

# Proyecto Fin de Carrera

## Ingeniería de Telecomunicación

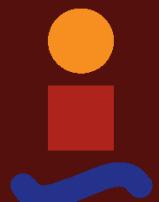
### Detección de anomalías en los registros de tráfico ofrecidos por IPFIX

Autor: Agustín Walabonso Lara Romero

Tutor: Rafael Estepa Alonso

Dpto. de Ingeniería Telemática  
Escuela Técnica Superior de Ingeniería  
Universidad de Sevilla

Sevilla, 2018





Proyecto Fin de Carrera  
Ingeniería de Telecomunicación

# **Detección de anomalías en los registros de tráfico ofrecidos por IPFIX**

Autor:  
Agustín Walabonso Lara Romero

Tutor:  
Rafael Estepa Alonso  
Profesor titular

Dpto. de Ingeniería Telemática  
Escuela Técnica Superior de Ingeniería  
Universidad de Sevilla  
Sevilla, 2013



Trabajo Fin de Grado: Detección de anomalías en los registros de tráfico ofrecidos por IPFIX

Autor: Agustín Walabonso Lara Romero

Tutor: Rafael Estepa Alonso

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal



*A mi familia*

*A mis profesores*

*A mis amigos*



# Agradecimientos

---

En primer lugar, quiero agradecer a mis padres, Agustín y María Victoria por haberme permitido realizar mis estudios, formarme como persona y apoyarme en todo lo que me he propuesto.

En segundo lugar, agradecer a todas las personas con las que paso mucho tiempo y siempre están conmigo, Manuel Ferrera, Andrés, Ismael, Juan, Curro, Juan Ángel, Francisco, Lourdes. En definitiva, a todos mis compañeros de clase por haber estado tan unidos, siempre disponibles para ayudar al que lo necesitaba.

En tercer lugar, agradecer a toda mi familia, con especial cariño a mi abuela, por todo el gran apoyo que me han brindado.

Gracias también a Mónica Pérez por estar todos los días ayudándome en todo, dándome un gran apoyo y saber que siempre estás ahí para todo.

Gracias a todos mis profesores por toda la formación y conocimiento proporcionado a lo largo de mi etapa de estudiante.

No se me olvida agradecer a mis compañeros de Everis el haberme permitido trabajar con ellos y enseñarme tanto en mis prácticas de empresa ¡Espero que podamos volver a trabajar juntos!

También agradecer a mis grandes compañeros de Hapkido que somos una gran familia.

Por último y no menos importante, agradecer al departamento de telemática por haber conseguido motivarme y querer trabajar en el mundo de la telemática. En especial, agradecer a mi tutor de proyecto, Rafael Estepa, por ayudarme y permitirme llevar a cabo este proyecto, así como motivarme en el mundo de la ciberseguridad. Gracias por su asesoramiento y corrección.

*Agustín Walabonso Lara Romero*

*Sevilla, 2018*



# **Resumen**

---

El de la ciberseguridad es un campo que siempre está evolucionando. Esto hace que sea uno de los escenarios más complejos, pues los enfoques y paradigmas cambian continuamente.

Cada año se encuentran un gran número de vulnerabilidades nuevas en los diferentes sistemas, y es por ello por lo que constantemente se desarrollan técnicas y software que eliminan, bloquean ó identifican estas vulnerabilidades.

El objetivo de este documento es la investigación y desarrollo de técnicas de detección de comportamientos anómalos en el tráfico de red, así como la realización de pruebas que permitan valorar el funcionamiento de los mismos.



# **Abstract**

---

Cybersecurity is a constantly evolving field. That makes it particularly complex since its approaches and paradigms change frequently. Each year, a large number of new vulnerabilities across different systems is identified. To tackle these, new techniques and software are constantly being developed as to eliminate, block or identify these vulnerabilities. This document's objectives are to research on and develop techniques to identify anomalous behaviour in network traffic, as well as running performance tests allowing for the evaluation of their functioning.

# Índice

---

<b>Agradecimientos</b>	<b>9</b>
<b>Resumen</b>	<b>11</b>
<b>Abstract</b>	<b>13</b>
<b>Índice</b>	<b>14</b>
<b>Índice de Tablas</b>	<b>16</b>
<b>Índice de Figuras</b>	<b>18</b>
<b>1 Introducción y objetivos</b>	<b>1</b>
1.1 <i>Introducción</i>	1
1.2 <i>Motivación</i>	2
1.3 <i>Metodología de trabajo</i>	3
1.4 <i>Objetivos</i>	4
<b>2 Estado del arte</b>	<b>7</b>
2.1 <i>Vulnerabilidades en la red.</i>	7
2.1.1 Man In The Middle.	8
2.1.2 Denegación de servicios (DOS).	8
2.1.3 Ransomware.	9
2.2 <i>Detección de anómalías en la red.</i>	10
2.2.1 Redes neuronales.	10
2.2.2 R&S®PACE 2.	11
2.2.3 Firewall de nivel de aplicación.	12
2.2.4 IDS/IPS.	12
2.2.5 Análisis y conclusiones.	14
<b>3 Conceptos</b>	<b>17</b>
3.1 <i>IPFIX y Netflow.</i>	17
3.2 <i>DPI.</i>	18
3.3 <i>nDPI.</i>	19
3.4 <i>wireshark.</i>	20
3.5 <i>Integración de nDPI con wireshark.</i>	20
3.6 <i>Recolector nProbe.</i>	21
<b>4 Sistema</b>	<b>23</b>
4.1 <i>Aspectos generales del sistema.</i>	23
4.2 <i>Fase 1: Obtención del tráfico.</i>	24
4.3 <i>Fase 2: Exportación del tráfico en formato IPFIX.</i>	24
4.4 <i>Fase 3: Procesamiento de IPFIX.</i>	26
4.4.1 Lenguaje de programación y librerías.	26
4.4.2 Diseño del sistema.	27
4.4.3 Cálculo de los indicadores.	31
4.5 <i>Fase 4: Base de datos.</i>	33
4.5.1 Diseño de la base de datos.	33
<b>5 Pruebas y resultados</b>	<b>41</b>
5.1 <i>Descripción de software usado para la realización de las pruebas</i>	41

<b>5.2</b>	<i>Pruebas realizadas</i>	<b>41</b>
5.2.1	Captación de los datos y exportación IPFIX	42
5.2.2	Preparación del archivo de configuración	42
5.2.3	Ejecución del script	43
5.2.4	Consultas tablas BBDD	44
5.2.5	Errores	46
5.2.6	Indicador Unknown	47
<b>6</b>	<b>Conclusiones y líneas futuras</b>	<b>11</b>
6.1	<i>Conclusiones</i>	11
6.2	<i>Líneas futuras</i>	12
<b>Referencias</b>		<b>11</b>
<b>ANEXO A: Instalación del proyecto</b>		<b>14</b>

# ÍNDICE DE TABLAS

---

<i>Tabla 1 Comparativa de diferentes DPI.</i>	15
<i>Tabla 2 Comparativa de los diferentes cortafuegos.</i>	16
<i>Tabla 3 Comparativa de diferentes IDS.</i>	16
<i>Tabla 4 Campos exportados con nProbe.</i>	25
<i>Tabla 5 Descripción de las opciones usadas en nProbe.</i>	25
<i>Tabla 6 Librerías utilizadas.</i>	26
<i>Tabla 7 Parámetros del fichero de configuración.</i>	29
<i>Tabla 8 Tratamiento de las direcciones IP del tráfico.</i>	29
<i>Tabla 9 Diferentes indicadores.</i>	30
<i>Tabla 10 Indicador de aplicaciones.</i>	35
<i>Tabla 11 Indicador de ICMP.</i>	36
<i>Tabla 12 Indicador Puertos destino.</i>	36
<i>Tabla 13 Indicador Puertos destino.</i>	37
<i>Tabla 14 Indicador relación de las diferentes ip.</i>	37
<i>Tabla 15 Número total de ip destinos.</i>	38
<i>Tabla 16 Número total de las diferentes aplicaciones.</i>	38
<i>Tabla 17 Tabla principal de indicadores.</i>	39
<i>Tabla 18 Logs.</i>	40
<i>Tabla 19 Máquina virtualizada con vmware.</i>	41



# ÍNDICE DE FIGURAS

---

<i>Figura 1-1 Supervisión de matrices de tráficos ofrecidos en las redes.</i>	1
<i>Figura 1-2 Funcionamiento de Netflow</i>	2
<i>Figura 1-3 Ubicación en el modelo de capas OSI.</i>	3
<i>Figura 1-4 Panel de waffle.io</i>	3
<i>Figura 2-1 Definición de riesgo en los sistemas de información.[1]</i>	7
<i>Figura 2-2 Ataque Man In The Middle</i>	8
<i>Figura 2-3 Ataque de denegación de Servicios</i>	9
<i>Figura 2-4 Ramsonware</i>	10
<i>Figura 2-5 Modelo de red neuronal</i>	11
<i>Figura 2-6 Diferentas etapas de procesamiento de un paquete en la librería R&amp;S®PACE 2 [9].</i>	11
<i>Figura 2-7 Concepto de Firewall de aplicación.</i>	12
<i>Figura 2-8 Arquitectura de IDS según la metodología.</i>	13
<i>Figura 2-9 Logo de Snort</i>	13
<i>Figura 2-10 Gráfico de las vulnerabilidades en los últimos 5 años.</i>	14
<i>Figura 2-11 Vulnerabilidades por tipo [16].</i>	15
<i>Figura 3-1 Información exportada a través de Netflow.[21]</i>	17
<i>Figura 3-2 Tráfico tunelado.</i>	18
<i>Figura 3-3 Identificación de patrones usando DPI.</i>	18
<i>Figura 3-4 Añadir protocolos en nDPI.</i>	19
<i>Figura 3-5 Ejemplo de algoritmo Aho-Corasick. [27]</i>	19
<i>Figura 3-6 Logo de wireshark.</i>	20
<i>Figura 3-7 Captura de paquetes con wirehsark y nDPI.</i>	20
<i>Figura 3-8 certificados SSL.</i>	21
<i>Figura 3-9 Uso de nProbe.</i>	21
<i>Figura 4-1 Esquema general del sistema.</i>	23
<i>Figura 4-2 Ejemplo de exportación con nProbe.</i>	24
<i>Figura 4-3 Ejecución nProbe.</i>	25
<i>Figura 4-4 Logo de Python [34].</i>	26
<i>Figura 4-5 Diseño del sistema.</i>	27
<i>Figura 4-6 Media móvil.</i>	31
<i>Figura 4-7 Forma de media exponencial móvil.</i>	32
<i>Figura 4-8 Diagrama funcional de la base de datos.</i>	34

