pruebas de caracterización de entornos 35
6.2. Ocupación del *buffer* para tráfico uniforme 1500*bytes vs* Tiempo (*ms*) . . . 39
6.3. Detalle ocupación del *buffer* para tráfico uniforme 1500*bytes vs* Tiempo (*ms*) 40
6.4. Velocidad de llenado y vaciado del *buffer* del *Switch Ethernet* 40

B.1. Diagrama de Gantt del proyecto. 48

C.1. Menu principal de configuración de la AXIS 2120 49
C.2. Menú de configuración de los parámetros de la red 50
C.3. Imagen enviada por la cámara IP 51

C.4. Configuración de la emisión del *streaming*. Paso 1 52
C.5. Configuración de la emisión del *streaming*. Paso 2 53
C.6. Configuración de la emisión *streaming*. Paso 3 53
C.7. Configuración de la recepción del *streaming*. Paso 1 54

Índice de tablas

4.1. Formatos de compresión de la cámara AXIS 2120 22
4.2. Resultados obtenidos ambos tamaños de imagen 24
4.3. Protocolos y códecs que ofrece VLC 26

5.1. Valores teóricos de número de conexiones calculados para vídeo 33

6.1. Límites de llenado y vaciado de los *buffer* estudiados 39

1. Introducción

1.1. Motivación

Debido al aumento de tráfico que ha supuesto la aparición de nuevas aplicaciones multimedia en Internet, se están desarrollando continuamente por parte de operadoras, desarrolladoras y proveedores herramientas para el análisis de los puntos críticos provocados por ese aumento del tráfico.

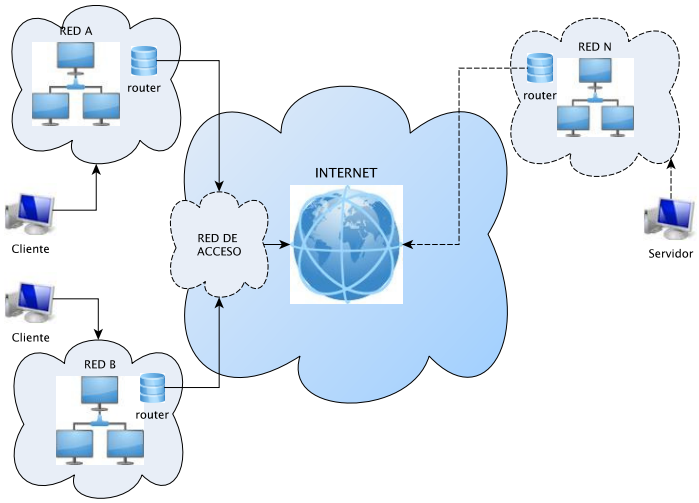


Figura 1.1: Esquema general

Según el esquema general de la figura, las operadoras tienen mucho más interés en conocer qué limitaciones y problemas se pueden dar en redes de interconexión como puede ser Internet, mientras que para proveedores y desarrolladoras, es muy útil poder conocer de antemano si la red, local o de acceso, será capaz de soportar el lanzamiento de un servicio así como conocer las limitaciones que puede suponer el uso de nuevas aplicaciones.

Por este motivo surge la necesidad de desarrollar un sistema enfocado al sector empresarial que brinde la posibilidad de analizar de manera automatizada y clara el comportamiento del tráfico de aplicaciones y servicios multimedia sobre un entorno de red, y que garantice la repetibilidad y escalabilidad en su uso.

1.2. Objetivo del proyecto

El objetivo del proyecto es desarrollar un sistema que siga una metodología clara y automatizada que permita capturar, modelar y generar tráfico multimedia tanto en entornos controlados de laboratorio como en entornos reales, de manera que permita el análisis de tráfico multimedia para obtener modelado de los flujos IP, determinación de puntos críticos en la red, medidas de calidad y caracterización de entornos.

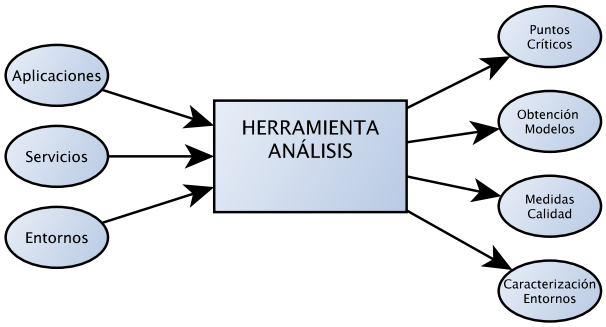


Figura 1.2: Inputs y Outputs herramienta desarrollada

Como podemos ver en la figura anterior, la herramienta desarrollada va a poder aplicarse tanto para aplicaciones y servicios multimedia como para entornos.

De las aplicaciones y servicios multimedia podemos obtener modelos y medidas de calidad, así como ser capaces de generar tráfico correspondiente a los modelos calculados.

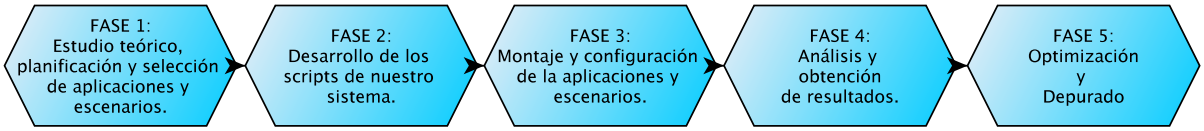
Para los entornos, se pueden obtener diferentes características del entorno, como pueden ser retardos, tamaños del *buffer*, velocidades de los enlaces, además de encontrar puntos críticos de la red.

Como estrategia para enfrentarse al problema planteado, nos hemos centrado en la automatización de un sistema que permita trabajar de forma desatendida. Además, debido a las fluctuaciones del tráfico sobre la red, el proceso deberá garantizar repetibilidad en su uso para que los resultados sean fiables.

En particular este sistema *software* (Fig.1.2) se ha desarrollado para facilitar el trabajo a aquellas empresas proveedoras que quieran conocer de antemano el impacto que sufrirán sus clientes a la hora de adquirir un servicio o aplicación (por parte de la empresa). Para ello, se ha colaborado directamente con una empresa proveedora comprobando la importancia en conocer previamente el comportamiento de una nueva aplicación sobre un entorno de red.

Con el cumplimiento de este objetivo se facilitará al entorno empresarial una herramienta que permitirá obtener información relevante a la hora de analizar distintos servicios y aplicaciones.

Las fases que se proponen para llevar a cabo el cumplimiento del mismo son las siguientes:



1.3. Contexto

Este proyecto de fin de carrera (PFC) se ha desarrollado en colaboración con la empresa Orbe Telecomunicaciones, una empresa Aragonesa que abarca desde el diseño e implantación hasta la puesta en marcha y posterior operación y mantenimiento en soluciones de los ámbitos de Telecomunicaciones. Todo el proceso se ha llevado a cabo bajo la supervisión de Julián Fernández Navajas, profesor titular de la Universidad de Zaragoza.

El desarrollo de este trabajo se realiza en el Grupo de las Tecnologías de las Comunicaciones (GTC) de la Universidad de Zaragoza, por lo tanto, la infraestructura, los equipos y las herramientas utilizadas son aquellas que se encuentran disponibles en dicho grupo de investigación.

1.4. Estructura

El presente proyecto se presenta dividido en 7 capítulos, el primero describe el porque de este estudio, presenta los objetivos que se quieren alcanzar y el entorno de trabajo en el que se desarrolla.

El segundo capítulo describe las principales herramientas empleadas en el proyecto, que nos han permitido realizar las medidas y los distintos análisis, las herramientas desarrolladas para automatizar el proceso y las aplicaciones multimedia escogidas.

El tercer capítulo expone la metodogía propuesta en el proyecto, con las distintas fases que lo componen así como una breve descripción de las mismas.

Los capítulos 4, 5 y 6 describen de forma más detallada los resultados obtenidos para cada una de las funcionalidades desarrolladas, mientras que en el último capítulo se exponen las conclusiones a las que hemos llegado y las posibles lineas futuras de investigación.

2. Herramientas utilizadas

En esta sección vamos a hablar de las herramientas y aplicaciones que hemos utilizado en el desarrollo del proyecto. El proceso de aprendizaje y utilización de las mismas ha sido continuo, y se ha debido poner especial cuidado en la decisión de las herramientas que se iban a utilizar ya que no todas trabajan con la misma exactitud ni obtienen los mismos resultados.

Las lecturas recomendadas para la compresión de este capítulo [1] [2] [3] [4] [5] se encuentran recogidas en la bibliografía del proyecto.

2.1. Herramientas para la captura de datos

Las herramientas de captura con las que hemos contado en el laboratorio donde se ha realizado el proyecto, son Tcpdump y Wireshark, para sistemas operativos de Linux y Windows.

La diferencia principal entre ambas herramientas es que Tcpdump no permite el análisis de protocolos. En una primera fase del proyecto, utilizamos Wireshark para exportar las capturas en CSV (*Comma Separated Value*) para su posterior análisis matemático con MatLab descartándose debido a que, aunque Tcpdump no posee un interfaz *friendly* para el usuario sí que permite proceder a la captura del tráfico de forma desatendida. Una vez escogido Tcpdump el resto del proyecto se ha realizado para Linux y las herramientas y aplicaciones de la que hablemos a continuación están orientadas a este entorno.

2.1.1. Tcpdump

Tcpdump es un herramienta en línea de comandos cuya utilidad principal es capturar el tráfico que circula por la red. Permite al usuario filtrar la información capturada de manera que se obtiene una salida con la información deseada.

Además, como trabaja en modo comando, permite capturar tráfico de forma desatendida mediante la programación de un *script*, lo que conlleva que nuestro proceso se realice de forma automatizada.

En contraposición a otras herramientas de captura, Tcpdump carece de interfaz gráfica, lo que lo hace un poco menos intuitivo a la hora de trabajar.

Tcpdump funciona en la mayoría de los sistemas operativos (la versión en Windows se llama WinDump) y hace uso de la biblioteca libpcap en el caso de los sistemas UNIX y Winpcap para Windows.

2.1.2. Wireshark

Wireshark es un analizador de protocolos *open-source* diseñado por Gerald Combs y que actualmente está disponible para plataformas Windows y Linux. Su principal objetivo es el análisis y captura de tráfico, además de ser una excelente aplicación didáctica para el estudio de las comunicaciones y para la resolución de problemas de red.

Una característica interesante de Wireshark es que implementa una amplia gama de filtros que facilitan la definición de criterios de búsqueda para los más de 1100 protocolos soportados actualmente; y todo ello por medio de una interfaz sencilla e intuitiva para el usuario, cosa que no nos sucede con TcpDump.

Otra característica de este analizador de protocolos es que permite exportar la información capturada a distintos y numerosos formatos de aplicación.

2.2. Herramientas para la generación y gestión de tráfico

Para la generación del tráfico de evaluación que se utiliza en los procesos de medidas de calidad y caracterización de entornos, hemos contado con ETG, una herramienta desarrollada por el grupo de trabajo. Mientras que para la gestión del tráfico se han utilizado dos herramientas disponibles para nuestro sistema Linux: TC y NETEM.

2.2.1. ETG/ETGAnalyzer

Esta herramienta [1] ha sido desarrollada por el grupo de trabajo en el cuál se enmarca el proyecto. Está orientada al estudio sistemático de comunicaciones de tráfico multimedia en tiempo real E2E (End-to-end) buscando la automatización de tareas. Además de

generar tráfico, la herramienta permite realizar medidas de parámetros objetivos y subjetivos de QoS (factor R, MOS) tanto en tráfico en un solo sentido como en tráfico de ida y vuelta en redes IP.

La arquitectura propuesta en esta herramienta es emisor-receptor que nos permitirá el estudio sistemático de redes de forma automática, que es lo que estamos buscando en todo momento. Mediante el análisis del tráfico recibido se obtendrán parámetros de retardo, pérdidas y jitter, además de estimar la calidad del sistema.

Esta herramienta nos facilita la repetibilidad ya que cada flujo se puede repetir cada cierto tiempo, permite generar tráfico equivalente al de diferentes servicios a partir de los modelos que dan información de los intervalos de transmisión de paquetes y el tamaño de los mismos.

La información necesaria para generar los flujos IP se consigue gracias a un fichero XML que se edita manualmente y que leerá el emisor para llevar a cabo el envío y la captura de tráfico. Los flujos de información están formados por varios atributos, destacando: dirección IP del emisor y receptor remoto, número de ráfagas a lanzar, número de paquetes por ráfaga, tamaño de los paquetes, así cómo tiempos de procesado (tiempo entre paquetes y tiempo entre ráfagas).

Más información en los anexos.

2.2.2. TC y NETEM

TC *Traffic Control* [2] [3] es la herramienta de control de tráfico de Linux que está contenida en el paquete *iproute2* y que permite al usuario acceder a las funciones de red. TC nos permite limitar el ancho de banda determinado y puede ser utilizado tanto para configurar las disciplinas de cola, cómo para configurar la clasificación de los paquetes en la disciplina de cola.

NETEM *Network Emulator* [4] es un emulador de red para Linux kernel 2.6.7 o versiones superiores que nos permite introducir de manera controlada en la red retrasos, pérdidas, paquetes duplicados y paquetes dañados, siendo NETEM una extensión de TC.

2.3. Herramientas desarrolladas

Recordando que el objetivo principal de este proyecto de fin de carrera, es que toda la metodología propuesta se pueda realizar de manera automatizada. Para poder alcanzar este objetivo, las herramientas de captura, generación y análisis se han integrado en un sistema *software* de manera que se obtiene un sistema final que nos permite automatizar el proceso en la medida de lo posible. Además, debido a la elección de trabajar con Linux,

estas herramientas han sido programadas para el citado sistema operativo.

Los *scripts* desarrollado se encuentran en el apartado de anexos del proyecto.

2.3.1. *Script* de captura (captura.sh)

El *script* desarrollado para este punto, nos permite capturar el tráfico generado por la aplicación multimedia o servicio a analizar, y almacenar los flujos de información para poder ser utilizados posteriormente. Este *script* se basa en la herramienta de captura Tcpdump, que nos permitirá que dicho proceso de captura se haga de manera desatentida.

Este *software* toma como parámetros la duración del proceso de captura, el número de capturas a realizar, el tiempo que debe transcurrir entre cada captura, la dirección IP de la aplicación que está generando el tráfico y el puerto.

Dispone de dos modos de uso: el primero de ellos trabaja de forma aislada y permite introducir los parámetros mediante teclado. El segundo es llamado por otro *script* que se encargará de pasarle los parámetros. En cualquiera de los dos casos nos permita realizar diversas capturas, a distintas horas sin tener que estar de manera física en el escenario.

