Universidad Zaragoza

Proyecto Fin de Carrera

Análisis de la eficiencia en túneles LISP con multiplexión y seguridad

Autor

Andrés García Cavero

Director

José María Saldaña

Ponente

Julián Fernández Navajas

ESCUELA DE INGENIERÍA Y ARQUITECTURA

2016 / 2017

Agradecimientos

A mis padres, a mi hermano, a Cynthia, a Lena, a Rubén, a Alberto y a Julián.

ANÁLISIS DE LA EFICIENCIA EN TÚNELES LISP CON MULTIPLEXIÓN Y SEGURIDAD

RESUMEN

Este proyecto tiene como objetivo el "Análisis de la eficiencia en túneles LISP con multiplexión y seguridad", con el propósito de comprobar las técnicas de Calidad de Servicio desarrolladas en el grupo Ceniteq de la Universidad de Zaragoza se ha diseñado un entorno de laboratorio adecuado a las necesidades de los protocolos a testear.

Con la aparición de nuevos servicios en Internet y el cambio de los perfiles de tráfico se descubrió la necesidad de garantizar una mínima capacidad a las transmisiones a través de una red Best Effort. Esta nueva necesidad se denominó Calidad de Servicio (QoS). Hoy en día, la Calidad de Servicio es una necesidad para las aplicaciones de tiempo real de los que disfrutamos actualmente en Internet y los dispositivos móviles.

Este proyecto se compone, primeramente, del estudio teórico de las técnicas, tecnologías y protocolos utilizados con el objetivo de mejorar la Calidad de Servicio en las comunicaciones mediante el enfoque de la Ingeniería de tráfico.

Más adelante se explica y aplica la metodología diseñada para evaluar la eficiencia de la red bajo la influencia de las diferentes técnicas utilizadas. Esta metodología comenzó siendo específica para el PFC "Análisis de la eficiencia en túneles LISP con multiplexión y seguridad", sin embargo, ha terminado siendo la definición de una propuesta metodológica para el estudio no solamente de las técnicas que atañen a este PFC, sino a cualquier configuración de red.

De forma paralela a esta metodología se ha desarrollado una herramienta capaz de ejecutarla de forma desatendida en los dispositivos que conforman la red.

Por último, se recogen los resultados y las posibles líneas futuras de investigación a seguir.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

Capítulo 1

Introducción

1.1 PROBLEMÁTICA

Los orígenes de Internet se remontan a 1969. En aquel entonces, la red ARPANET fue diseñada para el intercambio fiable de archivos, sin llegar a imaginar que podrían existir requerimientos temporales o congestiones de tráfico. La red de hoy, sin embargo, debe afrontar estos problemas continuamente, y los servicios que se prestan sobre ella tienen unos requerimientos que no fueron contemplados en un principio.

En el Internet actual, existe una figura de mérito, en la caracterización de una red, denominada Calidad de Servicio (Quality of Service, QoS, en sus siglas en inglés), que indica la eficiencia y la velocidad a la cual una red se compromete a entregar los paquetes de un servicio que transmite a través de su infraestructura. Esta figura se basa en valores como la tasa de pérdidas, el ancho de banda, el retraso introducido o la disponibilidad de capacidad.

Dicha figura define el rendimiento promedio de la red y también se puede expresar en términos de la calidad subjetiva percibida por los usuarios finales al usar un servicio: si existen cortes en la comunicación o si ésta se degrada y si son asumibles, aceptables, etc.

La Calidad de Servicio es particularmente representativa para aquellos servicios denominados de tiempo real, que presentan requerimientos temporales estrictos para la entrega de sus paquetes; como pueden ser, por ejemplo, la Voz sobre IP (Voice over IP, VoIP), las videoconferencias o los juegos online multijugador.

Dado que la red no fue diseñada para este tipo de servicios, presenta problemas para aplicar unas políticas que definan y garanticen la Calidad de Servicio, sobre todo en la interconexión de redes de diferentes sistemas autónomos o proveedores. Este problema se ha intentado abordar de diferentes maneras, siendo las más destacadas la arquitectura de servicios integrados (INTSERV), la arquitectura de servicios diferenciados (DIFFSERV) y la ingeniería de tráfico, que se ilustran en la Figura 1.

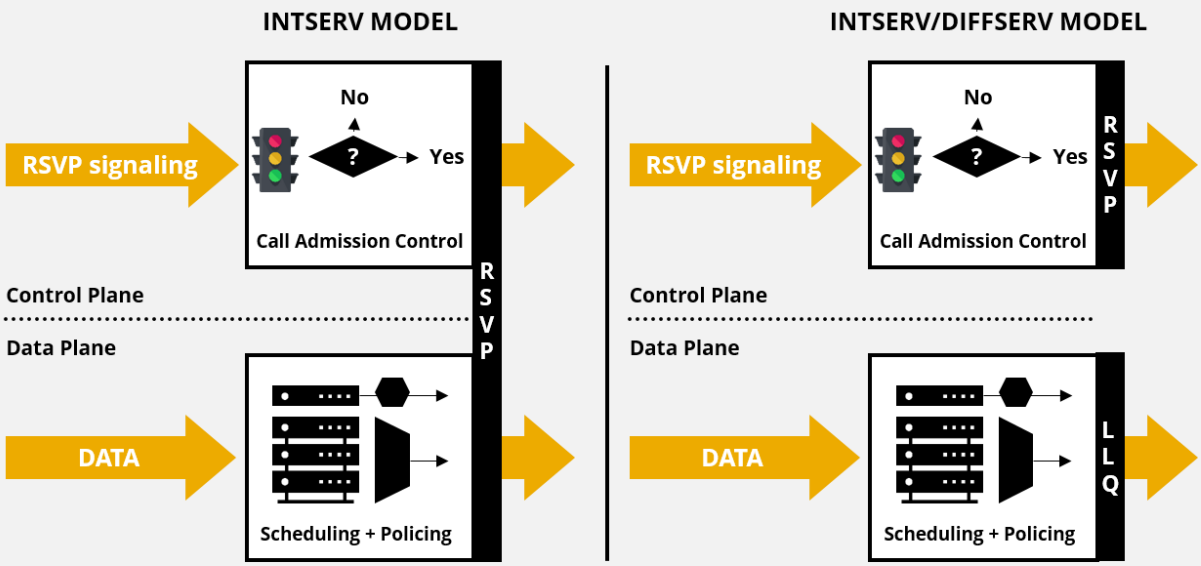


Figura 1 – Diferentes estrategias para proporcionar Calidad de Servicio

La primera de ellas, arquitectura de servicios integrados, busca una reserva de recursos de extremo a extremo, requiriendo el desarrollo de protocolos de control para cada comunicación entre actores y su implantación en todos los elementos de red. La arquitectura de servicios diferenciados propone mapear las necesidades de calidad de servicio en los paquetes transmitidos, utilizando 6 bits en el campo TOS (Type Of Service) en IPv4 y en el campo Traffic Class de IPv6, para así no tener que identificar cada comunicación.

En ambos casos, el requerimiento de que toda la infraestructura de red participe en estas arquitecturas ha provocado su estancamiento, presentando, no obstante, implantaciones que se encuentran habitualmente restringidas a un único sistema autónomo.

La ingeniería de tráfico, por otra parte, se compone de una serie de técnicas de diseño de rutas mediante la tunelización y el etiquetado de las comunicaciones para obtener Calidad de Servicio, aunque no existe garantía. Su enfoque no requiere de modificaciones en los elementos de red y es por ello el más utilizado hoy día. Además permite el uso de funciones adicionales, previas a la tunelización, como compresión de cabeceras, multiplexión o seguridad, que son las que aportan calidad.

1.2 OBJETIVOS DEL PRESENTE PFC

En el grupo de trabajo en el que se desarrolla el PFC, se propone la posibilidad de añadir compresión de cabeceras, multiplexión y seguridad a túneles LISP (Locator/Identifier Separation Protocol *[FARINACCI]*), mediante los protocolos SimpleMux *[IETF-SALDANA]* e IPSec *[SEO]*, de tal forma que se puede mejorar los parámetros de calidad de servicio y sacar un mayor partido a la

infraestructura ya desplegada, sin necesidad de realizar modificaciones en la red para su puesta a punto y funcionamiento. LISP es un protocolo de mapeo y encapsulación desarrollado por el Internet Engineering Task Force (IETF) que propone una nueva arquitectura de red y un conjunto de protocolos para el direccionamiento IP, y se basa en la idea de combinar las dos funciones principales que proporciona el nivel de red: enrutamiento y direccionamiento, pudiendo funcionar tanto sobre IPv4 como sobre IPv6.

El objetivo del proyecto supone la creación de una plataforma que permita la realización de pruebas, en un entorno similar a una red de comunicaciones real, para el estudio comparativo de la eficiencia, el retardo y las pérdidas (como parámetros de calidad) obtenidos en cada caso. Según se combinen los protocolos de multiplexión y seguridad sobre LISP se pueden obtener diferentes valores de eficiencia y retardo. Dicho estudio se basa en el análisis y comprobación de los métodos e ideas surgidos del proyecto TISFIBE (Túneles inteligentes y seguros para flujos IP con baja eficiencia, TIN2015-64770-R) que actualmente está desarrollando el grupo.

Para ello se ha diseñado y montado una red controlada de laboratorio con equipos enrutadores reales conectados mediante una conexión inalámbrica sobre la cual podemos medir la eficiencia, el jitter y las pérdidas provocados por el uso de LISP, según se combinen la compresión de cabeceras, la multiplexión y la seguridad.

Las comunicaciones a medir se realizan extremo a extremo, utilizando generadores de tráfico. Se registran los resultados para buscar la configuración óptima con respecto al rendimiento y la seguridad que presente la conexión.

Permite, por último, analizar los efectos finales a nivel de aplicación que producen las técnicas de multiplexión, tunelado y securización en los paquetes enviados a través de la red y presentar los resultados analizando su naturaleza y justificando las situaciones encontradas.

1.3 MARCO DEL PFC (PROYECTO TISFIBE)

El proyecto TISFIBE, desarrollado en el grupo CENITEQ, de la Universidad de Zaragoza, busca estudiar la optimización y securización del tráfico de servicios IP que generen flujos de baja eficiencia. Este tipo de servicios, denominados comúnmente “Small Packet Services”, sufre especialmente cuando necesitan ser transmitidos a través de tecnologías de acceso al medio compartido como puede ser una red WiFi.

1.4 ORGANIZACIÓN DE LA MEMORIA

El presente proyecto se encuentra dividido en cinco capítulos:

- En este primer capítulo introductorio se presenta el marco en el cual se engloba este proyecto, la motivación y los objetivos propios del PFC.
- El segundo capítulo describe la situación actual en el campo de la Calidad de Servicio, los túneles LISP y su securización.
- El tercer capítulo presenta el entorno de pruebas de laboratorio diseñado y los bancos de test ideados para medir las características de las configuraciones.
- El cuarto capítulo dispone las pruebas realizadas y los resultados obtenidos.
- En el quinto capítulo se exponen las conclusiones, tanto de los resultados obtenidos como del trabajo realizado durante el proyecto.

Se incluyen, además, a modo de anexos a la memoria, otras secciones con información complementaria sobre los procesos y metodología del estudio realizado.

Capítulo 2

Estado del arte

2.1 ESTUDIO DE LA EFICIENCIA DE LOS PERFILES DE TRÁFICO (APLICACIÓN DE LA MULTIPLEXIÓN, COMPRESIÓN Y SEGURIDAD)

Internet no fue diseñada para los servicios que disfrutamos actualmente. Tanto la Voz sobre IP (VoIP), las videoconferencias o los juegos online multijugador resultan muy exigentes en la Calidad de Servicio que demandan a la red. Además, esta nueva generación de servicios es a menudo ofertada por empresas Over The Top (OTT), es decir, empresas que conectan a la red sus sistemas para ofrecer estos servicios pero no son proveedores de Internet u operadores de telefonía, sino que transmiten sobre su red.