<i>Figura 4-9 Tablas que componen la base da datos.</i>	34
<i>Figura 5-1 Ejecución nprobe para la prueba.</i>	42
<i>Figura 5-2 Carpeta con los ficheros de flujos IP.</i>	42
<i>Figura 5-3 Configuración inicial.</i>	43
<i>Figura 5-4 Ejecución del sistema programado.</i>	43
<i>Figura 5-5 Indicadores temporales.</i>	44
<i>Figura 5-6 Consulta de tabla aplicaciones.</i>	45
<i>Figura 5-7 Consulta tabla estadísticos.</i>	45
<i>Figura 5-8 Segunda consulta a tabla estadísticos.</i>	46
<i>Figura 5-9 Ejecución del script con tráfico malicioso.</i>	46
<i>Figura 5-10 Consulta de logs.</i>	46
<i>Figura 5-11 Consulta en logs de nuevos indicadores.</i>	47
<i>Figura 5-12 Evolución temporal del indicador Unknown.</i>	48
<i>Figura 5-13 Evolución temporal indicador EMA.</i>	48
<i>Figura A-1 instalación del proyecto.</i>	14
<i>Figura A-2 Error en la instalación.</i>	14
<i>Figura B-1 Carpeta tfg.</i>	18
<i>Figura B-2 Carpeta indicadores.</i>	18
<i>Figura B-3 Carpeta IP.</i>	19
<i>Figura B-4 Carpeta 1.1.1.1</i>	19
<i>Figura B-5 Carpeta indicador aplicaciones.</i>	20







# 1 INTRODUCCIÓN Y OBJETIVOS

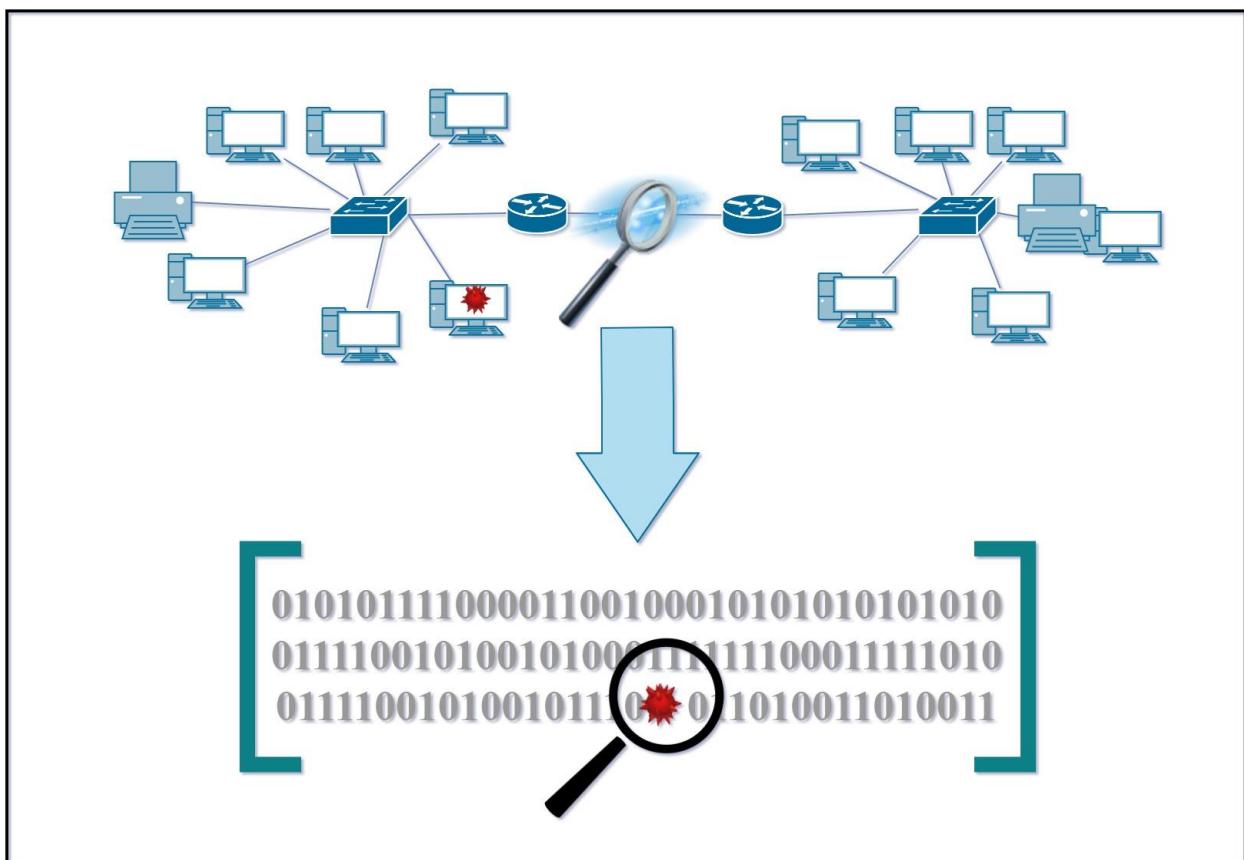
Creo que los virus informáticos deberían contar como vida. Creo que dice bastante sobre nosotros el hecho de que la única forma de vida que hemos logrado crear sea puramente destructiva. Hemos creado vida basada en nuestra imagen.

- Stephen Hawking-

## 1.1 Introducción

**E**n una red se realizan múltiples conexiones, son en estas dónde podemos detectar diferentes intrusiones ó software anómalos. Es por ello por lo que se realizan técnicas de tratamientos matriciales del tráfico. Dichas técnicas ofrecen la posibilidad de realizar estudios matemáticos con el fin de poder detectar anomalías en la información cursante de las redes.

Debido al gran número de vulnerabilidades existentes en las diferentes redes, justifica la gran necesidad del desarrollo continuo de sistemas capaces de llevar a cabo trabajos de detección de las mismas.



*Figura 1-1 Supervisión de matrices de tráficos ofrecidos en las redes.*

Un usuario que hace uso de dispositivos conectados a la red realiza una serie de acciones cotidianas que

pueden ser clasificadas y guardadas como patrones de comportamiento. Sobre estos, pueden llevarse a cabo estudios y toma de decisiones para comprobar la existencia de anomalías.

Las técnicas de detección de anomalías se basan en la clasificación y estudio de patrones. Cuando se identifica un patrón que contiene valores que difieren de su modelo, el sistema lo detectará como anómalo por lo que, a posteriori, se realizará un trabajo de supervisión para comprobar si resulta ser un falso positivo o, por lo contrario, se trata realmente de la detección de una anomalía.

Cabe destacar que estos sistemas siempre suelen alertar de falsos positivos, por lo que es muy importante llevar a cabo un buen estudio previo sobre las condiciones a trabajar y así poder configurar e implantar un modelo acorde tanto a las necesidades como a la escabilidad y el entorno sobre el que se desea operar.

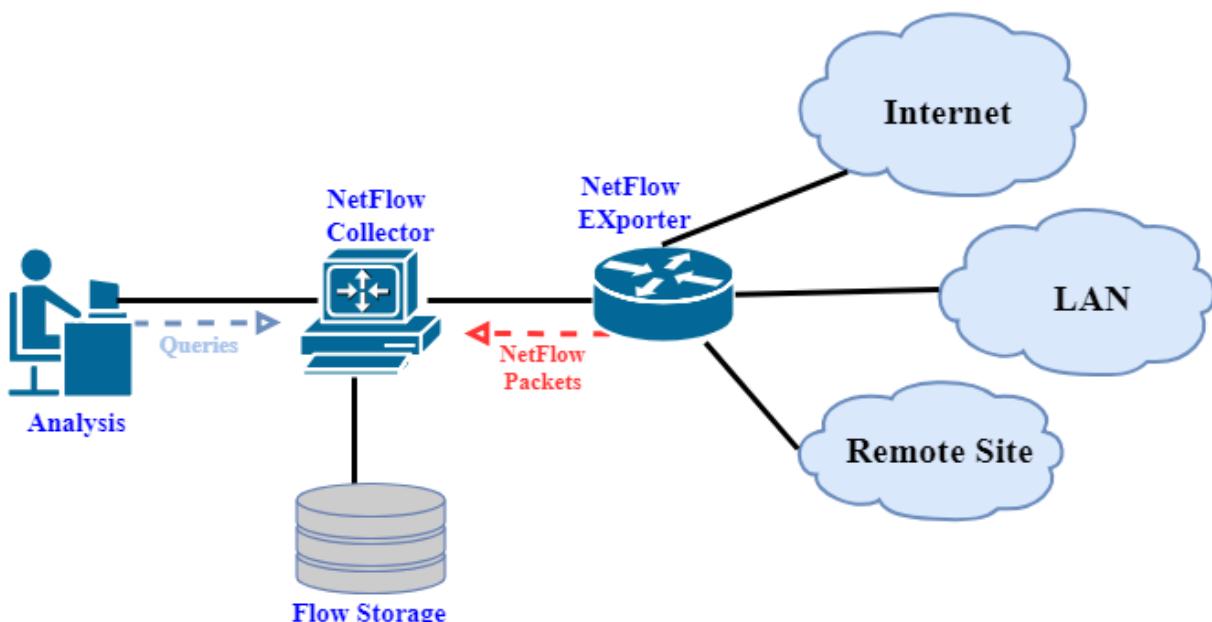
En una red de datos existe un intercambio continuo de flujos y en el presente trabajo se exponen modelos estadísticos así como la clasificación del flujo para la localización de anomalías.