Una vez introducidos todos los parámetros de configuración de la captura, se procede a su ejecución, almacenando el tráfico capturado en dos archivos. El primero contiene toda la información capturada de la red, mientras que el segundo sólo contiene el tráfico referente a la aplicación o servicio. Esto se consigue aplicando los filtros que Tcpdump nos ofrece, filtrando por dirección *IP*, puerto y por el protocolo utilizado (*ICMP, TCP, UDP...*).

La información obtenida de cada paquete se almacena en un línea independiente del fichero y se dispone por columnas con el siguiente orden: Tiempo de captura del paquete y el tamaño del mismo.

En el caso de ser utilizado por el usuario, una funcionalidad que se le ha añadido al *script* es la de representar los flujos de información almacenados, que pueden resultar interesantes para tener una primera idea del comportamiento de los mismos.

2.3.2. *Script* para la generación y captura del tráfico (generacion.sh)

El *script* que se ha desarrollado en este punto, nos va a permitir generar un tráfico de evaluación que será introducido por el usuario. Dicho tráfico servirá tanto para evaluar diferentes aspectos de calidad de una red como caracterizar entornos.

Para esta parte del proyecto se cuenta con la herramienta de generación de tráfico ETG desarrollada por el grupo, la cuál gracias a un fichero configuración en formato

XML permite generar flujos que tengan las mismas características que aquellos generados por aplicaciones y servicios concretos y que se hayan modelado previamente.

Por ello, el primer paso ha sido que el proceso de configuración de dicho fichero de configuración XML (*test*) se pueda realizar de diversas maneras. Se podrá editar manualmente, o mediante una ayuda opcional de edición incluida en el *script* en el caso de no conocer los comandos de edición de textos. Si se opta por la ayuda, el programa pedirá por teclado los diversos parámetros que hacen falta para que la herramienta genere los flujos de tráfico deseados y será la propia herramienta la que cree el fichero de configuración.

Los parámetros que se introducen por teclado son los siguientes:

- Nombre del proyecto.
- Nombre del test.
- Fecha de Inicio, en formato DD/MM/YYYY HH:mm:ss.
- Fecha de Fin, en formato DD/MM/YYYY HH:mm:ss.
- Tamaño de la cabecera.
- Tiempo de espera antes iniciar.
- Dirección Ip del emisor en formato XXX.XXX.XXX.XXX.
- Interfaz del emisor (eth0,eth1...).
- Puerto del emisor.
- Dirección Ip del *host* remoto en formato XXX.XXX.XXX.XXX.
- Interfaz remoto.
- Puerto remoto.
- Frecuencia (tiempo en segundos).
- Numero de ráfagas a lanzar.
- Tiempo entre ráfagas.
- Número de paquetes por ráfaga.
- Tiempo entre paquetes.
- Tamaño del paquete.
- Tráfico de ida o tráfico de ida y vuelta.
- Número de repeticiones.

El siguiente paso del *script* es intercambiar las claves públicas y privadas que permiten a nuestros equipos comunicarse entre sí de forma segura, mediante un *script* desarrollado por el grupo de trabajo (*configure.sh*). Después de que las máquinas han compartido las claves se comienza el proceso de sincronización de las mismas.

Una vez han compartido las claves y se han sincronizado entre sí los equipos, se comienza con el proceso de generación de tráfico gracias a la herramienta ETG, así como el proceso de captura. La captura se hace para todos los interfaces del equipo que genera el tráfico de evaluación (*any*) generando un archivo que contiene toda la información capturada en dichos interfaces.

En nuestro caso, se captura tanto en el interfaz de entrada, como en el interfaz de salida del sistema, generando un fichero con toda la información. Gracias a que la herramienta de generación de tráfico asigna un número de secuencia a cada paquete, podemos identificar en dicho archivo, los paquetes a la entrada y a la salida del sistema pudiendo de esta manera calcular retardos.

Aplicando una serie de filtros y procesando el archivo que contiene toda la información, se obtienen dos archivos independientes, uno con los flujos a la entrada del sistema, y otro con los flujos a la salida, quedando preparados de esta manera por si quieren utilizarse en el proceso de caracterización de entornos, donde en este caso la información esta organizada por columnas y en el siguiente orden: número de secuencia, tamaño del paquete y tiempo de emisión o recepción.

2.3.3. *Script* para la caracterizacion de entornos (entornos.sh)

El software desarrollado en este punto, permite caracterizar distintos entornos de red gracias al uso de una serie de algoritmos que analizan el comportamiento del tráfico. Para este punto contaremos con la herramienta de generación de tráfico ETG, que generará un tráfico de evaluación el cuál nos permitirá caracterizar el entorno, que es usada por nuestro *script* de generación y captura de tráfico (generacion.sh) y que genera dos archivos de información.

El *script* desarrollado trabaja con los dos archivos mencionados, uno con la información del tráfico a la entrada del sistema bajo análisis y el otro con la información del tráfico capturado a la salida del sistema. Ambos ficheros contarán con la información de número de secuencia, tamaño del paquete y tiempo de emisión o recepción dispuestos por columnas.

Con estos tres campos de información es suficiente para poder obtener distintos parámetros característicos de nuestro sistema, como pueden ser: número de paquetes perdidos y en consecuencia la tasa de pérdida, velocidad del enlace a la entrada y a la salida, retardos, tamaños de buffer presentes en la conexión a estudiar, así como otros.

El procesado de los archivos por el *software* desarrollado es el siguiente: una vez consolidados los archivos para que ambos presenten la información dispuesta en el mismo orden, estos se comparan entre sí generando un tercero que contará con la información total necesaria para la caracterización. En nuestro caso, el archivo total cuenta con 4 columnas de información de la siguiente manera: número de secuencia del paquete, tamaño, tiempo de emisión y tiempo de recepción.

Con el archivo anterior, ya sólo queda decidir qué características del sistema se quiere estudiar.

Se ha añadido al *script* el caso particular de cálculo del tamaño del *buffer* con tres

opciones diferentes, debido a que era una característica que se quería estudiar en el grupo. Para ello se cuenta con los tres métodos de análisis que se exponen a continuación:
Método 1: Algoritmo que cuenta el número de paquetes que hay en el *buffer* en el momento de la entrada de un nuevo paquete, basándose en los tiempos de entrada-salida.
Método 2: Algoritmo que estima el número de paquetes teniendo en cuenta el retardo de un paquete en el *buffer*, obtenido con base al tamaño del paquete y a la velocidad de salida.
Método 3: Algoritmo que estima las velocidades de llenado y vaciado del *buffer* con base en las relaciones temporales de los paquetes recibidos con respecto a la estimación del tiempo en el que fueron enviados. Para este algoritmo sólo se necesita el fichero de salida.

2.4. Herramientas matemáticas

En ciertos momentos de nuestro proyecto, se ha necesitado un entorno matemático en el que realizar cálculos con las capturas y los análisis de los flujos de información. Se ha optado por utilizar MatLab ya que es un herramienta ampliamente conocida y que proporciona gran potencia matemática.

2.4.1. Matlab

MATLAB es un software matemático que ofrece un entorno integrado con un lenguaje de programación propio. Está disponible para las plataformas Unix, Windows y Apple Mac OS X.

Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos *hardware*.

Es un *software* muy usado en universidades y centros de investigación y desarrollo, aunque es un producto propietario de MathWorks y por lo tanto sujeto a licencia. En el presente trabajo se ha utilizado principalmente para analizar las tramas de información obtenidas en las capturas del tráfico en la red, ya que en muchos casos la complejidad de ciertas operaciones matemáticas hizo necesario contar con una herramienta de este tipo.

2.5. Aplicaciones multimedia y servicios

Además de escoger las herramientas a utilizar en el desarrollo del proyecto, es necesario estudiar y seleccionar aquellas aplicaciones y servicios multimedia que despiertan mayor interés, tanto para nosotros como para el entorno empresarial.

2.5.1. Videovigilancia

La AXIS 2120 es una cámara fija de red, también conocida como *cámara IP*, para aplicaciones de videovigilancia y monitorización en forma remota. Proporciona múltiples secuencias fijas JPEG y *motion*-JPEG de forma simultánea, sea a frecuencia de imagen máxima o con calidades que pueden configurarse para que se adapten a las necesidades del servicio y a las restricciones del ancho de banda.

Permite obtener hasta 25/30 imágenes/seg (PAL/NTSC) a una resolución de 352×288, hasta 5 niveles distintos de compresión y 7 anchos de banda (desde 0,1 hasta 2 MBit/s) o eliminar la restricción del ancho de banda (*unlimited*).

El proceso de configuración de la cámara es muy sencillo. Básicamente se debe configurar la dirección IP desde la cuál se esta enviando los flujos de información, así como los parámetros mencionados: tipo de imagen (*single* o *motion*), compresión utilizada en la transmisión y ancho de banda utilizado.

El grupo dispone de una cámara accesible para la investigación, por lo que también se ha podido utilizar siempre que ha sido necesario probar las herramientas.

2.5.2. VLC

VLC media player es un reproductor multimedia y framework multimedia libre y de código abierto desarrollado por el proyecto VideoLAN. Es un programa multiplataforma con versiones disponibles para muchos sistemas operativos como Microsoft Windows, GNU/Linux, Mac OS X, BeOS...*etc.*

Es un reproductor de audio y vídeo capaz de reproducir muchos códecs y formatos, además de capacidad de streaming (uno de los principales usos que vamos a estudiar en este trabajo). Es software libre, distribuido bajo la licencia GPL.

Desde un primer momento se tuvo la limitación de que el presente proyecto se pudiera probar asiduamente en instalaciones empresariales, por lo que el manejo y estudio de esta aplicación, nos suponía una alternativa a no poder acudir a la empresa siempre que se necesite probar la herramienta. Fundamentalmente hemos utilizado VLC para realizar *streamings* de audio y vídeo.

2.5.3. Sistema de Videoconferencia Vidyo

Vidyo es un software de videoconferencia para empresas que utiliza la aquitectura adaptativa de vídeo en capas, ofreciendo una menor latencia y una alta calidad en la

definición. La arquitectura de Vidyo optimiza dinámicamente la calidad del vídeo según la red y las capacidades de los dispositivos terminales individuales para ofrecer experiencias con calidad de telepresencia a todos los participantes.

Vidyo utiliza tecnología de enrutamiento inteligente sensible a los medios con la flexibilidad y la solidez del estándar de compresión de vídeo H.264 SVC (*Scalable Video Coding*) para ofrecer colaboraciones y comunicaciones de vídeo, de altas prestaciones, en Internet, 3G/4G, WiFi y WiMAX. Esta innovadora solución ha desencadenado una revolución en el sector, haciendo que las videoconferencias con calidad de alta definición sean accesibles para todos, sobre cualquier red y en cualquier terminal.

Otra de las ventajas de Vidyo es que ofrece soporte de videoconferencia en sistemas de salas, equipos de sobremesa y dispositivos móviles y para distintos sistemas operativos.

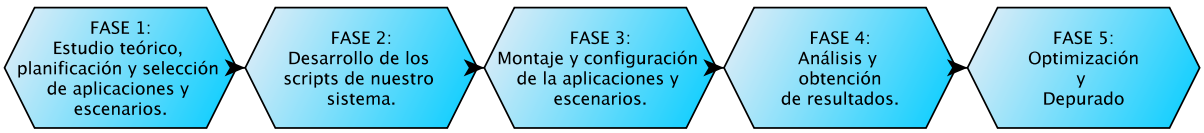
El sistema de videoconferencia Vidyo está formado básicamente por: VidyoRouter, VidyoDesktop y VidyoGateway donde el primero, es el elemento central del sistema VidyoConferencing, el segundo es la aplicación que se descarga en el dispositivo y el tercero es un sistema que permite adaptar nuestro nuevo sistema con otros sistemas de videoconferencia heredados.

Debido a que se trata de un sistema profesional, el uso está limitado a la disponibilidad que nos ofrece la empresa ORBE. S.L.

3. Metodología de trabajo

3.1. Descripción

El objetivo de este proyecto de fin de carrera, es desarrollar un sistema con una metodología clara y automatizada que nos permita capturar, modelar y generar tráfico multimedia en diferentes entornos reales. Para la consecución de este objetivo se llevan a cabo 5 fases principales:



En el anexo A.1 podemos encontrar el diagrama de Gantt correspondiente al trabajo realizado en este proyecto.

Las funcionalidades del sistema desarrollado para el proyecto son las siguientes:

- Captura de flujos de tráfico.
- Generación de tráfico de evaluación.
- Caracterización de entornos.

Cabe destacar que estas funcionalidades se pueden realizar todas como un proceso conjunto, o independientemente unas de las otras y en el orden que se quiera, dependiendo de las pruebas que quiera realizar el usuario.

Por último, en nuestro proyecto se han realizado las siguiente pruebas para comprobar el correcto funcionamiento de la metodología propuesta para el sistema.

- Modelado de flujos IP. Análisis y obtención de resultados.

- Medidas de calidad. Análisis y obtención de resultados.
- Caracterización de entornos. Análisis y obtención de resultados.

3.2. FASE 1: Estudio teórico, selección de aplicaciones y escenarios y planificación

Para esta fase en primer lugar se procede al estudio teórico de los servicios multimedia en los que se basa el proyecto de fin de carrera, obteniendo la información necesaria para llevar a cabo nuestro trabajo. El siguiente paso será elegir aquellas aplicaciones multimedia (VLC, televigilancia...) y tecnologías (WiFi, LAN...) que despiertan mayor interés, tanto para nosotros como para el entorno empresarial.

A continuación, hay que analizar las herramientas con las que contamos, eligiendo aquellas que resultan más útiles a la hora de conseguir mayor automatización en el proceso. Elegidos los servicios, las aplicaciones, tecnologías y las herramientas se definen los escenarios generales para la realización de las pruebas.

3.3. FASE 2: Desarrollo de los *scripts*

Como se ha podido ver en el capítulo 2 de este proyecto, se han desarrollado una serie de *scripts* que van a servir para automatizar el proceso en la medida de lo posible. Estos *scripts* cuentan a su vez con las herramientas de captura y generación de tráfico, que por sí solas no pueden trabajar de manera automatizada.

Los *scripts* han sido desarrollados en *shell*, y funcionan sobre sistemas operativos que utilizan el núcleo de Linux, como pueden ser Ubuntu y Debian (se han probado para los dos).

Para su desarrollo se ha usado *awk*; que es un lenguaje de programación diseñado para procesar datos basados en texto, ya sean ficheros o flujos de datos. La ventaja de utilizar *awk* es la rapidez de ejecución a la hora de tratar archivos grandes y va a ser de gran utilidad en el proyecto.

Estos *scripts* se han probado previamente sobre entornos muy controlados de laboratorio, para comprobar el correcto funcionamiento de los mismos, antes de pasar a los escenarios definidos y además han sido unificados todos en un sistema *software* fácil de entender por el usuario.

En la siguiente figura vemos el menú principal en el que aparecen las tres funcionalidades definidas para el PFC.