La diferencia se encuentra en que los OTT operan a nivel de aplicación o de transporte, es decir, por encima de la capa de red. No tienen control sobre la infraestructura sobre la cual transitan sus paquetes y no pueden disponer de mecanismos de reserva de recursos y ancho de banda o políticas de priorización de tráfico. Un ejemplo de servicio de operador versus servicio OTT sería una llamada de voz a través de un proveedor de telefonía móvil o a través de Skype o WhatsApp.

Dichos servicios suelen presentar necesidades de tiempo real, habitualmente porque la aplicación necesita de una interactividad entre los usuarios que se encuentra en los extremos de la conexión. Si aplicamos un perfil de tráfico tradicional a una llamada de voz, guardaríamos las muestras recogidas del micrófono durante, al menos, tres segundos para enviarlas posteriormente todas juntas en un único paquete de tamaño habitual. Esto introduciría un retardo adicional haciendo inviable la comunicación, pues habrá paquetes que resulten obsoletos al llegar al destino.

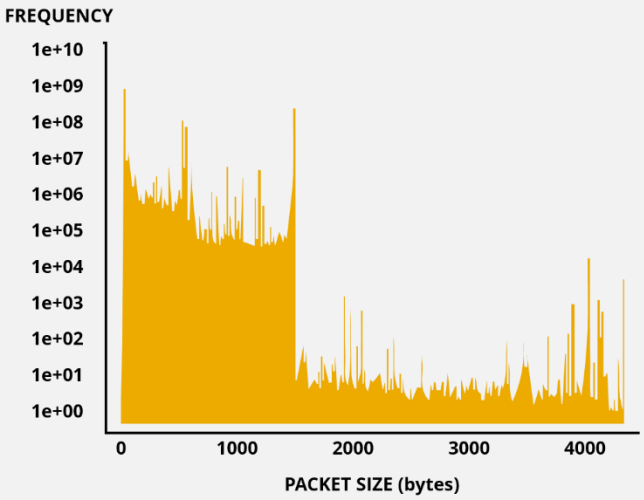


Figura 2 – Distribución media del tamaño de los paquetes en 1998.

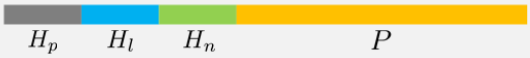
(Fuente: http://www.caida.org/publications/papers/1998/Inet98/Inet98.html#size)

Debido a ello, surgieron nuevos perfiles de tráfico que se adaptan mejor a las necesidades de los servicios de tiempo real. Estos nuevos esquemas se basan en la transmisión de paquetes pequeños a altas tasas, disminuyendo así los retardos introducidos por la espera para la generación de paquetes. En la Figura 3 podemos observar como ya en 1998 la red presentaba habitualmente paquetes pequeños. Esta tendencia continuaría e incluso se acrecentaría en el futuro. Las muestras de voz de nuestra llamada, utilizando este nuevo perfil, son ahora transmitidas prácticamente después de su generación, con la intención de ser enviadas lo antes posible al receptor y no provocar cortes de audio. Favoreciendo la interactividad.

Estos perfiles de tráfico han prosperado debido a la aparición de transmisiones de información diferencial con respecto al estado del servicio. Pongamos como ejemplo un juego online multijugador: la arquitectura de red del juego está diseñada de tal manera que no sea preciso enviar a otro jugador el estado completo de nuestra partida, lo cual supondría transmitir una gran cantidad de datos, sino solamente enviar los cambios con respecto a la última transmisión, como que un jugador se ha movido hacia la izquierda o hacia la derecha, siendo estos fácilmente codificables en pocos bytes. Las necesidades de tiempo real no permiten crear tramas grandes.

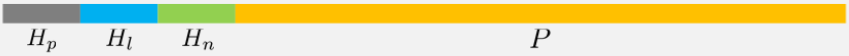
En la Figura 3 se muestran los cálculos analíticos genéricos correspondientes a las configuraciones descritas en el apartado junto con una relación visual de sus aplicaciones sobre las tramas.

Servicio genérico



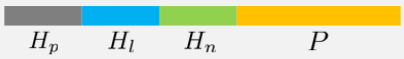
$$\eta = \frac{P}{H_p + H_l + H_n + P}$$

Servicio *Long packet service*



$$P \gg H_p + H_l + H_n \Rightarrow \eta \approx \frac{P}{P} \rightarrow 1$$

Servicio *Small packet service*

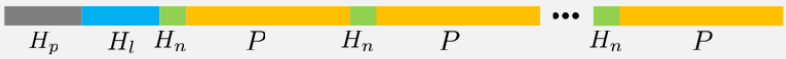


$$P \approx H_p + H_l + H_n \Rightarrow \eta \approx \frac{P}{2 * P} \rightarrow 1/2$$

Servicio con compresión de cabeceras

$$P > H_p + H_l + \frac{H_n}{C}$$

Servicio con compresión de cabeceras y multiplexión



$$\eta = \frac{n * P}{H_p + H_l + n * (\frac{H_n}{C} + P)}$$

Leyenda

$H_l : Cabecera\ de\ nivel\ de\ enlace$
$H_p : Cabecera\ de\ nivel\ fisico$
$H_n : Cabecera\ de\ nivel\ de\ red$
$P : Payload$
$C : Ratio\ de\ compresion$
$\eta : Eficiencia$
$n : Numero\ de\ paquetes\ multiplexados$

Figura 3 – Tramas generadas según las diferentes técnicas utilizadas

Aunque estos nuevos perfiles de tráfico resultan más adecuados para servicios con requerimientos de QoS, tienen la contrapartida de afectar al rendimiento de la red. Donde anteriormente teníamos paquetes grandes con pequeño overhead, ahora tenemos una distribución de paquetes cuyos datos útiles son del orden del tamaño de las cabeceras que los transportan, y por lo tanto ineficientes. Los Small Packet Services disminuyen la eficiencia de las transmisiones, no sólo porque ahora tenemos que enviar cabeceras y campos de control donde antes enviábamos datos útiles, sino porque ahora además requerimos de más capacidad de procesado en los elementos de red.

Este problema de eficiencia en los elementos de la red puede llegar a afectar al resto de transmisiones que deben ser tramitadas en ese momento, por ello, se han desarrollado varias técnicas para tratar de paliar sus efectos: la compresión de cabeceras y la multiplexación de varios paquetes pequeños en uno mayor.

La técnica de la compresión de cabeceras consigue la reducción del overhead de un paquete mediante el uso de un compresor al inicio del canal de transmisión y un decompresor a su final. Es una técnica comúnmente utilizada que permite la reducción del overhead enviado para cada paquete, lo cual aumenta la eficiencia y disminuye la carga de procesado de la red.

La multiplexión de varios paquetes pequeños propone el envío conjunto de varios paquetes pertenecientes a un Small Packet Service utilizando una única cabecera. Se aumenta así la eficiencia de la transmisión dado que estamos repartiendo el overhead que provoca una cabecera entre varios paquetes.

Esta multiplexión suele realizarse para una cantidad pequeña de paquetes, de manera que el retardo introducido no sea comparable con aquel que teníamos transmitiendo tramas grandes. Estamos esperando a varios paquetes para poder concatenarlos.

Hay que tener presente que, aunque estas técnicas son de ayuda para mejorar la eficiencia de la red, no son a coste cero. Ambas introducen un nuevo retraso, aunque pequeño, en la transmisión y, en el caso de la compresión un procesado en el equipo inicial y final del tránsito. Por ello es conveniente buscar el punto óptimo de compresión y multiplexión en una transmisión.

Cuando aplicamos compresión de cabeceras estamos recortando la información que necesita un paquete para llegar de un extremo a otro de la red, sin esa información no puede viajar. Entonces, necesitamos proporcionarle otro método para que sea enrutado a su destino. Esto lo podemos conseguir añadiendo una cabecera básica, de menor tamaño que la que hemos quitado, que guíe su camino hasta el descompresor. Estamos añadiendo de nuevo parte del overhead que hemos quitado, pero la mínima información necesaria y además se trata de una sola cabecera compartida por todos los paquetes, es decir, estamos creando un túnel entre compresor y descompresor.

Además de ello, cuando multiplexamos, concatenamos varios paquetes que comparten un destino o unas características. Lo hacemos para que viajen juntos durante una parte del trayecto. Por lo tanto, si van a viajar juntos, no necesitan de una cabecera cada uno de ellos, sino que podemos colocarles una cabecera para todos que los lleve hasta un demultiplexor. Estamos creando de esta manera otro túnel.

Ambas técnicas se pueden combinar, para aprovechar las sinergias que presentan entre ellas. En lugar de crear un túnel para compresión y otro para multiplexión, utilizamos un único túnel que realiza ambas funciones. Así pues, comprimimos las cabeceras de varios paquetes y los multiplexamos, obteniendo un mínimo tamaño de paquete para el túnel. Un túnel que es necesario para que los datos puedan viajar por la red.

Para este tipo de configuraciones es factible realizar una aproximación analítica para tecnologías como Ethernet, pero todo se complica cuando los interrogantes se trasladan al mundo inalámbrico. Los tiempos de guarda, de Backoff, CIFS y SIFS no resultan predecibles para una red no teórica.

En las comunicaciones de hoy en día la seguridad es necesaria. En todo el transcurso de nuestras transmisiones no hemos añadido una capa de seguridad en ningún momento, confiando en la seguridad que haya estimado la capa de aplicación. Esto podría permitir que cualquiera escuchase nuestra llamada de voz. Por lo tanto es imperativo contemplarla. Pero añadir seguridad presenta dos problemas:

Por un lado volvemos a incurrir en un excesivo overhead y en una ineficiencia de las transmisiones, puesto que nos vemos obligados a añadir una nueva cabecera para cifrar los datos y/o autenticarlos.

Por otro lado, además, podemos incurrir en un problema de doble securización. Este sucedería cuando una aplicación cifra sus datos antes de enviarlos, nosotros los transportamos por un túnel y les añadimos seguridad para garantizar su confidencialidad e integridad, cosa que ya se encontraba garantizada desde un principio. Es decir, estamos securizando unos datos dos veces, con el overhead añadido que eso supone y la disminución de la tasa útil que provoca.

2.2 PROTOCOLOS ESPECÍFICOS UTILIZADOS

LISP

LISP es un protocolo de enrutamiento que permite la creación de túneles dinámicamente en función de las necesidades de conexión, además de añadir otras funcionalidades como el multihoming o la movilidad.

Se basa en la separación semántica de las direcciones IP en dos tipos conjuntos diferentes: Endpoint Identifiers (EID) y Routing Locators (RLOC). Los RLOC son usados por los *router* LISP para crear túneles de direccionamiento, mientras que los EID se utilizan para identificar a un dispositivo final.

Puede funcionar sin ningún tipo de cambio en la pila actual de comunicaciones TCP/IP y puede ser implantado de manera gradual. Actualmente se considera como una de las posibles soluciones al problema de la escalabilidad del enrutamiento.

El protocolo LISP proporciona una serie de métodos para que los *router* intercambien información de mapeo entre EID, no enrutables públicamente, y RLOC. Además define las directivas que utilizan los *router* para encapsular paquetes IP dirigidos a EID utilizando RLOC como destino del túnel. Su funcionamiento es análogo al de DNS para la resolución de direcciones IP.