## 1.2 Motivación

Para la detección de anomalías es necesaria la manipulación de una inmensa cantidad de información, de modo que para su procesamiento es necesario el uso de modelos estadísticos adecuados.

Resulta de gran interés realizar un estudio profundo de los paquetes que viajan por la red, pues esto permite obtener una información útil para ser estudiada y poder definir comportamientos habituales.

En la actualidad existen colectores netflow que recogen la información que viaja por la red. NetFlow es un protocolo de red diseñado para recolectar la información sobre tráfico IP, la cual hace posible la monitorización de las redes.



*Figura 1-2 Funcionamiento de Netflow*

Debido a las vulnerabilidades existentes en la red, mucho del tráfico cursado se encuentra cifrado. Por ejemplo, un usuario hace uso de una aplicación web que utiliza el protocolo HTTPS (Protocolo seguro de transferencia de hipertexto). Este es un protocolo de transferencia de datos de forma segura, es decir, el contenido que transporta se cifra, esto permite que si un atacante consiguiera hacerse con estos datos, dicho atacante, a priori, no podría interpretar la información substraída.



Figura 1-3 Ubicación en el modelo de capas OSI.

Debido a que el tráfico se transporta de forma cifrada se crea un gran interés de la realización de inspección profunda de paquetes, para poder identificar la aplicación que se encuentra detrás del tráfico cifrado.

### 1.3 Metodología de trabajo

Para el desarrollo del presente trabajo se ha llevado a cabo la metodología de trabajo Scrum. En la cuál se recogen un conjunto de buenas prácticas para trabajar y llevar a cabo la realización de un Proyecto.

Esta metodología de trabajo se basa en la división del proyecto en una lista de tareas pequeñas ordenadas por prioridad y puntos de esfuerzos. Estas pequeñas tareas pasan por distintas etapas hasta la finalización de las mismas.

También se ha optado por el uso de GitHub como repositorio del proyecto. Esto hace posible que se pueda llevar un buen control de versiones y cambios en el software desarrollado.

A través del siguiente enlace se puede obtener todo el trabajo subido en GitHub  
<https://github.com/agularrom/TFG>

Para poder unificar GitHub y la metodología de trabajo se ha decidido trabajar con waffle. Esta es una herramienta de uso libre que permite la planificación de un proyecto así como la sincronización del mismo en GitHub.

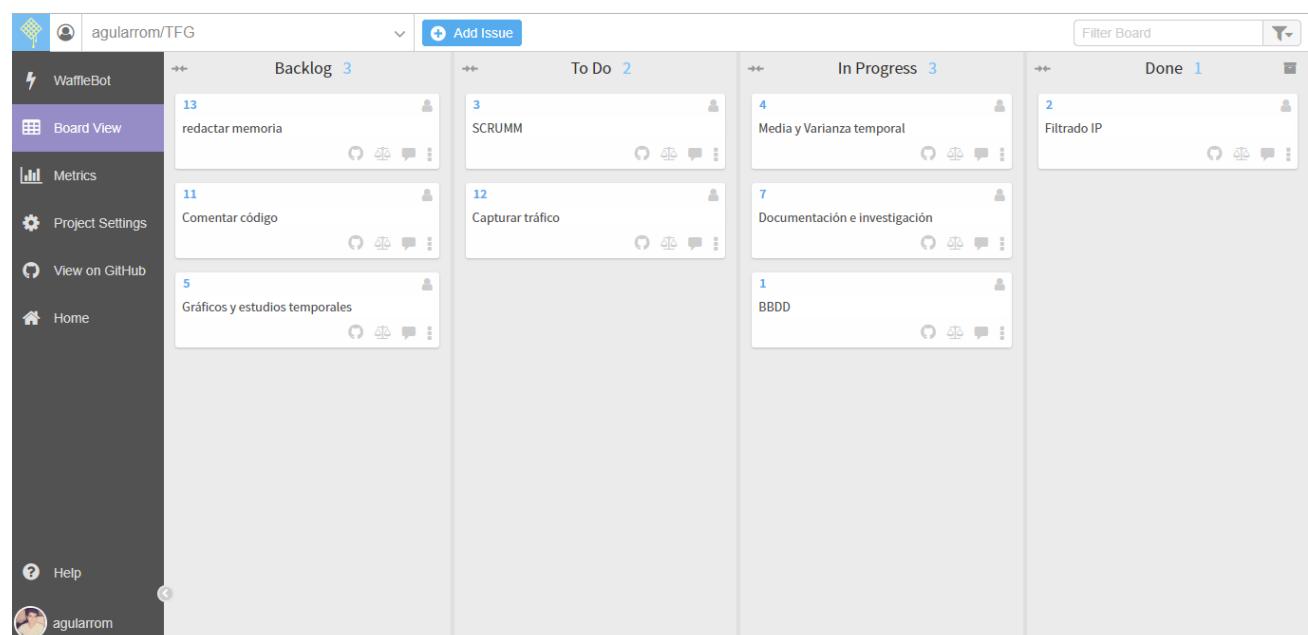


Figura 1-4 Panel de waffle.io

## 1.4 Objetivos

El enfoque de este trabajo consiste en el desarrollo de un sistema autónomo basado en la investigación estadística para solventar los problemas de detecciones de anomalías en redes de datos.

Los objetivos específicos de este trabajo se categorizan en el siguiente orden:

- Análisis de las diferentes tecnologías similares existentes.
- Análisis de librerías de software libre que realicen inspección profunda de paquetes de red y permitan identificar aplicaciones en el flujo de red.
- Análisis de los diferentes recolectores de flujo de red.
- Análisis de modelos estadísticos para la clasificación y toma de decisiones.
- Pruebas y funcionamiento de librerías.
- Desarrollo de un sistema real y escalable que pueda ser usado como método de defensa, así como la posibilidad de poder ser implantado en cualquier lugar.
- Búsqueda y generación de dataset para ser usado como entrada al modelo.
- Planteamiento de un escenario real.
- Instalación y configuración del escenario.
- Pruebas experimentales.
- Análisis y validación de los datos obtenidos.
- Conclusiones y posibles líneas de mejora.





## 2 ESTADO DEL ARTE

---

Una vez un ordenador me venció jugando al ajedrez, pero no  
me opusó resistencia cuando pasamos al kick boxing

- Emo Philips -

Este capítulo recoge el estudio de diferentes detectores de anomalías en tráfico de red, así como una introducción de las vulnerabilidades existentes en las redes. Para finalizar, se recoge una conclusión sobre las diferentes tecnologías mencionadas y la dirección que llevan.

### 2.1 Vulnerabilidades en la red.

Es muy fácil confundir vulnerabilidad con amenaza. Una vulnerabilidad se define como el fallo en un sistema de información. Dicho fallo compromete la integridad del sistema. Mientras que por otro lado, una amenaza es aquella que aprovecha una vulnerabilidad para acceder a un sistema de información con fines maléficos.[1]

Según el instituto de ciberseguridad de España, el riesgo en un sistema de información se encuentra en la intersección de la existencia de una vulnerabilidad, una amenaza y el propio sistema.



Figura 2-1 Definición de riesgo en los sistemas de información.[1]

En la red hay multiples vulnerabilidades ya sea inalámbrica o cableada. A continuación se explican diferentes ataques posibles que pueden ser llevados a cabo a través de una persona o empresa con fines maléficos.

### 2.1.1 Man In The Middle.

Este ataque (hombre en el medio) es una técnica que permite situar al equipo atacante en medio de la comunicación del equipo de la víctima y el router. Esto hace que el atacante substraiga información de la víctima. Este tipo de ataque resulta muy eficaz cuando la información que se transmite no se encuentra cifrada, puesto que, en caso contrario el atacante tendría que realizar técnicas de desencriptado para que la información substraída pueda ser legible. [2]

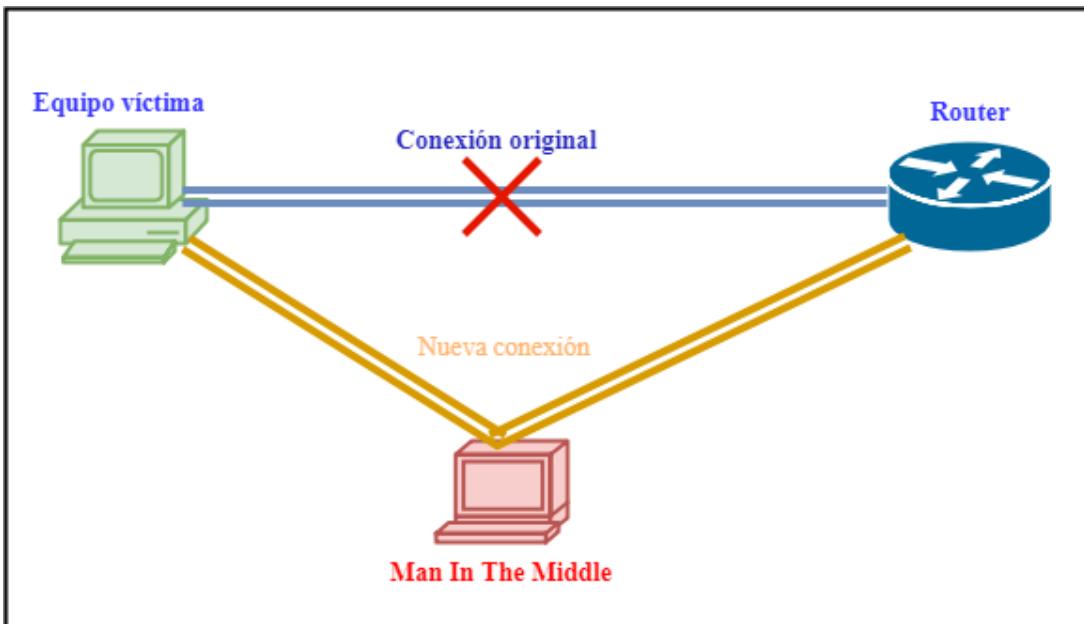


Figura 2-2 Ataque Man In The Middle

### 2.1.2 Denegación de servicios (DOS).

Este ataque es uno de los más conocido y extendido. Básicamente consiste en dejar inaccessible para los usuarios un servicio o recurso durante un cierto tiempo.