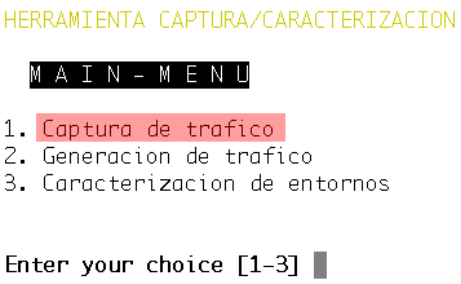


Figura 3.1: Menú principal de nuestro sistema *software*

3.4. FASE 3: Montaje y configuración de las aplicaciones y escenarios

El siguiente paso en nuestro estudio es acondicionar el entorno, comprobando las conexiones y configuraciones de los parámetros en las pruebas a realizar. Es necesario recalcar que el usuario del sistema debe poseer un control preciso sobre el entorno de pruebas, y que dicho entorno debe ser configurado acorde con las características de la tecnología.

Se debe tener también un control preciso a la hora de llevar a cabo las configuraciones de las aplicaciones y servicios, conociendo detalladamente el proceso de las mismas, puesto que para una misma aplicación se pueden modificar parámetros como ancho de banda, compresión del flujo de datos, resolución y que son de gran importancia a la hora de modelar el tráfico de la aplicación.

La configuración de las aplicaciones, las pruebas, los escenarios utilizados en las pruebas y los resultados específicos se pueden encontrar en los capítulos 4, 5 y 6 del presente proyecto.

3.5. FASE 4: Pruebas de análisis y obtención de resultados

3.5.1. Modelado de flujos IP

Una de las primeras pruebas que se realizaron en el proyecto, fue el modelado de tráfico*IP*. Para este punto necesitamos el servicio a estudiar y/o modelar (que actúa como

servidor) y un cliente que esté demandando el servicio además del equipo de captura y todos ellos conectados a un *hub* en el que capturaremos el tráfico.

La topología que se ha empleado es la que se muestra en la Fig. 3.2.

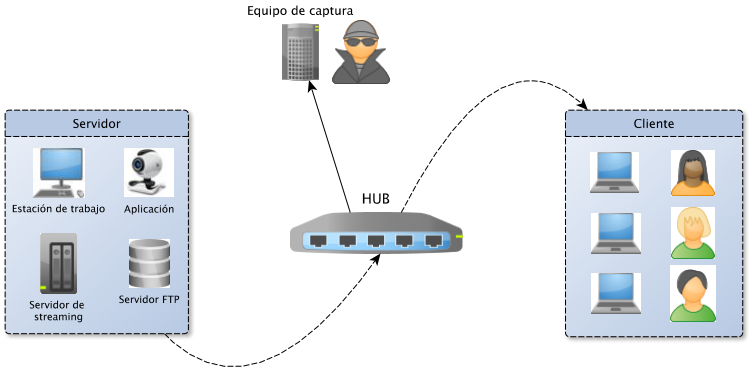


Figura 3.2: Entorno de pruebas

Una vez hecho el montaje, se debe configurar la aplicación o servicio que se va a estudiar. En este caso no se puede automatizar el proceso de configuración de la aplicación o servicio debido a que cada uno es diferente y deberá hacerse presencialmente.

El equipo de captura (*sniffer*) cuenta con el *script* de captura del tráfico (captura.sh), que se ha desarrollado para automatizar este proceso. Como hemos explicado previamente, el *sniffer* captura todo el tráfico perteneciente al interfaz que está escuchando, el cuál es filtrado por la herramienta para generar un archivo que nos da la información necesaria (tiempo y tamaño) para obtener el modelo de la aplicación.

Dicha captura se realiza con Tcpdump puesto que como hemos comentado previamente, ofrece la posibilidad de capturar tráfico de forma desatendida lo cuál permite, al menos, automatizar el proceso.

El orden de actuación para obtener la captura es el siguiente: el primer paso es lanzar desde el *sniffer* el *script* de captura de tráfico. A continuación, se lanza la aplicación a analizar, capturando y almacenando los flujos de información en archivos para su posterior estudio.

Con los datos obtenidos se procede a calcular un modelo para los flujos de información capturados. Este proceso no se ha podido automatizar puesto que se trata de una tarea compleja que requiere un desarrollo que se escapa a los objetivos del proyecto.

3.5.2. Medidas de calidad

La siguiente prueba, se centra en la obtención de un sistema que genera un tráfico de evaluación correspondiente al modelo obtenido de las aplicaciones y servicios del punto

anterior o a otros modelos previamente conocidos por el usuario y que nos permite evaluar diferentes aspectos de calidad de una red.

El esquema de la Fig. 6.4 expone el montaje realizado para este punto en donde el equipo A sirve tanto para generar como para capturar tráfico y está conectado tanto a la entrada como a la salida del sistema bajo análisis, mientras que el equipo B sólo recibe tráfico y está conectado sólo a la salida del sistema. En este caso, se va a utilizar el *script* de generación y captura de tráfico (generacion.sh), que se encuentra ubicado en el equipo A y que genera el tráfico por el interfaz 0 mientras que captura por el interfaz 0 y el interfaz 1.

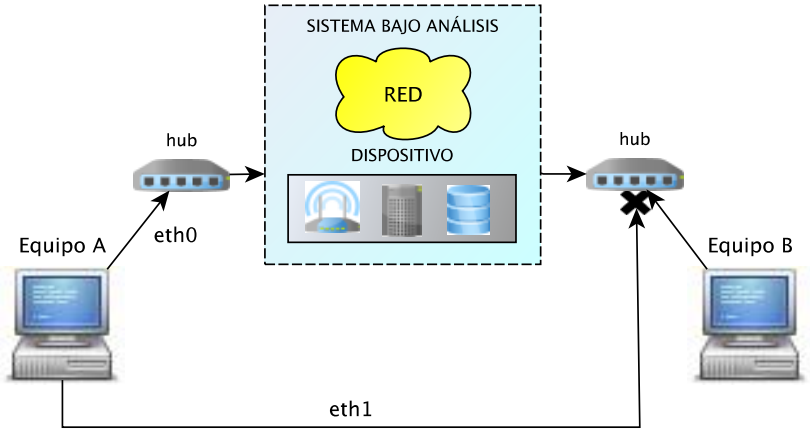


Figura 3.3: Montaje generación tráfico

Cabe destacar que este es el esquema completo para realizar medidas de calidad pero que en ocasiones se puede prescindir del equipo B para generación de flujos que no necesiten respuesta (como puede ser tráfico UDP).

Sobre el sistema bajo análisis se realizan dos tipos de pruebas de calidad: la primera de las pruebas realizadas consiste en medir la calidad de un servicio al atravesar un sistema mientras que la segunda obtiene el número de comunicaciones para las cuáles el sistema no presenta deterioro en la información. El tráfico para realizar las medidas deberá seguir el modelo de tráfico real que se ha modelado previamente.

El orden de actuación para obtener las medidas de calidad es el siguiente: Una vez tenemos montado el escenario de evaluación y configurados los equipos ejecutamos el script de generación y captura del tráfico, dando lugar a dos archivos con los que podremos medir el retardo y las pérdidas introducidas por el sistema, tanto para un sólo servicio como para comunicaciones que se realicen simultáneamente sobre el sistema bajo análisis.

3.5.3. Caracterización de entornos

Para esta útima prueba del proyecto, se utiliza el mismo escenario montado para las medidas de calidad, aunque en este caso después de la captura se procede a la caracterización de entornos de red basada en el análisis del tráfico y trabajaremos con los *scripts* desarrollados de generación y caracterización de entornos (generacion.sh y entornos.sh) que se encuentran ubicados en el quipo A.

El orden de actuación para el análisis de este escenario es el siguiente: en primer lugar se genera un tráfico de evaluación con la herramienta de generación de tráfico que tenemos en nuestro sistema, desde el equipo A (local) hacia el equipo B (remoto), tal y como se ha indicado anteriormente (igual que en el apartado de calidad) obteniendo los dos archivos de información, que serán la entrada del *script* de caracterización de entornos, el cuál tratara los flujos de información para obtener distintas características del entorno. Se ha añadido la funcionalidad de capturar un tercer archivo en el equipo B cuando éste se encuentre en otra localización y por lo tanto no se pueda conectar directamente, aunque en este proyecto no se vayan a realizar pruebas con el mismo.

El tráfico de evaluación para realizar la caracterización es de tipo uniforme, y es generado de manera que saturemos la entrada al sistema. Una vez obtenidos los ficheros ya sólo queda tratarlos para obtener las caracteríticas del sistema bajo análisis mediante los métodos descritos en el capítulo 2.

3.6. FASE 5: Optimización y depurado

Despues de haber completado las fases previamente comentadas, se ha realizado un proceso de optimización de los *script* y depurado del sistema para hacerlo mucho más manejable y fácil para el usuario.

Se han desarrollado una serie de manuales que pueden ser leídos en el propio terminal de Linux, donde se explica el funcionamiento de los *scripts* y se dan las pautas iniciales para su correcto funcionamiento, así como unos anexos de configuración de las aplicaciones que facilitan el proceso al usuario.

Se han modificado algunos algoritmos que hacían que el proceso se ralentizará para archivos de información de gran tamaño.

4. Modelado de flujos IP

Como se ha indicado anteriormente, para las empresas proveedoras es muy útil poder conocer de antemano el comportamiento que un nuevo servicio puede provocar en la red de acceso estudiando el impacto que tendría sobre un escenario real. Por ese motivo, una de las funcionalidades que se ha desarrollado en el presente proyecto, permite capturar y analizar los flujos de distintas aplicaciones multimedia sobre entornos reales.

Con esta primera funcionalidad, se han llevado a cabo una serie de pruebas que ratifiquen su perfecto funcionamiento a la hora de ser usado por la empresa proveedora, obteniendo a modo de ejemplo unos resultados para las aplicaciones analizadas.

Para esta sección de modelado de flujos *IP* de las tres funcionalidades desarrolladas se ha utilizado solamente la opción 1 (captura de tráfico). Se han hecho pruebas sobre dos de las aplicaciones mencionadas en el capítulo 2: sistema de televigilancia y sistema de *streaming*, ya que debido a la disponibilidad del sistema de videoconferencia sobre este no se han podido realizar las pruebas. A continuación se explica detalladamente el proceso de configuración de los escenarios de cada una de ellas y los resultados obtenidos.

Las lecturas recomendadas para la compresión de este capítulo [6] [7] [9] se encuentran recogidas en la bibliografía del proyecto.

4.1. Modelado del sistema de videovigilancia

Para esta parte del proyecto contamos con una cámara fija de red AXIS 2120 que proporciona múltiples secuencias JPEG (fijas y *motion*) pudiendo obtener hasta 25/30 imágenes/seg (PAL/NTSC) a una resolución de 352×288, sobre una red con anchos de banda de 10 Mbps.

El primer es el montaje del escenario que se corresponde con el montaje mencionado en el capítulo 3 y la configuración de las aplicaciones. El escenario específico utilizado para determinar el tráfico de la cámara es el mostrado en la Fig. 4.1.

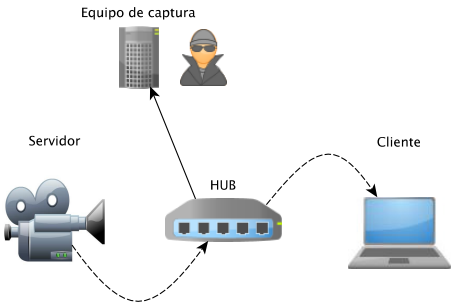


Figura 4.1: Escenario utilizado para el sistema de videovigilancia

En este apartado sólo vamos a trabajar con la opción 1 del sistema desarrollado que utiliza el *script* de captura (captura.sh). El sistema va a estar ubicado en el equipo de captura, y está conectado al Hub de 10Mbps en donde podremos capturar todo el tráfico generado por la cámara.

4.1.1. Configuración

El proceso de configuración de la cámara es muy sencillo. El primero paso es acceder a ella de forma remota desde el equipo de trabajo mediante un navegador web tradicional. A continuación configuramos la dirección *IP* desde la cuál la cámara está enviando los flujos de información. También debemos configurar los distintos parámetros que proporciona el fabricante, ya que el comportamiento del flujo de datos difiere en función de la configuración elegida. Los más significativos para nosotros para modelar el tráfico son: tipo de imagen (*single* o *motion*), compresión utilizada en la transmisión (ver tabla 4.1), tamaño de la imagen en *pixeles* (352×288 ó 704×576) y ancho de banda utilizado por la cámara ($2Mbps$,$1Mbps$,$0.5Mbps$...).

FORMATOS DE COMPRESIÓN AXIS		
Opciones	Factor de compresión	
	352×288	704×576
Lowest	$75K$	$300K$
Low	$13K$	$50K$
Medium	$8K$	$32K$
High	$4K$	$16K$
Very high	$3K$	$12K$

Tabla 4.1: Formatos de compresión de la cámara AXIS 2120

En los anexos podemos encontrar con más detalle los distintos parámetros de

configuración de la cámara. Una vez configurada la cámara y el escenario, comenzamos con el proceso de captura.

4.1.2. Proceso de captura

El orden de actuación es el siguiente: procedemos a generar tráfico desde la cámara *IP* y a continuación lanzamos desde el equipo de captura nuestro sistema *software*, eligiendo la opción 1 (captura del tráfico) el cuál nos pide una serie de parámetros para configurar la captura.

El sistema captura toda la información existente en la red en la que estamos trabajando. A continuación aplica un filtro por dirección *IP* de la fuente, puerto, tipo de tráfico a capturar y almacena los flujos de información obtenidos en un archivo en formato de texto. Además, también almacena la captura realizada por Tcpdump para su posterior consulta si fuese necesario y genera una gráfica del tráfico capturado.

Ya tenemos la información necesaria para poder trabajar en un entorno matemático que nos ayude a conocer el comportamiento del tráfico de la aplicación en nuestro entorno de red.

4.1.3. Resultados

Una de las primeras conclusiones a la que hemos llegado es que de los parámetros de configuración mencionados, el más importante a la hora de caracterizar los flujos de información es el tipo de compresión utilizada, afectando tanto a la calidad con la que el usuario percibe las imágenes como al comportamiento de los paquetes en la red.

Se han hecho pruebas para distintos anchos de banda (2M*bps*, 1M*bps* y 0,5M*bps*), distintos factores de compresión y distintos tamaños de imagen (352×288 ó 704×576).

Se ha concluido que el ancho de banda configurado en la cámara no es determinante en el número de paquetes enviados en cada ráfaga, permaneciendo constante para los tres anchos de banda con los que se han hecho las pruebas pero si es determinante en el tiempo entre ráfagas.

El nivel de compresión utilizado define el número de paquetes enviado en cada ráfaga, es decir el tamaño de la imagen en cada instante. El tamaño de los paquetes será siempre de 1500 *bytes*, excepto el último paquete que será de tamaño variable dependiendo del tipo de compresión elegida y de la resolución de la imagen.