LISP fue resultado del problema de que utilizar una única dirección para identificar unívocamente a un dispositivo y además determinar su localización requiere una optimización en dos ejes: Para que el routing sea eficiente, la dirección debe ser asignada topológicamente, pero para manejar los dispositivos de una manera eficiente, sin la necesidad de reasignación de direcciones ante un cambio topológico, la dirección IP debe ser independiente de la topología en la que se encuentre el dispositivo.

Tanto los RLOC como los EID son sintácticamente iguales a las direcciones IP, tanto en su versión 4 como en su versión 6. Sólo cambia la manera en la que se utilizan. Este hecho es el que permite a LISP trabajar sin ningún cambio en enrutadores intermedios o en dispositivos finales.

Debido a esta separación entre la topología y la identificación unívoca, necesitamos a un tercer actor en el escenario, un dispositivo que sea capaz de mapear las direcciones EID a su correspondiente enrutador RLOC. Este actor es el denominado Map Server, cuyas funcionalidades pueden ser divididas en Map-Server y Map-Register. Un sistema Map-Register es aquel al cual los RLOC informan de qué direcciones son capaces de enrutar o se encuentran conectadas a ellos, mientras que un Map Server es aquel a quien un RLOC pregunta por una determinada dirección de EID para saber a través de qué RLOC debe enrutar los datos.

LISP crea los túneles de direccionamiento bajo demanda, es decir, en una transmisión de la Red-A a la Red-B, hasta que un paquete no desea ir a la Red-B, LISP no consulta al Map-Register cuando es la dirección a la cual debe encaminar los paquetes dirigidos a la Red-B. Como el agente LISP que gestiona la salida de la Red-B opera de igual manera, no preguntará por el la dirección de enrutamiento de la Red-A hasta que reciba un paquete de su red con destino a ella. Como la resolución de estas direcciones supone una consulta al Map-Register, y esto lleva una conexión, cuando la dirección sea resuelta el primer paquete enviado ya habrá sido descartado por *timeout* antes de ser enviado. El segundo paquete, sin embargo, encontrará el túnel LISP creado y será enrutado eficazmente. Esto implica que, como vemos en la Figura 6, en una conexión entre las redes A y B, donde haya tráfico de ida y vuelta, se perderán dos paquetes. Uno resolviendo el

camino de ida a la Red-B y otro resolviendo el camino de vuelta desde la Red-B a la Red-A. Una vez resueltas, LISP guardará las direcciones de enrutamiento en una tabla para no volver a tener que realizar la petición al Map-Register.

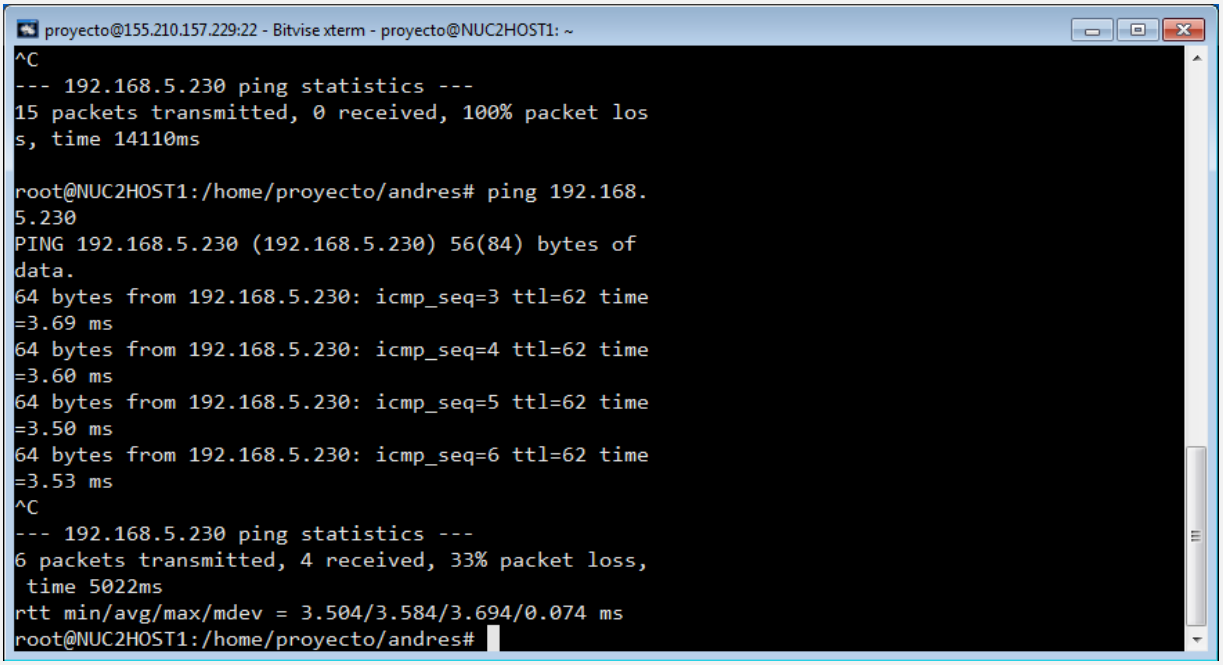


Figura 4 – Captura del envío de pings a través de un enlace LISP

IPSec

IPSec permite añadir seguridad a una comunicación al nivel de la capa de red. No se trata de un protocolo en sí, sino de un conjunto estandarizado de protocolos. Estos protocolos proporcionan la funcionalidad de cifrar y autenticar las comunicaciones entre dos dispositivos y la negociación necesaria para el intercambio seguro de claves. Puede funcionar entre dos dispositivos finales pero también involucrando elementos de red para crear un túnel entre dos redes. IPSec utiliza los siguientes protocolos subyacentes para proporcionar dichas funcionalidades:

Authentication Headers (AH): Proporciona integridad de los datos enviados en un paquete y del remitente para datagramas IP. Es un protocolo no orientado a conexión y es capaz de proteger de ataques de réplica y de inserción.

Protege todos aquellos campos que no varían en un salto de enrutamiento (DSCP/ToS, ECN, Flags, Fragment Offset, TTL y Header Checksum). Funciona de manera similar para IPv6. Funciona directamente sobre IP, usando el 51 como número de protocolo.

Encapsulating Security Payloads (ESP): Garantiza la confidencialidad, el origen de los datos y su integridad. Soporta configuraciones para solo autenticar o solo cifrar, o ambas cosas. Pero no se recomienda usar encriptación sin autenticación por considerarse inseguro. No proporciona integridad y autenticación si se usa en modo transporte, si lo hace en modo túnel, ya que el paquete original se encapsula dentro de un nuevo paquete IPSec. Utiliza el número 50 de protocolo, funciona sobre IP.

Security Associations (SA): Proporciona un conjunto de algoritmos y parámetros necesarios para la correcta operación de AH y ESP. Mediante ISAKMP (Internet Security Association and Key Management Protocol) intercambia las claves y parámetros relativos a la política de seguridad de cada dispositivo.

Para seleccionar el nivel de protección que requiere un paquete al ser enviado, IPSec utiliza el SPI (Security Parameter Index), un índice de la política en la base de datos de asociaciones de seguridad (SADB, Security Associations Database), así como la dirección IP de destino que identifican unívocamente una configuración de IPSec para ese paquete. El mismo mecanismo se utiliza para los paquetes entrantes.

Puede operar en dos modos diferentes:

- **Transporte**. Es un modo de host a host. En este modo, sólo los datos del paquete se cifran o autentican. Como la cabecera IP del paquete no se modifica, el enrutamiento es el mismo que sufriría un paquete sin IPSec. Sin embargo, si se utiliza junto con AH, la dirección IP no puede ser modificada, lo que provoca problemas en el uso de NAT.
- **Túnel**. En el modo de túnel de red, el paquete entero es cifrado y autenticado, creando un nuevo paquete IP exterior cuyos datos son el paquete original. Normalmente se utiliza para crear redes privadas virtuales (VPN). Puede trabajar entre dispositivos de red (de *router* a *router*), entre dispositivos finales (de host a host) y de manera híbrida (de host a network). El modo de túnel sí soporta NAT.

SimpleMux

SimpleMux es un protocolo de multiplexación genérica desarrollado por investigadores de la Universidad de Zaragoza para aumentar la eficiencia de las transmisiones. Es positivo, sobre todo, para paquetes de tamaño pequeño como los que presentan los Small Packet Services y pertenecientes a una misma conexión *[SALDANA].*

Su aplicación no está restringida a la multiplexión de paquetes de un solo protocolo, sino que posee un campo para especificar el protocolo que contiene cada uno de los paquetes incluidos, permitiendo de este modo la tunelización de protocolos dentro de un único paquete.

Su utilización mejora la calidad de servicio en transmisiones con necesidades de tiempo real, puesto que éstas se basan en el envío de una alta tasa de muestras transmitidas en paquetes de

tamaño bajo y cuya eficiencia de envío, el ratio entre la cantidad de recursos que necesita el envío de la cabecera y el del paquete en total, es muy baja.

También es apropiado para realizar ingeniería de tráfico, donde la multiplexión de paquetes pequeños puede permitirnos definir un camino concreto común para ellos. Además, esta aplicación nos permite convertir perfiles de tráfico donde el paquete medio tiene un tamaño bajo en perfiles de paquete medio grande, reduciendo el número de paquetes a gestionar por los equipos de red y permitiendo la aplicación de otros algoritmos destinados a datos de tamaño superior como puede ser la compresión.

Las consecuencias de esta multiplexión en una red son:

- Tunelización sencilla de caminos. Permite evitar el uso de protocolos específicos de tunelización que añadirían más overhead a la transmisión.
- Reducción del número de paquetes por segundo en una red, aliviando las necesidades de cómputo de los equipos de transmisión.
- Reducción del ancho de banda necesario. Una alta tasa de paquetes de tamaño pequeño provoca una eficiencia de red baja, ya que gastamos una alta proporción de los recursos de la red en enviar la información necesaria para gestionar una pequeña cantidad de información transmitida.
- Reducción de la energía gastada en los equipos de la red, dado que en su mayor parte se debe al procesamiento de las cabeceras más que a la transmisión de las señales.

SimpleMux crea un túnel de datos multiplexados, este túnel puede operar en una transmisión *end to end*, de host a host, o mediante equipos de transmisión intermedios como *router* o *switch*, siendo de esta manera transparente al usuario final y de innecesaria configuración en los equipos finales pudiendo, además, multiplexar tráfico de varios usuarios.

Capítulo 3

Realización de pruebas en entorno controlado

3.1 ENTORNO DE LABORATORIO

El entorno de pruebas ha sido diseñado para poder poner en práctica las diferentes configuraciones que posibilitan las técnicas descritas anteriormente. Con este objetivo, hemos levantado un escenario que busca replicar a escala la estructura de una interconexión de redes que atravesaría una transmisión a través de Internet.

De esta manera, hemos ideado una red sencilla en la que podamos controlar y medir los parámetros que nos permitan determinar la efectividad del envío de datos a través de una determinada configuración. En nuestro escenario, las redes LAN representan la infraestructura privada de dos sistemas autónomos conectados a Internet a través de la red WAN de un proveedor que les ofrece direccionamiento público.

Esta red común es la que nos permite simular el funcionamiento de LISP en Internet, configurando los dispositivos de salida de cada red LAN como elementos xTR dentro de un sistema LISP y ofreciéndoles un servidor LISP dentro del espacio de direccionamiento público.