Este ataque suele ser usado como distracción para los administradores de la red y así poder realizar otro ataque más potente y con un claro objetivo.

Llama la atención que el 80% de la veces que se realiza este ataque resulta ser desde el interior de la red [3].

Un DOS típico, es la de dejar sin recursos a un servidor inundándolo a peticiones falsas con el fin de dejar inaccesible el servicio que ofrece.

Para conseguir que este ataque sea muy eficaz. Los atacantes suelen infectar una red de ordenadores para poder realizar un DOS masivo.

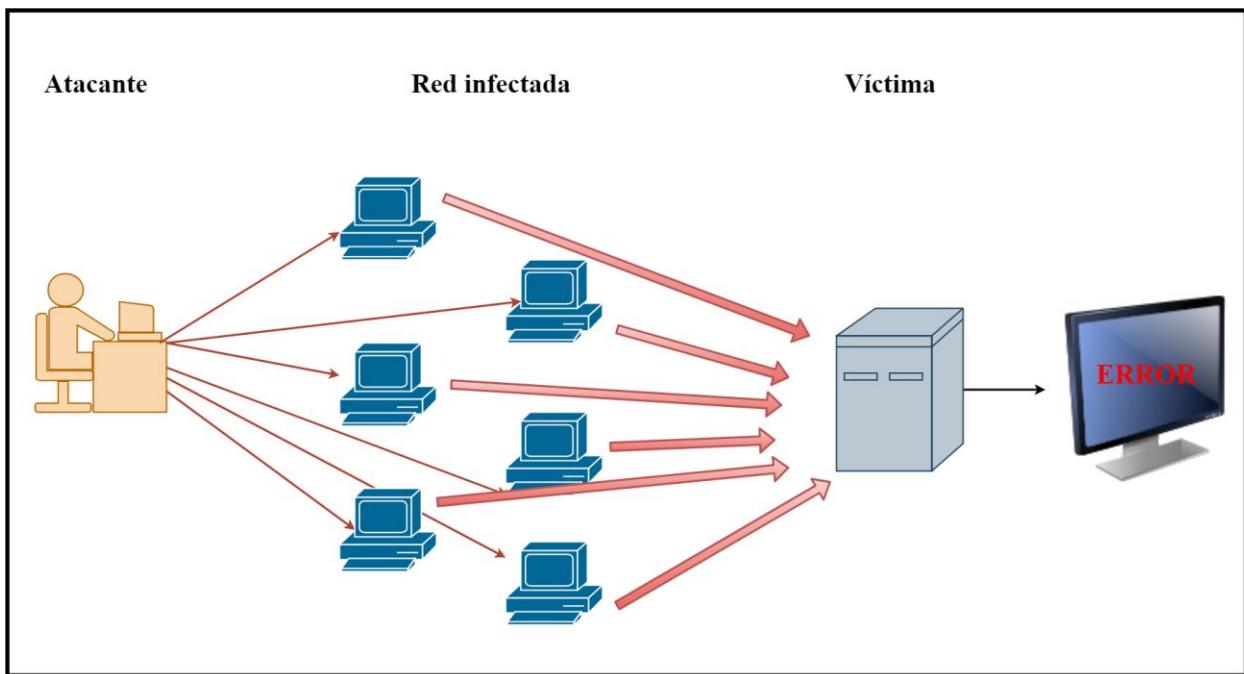


Figura 2-3 Ataque de denegación de Servicios

### 2.1.3 Ransomware.

Se clasifica este ataque como un malware capaz de acceder a documentos de la víctima con el fin de impedir el acceso a los mismos. El fin de este ataque es el de pedir un rescate por la recuperación de los documentos.

Una de las propiedades que tiene es la facilidad de propagación. Uno de los medios utilizados más habituales es el envío de correos maliciosos. [4]

El rescate siempre se exige en bitcoins, ya que es una criptomoneda que hace que sea difícil seguir el rastro de la trasacción llevada a cabo.

Para que el ataque sea efectivo los hackers realizan mucha ingeniería social para aprovechar las brechas de seguridad.

A continuación se citan algunas de las diferentes técnicas realizadas para conseguir infectar con el malware Ransomware:

- Obtención de cuentas con privilegios de administrador.
- Envío de spam.
- Suplantación de identidad para descargar un archivo infectado.



*Figura 2-4 Ramsonware*

## **2.2 Detección de anómalías en la red.**

Actualmente existen numerosas aplicaciones, librerías y técnicas para lograr detectar anomalías en el tráfico circulante. Para llevar a cabo esta tarea es muy importante realizar modelos estadísticos de tráfico limpio. Debido a que un modelo un incorrecto implica errores en la detección. Pudiendo causar un alto número de falsos positivos ó incluso la no detección de positivos. [5]

### **2.2.1 Redes neuronales.**

Los sistemas inteligentes cada vez sirven más de ayuda para dar solución a problemas cotidianos en todos los campos. Por ejemplo, en el mundo de la ciberseguridad son muy importantes para la detección de anomalías. Actualmente existen trabajos que hacen uso de redes neuronales para la predicción del tráfico [6], [7]. Esta labor se consigue a través del entrenamiento de un modelo neuronal del tráfico circulante en una red.

Estos sistemas contienen formados por neuronas divididas en tres capas [8]:

- Una primera capa de entrada que introduce los patrones al sistema.
- Una segunda capa denominada como ‘capas ocultas’ que son formadas por neuronas interconectadas entre sí formando multicapas.
- Por último una tercera capa constituida por los valores de salidas del sistema.

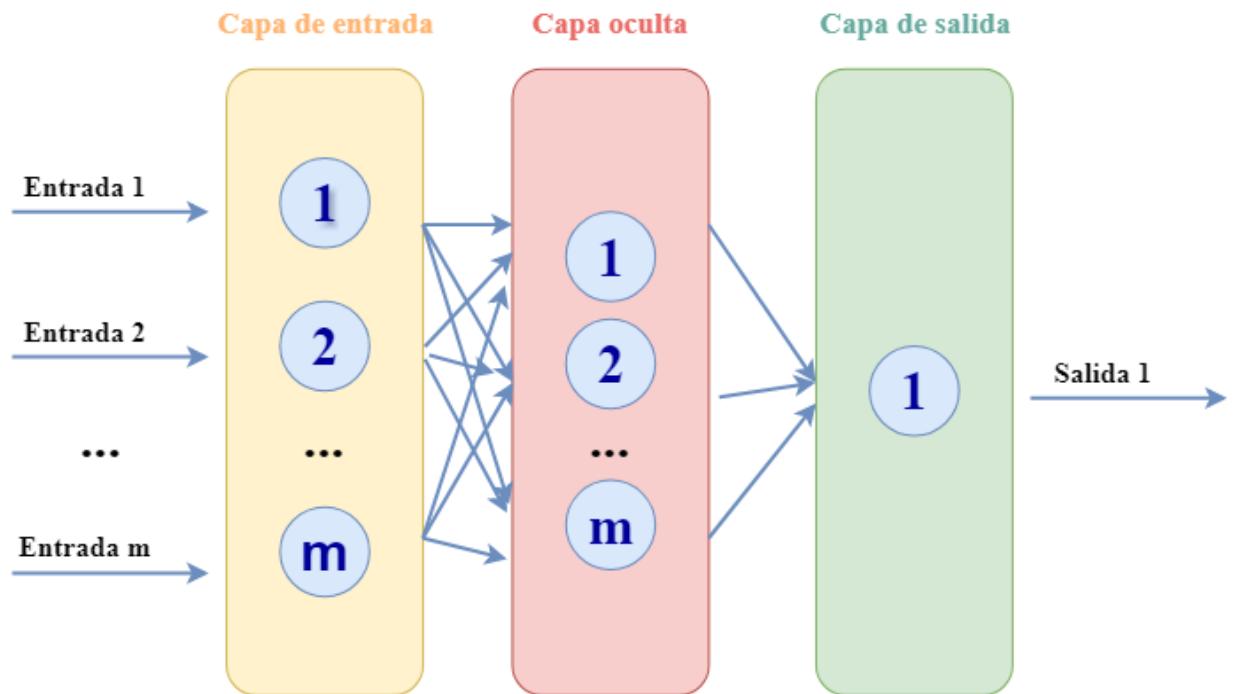


Figura 2-5 Modelo de red neuronal

## 2.2.2 R&S®PACE 2.

PACE es una librería software capaz de detectar múltiples aplicaciones y protocolos de red haciendo uso de diferentes técnicas de detección: [9]

- Inpección profunda de paquetes.
- Análisis de comportamiento heurístico y estadístico.

Esta librería es capaz de detectar aplicaciones y extraer metadatos de tráfico cifrado. Para conseguir todo esto, el software realiza los siguientes pasos:

1. Preparación de los paquetes.
2. Reordenación de los paquetes.
3. Clasificación de paquetes.
4. Decodificación.
5. Tiempo de espera de manejo.

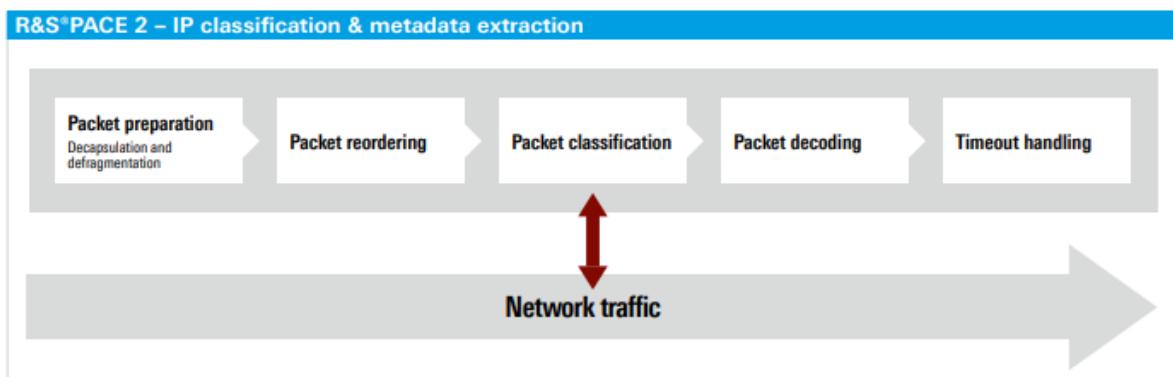


Figura 2-6 Diferentas etapas de procesamiento de un paquete en la librería R&S®PACE 2 [9].