Podemos ver en la tabla 4.2 el número de paquetes/ráfaga y el tiempo entre ráfagas que se obtienen en función de la resolución, del factor de compresión y del ancho de banda escogidos:

Resolución	Nivel de compresión	Paquetes/ráfaga - Tiempo ráfagas(ms)		
		$2Mbps$	$1Mbps$	$0.5Mbps$
352×288	$75K$	16 - 280	16 - 280	16- 600
	$13K$	(7-9) - 40	(7-9) - 80	(7- 9) - 160
	$8K$	5 - 40	5 - 80	5 - 160
	$4K$	3 - 40	3 - 40	3 - 80
	$3K$	3 - 40	3 - 40	3 - 80
704×576	300	38 -240	38 - 480	38 - 900
	$50K$	25 - 160	25 - 280	25 - 400
	$32K$	15 - 160	15 - 160	15 - 400
	$16K$	9 - 160	9 - 160	9 - 200
	$12K$	9 - 160	9 - 160	9 - 200

Tabla 4.2: Resultados obtenidos ambos tamaños de imagen

Los flujos de información de la cámara se realizan siempre por ráfagas, compuestas a su vez por el número de paquetes que proporcione el tipo de compresión utilizada. Las tramas van seguidas dentro de la ráfaga y el tiempo entre ráfagas será también variable dependiendo de la configuración elegida.

En la Fig. 4.2 podemos ver una representación del tráfico capturado, para un ancho de banda de 2Mbps, para una resolución de imagen pequeña y utilizando un factor de compresión medio (8K). Se pueden diferenciar el número de paquetes enviados en cada ráfaga, asi como el tiempo entre ráfagas, que con la configuración ya descrita podemos observar que es de 40 ms y se puede concluir que el tráfico presenta un comportamiento uniforme.

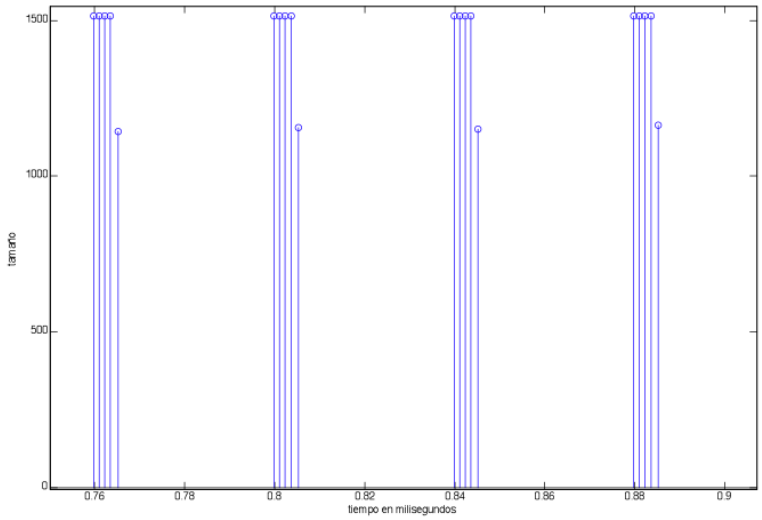


Figura 4.2: Detalle de una cáptura de tráfico de la AXIS 2120

Pero al contrario que el número de paquetes/ráfaga que permanecia constante independientemente del ancho de banda utilizado, no lo hace para el tiempo entre ráfagas siendo distinto dependiendo del ancho de banda.

Cabe destacar que para esta aplicación se han hecho bastantes pruebas y se han obtenido más resultados que para las otras debido a la simplicidad de la aplicación como a la simplicidad de los modelos que genera.

4.2. Modelado VLC

En este apartado contamos con VLC *media player* que es un reproductor multimedia y framework, capaz de reproducir muchos *códecs* y formatos, además de capacidad de *streaming*.

Se va a realizar la prueba de captura de tráfico con nuestro sistema desarrollado para una aplicación de *streaming* entre dos equipos de nuestro laboratorio que están interconectados entre sí mediante un hub de 100*Mbps*. Siguiendo el esquema general del capítulo 3 la topología necesaria para esta prueba es la mostrada en la figura 4.3.

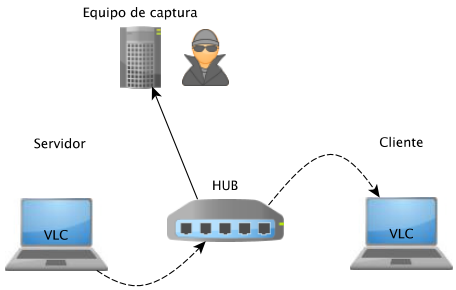


Figura 4.3: Escenario utilizado para el servicio de *streaming*.

El programa VLC debe estar instalado en ambos equipos, tanto en el servidor como en el cliente. En nuestro equipo de captura tendremos ubicado el sistema desarrollado y utilizaremos la opción de captura de tráfico (opción 2).

4.2.1. Configuración

Tanto el servidor como el cliente se configuran de manera diversa y por tanto hay que tener cuidado a la hora de la configuración del servicio VLC en ambos equipos.

En el servidor, debemos tener el archivo de audio o vídeo con el que vamos a hacer el *streaming*, y en las opciones que nos proporciona la herramienta debemos elegir emisión. Una vez elegido, la aplicación nos pide el protocolo que queremos utilizar para hacer el

PARÁMETROS CONFIGURACIÓN STREAMING	
PROTOCOLOS	CÓDECS
RTP	MPEG-1,2,4
UDP	DIVX 1,2,3
Archivo	H.264, H.263
HTTP	Theora
MMS	MP4, WAV y MOV
IceCast	AAC, Vorbis y MP3

Tabla 4.3: Protocolos y códecs que ofrece VLC

streaming y si queremos que haya trascodificación o no (se utiliza alguno de los *códecs* que posee la herramienta). Los protocolos y *códecs* se pueden consultar en la tabla que aparece a continuación (4.3). Ahora sólo queda introducir la dirección del equipo con el que vamos a hacer *streaming* y el puerto solicitado por el protocolo.

Debemos configurar la aplicación VLC en el equipo que actúa como cliente. En este equipo deberemos elegir la opción de volcado de red y después introducir el protocolo elegido para la emisión. En este caso no se debe poner dirección *IP* debido a que es el cliente el que esta recibiendo la información.

Una vez configuradas las dos máquinas, ya sólo queda comenzar con el *streaming* comenzando con el volcado de red en el cliente y a continuación la emisión en el servidor. En los anexos podemos encontrar con más detalle el proceso de configuración de la aplicación *vlc* para el realizado del *streaming*.

4.2.2. Proceso de captura

El proceso es similar al utilizado en la captura del sistema de videovigilancia. Lo primero es comenzar a recibir el *streaming* en el equipo que actúa como cliente con la opción de volcado de red. A continuación, desde el servidor comenzamos el proceso de emisión del archivo de vídeo o audio elegido. Una vez se esta realizando el *streaming* podemos proceder a capturar los flujos de tráfico desde nuestro equipo de captura.

Igual que en el caso anterior nuestra herramienta captura los flujos de tráfico que a continuación son filtrados, por dirección *IP* del emisor, puerto y tipo de tráfico capturado para obtener nuestro archivo de texto listo para ser tratado con un entorno matemático.

4.2.3. Resultados

Se han realizado varias capturas, modificando tanto el protocolo utilizado en la transmisión como el tipo de codificación, y la primera conclusión a la que se ha llegado, es que el tráfico no presenta un comportamiento uniforme, siendo bastante complejo el modelo de tráfico que utiliza.

Para un *streaming* con un protocolo UDP y sin codificación, el tiempo entre tramas obtenido no es constante llegando a variar desde 0,20ms entre algunas tramas hasta 2ms entre otras de ellas. Como podemos observar, para el *streaming* no es fácil obtener a primera vista un modelo de tráfico.

En el *streaming* de vídeo realizado en este caso, el único parámetro que permanece constante es el tamaño del paquete que para un protocolo UDP y sin codificación presenta un tamaño de 1370$bytes$. Esto se debe a que transmitimos información con TS (*Transport Stream*), protocolo que divide la información en *streams* de 188 *bytes* fijos y los paquetes IP pueden contener múltiplos de estos, por lo tanto el mayor tamaño posible sería 1370 *bytes*.

5. Medidas de calidad

La calidad con la que un sistema soporta un nuevo servicio así como la calidad con la que un sistema soporta una o varias comunicaciones simultáneas sin que se presente deterioro en la información son dos medidas que van a ser de interés tanto para una empresa proveedora como para el usuario del servicio. Es por ello que surge la necesidad de desarrollar una herramienta que permita evaluar dichos aspectos de calidad sobre un entorno real.

Para esta sección de medidas de calidad, se utiliza solamente de las tres funcionalidades desarrolladas la opción 2 (generación y captura de tráfico). En este caso se analiza la calidad de nuestro sistema *software* enviando sobre él un tráfico de evaluación que seguirá el modelo introducido por el usuario. Este modelo, puede ser uno de los calculados previamente o alguno conocido por el usuario.

Se van a realizar dos medidas de calidad: la primera de las pruebas realizadas consiste en medir la calidad de un sistema mientras que la segunda obtiene el número de comunicaciones para las cuáles el sistema no presenta deterioro en la información. Para la realización de estas dos medidas se va a utilizar la topología siguiente:

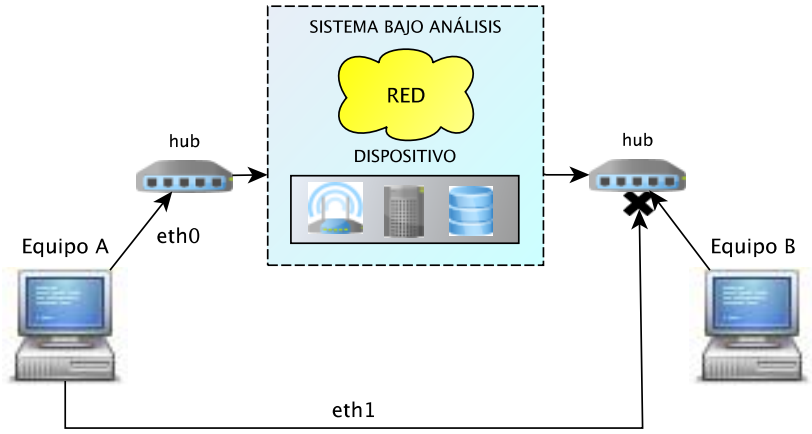


Figura 5.1: Montaje pruebas medida de calidad

Vamos a estudiar dos dispositivos con los que contamos en el laboratorio: un *Switch*

Ethernet que trabaja a una velocidad de 100*Mbps* y dos puntos de acceso (Linksys WAP54G) conectados entre sí mediante un enlace inalámbrico.

El primer paso es comprobar el correcto funcionamiento del *software* desarrollado, comprobando que el tráfico generado sigue el modelo que queríamos obtener. A continuación se detalla el proceso de configuración de las aplicaciones y del escenario, así como los resultados obtenidos.

Las lecturas recomendadas para la compresión de este capítulo [8] [9] se encuentran recogidas en la bibliografía del proyecto.

5.1. Configuración

Una vez que tenemos el montaje del escenario, se debe comenzar con el proceso de configuración del modelo del tráfico que se va a generar, y por lo tanto la configuración del *test* (archivo *XML*). El usuario debe conocer de antemano cuales van a ser los parámetros que va a necesitar para la generación del tráfico de evalución. Estos parámetros se encuentran recogidos en el capítulo 2 del presente proyecto.

La configuración del *test* se puede hacer manualmente o mediante una ayuda que inlcuye nuestro *script* de generación y captura de tráfico. Si se opta por configurarlo manualmente se debe abrir el archivo mediante un editor de textos de los que se disponen en el sistema operativo y proceder a su modificación. Si se configura con la ayuda del *script* el programa pedirá por pantalla los parámetros que se deben introducir mediante teclado y que son necesarios para realizar la generación.

La diferencia principal en la configuración del *test* para las dos medidas de calidad que vamos a realizar es el valor de parámetro *Número de Repeticiones*. Este parámetro es el número de comunicaciones que se van a generar, y por lo tanto será 1 para la primera prueba de calidad, mientras que para la segunda iremos aumentando dicho número para comprobar cuantas comunicaciones puede aguantar nuestro sistema sin presentar deterioro.

Otro de los parámetros importantes a la hora de configurar el *script* es el camino que va a seguir el tráfico, ya que puede ser sólo de ida (*one way*) o de ida y vuelta (*round trip*). Para nuestras pruebas elegeriremos la opción *one way* ya que una de las medidas que queremos obtener es el retardo introducido por el sistema.

Para nuestras pruebas, los flujos IP generados corresponden al modelo del sistema televigilancia obtenido en el capítulo anterior. Se va a utilizar este debido a que presenta un modelo sencillo e uniforme que nos facilitará el análisis. La herramienta de generación de tráfico a utilizar va a ser ETG.

Una vez configurado el *test* debemos crear un camino seguro de intercambio de información entre servidor y cliente. Si es la primera vez que las máquinas que actúan

como servidor y cliente van a compartir información, se debe ejecutar entre ellas un *script* de configuración (configure.sh) que permite compartir las claves entre las dos máquinas de manera segura, debido a que sino no permite el envío de información de una a otra. Este paso se hará de forma automática por la herramienta desarrollada que simplemente pregunta si es la primera vez que las máquinas comparten información.

En este caso, el *software* desarrollado se encuentra ubicado en el equipo que va a actuar como generador del tráfico y como equipo de captura, que es el equipo A. Una vez tenemos el montaje del escenario, está configurado el archivo de generación de tráfico (XML) y las máquinas han compartido las claves, se puede proceder a la generación y captura del tráfico.

5.2. Proceso de generación y captura

Descripción del proceso:
Lanzamo la herramienta y elegimos la opción 2: generación de tráfico. Esta a su vez, lanza el *script* de generación y captura que incialmente nos pedirá que configuremos el *test* tal y como hemos explicado en el punto anterior. Una vez se tiene el archivo XML para la generación del tráfico, el *script* lanza el programa de compartición de claves (configure.sh) entre las máquinas creando un canal seguro entre ellas para el intercambio de información. A continuación se lanza ETG que es la herramienta desarrollada por el grupo que va a permitir generar el tráfico que hemos modelado en el *test*.

Mientras se esta generando el tráfico con la herramienta ETG, debemos capturar tanto a la entrada cómo a la salida del sistema bajo análisis. Gracias a que el equipo que actúa como servidor está conectado tanto en el interfaz de entrada como en el interfaz de salida, y que la herramienta ETG captura en todos los interfaces que tiene conectados (opción *any*) se crea un archivo único en el que obtendremos toda la información que estamos buscando, en el que los paquetes se identifican gracias al número de secuencia.