En la Figura 9 se puede observar un esquema del entorno de pruebas de laboratorio. Este entorno está formado por las redes LAN-4 y LAN-5, unidas por una red LAN-70 que realiza las funciones de red WAN. La tecnología utilizada en las redes LAN-4 y LAN-5 es Ethernet mientras que la red LAN-70 está basada en WiFi y ha sido configurada en modo Ad-Hoc.

Hemos decidido utilizar WiFi como tecnología para la red troncal dado que nos proporciona un entorno hostil (para la simulación en laboratorio, más parecido a lo que sería una transmisión real a través de Internet, donde tendríamos que lidiar con las transmisiones de otros usuarios y sufriríamos pérdidas de paquetes y retransmisiones, que son modelados en nuestro entorno como colisiones en el acceso al medio inalámbrico compartido, permitiéndonos además probar

las técnicas teóricas en condiciones adversas). Sin embargo, esto ha supuesto no poder desarrollar los cálculos analíticos más allá de los mostrados en el Capítulo 2.

Cada una de las redes LAN está conectada a la red WAN a través de un equipo tipo Raspberry Pi que realiza las funciones de enrutador. Este equipo es el encargado de proporcionar a los equipos situados en su red la posibilidad de conectar con la red WAN, lo cual le proporciona a su vez conectividad con la otra red LAN.

Hemos decidido utilizar este tipo de equipos puesto que nos permitían añadir al entorno de test un elemento que impusiera un límite basado en la capacidad de computación a la transmisión de los paquetes en la red, pudiendo testear su influencia en los resultados si fuese necesario.

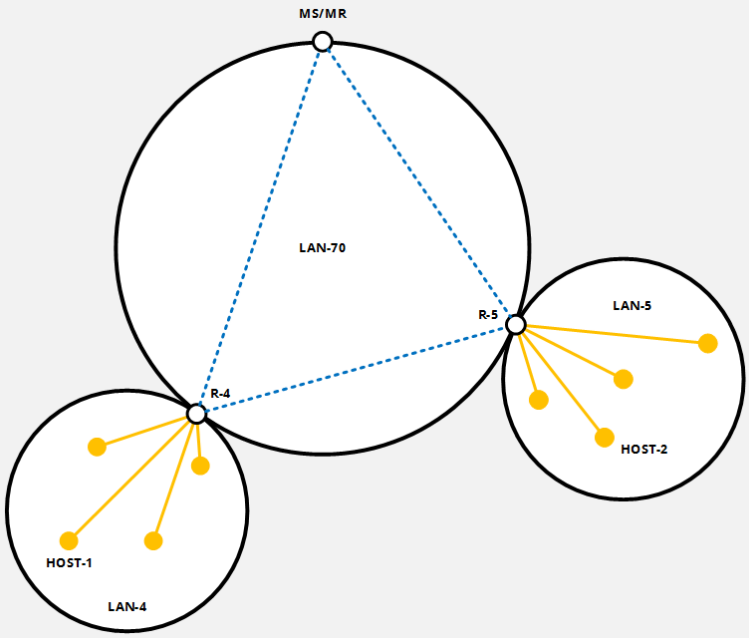


Figura 5 - Esquema del entorno de red montado en el laboratorio

En la red WAN podemos encontrar un tercer dispositivo además de los dos equipos enrutadores. Este equipo, conectado directamente a la red WiFi realiza las funciones de Map-Server y Map-Register del ecosistema LISP. Es el equipo al cual los enrutadores enviarán la información acerca de qué redes son capaces de alcanzar y las peticiones para resolver el direccionamiento a las redes a las que no estén conectados por sí mismos.

Los equipos situados en cada una de las redes LAN tienen configurado el direccionamiento necesario para alcanzar sus respectivos equipos de salida, pero desconocen el camino a la red LAN contraria. Por lo tanto, necesitarán realizar una resolución de direccionamiento LISP para llegar a un destino no perteneciente a su misma red.

Hemos querido crear dos redes autónomas en este entorno para poder testear transmisiones de extremo a extremo. Esto nos permite comprobar cómo afectan las técnicas diseñadas a la percepción que tendría un usuario final, enviando información del HOST-1 al HOST-2 y analizando los parámetros deseados a nivel de aplicación, pero también pudiendo tomar medidas en los equipos de transmisión oportunos si fuese necesario.

3.2 DIFERENTES ESQUEMAS DE MONTAJE

Figura 6 – Diferentes esquemas de montaje realizados en el laboratorio

Dentro de esta red hemos planteado diferentes esquemas de configuración, que se muestran en la Figura 10. Esto nos permite evaluar las medidas contra una referencia y ser capaces de expresar los resultados no en función de las características de la red, que suponen una medida única para esta infraestructura, sino en la variación de las mismas. Pudiendo tomar como origen una red limpia, con la configuración básica para el envío para expresar la mejora o empeoramiento porcentual referenciado con respecto a ella.

Montaje A:
Conexión extremo a extremo limpia a través de la red WiFi.
Para ello debemos establecer un direccionamiento entre los equipos enrutadores, ya que en este caso no se va a realizar la resolución del direccionamiento a través del servidor LISP.

Montaje B:
Conexión extremo a extremo a través de un túnel LISP sin seguridad.

Montaje C:
Conexión extremo a extremo a través de un túnel LISP con seguridad entre los equipos enrutadores.

Montaje D:
Conexión extremo a extremo con seguridad de extremo a extremo a través de un túnel LISP sin seguridad.

Montaje E:
Conexión extremo a extremo con seguridad de extremo a extremo a través de un túnel LISP con seguridad entre los equipos enrutadores. Este montaje presenta una doble securización de los datos a través del canal de transmisión inalámbrico, lo que supone una pérdida de eficiencia.

Montaje F:
Conexión extremo a extremo con seguridad a través de dos túneles LISP, uno con seguridad entre los equipos enrutadores y otro sin seguridad.

3.3 HERRAMIENTAS UTILIZADAS

LISPmob con SimpleMux

LISPmob es la implementación de LISP para distribuciones Linux. LISPmob con SimpleMux es la versión modificada de LISPmob que incorpora el protocolo SimpleMux desarrollado en el Grupo

Ceniteq de la Universidad de Zaragoza. Posee todas las funcionalidades de LISP más la capacidad de realizar multiplexión de paquetes en función de la dirección de origen o de destino.

» Disponible con licencia libre en el repositorio del proyecto en GitHub: https://github.com/simplemux/lispmob-with-simplemux
» Instrucciones de instalación y configuración en el Anexo C: Instalación, configuración y uso de herramientas.

IPSec-tools

IPSec-tools es un paquete de Linux que proporciona todas las herramientas necesarias para trabajar con IPSec en Debian. Incorpora todos los protocolos subyacentes y es fácilmente configurable a través de un fichero de configuración.

» Disponible en el repositorio de paquetes de Debian, a través tanto del gestor de paquetes Aptitude como de su página web: https://packages.debian.org/iperf y en el repositorio de Sourceforge: http://ipsec-tools.sourceforge.net/
» Instrucciones de instalación y configuración en el Anexo C: Instalación, configuración y uso de herramientas.

D-ITG

D-ITG (Distributed Internet Traffic Generator) *[DITG]* es un software desarrollado por la Universidad de Nápoles Federico II para generar tráfico de pruebas según unos patrones configurables. Funciona bajo una arquitectura cliente-servidor, la cual nos permite tanto la generación de tráfico como su comprobación tras ser recibida en el destino. Para ello debemos ejecutarlo en ambos dispositivos.

Con D-ITG podemos analizar tanto el rendimiento real que presenta una conexión como el nivel de pérdidas que sufre. Algunas de sus funciones más interesantes son:

- Generación de tráfico según intervalos configurables.
- Definición de diferentes tamaños de paquetes de testeo.
- Posibilidad de transmisión tanto sobre UDP como sobre TCP.
- Envío de tramas capturadas como datos de testeo, pudiendo reproducir el tráfico creado por aplicaciones como juegos o VoIP.
- Posee una API con la que interactuar programáticamente.

Está compuesto de varios programas:

- ITGSend: Generar el envío de una transmisión.
- ITGRecv: Recibir el envío de una transmisión.

- ITGDec: Decodificar el archivo binario generado por ITGRecv para el envío de una transmisión en datos legibles por el usuario o calcular una salida según un intervalo de muestreo y un formato de salida estructurado.

» Disponible en la página web del proyecto: http://traffic.comics.unina.it/software/ITG/.
» Instrucciones de instalación en el Anexo C: Instalación, configuración y uso de herramientas.

TCPDump

TCPDump es una herramienta de captura de paquetes de red con interacción mediante línea de comandos. Permite observar el funcionamiento de una red en tiempo real, mostrando tanto los paquetes transmitidos como los recibidos.

Fue escrita por Van Jacobson, Craig Leres y Steven McCanne en el Grupo de Investigación de Red del Laboratorio Lawrence Berkeley. Es posible ejecutarlo sobre la mayoría de los sistemas operativos aunque en algunos puede ser necesario tener privilegios de administrador.

Es capaz de recibir una serie de parámetros para definir los filtros a aplicar sobre las capturas. Un filtro es una expresión que permite seleccionar los paquetes que capturamos. En ausencia de filtros se capturan todos los paquetes.

Comúnmente se utiliza para depurar aplicaciones y firewalls. Aunque también puede ser usado maliciosamente para capturar tráfico no cifrado en una red.

» Disponible en el repositorio de paquetes de Debian, a través tanto del gestor de paquetes Aptitude como de su página web: https://packages.debian.org/tcpdump.
» Instrucciones de instalación en el Anexo C: Instalación, configuración y uso de herramientas.

3.4 CREACIÓN DE UNA HERRAMIENTA DE AUTOMATIZACIÓN DE TESTS

Normalmente las herramientas de testeo actúan en modo cliente-servidor. Una aplicación es lanzada desde un dispositivo y conecta contra otro para realizar una serie de acciones o pruebas. Este tipo de arquitectura requiere el lanzamiento de pruebas desde cada uno de los terminales desde los que se desee testear.

Figura 7 – Logotipo de *Multimeter*, la herramienta desarrollada

En una red grande y dependiendo de la batería de pruebas a realizar, esto puede ser costoso. Por ello se planteó el desarrollo de una herramienta de test descentralizada que fuese capaz de actuar en todos los dispositivos de la red. La herramienta actúa como un cliente de todos los dispositivos de la red, lo cual permite realizar test desde cada uno de los nodos.

La herramienta (Figura 11 y 12) es un cliente que podemos ejecutar en nuestro PC sin necesidad de que sea uno de los dispositivos que conforman la red. Podemos especificar desde ella un test y en el momento de lanzarlo ella se ocupará de traducir las especificaciones de nuestro test en las apropiadas acciones dentro de cada uno de los dispositivos.