Gracias al uso de diferentes técnicas de detección, esta librería consigue minimizar falsos positivos y negativos en la detección. Después de las pruebas realizadas por parte de la empresa del software, la marca afirma la identificación del 95% del tráfico circulante en la red. [9]

### 2.2.3 Firewall de nivel de aplicación.

Los firewall han evolucionado mucho en los últimos años y ya son capaces de actuar en la capa de aplicación del modelo OSI. Este tipo de firewall evalúa la capa de aplicación en los paquetes antes de permitir una conexión, también realiza un seguimiento de las conexiones así como la secuencia, con lo cual se consigue permitir tener un buen control de las conexiones e imponer un impedimento a las aplicaciones no deseadas.

Estos tipos de cortafuegos son conocidos como Proxy Firewall ya que son capaces de examinar y evaluar contraseñas, así como solicitudes de servicios que se encuentran en los datos de la capa de aplicación. Aplican por ejemplo reglas de segmentación en protocolos tales como HTTP, FTP, Telnet, etc.

También existe una variante de estos cortafuegos, denominados como firewall DPI. Estos son los más sofisticados. Pueden llegar a filtrar tipos de archivos específicos como XML ó SOAP.[10]

Aunque un firewall de aplicación proporciona bastante seguridad frente a un firewall clásico de filtrado de paquetes, el firewall de aplicación es más lento debido a que el procesamiento de los datos en el nivel de aplicación resulta ser más costoso. Esto provoca una escasa escalabilidad. [11]

ConfigServer Security Firewall (CSF) sería un ejemplo de firewall de nivel de aplicación.

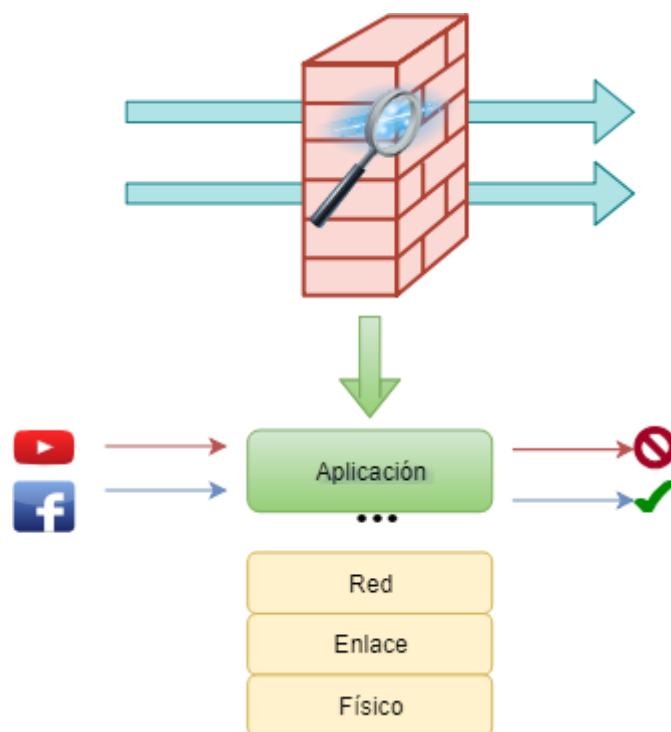


Figura 2-7 Concepto de Firewall de aplicación.

### 2.2.4 IDS/IPS.

La mayoría de las empresas u organizaciones desean prevenir intrusiones en su red. Los cortafuegos evitan ataques con procedencia del exterior, pero cada vez son más frecuentes ataques desde el interior de la red. Es por ello por lo que es tan importante la implantación de un sistema de detección de intrusos (IDS). Así como un sistema de prevención de intrusos (IPS) que una vez detecta y aprende un ataque, lo bloquea y genera

documentación del mismo.

Los IDS detectan ataques aplicando diferentes metodologías [12], [13]:

- Basado en firmas:
  - Realizan comparaciones de ataques con una base de datos de reglas ó firmas.
  - Posee limitaciones a la hora de la detección de nuevos ataques.
  - Generación de logs.
- Basado en anomalías:
  - Emplea un algoritmo de aprendizaje.
  - Posibilidad de detectar nuevos ataques con una probabilidad de acierto.
  - Generación de logs.

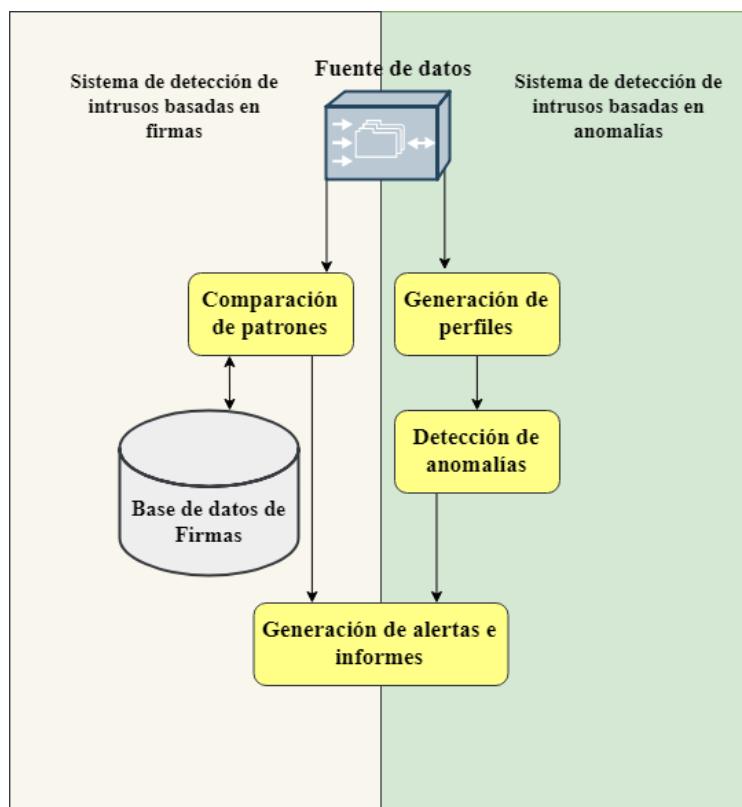


Figura 2-8 Arquitectura de IDS según la metodología.

Snort es un ejemplo claro de IDS que podemos encontrar en la actualidad. Tiene un motor de búsqueda y su metodología es basada en el análisis de firmas. Inspecciona el tráfico de red y permite la generación de alarmas. [14]



Figura 2-9 Logo de Snort

Snort funciona bajo licencia GPL, es decir, se trata de software libre. Contiene la característica de ser

multiplataforma, puesto que es posible ser ejecutado en Windows y UNIX/Linux. Así mismo, Snort lleva por defecto tanto reglas como patrones de vulnerabilidades y ataques ya conocidos e identificados.

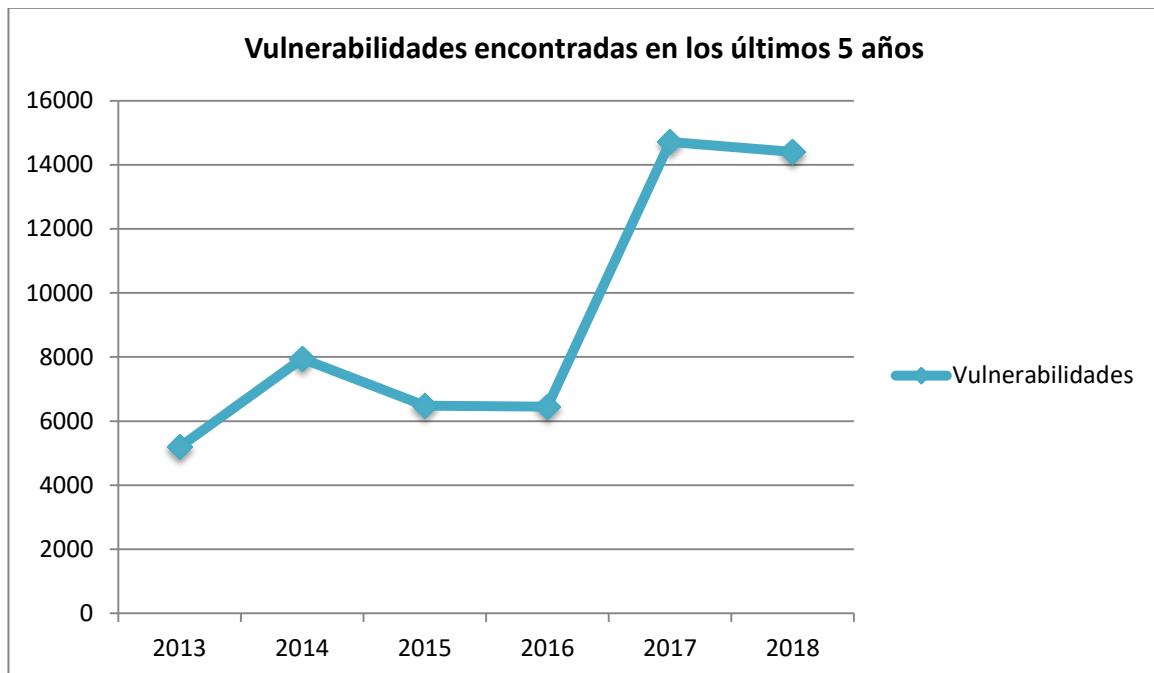
### 2.2.5 Análisis y conclusiones.

Según los datos obtenidos por CVE Details [15] podemos observar un aumento de las vulnerabilidades encontradas en los últimos 5 años. Resulta interesante observar el gran aumento de vulnerabilidades que hubo en el año 2017 respecto al año anterior.