Una vez obtenido el archivo, este se filtra para generar dos archivos independientes, uno que posee la información del tráfico generado a la entrada del sistema bajo análisis, y otro con la información a la salida. El primer paso es convertir el número de secuencia de hexadecimal a decimal. A continuación se reordena la información para que esté dispuesta por columnas en el siguiente orden: número de secuencia, tiempo de emisión o recepción (según el archivo de captura sea de la entrada o de la salida del sistema) y tamaño del paquete. El último paso es generar los dos archivos independiente.

Con estos dos archivos de información, y mediante un proceso de análisis podemos obtener medidas de calidad del sistema como pueden ser retardos introducidos por el sistema y tasa de pérdida.

5.3. Resultados

1. Calidad del sistema bajo análisis:
Los resultados obtenidos en este apartado tanto para el *Switch Ethernet* cómo para los puntos de acceso no son instructivos ya que al trabajar con un entorno controlado, estos dispostivios estan preparados para trabajar con control de flujo, y por lo tanto no introducen pérdidas ni retardos. Por tanto mediante la utilización de la herramienta, podemos comprobar que la calidad de un servicio al atravesar el sistema es óptima, ya que no se pierde información ni se introducen retardos apreciables.

2. Calidad del sistema para múltiples comunicaciones:
Para esta parte se ha realizado el cálculo teórico del número mínimo y máximo de comunicaciones que puede soportar el sistema antes de que se produzca deterioro en la información. De esta manera tenemos un dato teórico con el que contrastar los resultados obtenidos de manera experimental.

Para el caso de los puntos de acceso, la topología específica sobre las que se han realizado las pruebas es la siguiente:

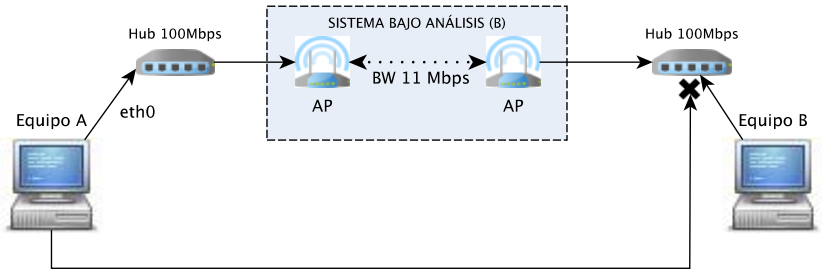


Figura 5.2: Medida de calidad sobre puntos de acceso

Si hacemos un cálculo teórico sobre el número de conexiones de vídeo (sistema de videovigilancia) que el sistema puede soportar, con velocidades de entrada y salida de $100Mbps$, para distintos anchos de banda en el enlace inálambrico y para un modelo de tráfico de vídeo con compresión $4Kbytes$, con un tamaño de paquete 1472 *bytes* y un tiempo de duración de la ráfaga de 40 *ms* se obtienen los siguientes valores:

Se han hecho pruebas de calidad con la herramienta para un $BW{=}11Mbps$ consiguiendo un número máximo de conexiones antes de que se empiece a deteriorar la información de 7 conexiones. Cómo vemos entra en el intervalo de conexiones calculadas de manera teórica.

BW	Valor teórico		
($Mbps$)	mínimo	medio	máximo
1	1	1	1
2	1	1	2
5.5	4	4	5
11	6	7	9
24	9	12	17
54	11	16	27

Tabla 5.1: Valores teóricos de número de conexiones calculados para vídeo

6. Caracterización de entornos

El retardo que introduce un equipo, la tasa de pérdida del mismo y el tamaño del *buffer* del equipo a caracterizar, son parámetros que resultan determinantes a la hora de realizar una comunicación. El tamaño de *buffer* que posee un equipo (como por ejemplo un *router*) suele ser un parámetro no especificado por muchos fabricantes. Dependiendo de este tamaño y de los enlaces del escenario real se pueden producir cuellos de botella que nos hagan perder paquetes si la capacidad de memoria del *buffer* es pequeña, o latencias elevadas si la capacidad del *buffer* es grande.

Por ello, una de las funcionalidades que se han desarrollado sirve para caracterizar entornos reales de manera que podemos obtener retardos, pérdidas y tamaños de *buffer* de los equipos. Esta última prueba consiste en utilizar nuestra herramienta para obtener dichas características, analizando el comportamiento del tráfico que atraviesa el sistema.

La topología utilizada para las pruebas es la mostrada en la Fig. 6.1 en donde el equipo de generación y captura (equipo A) esta conectado tanto a la entrada como a la salida del sistema bajo análisis y en el que utilizamos tanto el *script* de generación y captura de tráfico, como el *script* de caracterización de entornos (opciones 2 y 3 de la herramienta).

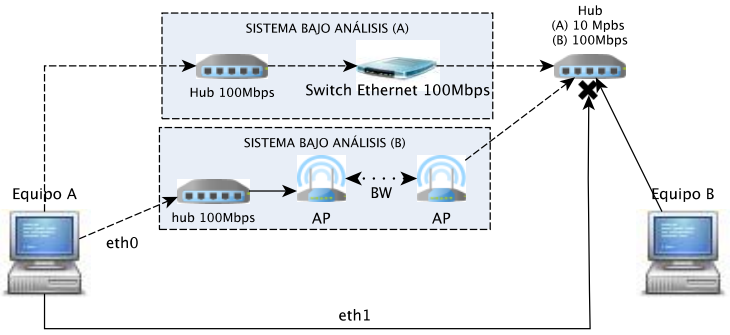


Figura 6.1: Montaje específico pruebas de caracterización de entornos

Para este punto se han hecho pruebas sobre dos sistemas distintos. El primer sistema a caracterizar (A) va a ser un *Switch Ethernet* mientras que el segundo (B) van a ser dos puntos de acesso (*Access Point*) con los que contamos en el laboratorio (Linksys WAP54G).

El Hub de salida del sistema es un Hub de 10*Mbps* para el caso del *Switch Ethernet* ya que al presentar una velocidad en el enlace de salida más baja se consigue saturar el *buffer* del propio *switch* , mientras que para los puntos de acceso contaremos con un Hub tanto a la entrada como a la salida del sistema, siendo tanto el Hub de entrada como el de salida de 100*Mbps* con lo que la comunicación de los puntos de accesso está solamente limitada por el ancho de banda del enlace inalámbrico y para saturar el punto de acceso simplemente se deberá poner un ancho de banda menor a 100*Mbps*.

Con este montaje obtendremos retardos, tasa de pérdida del sistema y el tamaño del *buffer* de los dos sistemas expuestos obteniendo los límites de llenado y vaciado del mismo.

El primer paso a realizar es el montaje del escenario y la comprobación de las distintas conexiones que lo constituyen. A continuación se detalla el proceso de configuración de los dispositivos y aplicaciones.

Las lecturas recomendadas para la compresión de este capítulo [10] [11] se encuentran recogidas en la bibliografía del proyecto.

6.1. Configuración

Una vez que tenemos el montaje de nuestro escenario real y se ha comprobado el correcto funcionamiento de las conexiones, vamos a proceder a la configuración de los distintos dispositivos y aplicaciones con las que contamos en la topología propuesta.

El *Switch Ethernet* debe configurarse de manera que no se realice control del flujo en el mismo. Debemos desactivar esa opción puesto que lo que nosotros queremos es que el *buffer* del dispositivo se sature y eso no lo conseguiremos con el control de flujo activo. Se deben configurar los dos puntos de acceso (AP) de manera que se puedan transmitir datos entre ambos. Para ello, además de configurar sus direcciones *IP* se debe crear entre ellos una red inalámbrica que los conecte (de la cuál variaremos el ancho de la conexión).

Respecto a los otros dispositivos, el equipo A va a ser el encargado de generar el tráfico de evaluación y capturar los flujos de información para su posterior análisis. Este tráfico de evaluación va a ser uniforme y constante, para que sea mucho más fácil de analizar a la hora de caracterizar el entorno. Además, gracias a la herramienta de generación de tráfico *ETG* los flujos de información generados llevan asociado un número de secuencia que nos va a servir para identificar los paquetes a la entrada y a la salida del sistema, lo que nos facilita el análisis.

Tanto la herramienta para la generación de flujos desarrollado por el grupo (ETG) como el *software* desarrollado para este proyecto se encuentran ubicados en el equipo A, donde se almacenan también los resultados obtenidos tanto de la captura de tráfico como los de su posterior análisis.

Cabe destacar que para las pruebas realizadas, hemos contado con los dos equipos en

el laboratorio, pero se están realizando otras medidas dentro del grupo, donde el equipo B puede no encontrarse en la misma ubicación, siendo entonces imposible realizar la conexión directa del interfaz eth1 como captura.

6.2. Proceso de generación, captura y análisis

El proceso de trabajo de esta última prueba del proyecto para ambos sistemas a caracterizar va a ser el siguiente:

1. Generación de tráfico uniforme con la opción 2 de la herramienta.
2. Captura del tráfico con la opción 2 de la herramienta.
3. Análisis del tráfico del sistema a caracterizar con la opción 3 de la herramienta.

Descripción del proceso:

Una vez tenemos el montaje y hemos configurado los dispositivos, lanzamos nuestra herramienta eligiendo la opción 3: Caracterización de entornos. Esta herramienta a su vez lanza la opción 2 de generación y captura del tráfico. El proceso a seguir para la generación y captura es el mismo que el utilizado para medidas de calidad. Una vez se ha acabado con la captura del tráfico de evaluación generado y se han creado los dos archivos con lo que trabajaremos se comienza con el análisis del tráfico para obtener las distintas características del sistema.

Los archivos creados en el paso anterior poseen información de número de secuencia, tiempo de emisión o recepción y tamaño del paquete. Con estos tres campos de información es suficiente para poder obtener distintos parámetros característicos de nuestro sistema, como pueden ser: número de paquetes pérdidos y en consecuencia la tasa de pérdida, tamaño del buffer del sistema, velocidad del enlace a la entrada y la salida, retardos, así como otros.

El siguiente paso a realizar por el *script* de caracterización es la consolidación de los datos en un solo archivo. Esto se ha hecho, debido a que a la hora de trabajar con dos archivos independientes, se aumentaba mucho el tiempo de procesado de los mismos y la capacidad del sistema, por tanto se decidió consolidar los dos archivos en uno solo que cuenta con la siguiente información:

Número de secuencia	Tamaño paquete	Tiempo de emisión	Tiempo de recepción

Gracias a que tenemos los tiempos de emisión y recepción de los paquetes, el cálculo del retardo que introduce el sistema es un paso muy fácil. El programa calcula la resta de

ambos tiempos para cada par de paquetes con el mismo número de secuencia y finalmente calcula la media de todos los paquetes para obtener un retardo medio.

El archivo consolidado ya no cuenta con los paquetes que se han perdido en el sistema bajo análisis, sino que en este archivo sólo tenemos información de aquellos paquete que han salido del sistema. Por lo tanto la manera de calcular la tasa de pérdida es la siguiente: el *script* va comprobando el número de secuencia (primera columna) del archivo consolidado hasta que se produce un salto de más de un número de secuencia. Ahí es donde tenemos los paquetes perdidos, por lo que sólo falta hacer una operación matemática.

Para la obtención del tamaño del *buffer* se han desarrollado 3 métodos en el grupo (de los cuáles se utilizan dos en el presente proyecto):

- *Método 1:* Algoritmo que cuenta el número de paquetes que hay en el *buffer* en el momento de la entrada de un nuevo paquete, basándose en los tiempos de entrada-salida.

- *Método 2:* Algoritmo que estima el número de paquetes teniendo en cuenta el retardo de un paquete en el *buffer*, obtenido con base al tamaño del paquete y a la velocidad de salida.

- *Método 3:* Algoritmo que estima las velocidades de llenado y vaciado del *buffer* con base en las relaciones temporales de los paquetes recibidos con respecto a la estimación del tiempo en el que fueron enviados. Para este algoritmo sólo se necesita el fichero de salida.

En este proyecto y para estas pruebas se han utilizado los dos primeros, ya que el tercero se usa cuando uno de dos equipos se encuentra en otra ubicación (remoto).

Una vez se ha calculado el tamaño del *buffer* la propia herramienta genera una gráfica en la que podemos apreciar el mecanismo de llenado y vaciado de*buffer* caracterizado.

6.3. Resultados

Se han hecho pruebas para tres tráficos de evaluación de tamaños de paquete distintos: 1500, 800 y 200 *bytes* para el caso del *Switch Ethernet* y tamaños: 1300, 800 y 300 *bytes* para el caso de los puntos de acceso. Además, el ancho de banda utilizado en el enlace inalámbrico para las pruebas en el segundo sistema bajo análisis corresponde a BW=11*Mbps* consiguiendo así saturar el dispositivo.

Se ha conluido que ambos sistemas poseen un *buffer* que presentan una gráfica de llenado y vaciado en forma de diente de sierra, por lo tanto ambos presentan un límite

Sistema bajo análisis	Límite superior	Límite inferior
Switch Ethernet	116	86
Access Point	56	30

Tabla 6.1: Límites de llenado y vaciado de los *buffer* estudiados

superior de llenado y un límite inferior de vaciado, descartándo paquetes cuando el *buffer* lleno. Estos límites pueden consultarse en la siguiente tabla:

A continuación podemos ver el mecanismo de llenado y vaciado del *Switch Ethernet* en donde se aprecian los límites superior e inferior de llenado y vaciado del *buffer*. Esta gráfica es la generada por nuestra herramienta, y debido a la cantidad de paquetes capturados, no se puede ver con tanto detalle cómo esperábamos. Además, se han generado varias ráfagas y se ha capturado y graficado el tiempo entre dos de ellas. Podemos ver que la velocidad de llenado del buffer en este caso es mayor que la velocidad de vaciado:

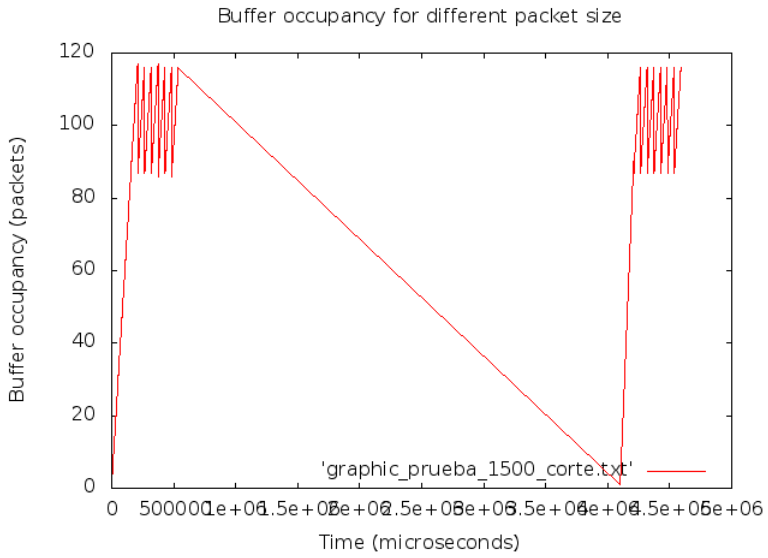


Figura 6.2: Ocupación del *buffer* para tráfico uniforme 1500*bytes vs* Tiempo (*ms*)