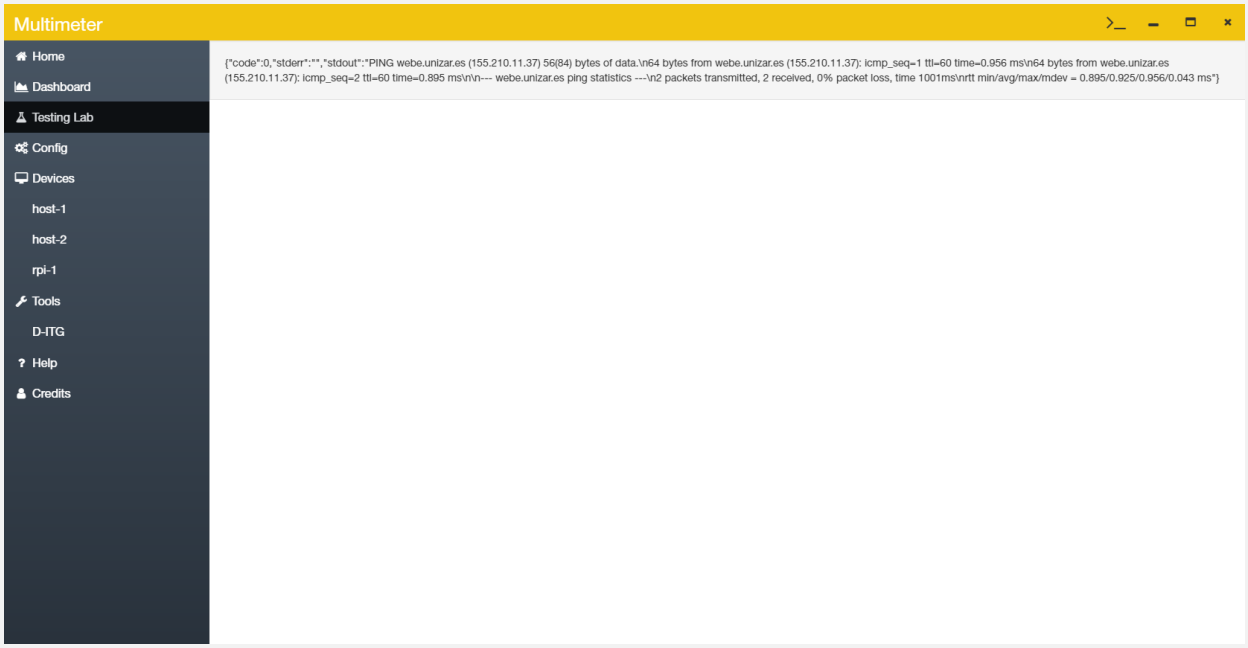


Figura 8 – Captura de pantalla de Multimeter

¿CÓMO FUNCIONA?

En la figura La herramienta se compone de una interfaz de usuario (UI) y de un “backend”, ambos ejecutados en la misma máquina. La interfaz de usuario es la que permite al testeador especificar las acciones que quiere realizar, estas acciones son comunicadas al “backend” mediante comunicación entre procesos (IPC: Inter Process Communication), que es quien se conecta mediante SSH a los distintos dispositivos, ejecuta comandos en ellos y devuelve los resultados formateados a la librería de gráficos de la UI. Permite el uso de herramientas de testeo como son Ping, IPerf o D-ITG.

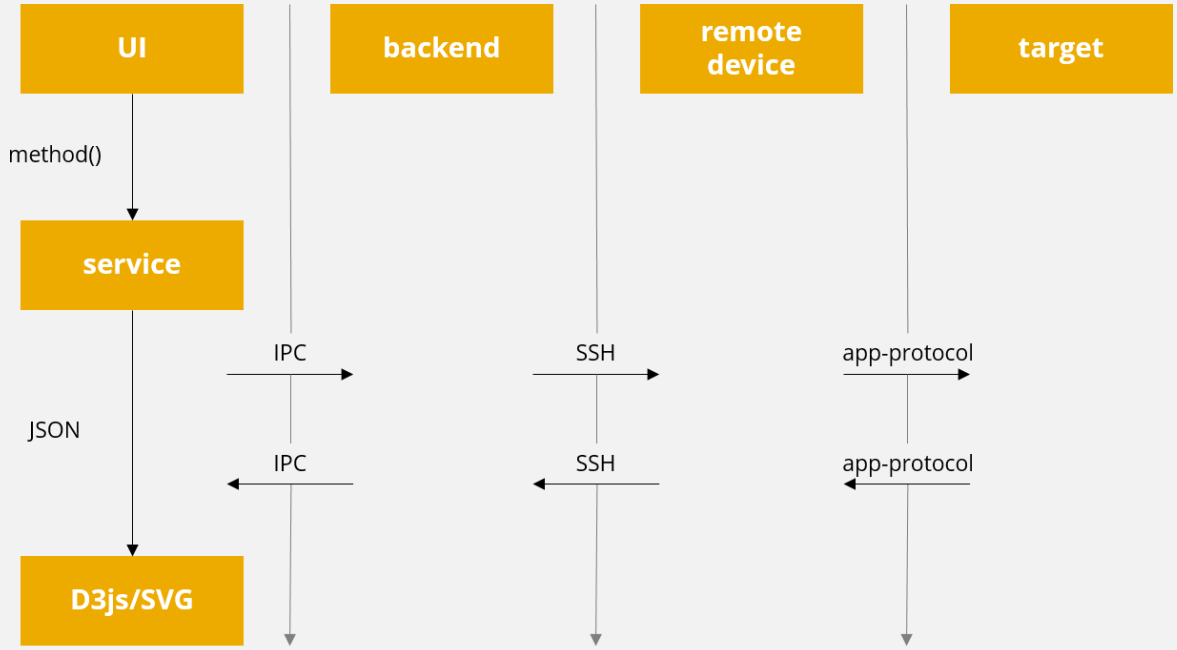


Figura 9 – Esquema de funcionamiento de Multimeter

DESARROLLO

Figura 10 – Arquitectura de tecnologías utilizada en el desarrollo de Multimeter

El desarrollo ha sido realizado utilizando el framework Electron. El cual permite la utilización de tecnologías de desarrollo web para el desarrollo de aplicaciones nativas de escritorio. Detrás de su desarrollo se encuentra GitHub Inc., es de código libre y puede descargarse desde la siguiente página: https://electron.atom.io/

Se compone de dos módulos principales que corren en procesos diferentes del sistema. Estos procesos son el *main*, para el backend y los *renderers* para la interfaz de usuario. Son independientes pero pueden comunicarse entre sí mediante el protocolo IPC. Estos procesos utilizan diferentes tecnologías:

Node JS como lenguaje de backend y de comunicación con el sistema operativo.

Chromium como motor de renderizado con soporte equivalente al del navegador Google Chrome para HTML5, CSS3 y Javascript, para la interfaz de usuarios.

Este framework se ha elegido dado que trabajar con él tan solo requiere conocimientos base de HTML, CSS y JS. Además, el uso del framework permite la ejecución de las aplicaciones en las tres plataformas de escritorio mayoritarias: Windows, Mac OS y Linux.

Al tratarse de tecnologías web, presenta la posibilidad de migrar una aplicación realizada sobre él para ser alojada y servida en un servidor web o para ser integrada en herramientas de generación de aplicaciones móviles como Apache Cordova con mínimas modificaciones.

Además el uso de Node JS como lenguaje de Backend permite la reutilización de librerías de terceros a través del gestor de paquetes NPM (Node Package Manager: https://npmjs.com) para la realización de operaciones más complejas como, por ejemplo, las conexiones SSH. Node JS se ejecuta bajo el motor V8 de Javascript desarrollado por Google para el navegador Chrome. Lo que lo hace compatible con el motor de renderizado de Electron.

Node JS presenta una especial facilidad para abordar el desarrollo de aplicaciones complejas que interactúan con Internet, presenta una eficiente escalabilidad y un conciso manejo de entradas y salidas de manera tanto síncrona como asíncrona. Mientras que gracias a su uso extendido presenta una facilidad de acceso a la documentación.

El código de *Multimeter* se encuentra disponible para descarga gratuita bajo licencia open source en el siguiente repositorio de GitHub: https://github.com/andres-garcia-cavero/pfc. Así mismo se proporcionan instrucciones de instalación y uso en el Anexo D: Instalación, configuración y uso de la herramienta Multimeter.

Capítulo 4

Metodología de las pruebas

Por claridad y simplicidad en la presentación de los resultados, en este PFC se han realizado test con distribución uniforme de tráfico. Pero debe tenerse en cuenta que las distribuciones de los servicios con requerimientos de Calidad de Servicio y el tráfico de Internet pueden seguir también otros patrones.

4.1 PRUEBAS DE SATURACIÓN DEL ENLACE

El primer tipo de pruebas planteadas son de saturación del enlace. Este tipo de pruebas nos permiten analizar cómo afecta la variación del overhead añadido por las diferentes técnicas a la transmisión. Consisten en un barrido de envíos de paquetes para diferentes tasas y tamaño de datos. Después identificamos según el bitrate recibido experimentalmente si el enlace se ha saturado.

Dado que estamos intentando saturar el enlace, vamos a fijarnos en la tasa de paquetes perdidos. Debemos pues establecer un umbral a partir del cual consideremos que hemos sufrido pérdida de paquetes, puesto que al realizar este test en una red WiFi siempre vamos a tener pérdidas debidas al canal. Es decir, definiremos una tasa de pérdidas de referencia. Se ha tomado un valor de ___

Esta prueba consiste en testear el envío a través de una red con diferentes configuraciones buscando la configuración umbral a partir de la cual obtenemos una disminución significativa del bitrate en las transmisiones. Para ello se utiliza un script que lanza un test de transmisión UDP de D-ITG en función de la tasa de envío en paquetes por segundo y el tamaño de dichos paquetes. Cada test realiza una prueba de ancho de banda con los parámetros especificados, y variando dichos parámetros somos capaces de mapear el comportamiento de la red para diferentes combinaciones de tamaño de los paquetes y de tasa de envío.

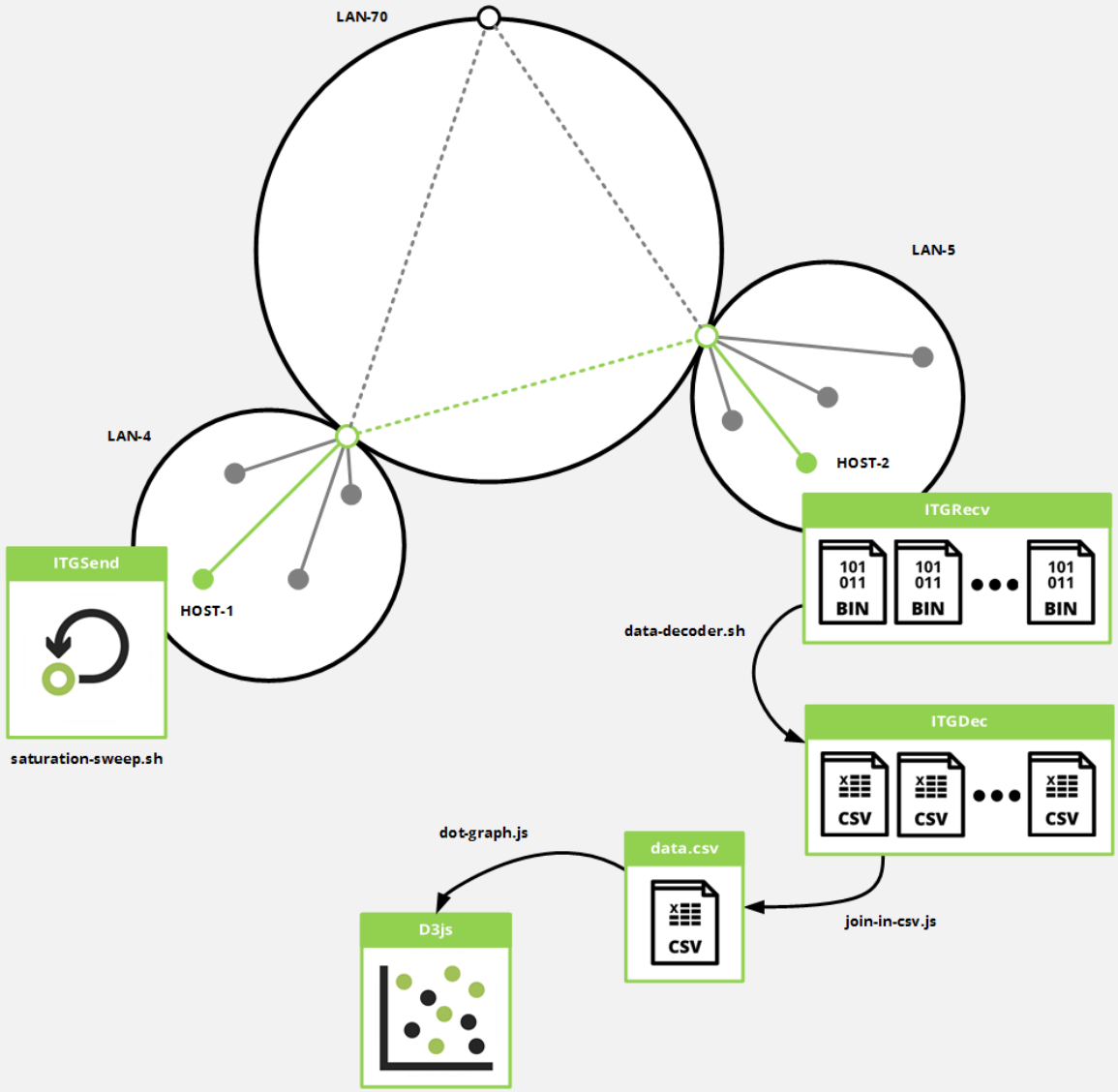


Figura 11 – Esquema de un test de barrido de saturación

Como se muestra en la Figura 15, lo que obtenemos de esta prueba es un fichero de resultados para cada uno de los test de envío, El siguiente paso consiste en decodificar la información de este fichero, que contiene todos los datos del test en un formato binario, para quedarnos únicamente con la información que nos es relevante para el estudio. En este caso se trata de los datos referentes a los resultados medios de la transmisión.