Cabe señalar que a pesar de una pequeña minoría de vulnerabilidades en el año actual a la redacción del presente documento (2018) respecto al año 2017, aún sigue existiendo bastantes. Esto se debe al gran aumento de los diferentes dispositivos y tecnologías existentes en el mercado actual, los cuales hacen que sea de mayor envergadura el abanico de opciones para la localización y clasificación de nuevas vulnerabilidades.

Analizando estos datos cabe destacar la gran importancia de la investigación y desarrollo de nuevos sistemas para mitigar dichas brechas de seguridad. Por ello, existen un gran número de organizaciones en los diferentes sectores tanto públicos como privados trabajando e investigando en el desarrollo de nuevas técnicas para la detección y mitigación de las nuevas vulnerabilidades.

En el siguiente gráfico se muestra una evolución temporal de las vulnerabilidades en los últimos 5 años.



*Figura 2-10 Gráfico de las vulnerabilidades en los últimos 5 años.*

Según CVE Details [16] existen 13 tipos de vulnerabilidades, donde las más comunes son las de denegaciones de servicios ó la ejecución de código malicioso.

Debido a la gran variedad de vulnerabilidades, es muy importante el desarrollo de sistemas inteligentes capaces de detectar aplicaciones y datos no deseados.

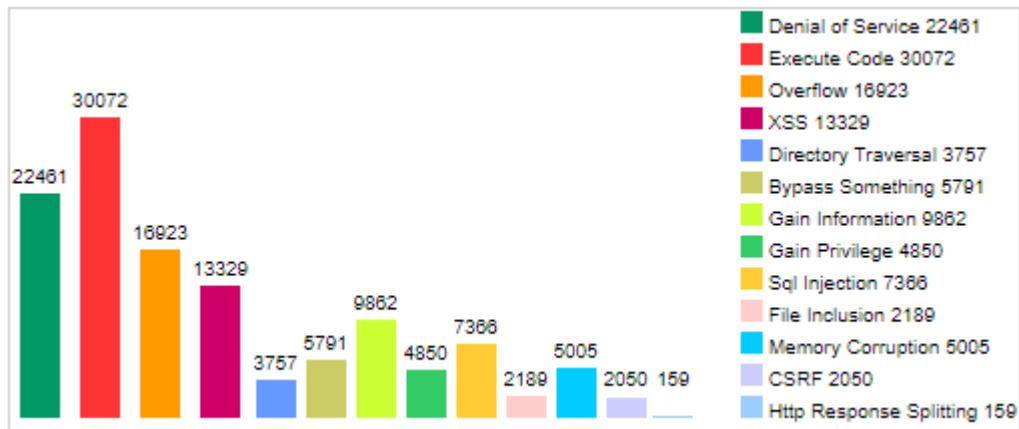


Figura 2-11 Vulnerabilidades por tipo [16].

Después de la documentación y análisis de diferentes librerías de software para la detección de aplicaciones en tráfico de red. Se puede garantizar que nDPI puede ser la mejor opción. Existe un artículo [17] donde se discuten las diferentes librerías existentes y se puede observar como nDPI resulta ser la mejor, puesto que es una librería de software libre y aún que a priori no es la librería que más aplicaciones puede detectar, se puede configurar como se desee para poder identificar protocolos y aplicaciones nuevas.

Existe una librería comercial llamada PACE que detecta numerosas aplicaciones. Tanto nDPI como PACE se basan en una antigua librería llamada OpenDPI.

Es cierto que en sus orígenes nDPI tenía números falsos positivos, pero tanto en las actualizaciones como en el desarrollo nuevas versiones esta eficiencia ha sido mejorada considerablemente.

A continuación se puede observar una comparativa de las librerías más famosas usadas en la actualidad.

Nombre	Versión	Aplicaciones	Sofware Libre
PACE	2	2800	NO
nDPI	2.4	185	SI
L7-Filter [18]	(May 2009)	110	SI
NBAR2 [19]	(Jun 2013)	~1000	NO

Tabla 1 Comparativa de diferentes DPI.

Como se ha mencionado anteriormente los cortafuegos han sufrido un gran cambio en los últimos años, debido a esto resulta de gran interés realizar un análisis y comparativa de los diferentes enfoques que puede tener un firewall.

En la siguiente tabla se puede observar una comparativa de los mismos.

Funcionalidad	Firewall clásico	Firewall stateful	Firewall Aplicación	Firewall DPI
Reglas básicas	SI	SI	SI	SI

<b>Reglas con relación entre paquetes</b>	NO	SI	SI	SI
<b>Reglas de aplicación</b>	NO	NO	SI	SI
<b>Eventos y logs</b>	SI	SI	SI	SI

*Tabla 2 Comparativa de los diferentes cortafuegos.*

Para finalizar se ha realizado un estudio de dos software muy usados en la actualidad para la implementación de sistemas de detección de intrusos (IDS). Cabe destacar que ambos poseen características muy similares.

Tanto Snort como Suricata pueden implementarse para ser sistemas de prevención de intrusos (IPS) y poder cortar ataques en tiempo real.

A continuación se muestra una comparativa de los softwares mencionados.

Nombre	Enfoque	Sofware libre	Registro de eventos	Detección de aplicaciones	Versión	Multihilo
<b>Snort</b>	Basado en firmas	SI	SI	SI	2.9.12	NO
<b>Suricata</b>	Basado en firmas	SI	SI	SI	4.0.5	SI

*Tabla 3 Comparativa de diferentes IDS.*

# 3 CONCEPTOS

You see, wire telegraph is a kind of a very, very long cat. You pull his tail in New York and his head is meowing in Los Angeles. Do you understand this? And radio operates exactly the same way: you send signals here, they receive them there. The only difference is that there is no cat.

- Albert Einstein -

En la investigación llevada a cabo se han visto involucradas diferentes tecnologías y mecanismos. Es por ello que en este capítulo se tratan los conceptos y metodologías empleadas para el desarrollo del proyecto.

## 3.1 IPFIX y Netflow.

IPFIX proviene de las siglas en inglés *IP Flow Information Export* [20], el cuál especifica un mecanismo estándar de exportación de datos sobre el flujo de red, tanto en comutadores como en routers.

Se puede definir como flujo de red un conjunto de paquetes IP pertenecientes a una misma conexión.

La compañía CISCO Systems desarrolló en 1996 una tecnología con el fin de mejorar el encaminamiento de los routers. Esta tecnología denominada Netflow, se basa en la identificación de los flujos establecidos entre distintos dispositivos para así poder agilizar el encaminamiento de paquetes IP. [21]

Un flujo de datos se caracteriza por la combinación de diversos atributos en un intervalo de tiempo. Estos atributos suelen ser direcciones, puertos, protocolo identificado, servicio, así como la interfaz de entrada.

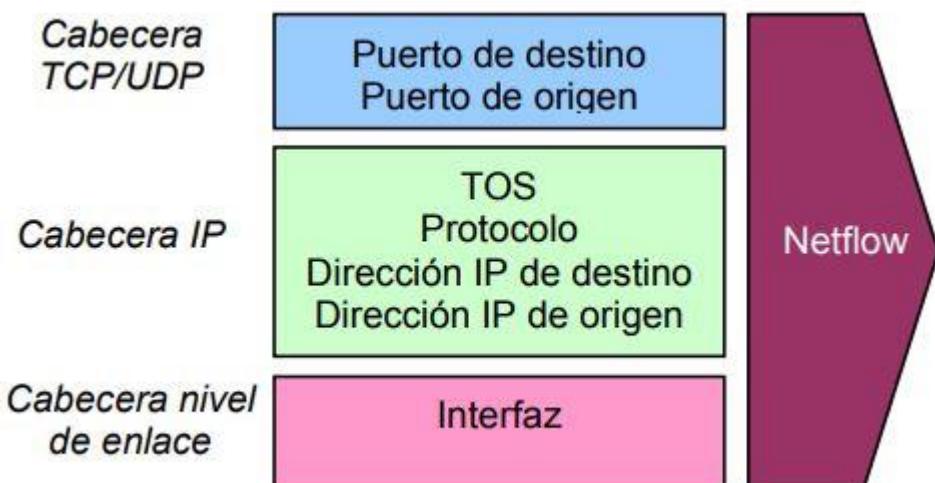


Figura 3-1 Información exportada a través de Netflow.[21]

Cuando un router identifica un nuevo flujo, Netflow lo guarda junto a la interfaz de salida asociada al flujo, esto hace que no sea necesario realizar consultas de encaminamiento para paquetes posteriores correspondientes al flujo. Consiguiendo disminuir el coste computacional que implica las consultas a las tablas de encaminamiento.

Una gran ventaja que provee esta característica, es la posibilidad de poder realizar mediciones y caracterizaciones del tráfico cursante en tiempo real.

Netflow es usado comúnmente para realizar análisis de la calidad del servicio a través de definiciones de métricas.

## 3.2 DPI.

Las primeras técnicas usadas para la identificación y clasificación de flujos de red estaban basadas en su mayoría, en el uso de puertos conocidos.[22] Es decir, se analizaban los encabezados de los paquetes para identificar un servicio en base al puerto. Por ejemplo, si llegaba tráfico asociado al puerto 80, dicho tráfico sería clasificado e identificado como HTTP. Este comportamiento presenta debilidades debido a que es impreciso. El tráfico cursante podría ir tunelado [23] consiguiendo así evitar reglas de un firewall.