Queremos representar el archivo anterior (graphic_prueba_1500.txt) con un poco más de detalle. Gracias a que nuestra herramienta genera una versión del archivo en .CSV (*Comma Separated Value)* se puede tratar con un entorno matemático y obtener una imagen como la que presentamos a continuación:

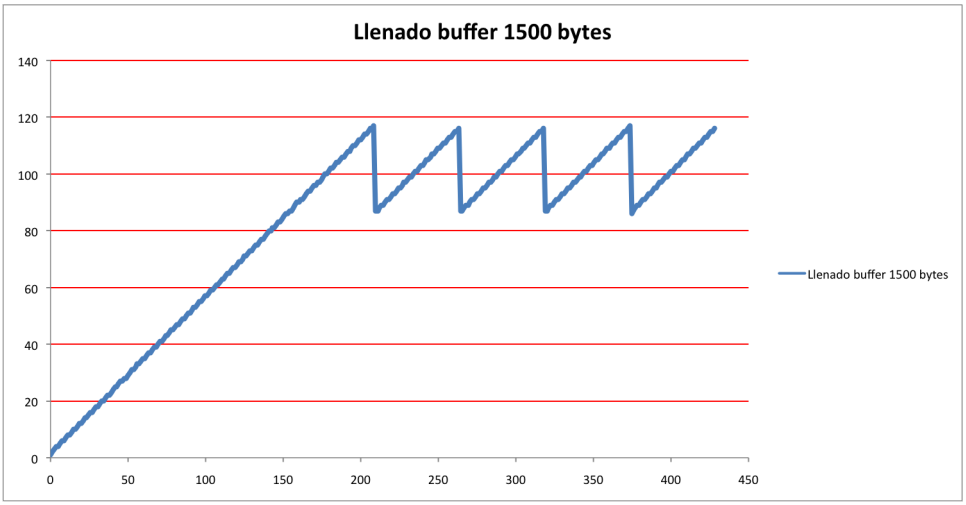


Figura 6.3: Detalle ocupación del *buffer* para tráfico uniforme 1500*bytes vs* Tiempo (*ms*)

El mecanismo de llenado así como los límites superior e inferior es exactamente igual para los tres tamaños de paquetes en ambos sistemas, por lo que se concluye que el *buffer* de ambos dispositivos esta definido a nivel de paquetes y no de *bytes*.

El tamaño del paquete no influye en los límites de llenado y vaciado del *buffer* al estar éste definido por número de paquetes y no por *bytes* pero sí que influyen en la velocidad con la que se llena o se vacía dicho *buffer*. En la gráfica siguiente, podemos comprobar como para dos tráficos con tamaños distintos de paquete (1500 *vs* 800) el *Switch Ethernet* posee un *buffer* con límites superior e inferior de 116 y 86 para ambos, pero podemos obrservar que el tiempo que tarda en llenarse y vaciarse no es el mismo, llenándose el *buffer* mucho antes para el tamaño de paquete de 800 *bytes*.

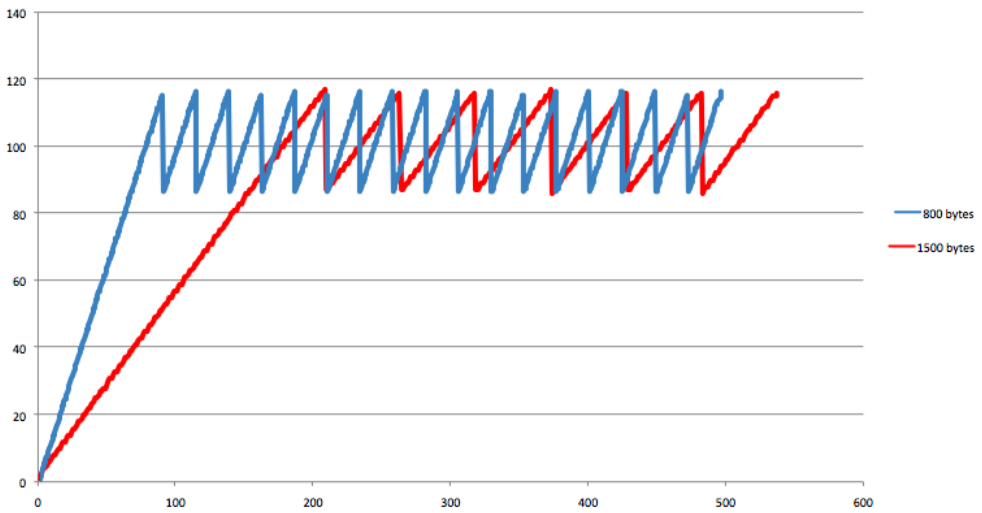


Figura 6.4: Velocidad de llenado y vaciado del *buffer* del *Switch Ethernet*

Para el punto de acceso el ancho de banda de la transmisión va a jugar un papel importante, no en los límites superior e inferior del *buffer*, puesto que al igual que para el *Switch* permanecen constantes para cualquier ancho de banda y tamaño de paquete, pero si lo hará en la velocidad de llenado y vaciado del mismo y para un mismo tamaño de paquete.

7. Conclusiones y Líneas futuras

7.1. Conclusiones

La metodología de trabajo propuesta, nos ha permitido realizar una serie de pruebas sobre diversos flujos de tráfico. Con todo ello hemos obtenido resultados de comportamiento del tráfico en la red, aspectos de calidad de entornos reales y la identificación de parámetros característicos de dichos entornos.

Además se ha buscado desarrollar una herramienta de trabajo que facilite al usuario la captura y análisis de distintos flujos de información automatizando, en la medida de lo posible, los procesos necesarios.

Los objetivos presentados al comienzo del proyecto se fueron modificando durante su desarrollo para adecuarlos al trabajo que se estaba realizando y a los resultados que se estaban obteniendo.

De esta manera, se ha conseguido una herramienta que permite capturar el tráfico de una aplicación o servicio en la red de forma automatizada y desatendida para el usuario. A modo de ejemplo, cómo resultado de las pruebas hemos obtenido el modelo que presentan los flujos de tráfico para varias aplicaciones multimedia, y extrayendo conclusiones sobre los mismos, como son la mayor sencillez de los modelos para tráfico de videovigilancia frente a la complejidad presentada para el sistema de streaming.

Se ha automatizado también el proceso de generación de flujos de tráfico previamente conocidos, utilizándolos para realizar pruebas de calidad tanto de aplicaciones como de sistemas. Así se ha comprobado la calidad que presenta una aplicación al atravesar un sistema y el número de aplicaciones que presenta dicho sistema antes de provocar deterioro en la información.

Por último y gracias a estas pruebas previas, se ha desarrollado una herramieta que permite al usuario caracterizar entornos reales, basándose en el análisis del tráfico que atraviesa el sistema y que previamente ha sido capturado y generado. Con estas pruebas se han obtenido características como retardos, pérdidas y tamaños de *buffer*, posibilitando en todo momento la automatización del proceso.

Como en los casos anteriores, a modo de ejemplo se concluyó que el mecanismo de llenado y vaciado tanto del punto de acceso WIFI, cómo del *switch ethernet* utilizado es de tipo diente de sierra con unos límites superiores e inferiores constantes en ambos casos y con una velocidad de llenado mucho mayor que la velocidad de vaciado del *buffer*.

7.2. Líneas futuras

Como se ha dicho en el apartado de conclusiones la finalización del proyecto ha permitido el desarrollo de una herramienta de análisis para flujos de tráfico. En el futuro, podrían ser llevadas a cabo diversas mejoras y ampliación de las funcionalidades de dicha herramienta. Algunas mejoras se nombran a continuación:

1. Se podría añadir la funcionalidad a la herramienta que permitiera la obtención de forma automatizada de los modelos que siguen los flujos de información analizados; ya que esta tarea se salía de los objetivos del proyecto.
2. Se podría mejorar la funcionalidad de generación y captura de tráfico de manera que no fuera necesario disponer de una conexión física en el equipo de salida del sistema a caracterizar pudiendo encontrarse en otra ubicación. Actualmente se esta trabajando en el grupo en esta mejora.
3. Se podrían realizar medidas de calidad de los dispositivos analizados utilizando un tercer equipo que nos permita introducir pérdidas en el sistema.
4. Podrían añadirse otros parámetros a capturar en el *script* de caracterización de entornos, ya que ahora mismo sólo se dispone de retardos, tasa de pérdida y tamaño del *buffer*.

Además, las pruebas realizadas se han hecho para escenarios comunes y para equipos y aplicaciones de las que disponíamos en el laboratorio. Sería interesante probar la herramienta en otros escenarios y para otras aplicaciones multimedia, así como caracterizar los *buffers* de otros equipos y fabricantes que debido a la limitación de tiempo no se ha podido realizar.

Decir por último que la herramienta desarrollada va a ser utilizada por el grupo para realizar capturas y medidas de calidad en el sistema de videoconferencia de una manera más exhaustiva y específica pues se estan realizando otros estudios sobre dicho sistema Vydio.

Bibliografía

[1] L.A. Casadesus Pazos, J.Fernandez Navajas, J. Ruiz Mas, J.M. Saldaña Medina, J.I. Aznar Baranda, y E. Viruete Navarro, *Herramienta para automatización de medidas de tiempo real extremo a extremo,* Actas del XXVI Simposium Nacional de la Union cientifica Internacional de Radio (URSI 2011), Leganés (España). ISBN 9788493393458. Septiembre 2011.

[2] W. Almesberger, Linux Network Traffic Control - Implementation Overview.

[3] STANIC, Milan P. tc- traffic control - Linux QoS control tool.

[4] A. Keller, *Manual tc Packet Filtering and netem* ETH Zurich, July 20, 2006.

[5] C. Villamizar and C. Song , *High performance tcp in ansnet,* SIGCOMM Comput Commun. Rev., vol. 24, pp.45-60, October 1994. [Online] Avaible: http://doi.acm.org/10.1145/205511.205520

[6] A. Golaup, H. Aghvami *A multimedia traffic modeling framework for simulation-based performance evaluation studies* Computer Networks, vol. 50, no. 12, pp. 2071-2087, 2006, network Modelling and Simulation.

[7] http://www.videolan.org/.

[8] IEEE Standard for Information Tecchnology. Wireless lan medium access control (mac) and physical layer (phy) specifications. Technical report, IEEE Computer Society, 2007.

[9] Telecomunication Standardization Sector of ITU. G.729 coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (cs-acelp). Technical report, International Telecomunication Union, 2007.

[10] G. Appenzeller, I. Keslassy, N. McKeown emphSizing Router Buffers, SIGCOMM 2004. [Online] yuba.stanford.edu/ nickm/papers/sigcomm2004.ps

[11] A Vishwanath, V Sivaraman, and G N Rouskas. Considerations for sizing buffers in optical packet switched networks. IEEE INFOCOM 2009 The 28th Conference on Computer Communications, pages 1323−1331, 2009.

A. Acrónimos y términos

A.1. Acrónimos

GNU GPL (*GNU General Public License*).
NETEM *Network Emulator* es un emulador de red.
CSV (*Comma Separated Value*). **TC** (*Traffic Control*).
Hub Concentrador o ethernet hub, un dispositivo para compartir una red de datos o de puertos USB de un ordenador.
MOS *Mean Opinion Score* (Nota Media de Resultado de Opinión).
jitter (variación de retardo). Es un término que se refiere al nivel de variación de retardo que introduce una red. Una red con variación 0 tarda exactamente lo mismo en transferir cada paquete de información, mientras que una red con variación de retardo alta tarda mucho más tiempo en entregar algunos paquetes que en entregar otros. La variación de retardo es importante cuando se envía audio o vídeo, que deben llegar a intervalos regulares si se quieren evitar desajustes o sonidos inintelegibles.
Sniffer un analizador de paquetes es un programa de captura de las tramas de una red de computadoras.
router (encaminador, enrutador). Dispositivo que distribuye tráfico entre redes. La decisión sobre a donde enviar los datos se realiza en base a información de nivel de red y tablas de direccionamiento. Es el nodo básico de una red IP.
QoS *Quality of Service* (Calidad de Servicio)
RTP *Real Time Protoco*l (Protocolo de Tiempo Real).
TCP *Transmission Control Protocol* (Protocolo de Control de Trasmisión).
UDP *User Datagram Protocol* (Protocolo de Datagramas de Usuario).

A.2. Términos

codec Algoritmo software usado para comprimir/descomprimir señales de voz o audio.
streaming Consiste en la distribución de audio o video por Internet, esta palabra hace referencia a una transmisión en forma continua de envío de voz por redes de conmutación de paquetes utilizando TCP/IP, tales cómo Internet.
Ancho de banda Capacidad de transmisión de datos que tiene un medio determinado, generalmente cuantificado según el número de bits que se transmiten en un segundo.

B. Desarrollo

Aunque se han diferenciado 5 fases principales en este proyecto de fin de carrera, en la figura B.1, se muestra el diagrama de Gantt correspondiente al trabajo realizado en este proyecto, en el que se pueden observar con más detalle la duración de las tareas realizadas.

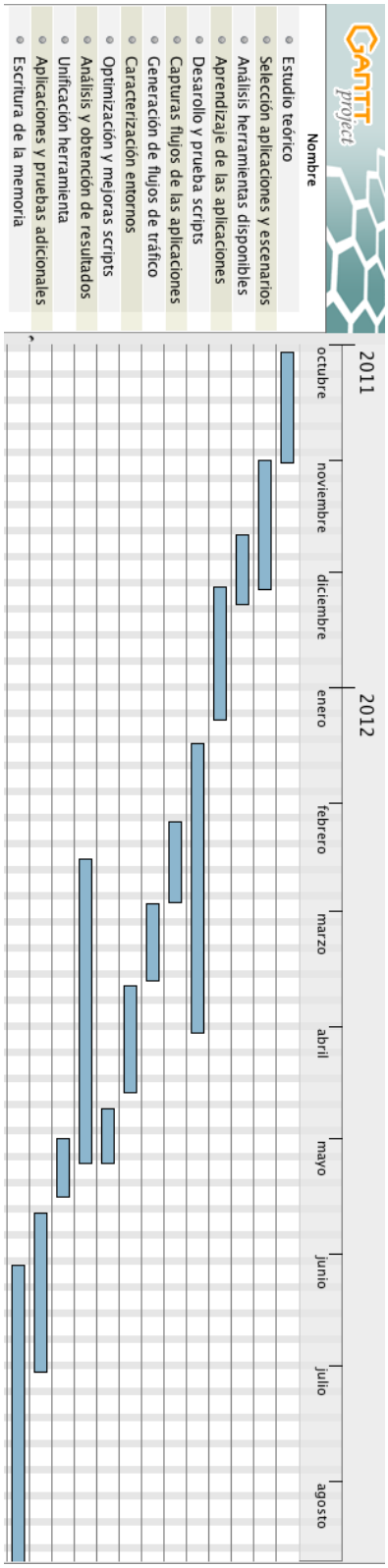


Figura B.1: Diagrama de Gantt del proyecto.