Esto lo hacemos utilizando la utilidad ITGDec, que nos permite extraer datos de los test realizados con D-ITG. Este proceso da como resultado los datos deseados en formato .CSV, en un fichero diferente para cada test. Después tratamos estos datos con un script de Node JS creado a tal

efecto, cuya funcionalidad es la de concatenar todos los ficheros en uno solo, manteniendo su formato.

Con este conjunto de test podemos recopilar estadísticas acerca de cómo se comporta la red para un determinado perfil de tráfico. Después, dado que hemos introducido dos variables independientes en los datos, podemos representar las magnitudes medidas en los test en una gráfica en dos dimensiones como la que se puede observar en la Figura 12:

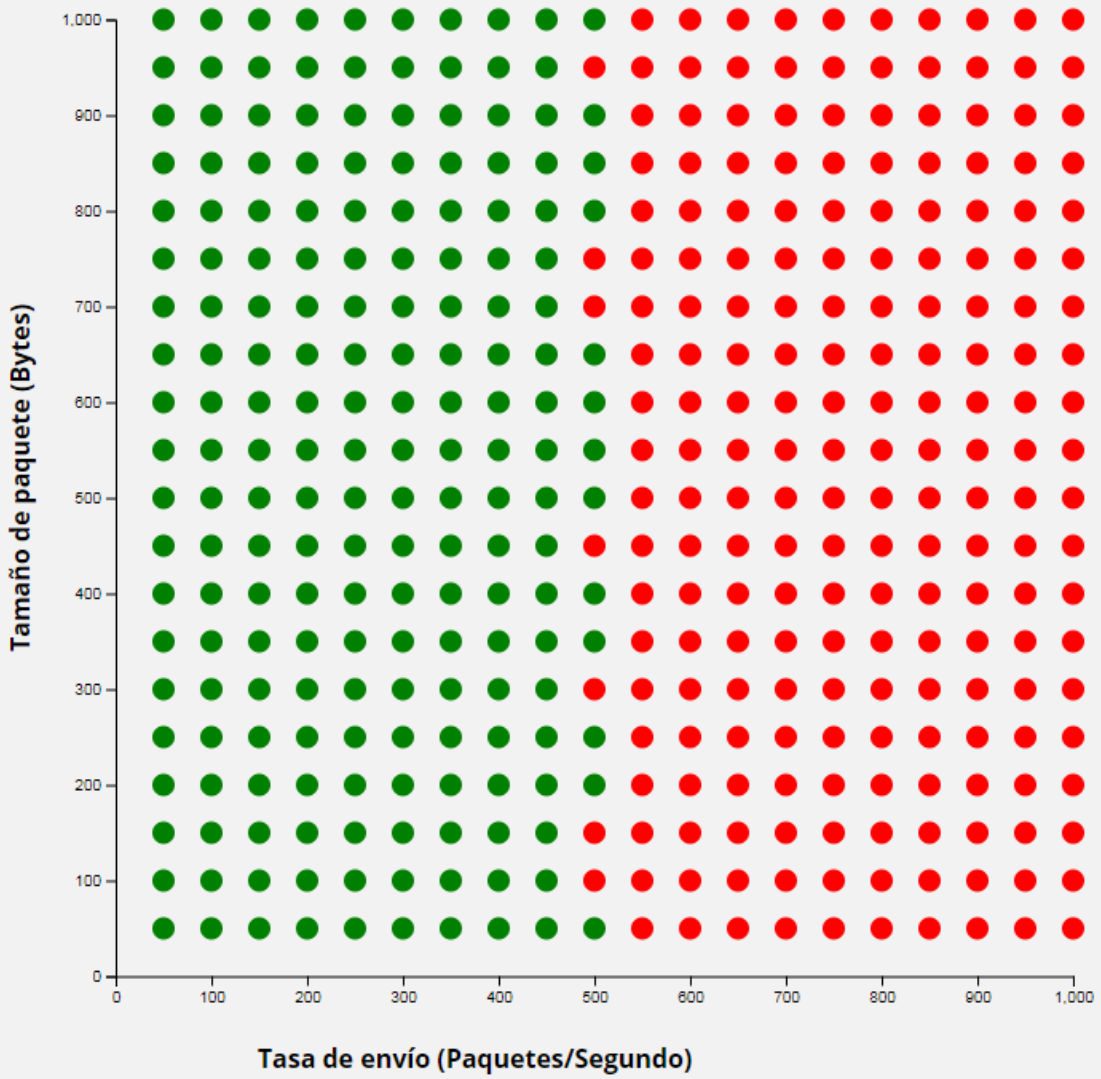


Figura 12 – Gráfica de éxitos y fallos obtenida para el Montaje - A

Representamos en ella cada uno de los pares de datos (tasa, tamaño) y le asignamos un color verde si se encuentra por debajo de un umbral definido como aceptable y rojo si se encuentra por encima de este. Gracias a esta representación podemos diferenciar dos áreas en la figura, el área que forman los puntos verdes, sin errores en la transmisión y el área formada por los puntos rojos, que representan los test que sufrieron errores de transmisión.

Debemos tener en cuenta que aunque nuestra gráfica se encuentren representados los ejes x, tasa de envío, e y, tamaño de paquete, linealmente, la saturación del enlace queda condicionada por el ancho de banda, siendo este la multiplicación de ambas magnitudes, es decir el área cubierta desde el punto origen y multiplicado por ocho, para realzar la conversión de bytes a bits.

Analizando los datos de esta gráfica, podemos definir una frontera de áreas que divida el área de éxitos en el test del área de fallos. Calculamos esta frontera como el punto medio entre el último test exitoso y el primer test fallido para los tests exitoso colindantes con fallos y viceversa, teniendo en cuenta todos los vecinos de estado opuesto para calcular los puntos medios.

El cálculo de esta frontera es útil para realizar una comparación entre escenarios. Podemos definir el área de éxito como el espacio que encierra la frontera a su izquierda, es decir, aquellos test que no han sufrido pérdidas de saturación, y el área de fallos como aquellos test que han visto sus datos afectados al ser transmitidos.

De esta manera, una disminución del área de éxitos equivale a un empeoramiento de la capacidad de transmisión de la red en cuanto a rango de paquetes. Al disminuir el área de éxitos, saturaríamos antes la red y sufriríamos pérdidas de paquetes y retransmisiones de los protocolos de capa superior. Asimismo, podemos definir un cálculo gráfico aproximado de la eficiencia como la relación entre el área de éxitos y el área total cubierta por los test; teniendo en cuenta que el área total cubierta por los test puede no representar todas las combinaciones de envío posibles en una red, que abarcarían hasta la MTU en el eje Y, tamaño de paquete, y hasta el número máximo de paquetes por segundo que pudiera mandar el más lento de los dispositivos que intervengan en cada ruta de transmisión.

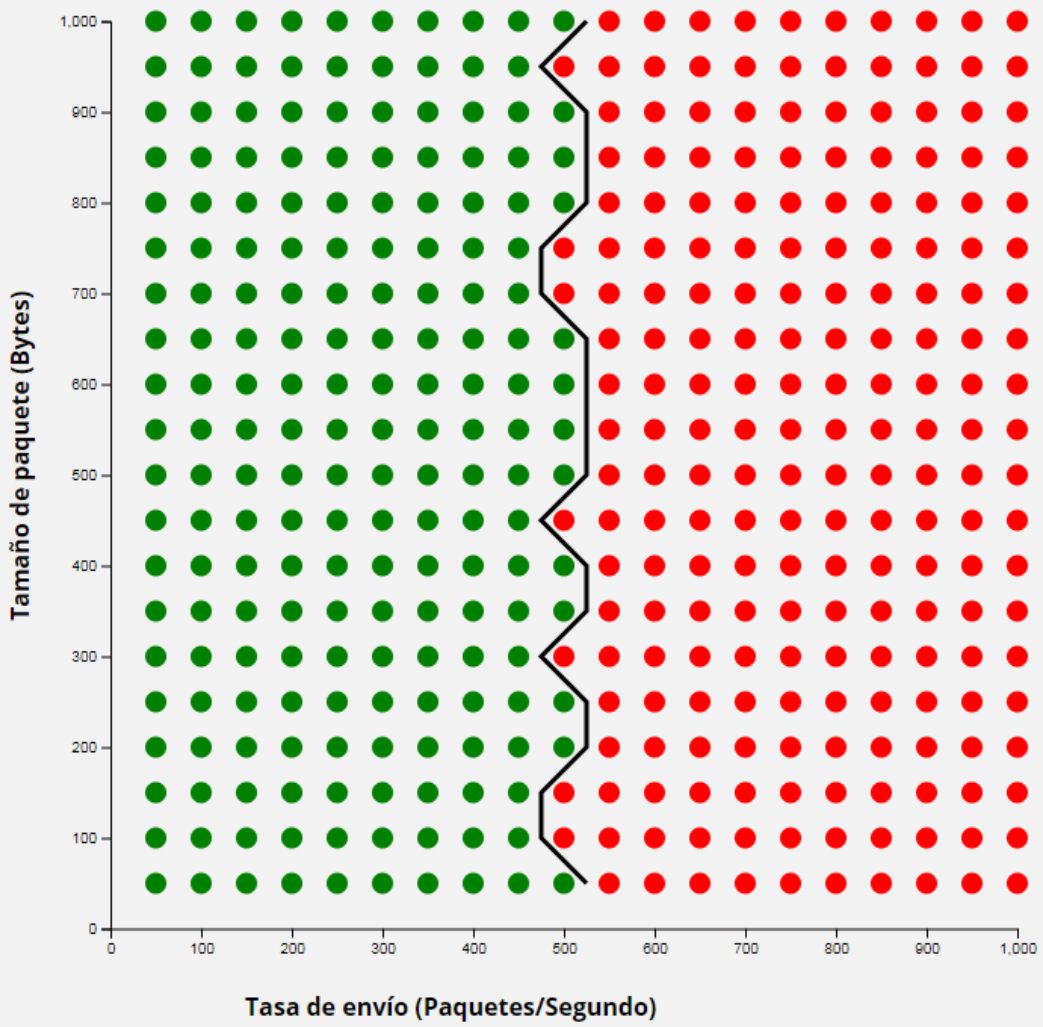


Figura 13 – Cálculo de la frontera umbral para el Montaje - A

Utilizando este procedimiento, hemos obtenido la gráfica para el canal WiFi limpio en las menores condiciones de tráfico posibles, por ejemplo, a horas nocturnas cuando nadie usa esos canales inalámbricos. Estos resultados son los que nos servirán como medida comparativa para las siguientes mediciones. Puesto que no queremos expresar la mejora o empeoramiento debidos al uso de las diferentes técnicas en medidas absolutas, pues estas serían relevantes únicamente para nuestra red y configuración, sino que las queremos expresar en medidas relativas como el porcentaje de variación con respecto a un canal limpio. A primera vista en estas gráficas vemos que es mayor la influencia de la tasa de envío que la del tamaño de los paquetes, esto es debido a que mayor tasa implica más computación, siendo esta última el cuello de botella de los elementos de red.