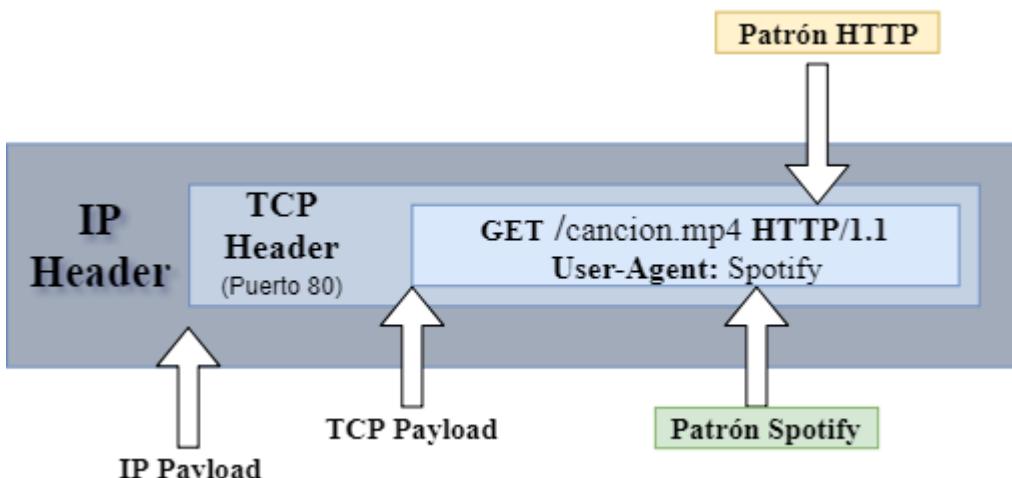


*Figura 3-2 Tráfico tunelado.*

Según un artículo [22] menos del 30% del tráfico es clasificado correctamente haciendo uso de identificación basada en puertos. Debido a esto, los sistemas han reemplazado la metodología basada en puertos por la inspección profunda de paquetes, consiguiendo así un gran aumento en la precisión.

DPI consigue analizar la carga útil de un paquete y encontrar patrones que se identifiquen con un servicio.

Cabe destacar, que este tratamiento puede llegar a ser muy costoso, por lo que los patrones son identificados por expresiones regulares, ya que así se consigue disminuir el coste computacional.



*Figura 3-3 Identificación de patrones usando DPI.*

Con este tratamiento los ISP [24] pueden proporcionar calidad de servicio a una aplicación en concreto, ó realizar cobros según el servicio que se esté utilizando.

### 3.3 nDPI.

Como ya se mencionó anteriormente nDPI es una librería capaz de realizar inspección profunda de paquetes, programada en **lenguaje C**.

Esta librería es multiplataforma, puesto que puede ser usada tanto en UNIX como en Windows.

Aunque por defecto es capaz de detectar numerosas aplicaciones, es muy fácil añadir nuevos servicios que no se puedan identificar a priori.

Consultando el manual [25] podemos observar un ejemplo de como añadir nuevos protocolos para que sean identificados en el flujo de red.

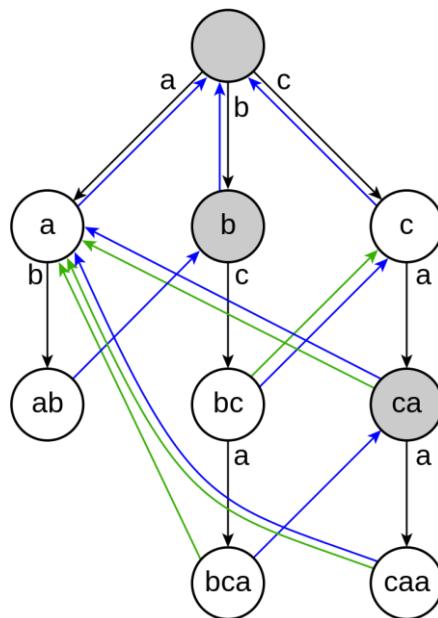
```
# Subprotocols
# Format:
# host:<value>,host:<value>,....@<subproto>
host:"googlesyndication.com"@Google
host:"venere.com"@Veneer
```

*Figura 3-4 Añadir protocolos en nDPI.*

Debido a que actualmente la mayoría de las conexiones se encuentran cifradas, nDPI contiene un decodificador SSL [26], el cual es capaz de extraer el nombre del host del certificado perteneciente al servidor. Esta información se añade a los metadatos del flujo de red, consiguiendo así ser identificado.

La librería nDPI implementa el algoritmo de **Aho-Corasick** para encontrar patrones dentro de un texto. Con el fin de ser eficiente a la hora de analizar la carga útil de los paquetes circulantes en la red.

Este algoritmo implementa un autómata similar a una estructura de datos de tipo árbol. A continuación se muestra un ejemplo del algoritmo cuando los patrones son: {**a, ab, bc, bca, c, caa**}



*Figura 3-5 Ejemplo de algoritmo Aho-Corasick. [27]*

La característica más interesante que posee este algoritmo, resulta ser la capacidad de poder buscar patrones de forma simultánea.

Esta implementación es usada mucho en el mundo de la ciberseguridad, como es por ejemplo para las bases de datos de virus.

### 3.4 wireshark.

Wireshark es un software usado para analizar protocolos y pasos de mensajes. Esta aplicación permite capturar todos los paquetes de una red con el fin de poder identificar problemas de configuración en las redes de comunicación.



Figura 3-6 Logo de wireshark.

Esta aplicación ha tenido gran importancia en el estudio realizado, puesto que ha sido utilizada para capturar el tráfico que será usado como entrada al sistema desarrollado.

### 3.5 Integración de nDPI con wireshark.

Wireshark posee la posibilidad de ser integrada junto con la librería nDPI en los sistemas UNIX. Esta nueva funcionalidad fue presentada en el SharkFest de 2017 [28]. Y permite la identificación de las aplicaciones en los paquetes capturados.

Para hacer esto posible, es necesario descargar de github los ficheros *ndpiReader.c* junto a *ndpi.lua*.

A continuación se muestra un ejemplo donde se ha capturado tráfico de **Google** y se puede comprobar cómo se ha añadido la información en el campo protocolo. Además, también se puede realizar filtros para mostrar sólo cierta aplicación. Un ejemplo sería: *ndpi.protocol.name == HTTP.Google*

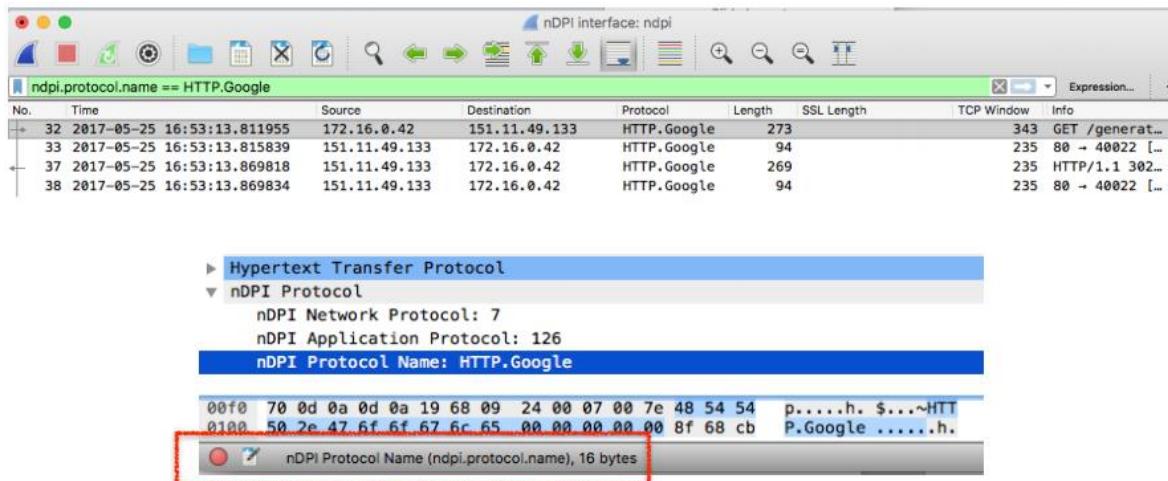


Figura 3-7 Captura de paquetes con wireshark y nDPI.

Una de las ventajas de poder capturar identificando las aplicaciones, es la de poder capturar sólo los paquetes de ciertas aplicaciones. Esto permite una mejora en la capacidad de respuesta de wireshark reduciendo el uso de recursos.

Este plugin también es capaz de llevar a cabo un seguimiento de los certificados del servidor, así como el conteo de números de conexiones SSL por certificado.

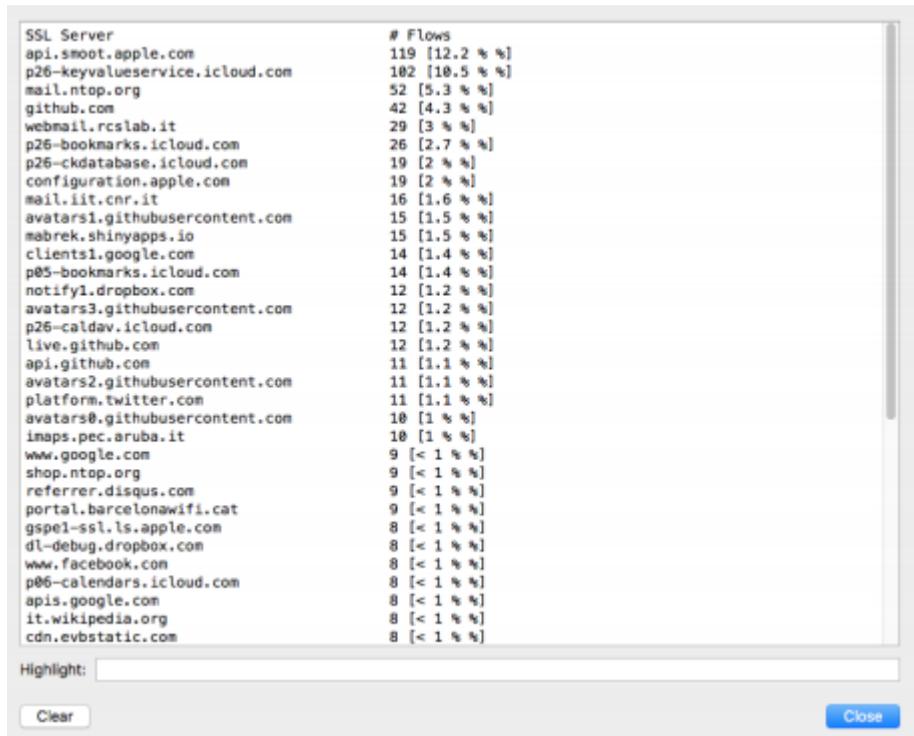


Figura 3-8 certificados SSL.