C. Configuración de las aplicaciones

A continuación se incluyen con más detalle algunos de los pasos que se ha realizado a la hora de configurar las aplicaciones de las que hemos capturado los flujos de tráfico y de las que han estudido las distribuciones que presentan los flujos de tráfico que las componen.

C.1. AXIS 2120

Accedemos a la cámara a través de un navegador tradicional desde el equipo donde vamos a recibir las imágenes. Para estas pruebas tenemos la cámara ubicada en la dirección *IP*: http://192.168.7.5:7453/ por lo tanto iremos al navegador y escribiremos la dirección de la cámara.

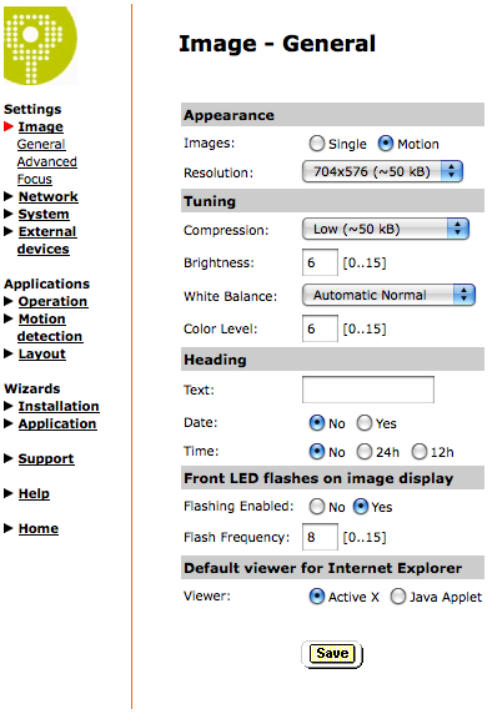


Figura C.1: Menu principal de configuración de la AXIS 2120

Una vez que hemos accedido de forma remota a ella, en la pantalla principal elegiremos la opción de *Settings* en donde vamos a configurar los parámetros que van a definir la imagen asi como el ancho de banda de la cámara.

En el menu desplegable de la izquierda elegiremos primeramente la opción *Image→General* como vemos en la Fig.C.1 en donde aparacen los parámetros de configuración del tipo de imagen a enviar, como son tipo de imagen (*single o motion*, tamaño de la imagen (*resolution*) y compresión elegida.

Del resto de parámetros que la cámara *Ip* da opción a modificar, nosotros no vamos a trabajar con ellos así que los dejaremos tal y como muestra la imagen anterior. A continuación en el menu desplegable de la izquierda elegimos la opción *Network→TCP/IP* en donde aparecen los parámetros de configuración de la red de la cámara. En este menu solamente vamos a trabajar con la opción de ancho de banda:

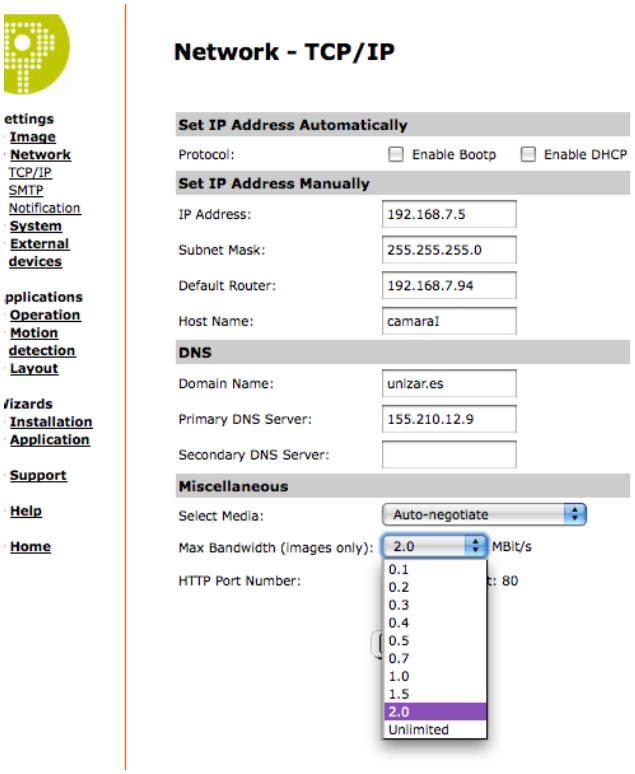


Figura C.2: Menú de configuración de los parámetros de la red

Una vez hemos configurado todos los parámetros de nuestra primera captura, guardamos las opciones y el menu volverá a la ventana principal en donde obtendremos las imágenes que la cámara esta capturando.

Figura C.3: Imagen enviada por la cámara IP

Ahora ya tenemos las cámara configurada para comenzar con el proceso de captura y el estudio de los flujos de tráfico.

C.2. VLC

Para la prueba de VLC debemos tener el *software* instalado en ambos equipos para realizar el *streaming* de audio o video.

Equipo Servidor:

Accedemos a la aplicación VLC desde el equipo que va a actuar como servidor y desde el cuál emitiremos el *streaming*. El archivo que vamos a enviar debe estar ubicado en este equipo en una ubicación conocida.

Una vez hemos lanzado la aplicación, nos aparece la ventana principal del reproductor VLC. Vamos a proceder a la emisión del archivo de *streaming*. Para ello iremos a la opción de menu *Medio*→ *Emisión*.

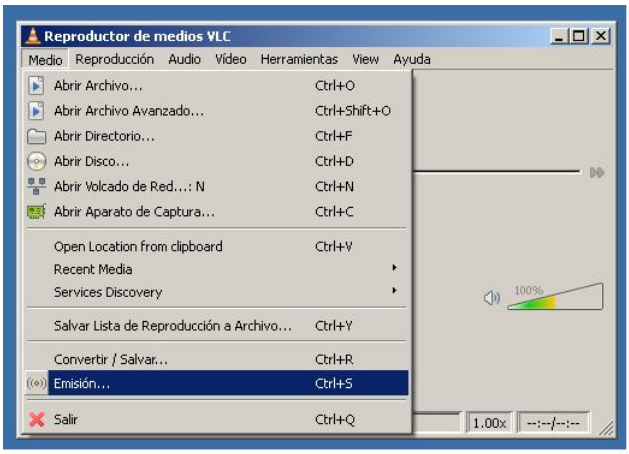


Figura C.4: Configuración de la emisión del *streaming*. Paso 1

Ahora la aplicación pedirá la ruta donde se encuentra el archivo con el que vamos a realizar el *streaming*. Una vez le hemos indicado el archivo le damos a emitir. Los pasos que vienen a continuación forman parte del proceso de configuración de la aplicación. Vamos a introducir el tipo de protocolo elegido para la transmisión así como la codificación elegida tal y como se ve en la siguiente imagen:

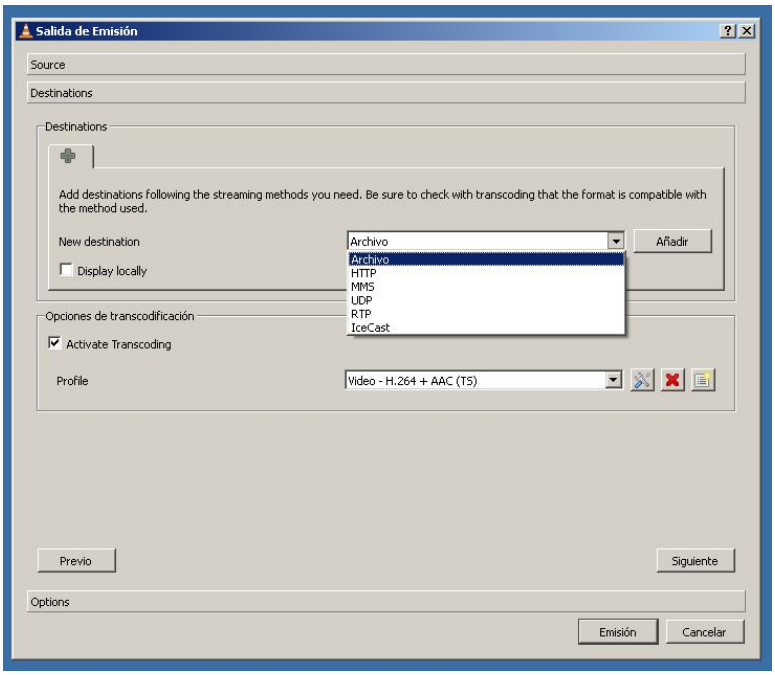


Figura C.5: Configuración de la emisión del *streaming*. Paso 2

La opción *Display locally* nos permite reproducir el archivo en local, pudiendo comprobar que se esta emitiendo el archivo deseado y de manera correcta. Vamos a elegir el tipo de protocolo a utilizar de la lista de protocolos posibles. Una vez elegido debemos darle al boton añadir pues nos pedirá información distinta para cada protocolo para poder realizar el *streaming*.

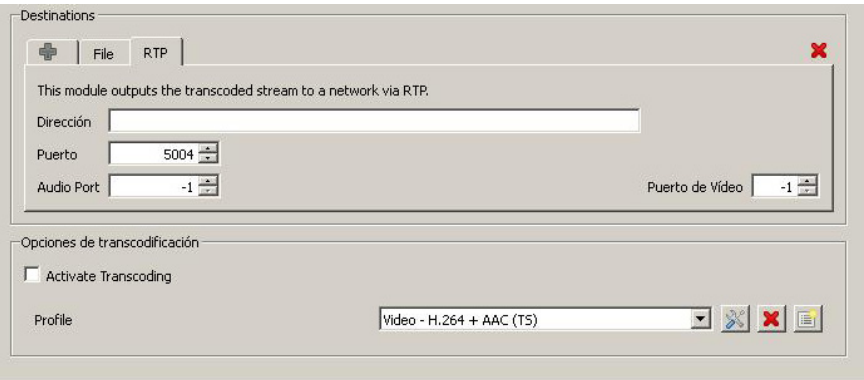


Figura C.6: Configuración de la emisión *streaming*. Paso 3

Para el ejemplo de la imagen se ha elegido un protocolo de transmisión RTP. Debemos introducir la dirección de la máquina que va a actuar como cliente y el puerto de la misma (en nuestro caso hemos utilizado un puerto situado entre el 5000 y 8000). Además debemos elegir la codificación utilizada de entre las existentes. En nuestro caso, no hemos activado la codificacón porque queremos obtener los flujos de información del *streaming* sin ninguna codificación de los mismos.

Cuando hemos configurado los parámetros del *streaming* le damos a emisión. Por último, aparece una ventana donde aparecen los parámetros configurados en modo comando. Antes de finalizar y darle a emitir el archivo seleccionado debemos configurar y lanzar la recepción en el equipo que actúa como cliente.

Equipo Cliente:
Este paso es más rapido y sencillo pues sólo debemos lanzar la recepción del *streaming* en nuestro equipo. Para ello, una vez lanzada la aplicación VLC en el equipo, iremos de nuevo al menu principal y elegiremos la opción *Medio→Abrir Volcado de Red...:N*

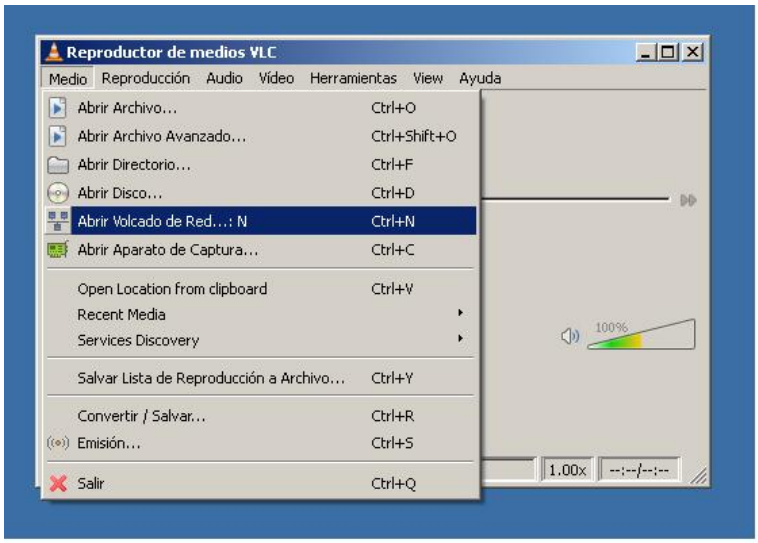


Figura C.7: Configuración de la recepción del *streaming*. Paso 1

Ahora nos aparece una ventana en la que tenemos que introducir los mismos parámetros utilizados para configurar la emisión. Por lo tanto elegiremos como protocolo usado RTP, pero en este caso no debemos introducir dirección ni puerto pues es este equipo el que esta recibiendo. A continuación le damos al botón Abrir, y ya tenemos el equipo cliente preparado para recibir el *streaming*, por lo tanto ya sólo debemos volver al equipo servidor y darle al botón emisión.