Después de haber obtenido esta gráfica, creamos un túnel LISP utilizando el script *launch-lisp.sh* (incluido en el Anexo E: Scripts utilizados en los tests) y volvemos a realizar las pruebas de barrido de saturación. Podemos entonces medir la variación provocada por el uso de un túnel LISP en la transmisión y saber si se trata de una medida beneficiosa y para qué perfil de tráfico y en qué medida.

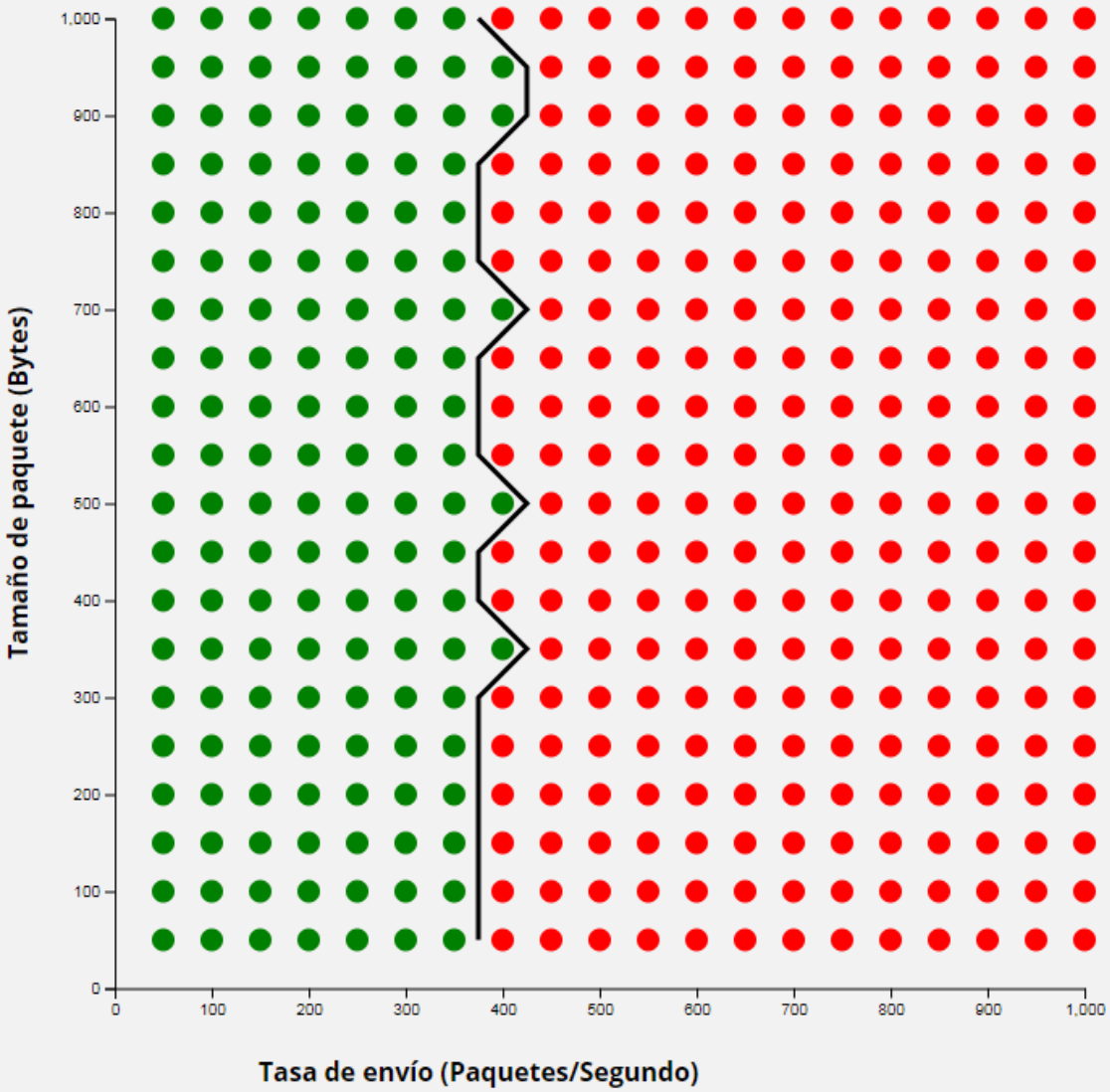


Figura 14 – Gráfica de éxitos y fallos obtenida para el Montaje - B

Aplicando los criterios previamente descritos, podemos observar cómo la frontera se ha desplazado hacia la izquierda. Al añadir LISP estamos añadiendo una cabecera nueva al paquete

y tratando la cabecera inicial como parte de los datos del túnel, es decir, estamos aumentando el tamaño del Payload, luego es lógico que el canal se sature antes teniendo en cuenta que la tasa y el tamaño de paquete hacen referencia a los datos útiles. El canal se sigue saturando a las mismas condiciones, pero ahora está transportando paquetes menos eficientes: más cabeceras y menos datos.

Añadimos ahora seguridad al escenario configurando el Montaje – C, que incluye el uso de un túnel IPSec para transmitir los datos que van a circular por el túnel LISP. Para securizar estos datos, estamos añadiendo una cabecera adicional, la cabecera ESP y un segmento de comprobación de la autenticación al final de los datos si utilizamos el modo transporte, además de una cabecera extra IP si utilizamos el modo túnel.

Por ello, al realizar las pruebas de barrido obtenemos una frontera desplazada hacia la izquierda. Estamos sufriendo el mismo efecto que en el cambio del Montaje – A al Montaje –B, la disminución de la eficiencia debido al incremento del tamaño relativo de cabeceras con respecto al total del paquete enviado.

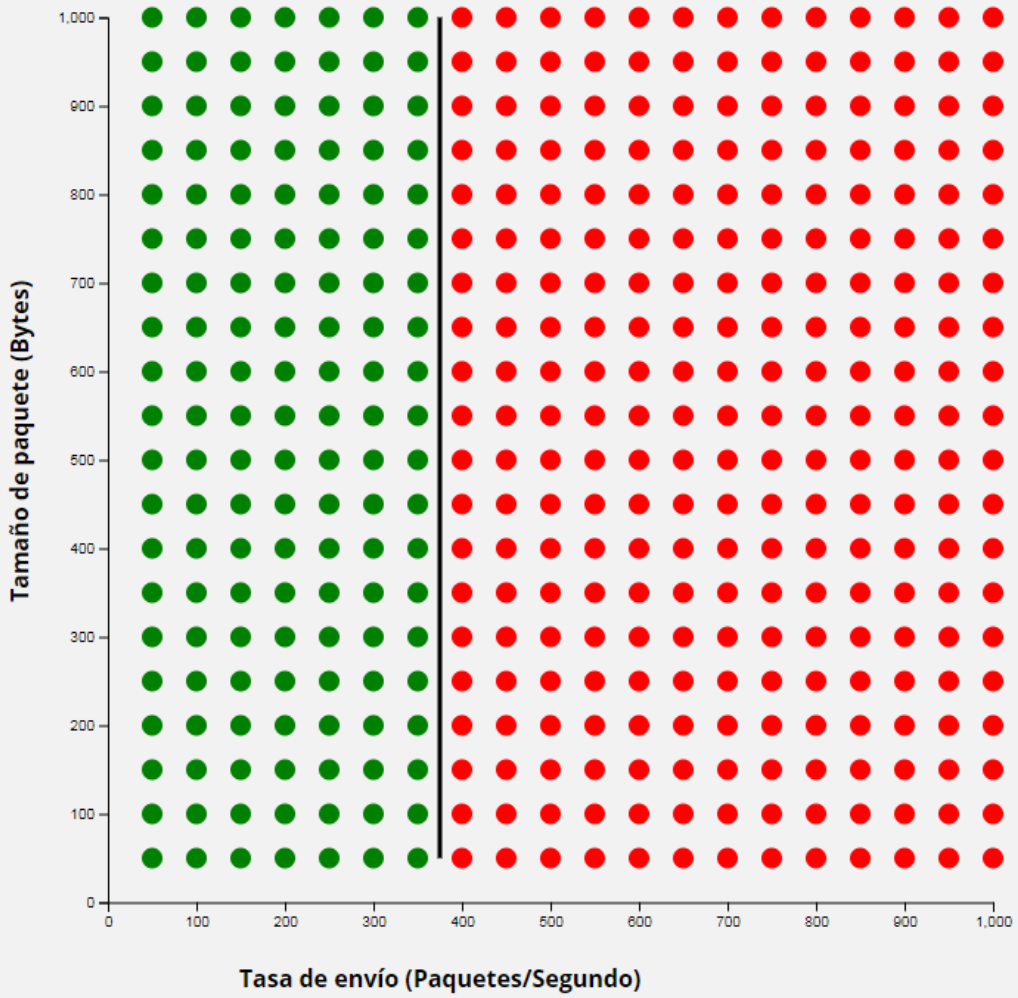


Figura 15 – Gráfica de éxitos y fallos obtenida para el Montaje - C

Preparamos ahora el Montaje – D añadiendo seguridad extremo a extremo en el escenario. Esto es desde el Host – 1 al Host -2 para las pruebas que nos ocupan. Los test de saturación nos indican en este caso que hemos vuelto a perder eficiencia al incluir una nueva cabecera a los datos. En este escenario estamos securizando dos veces los datos que circulan por el enlace WiFi, una vez debida a la securización extremo a extremo entre el Host-1 y el Host-2 y otra debida a la securización de la conexión LISP.