D. Scripts

D.1. *Script* de captura de tráfico (captura.sh)

```
echo “PROCESO DE CAPTURA DE DATOS ”
echo “Cuanto tiempo quiere que duren las capturas (en minutos)?”
read TIEMPO
let A=$TIEMPO
let B=60
let C=$A*$B
echo “Cuantas capturas quiere realizar?”
read CAPTURAS
echo “Cuanto tiempo quiere que pase entre cada captura (0 en caso de continuas)?”
read ESPERA
echo “Escriba la direccion de la fuente”
read DIRECCION
echo “La direccion es del tipo X.X.X.X: y/n?”
read RESPUESTA
echo “Escriba el puerto de la dirección fuente: ”
read PUERTO
echo “Que interfaz quiere capturar (eth0,eth1,...)?”
read INTERFAZ
case $RESPUESTA in
        Y|y)
        echo ‘PUEDE CONTINUAR CON EL PROCESO”
        ;;
        n|N)
        echo “NO ELIGIO NINGUNA OPCION VALIDA”
        exit
        ;;
esac
echo “Elija el protocolo usado: TCP, ICMP, IP, RTP, UDP”
read PROTOCOLO
case $PROTOCOLO in
```

```
                icmp|ICMP)
                ;;
                ip|IP)
                ;;
                tcp|TCP)
                ;;
                rtp|RTP)
                ;;
                udp|UDP)
                ;;
                *) echo "SE HA EQUIVOCADO"
                exit ;;
esac
k=0
while [ $k -lt $CAPTURAS ]; do
                tcpdump -i $INTERFAZ -w programawk_$k&
                sleep $C
                rm ./fichproc
                ps aux | grep "tcpdump"|cut -b 10-14 > ./fichproc
                fich="./fichproc"
                read PROC<$fich
                kill -15 $PROC
                cp programawk_$k ./captures
                k=$(($k+1))
                sleep $ESPERA
done
rm programawk*
cd ./captures
echo "Procesado de las capturas"
for x in `ls`
        do
                tcpdump -i $INTERFAZ src $DIRECCION and src port $PUERTO and ip
proto
$PROTOCOLO -r $x>$x.txt
                awk '{print $1,$12,$14}' $x.txt > final1.txt
                tr -d : <final1.txt>final2.txt
                tr -d , <final2.txt>final3.txt
                tr -d . <final3.txt>$x.txt
                rm $x
done
rm final*
clear
echo "GENERACION GRAFICAS JPEG "
echo "set terminal png"> graphic.gnuplot
```

```
echo “set autoscale # scale axes automatically”>> graphic.gnuplot
echo “unset log # remove any log-scaling”>> graphic.gnuplot
echo “unset label # remove any previous labels”>> graphic.gnuplot
echo “set xtic auto # set xtics automatically”>> graphic.gnuplot
echo “set ytic auto # set ytics automatically”>> graphic.gnuplot
echo “set output ‘salida.png’ ”>> graphic.gnuplot
echo “set key bottom right nobox”>> graphic.gnuplot
echo “set title ‘Modelado de trafico IP’ ”>> graphic.gnuplot
echo “set xlabel ‘Time (microseconds)’ ”>> graphic.gnuplot
echo “set ylabel ‘Tamaño (packets)’ ”>> graphic.gnuplot
echo “plot ‘pincho1.txt’ ”w impulses >> graphic.gnuplot
gnuplot < graphic.gnuplot
```

D.2. *Script* de generación y captura (generacion.sh)

```
# Compartición de claves # Este proceso solo debe realizarse una vez asi que
preguntara al usuario si es la #primera vez y compartira las claves, sino NO
read -p “Es la primera vez que va a generar trafico de evaluacion (si/no)?”
RESPUESTA_CONFIGURE
case $RESPUESTA_CONFIGURE in
        si|S|s|S)
        read -p “Remote username:” USERNAME
        read -p “Remote ip:” REMOTEIP
        ./configure $USERNAME $REMOTEIP ;;
        no|NO|n|N)
        ;;
esac
mkdir capturas_generacion
# Configuracion del XML
echo “A continuacion debe introducir los datos de generacion de los flujos”
read -p “Nombre del proyecto: ”NOMBRE
read -p “Nombre del test: ”NOMBRE_TEST
read -p “Fecha de Inicio, en formato DD/MM/YYYY HH:MM:SS: ”FECHA_INICIO
read -p “Fecha de Fin, en formato DD/MM/YYYY HH:MM:SS: ”FECHA_FIN
read -p “Tamaño de la cabecera (en bytes): ” CABECERA
read -p “Tiempo de espera antes iniciar (en segundos): ” ESPERA
read -p “Local Host, en formato XXX.XXX.XXX.XXX: ”HOST
read -p “Local Interface (eth0,eth1...): ” INTERFAZ
read -p “Local Port: ”PORT_LOCAL
read -p “Remote Host: ” REMOTO
read -p “Remote Interface” INTERFAZ_REMOTO
read -p “Remote Port: ”PORT_REMOTO
```

read -p “Frecuencia (tiempo en segundos): ”FRECUENCIA
read -p “Numero de Rafagas: ”NUM_RAFAGAS
read -p “Tiempo entre rafagas (en segundos): ”TIME_RAFAGAS
read -p “Numero de paquetes por rafaga: ”PAQUETES
read -p “Tiempo entre paquetes (en segundos): ”TIME_PAQUETES
read -p “Tamaño del paquete: ”TAMANO_PAQUETE
read -p “Trafico de ida (pulse 0) o trafico de ida y vuelta (pulse 1): ”TIPO
read -p “Numero de repeticiones: ” REPETICIONES
echo “Ahora debe introducir los datos de la captura”

read -p “Numero de capturas: ” CAPTURAS
read -p “Cuanto tiempo quiere capturar (en segundos): ”TIEMPO_CAPTURA
read -p “Donde quiere capturar el trfico: 0 destino, 1 origen, 2 ambos:”DONDE
read -p “Cuanto tiempo de guarda quiere aplicar a la captura?”TIEMPO_GUARDA
Proceso de generación del archivo XML con el que generaremos el tráfico de evaluación
echo “<test>”> test.xml
echo “<description>”>> test.xml
echo “<id> $NOMBRE</id>< −− id −− >”>> test.xml
echo “<text>$NOMBRE_TEST</text><! − − Test desciption −− >”>> test.xml
echo “<beginningDate>$FECHA_INICIO</beginningDate> <! − − “DD/MM/YYYY HH:mm:ss”−− >”>> test.xml
echo “<endDate>$FECHA_FIN</endDate> ¡−− “DD/MM/YYYY HH:mm:ss”−− >”>> test.xml
echo “<headerSize>$CABECERA</headerSize>< −− Tamaño del header del paquete de los paquetes, para calcular ancho de banda −− >”>> test.xml
echo “</description>”>> test.xml
echo “<stream>”>> test.xml
echo “<beginWait>$ESPERA</beginWait><! − − time in seconds. iniciar este tiempo despues de la hora de inicio −− > ” >> test.xml
echo “<localHost>$HOST</localHost>”>> test.xml
echo “<localInterface>$INTERFAZ</localInterface>”>> test.xml
echo “<localPort>$PORT_LOCAL </localPort>”>> test.xml
echo “<remoteHost>$REMOTO</remoteHost>”>> test.xml
echo “<remoteInterface>$INTERFAZ</remoteInterface>”>> test.xml
echo “<remotePort>$PORT_REMOTO</remotePort>”>> test.xml
echo “<frequency>$FRECUENCIA</frequency><! − − time in seconds. 0 just once and stop −− >”>> test.xml
echo “<numberOfBursts>$NUM_RAFAGAS</numberOfBursts>”>> test.xml
echo “<timeBetweenBursts>$TIME_RAFAGAS</timeBetweenBursts><! − − time in microseconds −− >”>> test.xml
echo “<numberOfPackets>$PAQUETES</numberOfPackets>”>> test.xml
echo “<timeBetweenPackets>$TIME_PAQUETES</timeBetweenPackets><! − − time in microseconds −− >”>> test.xml

```
echo "<packetsSize>$TAMANO_PAQUETE</packetsSize><! − − Tamaño del payload del paquete −− >">> test.xml
echo "<rtt>$TIPO</rtt><! − − Indica si el trafico va a ser capturado en el emisor. 1 si (rtt). 0 no (one way).−− >">> test.xml
echo "<file></file><! − − nombre del fichero de tamaños y tiempos, dejarlo vacio si no es necesario −− >">> test.xml
echo "<burstFile></burstFile¿">> test.xml
echo "<numberOfRepetitions>$REPETICIONES¡/numberOfRepetitions¿¡!– repeat this communication. 0 just one thread –¿">> test.xml
echo "<codec></codec><! − − codec en caso de audio. Admite G729a y G711, dejar en blanco si no es audio −− >">> test.xml
echo "<dejitterBufferDelay></dejitterBufferDelay><! − − en caso de audio −− >">> test.xml
echo "</stream>">> test.xml
echo "<capture>">> test.xml
echo "<localHost>$LOCAL_HOST</localHost>">> test.xml
echo "<remoteHost>$REMOTE_HOST</remoteHost>">> test.xml
echo "<frequency>$FRECUENCIA< /frequency><! − − time in seconds. 0 just once and stop −− >">> test.xml
echo "<timeBetweenBursts>$TIME_RAFAGAS</timeBetweenBursts><! − − time in seconds −− >">> test.xml
echo "<numberOfCaptures>$CAPTURAS< /numberOfCaptures><! − − number of captures −− >">> test.xml echo "<time>$TIEMPO_CAPTURA</time><! − − time to capture in seconds –¿»¿test.xml
echo "<file></file><! − − tiempo a capturar, frecuencia −− >">> test.xml
echo "<capture>$DONDE</capture><! − − Indica donde el trafico va a ser capturado. O destino, 1 origen, 2 ambos extremos−− >">> test.xml
echo "</capture>">> test.xml
echo "</test¿">> test.xml
# Lanzamos el ETG, que generara tráfico y me hara una captura del origen
./etg test.xml
tcpdump -i eth0
cp D* ./capturas_generacion
# Ahora se ha generado un archivo donde tendremos toda la información empieza por D
# y lo copiamos en la carpeta de resultados
cd ./capturas_generacion
for x in 'ls' do tcpdump -r $x -nn -tt -x dst $REMOTO and dst port $PORT_REMOTO > pincho1.txt
# NUMERO DE SECUENCIA EN HEXADECIMAL #
# Aqui lo que vamos a obtener es el numero de secuencia en hexadecimal de la trama obtenida de la captura para el trafico de entrada al buffer awk '
BEGIN {
    FS=" "
    cont=1
```

```
}
{
if (cont==1) {
    printf $1 “\t”
    printf $9 “\t”
}
if (cont==3) {
    printf $9
}
if (cont==4) {
    printf $2
    printf $3 “\ n”
}
cont++
if (cont==7) {
    cont=1
}
}
’ pincho1.txt > entrada.txt
# swap and erasing ( size ) and erasing .
# Obtenemos un fichero del trafico a la entrada del buffer que tendra en columnas:
# Columna 1 Columna 2 Columna 3
# Numero de secuencia (hexadecimal) Tiempo origen Tamaño
awk ’{printf $3 “\t”$1 “\t”$2 “\n”}’ entrada.txt > temp.txt
tr -d . <temp.txt>temp2.txt
sed -i “s/)//g”temp2.txt
sed -i “s/(//g”temp2.txt
cp temp2.txt entrada1.txt
rm temp.txt
rm temp2.txt
# NUMERO DE SECUENCIA EN DECIMAL #
# Obtenemos un fichero del trafico a la entrada del buffer que tendra en columnas:
# Columna 1 Columna 2 Columna 3
# Numero de secuencia (decimal) Tiempo origen Tamaño
awk ’
BEGIN {
    FS=“ ”
    pos=1
}
{
while (pos<13) {
    if (substr($pos,1) == “3”) {
            pos=pos+1
            printf $pos
```

```
            pos=pos+1
}
    if (substr($pos,1) == "2") {
            pos=13
}
}
printf "\t"
printf $2 "\t"$3 "\n"
pos=1
}
' entrada1.txt > entradaNStemp.txt
# CREO ARCHIVO FINAL ENTRADA #
awk '
BEGIN{ }
{
getline a < "entradaNStemp.txt"
printf $2 "\t"$3 "\t"a "\n"
}
' entrada1.txt > entrada2.txt
# Ahora tengo el archivo con información de entrada y salida en uno lo tengo que
desglosar en dos
# archivo de entrada
awk '
BEGIN {
igual=0
}
while ($1=igual)
printf $1 "\t"$2 "\t"$3 "\n"
igual++
}
}' entrada2.txt > entradaNS.txt
# archivo de salida
awk '
BEGIN{
igual=0
}
while ($1=igual){
igual++
}
if ($1==igual){
printf $1 "\t"$2 "\t"$3 "\n"
}
}' entrada2.txt > salidaNS.txt
}' entrada2.txt ¿salidaNS.txt
```

D.3. *Script* de caracterización (entornos.sh)

ejecuta_todo_1.sh

```
cd ./generacion
./generacion.sh
echo “Done”
cd ..
cd ./entornos
cd ./captures_dual/
echo “Proccessing dual captures”
for x in ‘ls entrada*‘
        do
                ../proceso_awk.sh $dst_ip_pub $puerto $x filtrado_$x
                cp $x.csv ../results_dual
                cp $x.txt ../results_dual
                cp graphic_$x.csv ../results_dual
done
rm *.csv
rm *.txt
echo “Done”
cd ..
cd ./results_dual
echo “Proccessing dual graphics”
for x in ‘ls graphic_*.csv‘
        do
                cp $x $x.txt
                sed -i “s/,/ /g”$x.txt
                echo “set terminal png”> graphic.gnuplot
                echo “set autoscale # scale axes automatically”> > graphic.gnuplot
                echo “unset log # remove any log-scaling”> > graphic.gnuplot
                echo “unset label # remove any previous labels”> > graphic.gnuplot
                echo “set xtic auto # set xtics automatically”> > graphic.gnuplot
                echo “set ytic auto # set ytics automatically”> > graphic.gnuplot
                echo “set output ’$x.png’”> > graphic.gnuplot
                echo “set key bottom right nobox”> > graphic.gnuplot
                echo “set title ’Buffer occupancy for different packet size’”> >
graphic.gnuplot
                echo “set xlabel ’Time (microseconds)’”> > graphic.gnuplot
                echo “set ylabel ’Buffer occupancy (packets)’”> > graphic.gnuplot
                echo “plot ’$x.txt’ w lines”> > graphic.gnuplot
                gnuplot < graphic.gnuplot
                rm $x.txt graphic.gnuplot
done
```

```
echo 'Done"
```

proceso_awk.sh

```
# NUMERO DE SECUENCIA EN DECIMAL #
# Obtenemos un fichero del trafico a la entrada del buffer que tendra en columnas:
# Columna 1 Columna 2 Columna 3
# PROCESO DE ANALISIS DE LAS TRAMAS OBTENIDAS #
# Obtenemos un unico fichero que tendra:
# Columna 1 Columna 2 Columna 3 Columna 4
# Numero de secuencia (decimal) Tiempo origen Tamaño Tiempo Destino
awk '{print $3,$1,$2}' entradaNS.txt > mezclar2.txt
awk '{print $3,$1}' salidaNS.txt > mezclar4.txt
join mezclar2.txt mezclar4.txt > unificado.txt
# OPCION UNO: CONOCIENDO VELOCIDAD
awk '{
resta=$4-$2
division=resta/(($3*8)/10)
print $1,$2,$3,$4,division
resta=0
}' unificado.txt > unificado2.txt
awk '{
print $4
}' unificado2.txt > tiempos.txt
awk '{
if ($NR==0){
getline linea < "tiempos.txt"
}
igual=0
while (igual==0){
        if ($2 < linea){
        resta =NR-contador
        print $1,$2,$3,$4,$5,resta
        igual=1
}
        if ($2 ¿= linea) {
        getline linea < "tiempos.txt"
        contador++
}
}
}' unificado2.txt > $3.txt
awk '{printf $1 ","$2 ","$3 ","$4 ","$5 ","$6 "\n"}' $3.txt > $3.csv
# LO CONVIERTO A CSV PARA LAS GRAFICAS MATLAB O COMO SE QUIERA

awk '{printf $4  $6 "\n"}' $3.txt > graphic_$3.csv
```

```
i=0
awk '
{
if (i==0) {
var=$1
i++
}
# printf var  $2 "\n"
printf $1-var  $2 "\n"
}
' $3.txt ¿graphic_$3.csv
```