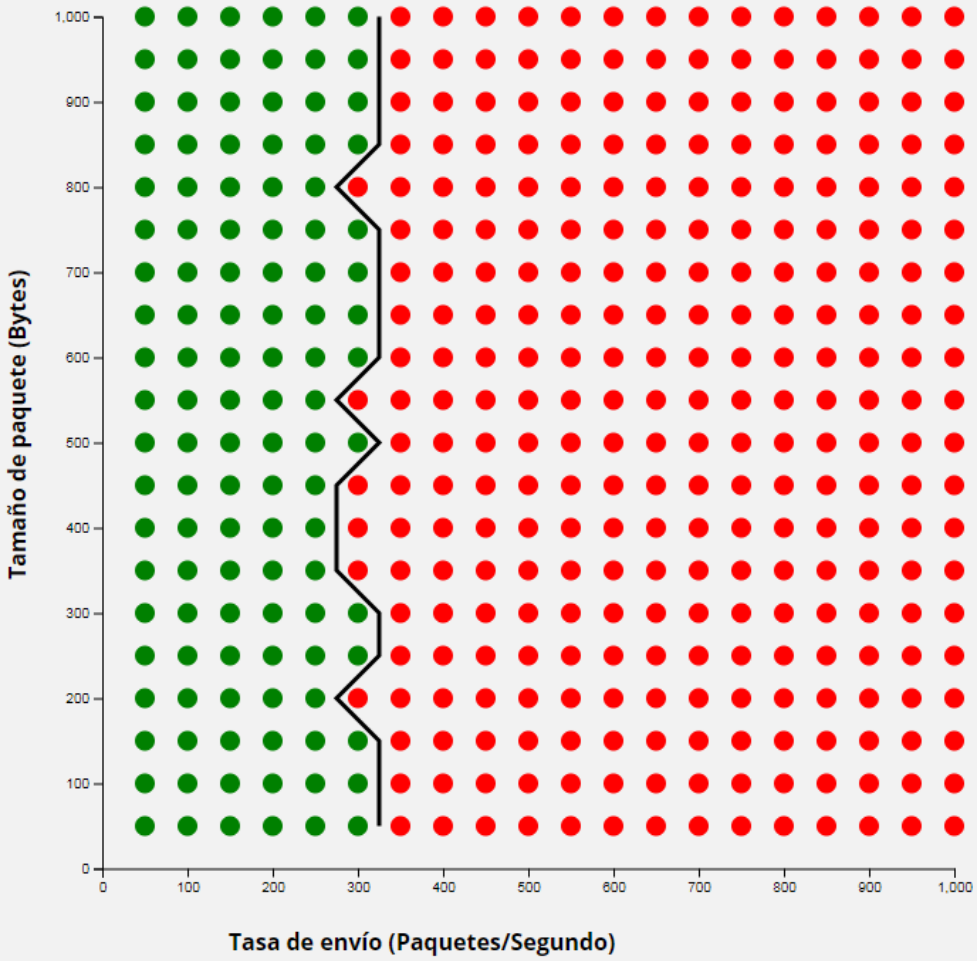


Figura 16 – Gráfica de éxitos y fallos obtenida para el Montaje - D

Para intentar mejorar esa pérdida, podemos prescindir a la securización de los datos en la conexión LISP confiando en la seguridad extremo a extremo. Esto tiene sus ventajas y sus inconvenientes. Por un lado mejoramos la eficiencia de la transmisión en la red al reducir una cabecera IPSec, por otro lado estamos delegando la seguridad de las transmisiones en el nivel de aplicación del host que realice el envío. La aplicación que quiera transferir datos seguros tendrá la responsabilidad de añadir una capa de seguridad entre el nivel de aplicación y el nivel de transporte para proporcionar cifrado y/o autenticación a la información.

En el Montaje – E, recuperamos eficiencia al eliminar cabeceras, y por ello nuestra frontera de áreas se desplaza hacia la derecha.

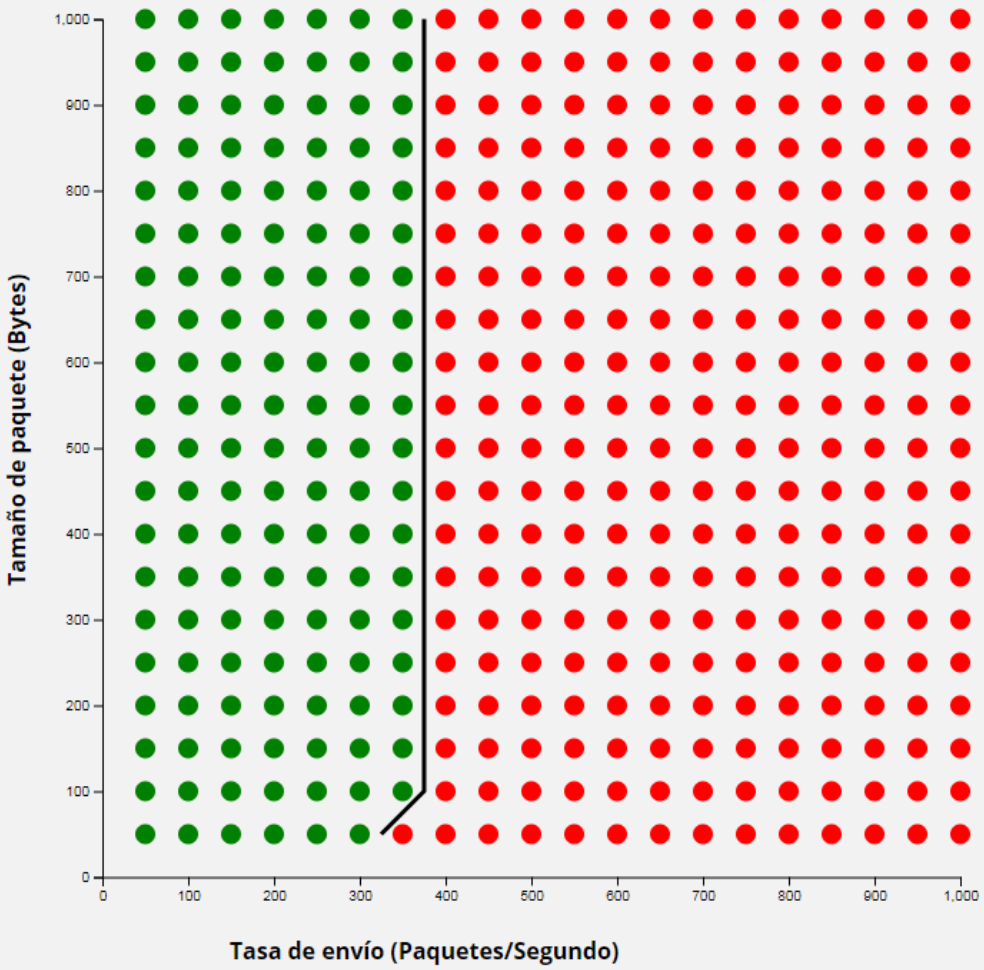


Figura 17 – Gráfica de éxitos y fallos obtenida para el Montaje - E

Por último, planteamos el Montaje – F como la mejora con respecto al Montaje – E al volver a añadir la seguridad a la conexión del túnel LISP pero sin recurrir al incremento del overhead que hacía disminuir la eficiencia en el Montaje – D.

Para ello, configuramos un escenario en el cual la conexión LISP entre R-4 y R-5 crea dos túneles. Uno de ellos con seguridad IPSec y el otro sin ella. Ambos túneles presentan las mismas direcciones de origen y destino para ser transparentes al resto de la red pero con diferentes políticas.

Una primera aproximación sería direccionar, por un lado, por el túnel configurado sin seguridad todo aquel tráfico que podamos garantizar que ya se encuentra securizado. Este es el caso de los protocolos de securización que mapean su estado en el campo protocolo de los paquetes IP. El

protocolo ESP lo hace especificando el valor 50 en este campo, y el protocolo AH, aunque siendo un protocolo de autenticación, no de encriptación, lo hace especificando el número de protocolo 51.

Por otro lado redireccionaríamos por el túnel al que le añadimos seguridad, todo aquel tráfico que no podamos garantizar que es seguro. Esto incluye tanto el tráfico no seguro, como el tráfico seguro que no mapee sus condiciones en los paquetes IP. Esta aproximación, denominada *Flipsec*, conlleva la necesidad de detección del tráfico a su llegada al equipo que gestiona la salida LISP y su gestión para reencaminarla según la salida LISP que le corresponda. Podemos realizar este procedimiento mediante el uso de reglas de firewall IPTables, somos capaces de enrutar el tráfico según sus características.

Como segunda aproximación, podemos tratar de identificar los paquetes securizados que no mapeen su condición en los campos de IP. Para ello podemos recurrir a la utilización de la librería *libnetqueue_filter*, para sistemas Linux, que actúa sobre las colas de IPTables, proporcionándoles acceso al espacio de ejecución de aplicaciones. Es decir, un paquete llega al dispositivo, mediante filtros de IPTables lo dirigimos al espacio de usuario y allí evaluamos si se encuentra securizado o no mediante la inspección con herramientas de nivel de transporte.

4.2 PRUEBAS DE MEJORA DE CAUDAL

Mediante esta aproximación lo que hacemos es estudiar qué incremento de datos pueden llegar a enviarse en una transmisión que no llega a estar en saturación. Las técnicas de ingeniería de tráfico nos permiten aumentar la tasa de salida.

4.3 PRUEBAS DE MULTIPLEXACIÓN

Las pruebas de multiplexación nos permiten analizar el comportamiento de la red en función de los parámetros de SimpleMux. Podemos así comprobar el efecto que tiene restringir la multiplexación a un número determinado de paquetes, por ejemplo, dos. Reduciendo con ello el tiempo de espera para la generación y envío de un nuevo paquete multiplexado.

Capítulo 5

Conclusiones y líneas futuras

5.1 CONCLUSIONES

Hemos verificado que los razonamientos basados en los cálculos analíticos teóricos seguían un razonamiento lógico y son transportables de los modelos teóricos a los modelos físicos.

Hemos creado una plataforma de pruebas adecuada para profundizar en el estudio de la eficiencia para técnicas de Ingeniería de tráfico con objeto de mejorar la Calidad de Servicio en las transmisiones.

Hemos desarrollado una metodología gráfica de análisis de la eficiencia aplicable genéricamente sobre la que poder razonar la caracterización de una red.

Hemos encontrado una configuración alternativa (*Flipsec*) a las hasta ahora habituales (Montaje - F) que proporciona los beneficios de un cifrado *End2End* y de un cifrado del túnel de canal sin incurrir por ello en la pérdida de eficiencia por doble túnel.

5.2 LÍNEAS FUTURAS

Incluir en el banco de pruebas la posibilidad de utilizar distintas distribuciones estadísticas o ficheros de capturas de aplicaciones reales como videojuegos o Voz sobre IP.

Profundizar en los resultados mediante el desarrollo de las pruebas de mejora de caudal y las pruebas de multiplexación y representar sus resultados con el mismo diagrama de áreas que hemos utilizado en este PFC para comprobar la verosimilitud de las técnicas gráficas descritas en el Capítulo 4: Metodología.

Mejorar los scripts de pruebas desarrollados con el objetivo de hacerlos adaptativos a sus medidas, mejorando la resolución haya donde se midan variaciones más significativas de las magnitudes a estudiar.

Ponderar los resultados según la distribución estadística del paquete medio actual (similar a la vista en la Figura 2), dado que en el cálculo de la eficiencia por el método gráfico se tienen en

cuenta todos los test por igual, sin embargo, sabemos que en las redes de hoy en día predominan los paquetes pequeños. Por lo tanto sería más exacto dictaminar la eficiencia ponderando los resultados según la probabilidad de aparición de los paquetes.

Cuantificar los movimientos de la frontera para las distintas configuraciones analíticamente.

Continuar el desarrollo de la herramienta *Multimeter* desarrollando nuevas funcionalidad, como podrían ser:

- Integración con más aplicaciones: IPerf, Wireshark, MatLab, Netcat, SNMP, Sistemas de monitorización de la red (NMS), Traffic control y otras.
- Guardar los resultados en una base de datos para facilitar un posterior análisis más detallado y utilización como datos de entrada para otras herramientas.

Utilizar el banco de pruebas de saturación del enlace y sus resultados como la función de evaluación de un algoritmo genético que variando las distintas configuraciones de los protocolos utilizados para ingeniería de tráfico sea capaz de encontrar la mejor configuración para la red dado un perfil de tráfico concreto.

Portación del código de la herramienta a dispositivos móviles y a un servidor HTTP en la forma de aplicación web.