### 3.6 Recolector nProbe.

nProbe es una sonda NetFlow v5/v9 IPFIX que recompila, analiza y exporta información del tráfico de red. Este software posee muchas funcionalidades, pero en el estudio llevado a cabo se ha usado para poder exportar los flujos del tráfico capturado en formato IPFIX. Se puede encontrar una gran información sobre este software en el manual [29].

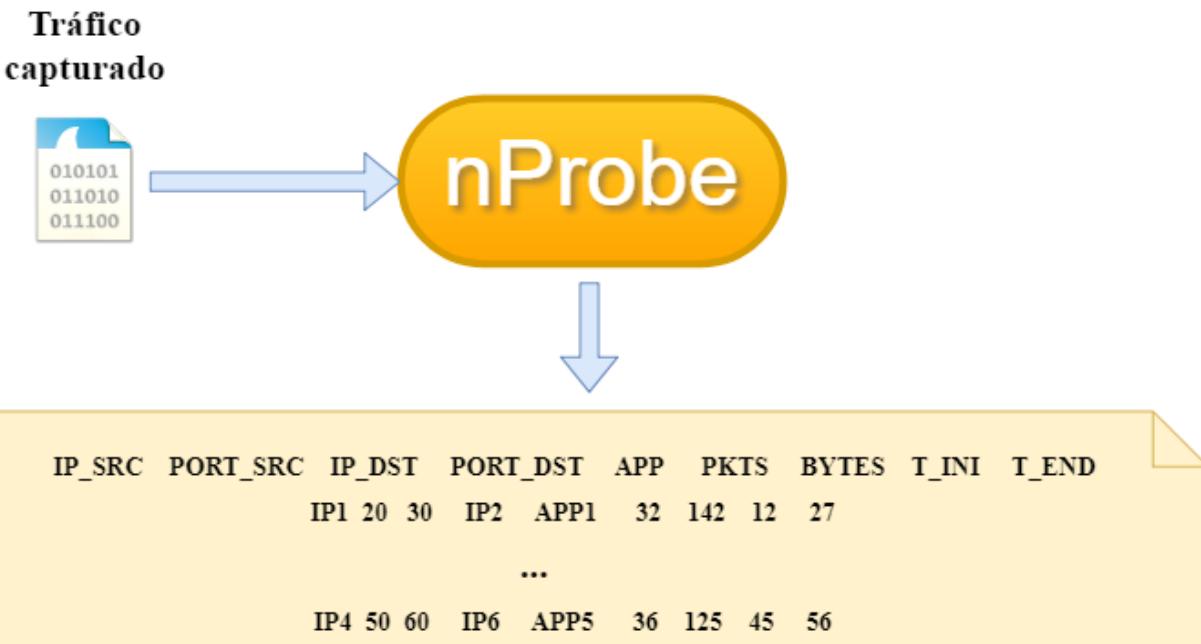


Figura 3-9 Uso de nProbe.

La gran ventaja del uso de nProbe es la posibilidad de exportar la información que deseé. Además nProbe integra internamente la librería nDPI, por lo que se puede añadir la información de que aplicación corresponde al flujo.



# 4 SISTEMA

My work on free software is motivated by an idealistic goal: spreading freedom and cooperation. I want to encourage free software to spread, replacing proprietary software that forbids cooperation, and thus make our society better.

- Richard Stallman -

**E**n este capítulo se verá el diseño del sistema desarrollado, las partes en las que se compone, así como las principales funcionalidades que posee. Cabe destacar que el sistema podría ser usado en casos reales debido a la implementación realizada.

## 4.1 Aspectos generales del sistema.

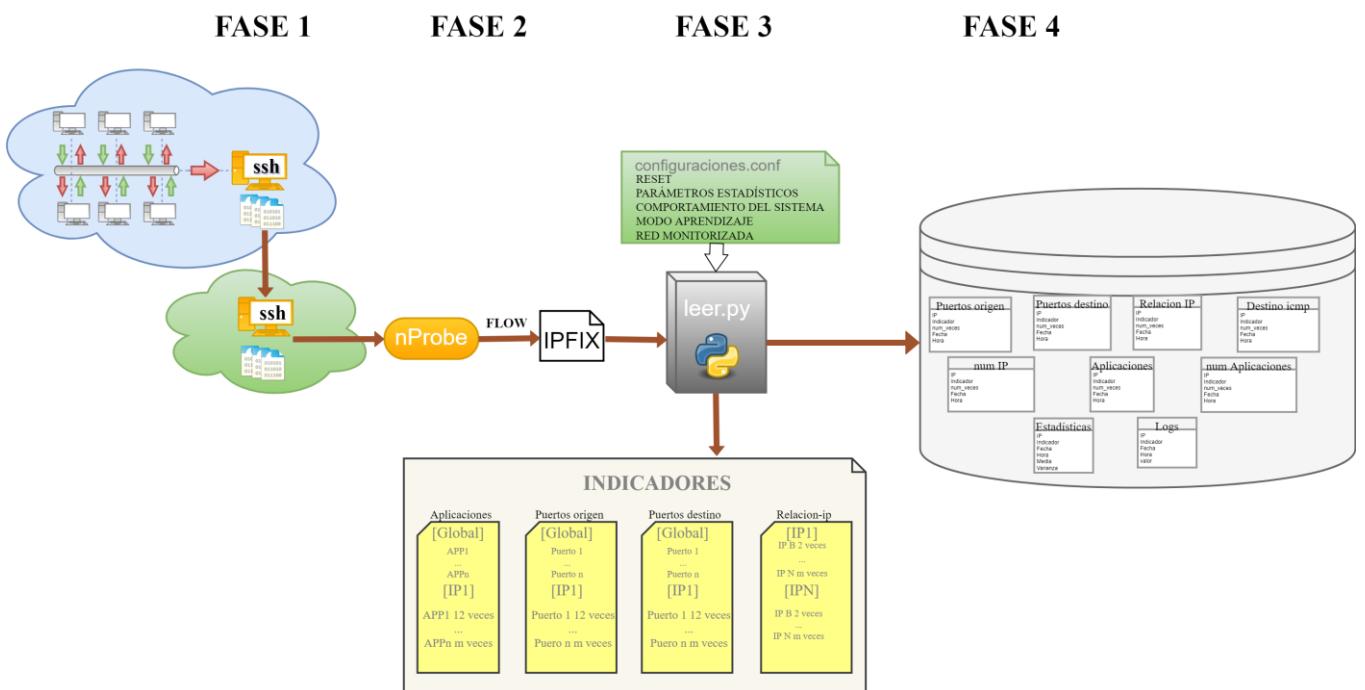


Figura 4-1 Esquema general del sistema.

Debido a la complejidad que posee el sistema desarrollado, para una mejor compresión del mismo, este puede ser descompuesto en 4 fases:

- Una primera en la que se capturan los datos.
- Una segunda parte en la que se exporta la información en formato IPFIX.
- Una tercera encargada del procesamiento de todos los datos.
- Y por último la exportación final del tratamiento de los datos.

## 4.2 Fase 1: Obtención del tráfico.

Primero será necesario capturar todo el tráfico que circula en una red. Para ello se hace uso de `tcpdump`[30] en modo promiscuo.

Una vez capturados los paquetes estos son guardados en un archivo de tipo *pcap*.

El equipo encargado de realizar las capturas, guardará un archivo *pcap* por cada franja horaria, consiguiendo así realizar un sondeo de la red a cada hora.

Estos archivos son enviados a un segundo equipo ubicado en una red externa, puesto que si procesaramos la información en el mismo equipo, podríamos sobrecargarlo y no conseguir el fin deseado.

Es muy importante que para el envío de los archivos de capturas, se realice a través de un medio seguro como es por ejemplo **SCP** [31]. El cuál permite transferencias seguras de ficheros a través de **SSH** [32].

## 4.3 Fase 2: Exportación del tráfico en formato IPFIX.

Hasta ahora la información del tráfico circulante en la red se encuentra en bruto en archivos **PCAP**. En esta fase se usa como entrada a nProbe estos ficheros, con el fin de poder exportar la información en archivos **CSV** [33].

La información exportada representa las comunicaciones en la red a modo de flujos IP en formato IPFIX. A continuación se muestra un ejemplo:

```
TPV4_SRC_ADDR,L4_SRC_PORT,TPV4_DST_ADDR,L4_DST_PORT,L7_PROTO,L7_PROTO_NAME,IN_PKTS,IN_BYTES,FLOW_START_MILLISECONDS,FLOW_END_MILLISECONDS
91.228.166.150,443,192.168.0.26,2594,188,QUIC,5,637,1537728302979,1537728302979
192.168.0.26,2594,91.228.166.150,443,188,QUIC,8,3448,1537728302979,1537728302979
192.168.0.26,2595,91.228.166.150,443,91,SSL,16,3837,1537728302979,1537728302980
91.228.166.150,443,192.168.0.26,2595,91,SSL,10,4355,1537728302979,1537728302980
192.168.0.26,63337,239.255.255.250,1900,12,SSDP,129,21285,1537728302980,1537728303080
192.168.0.26,2596,91.228.166.150,443,91,SSL,16,4168,1537728302980,1537728302980
91.228.166.150,443,192.168.0.26,2596,91,SSL,11,4391,1537728302980,1537728302980
192.168.0.26,55478,172.217.16.238,443,188,QUIC,1,51,1537728302980,1537728302980
172.217.16.238,443,192.168.0.26,55478,188,QUIC,1,48,1537728302980,1537728302980
192.168.0.26,1837,92.122.188.120,443,188,QUIC,5874,392245,1537728302980,1537728303080
92.122.188.120,443,192.168.0.26,1837,188,QUIC,24710,36184858,1537728302980,1537728303080
192.168.0.26,2515,35.186.224.53,443,188,QUIC,5,203,1537728302980,1537728303002
35.186.224.53,443,192.168.0.26,2515,188,QUIC,6,339,1537728302980,1537728303002
```

Figura 4-2 Ejemplo de exportación con nProbe.

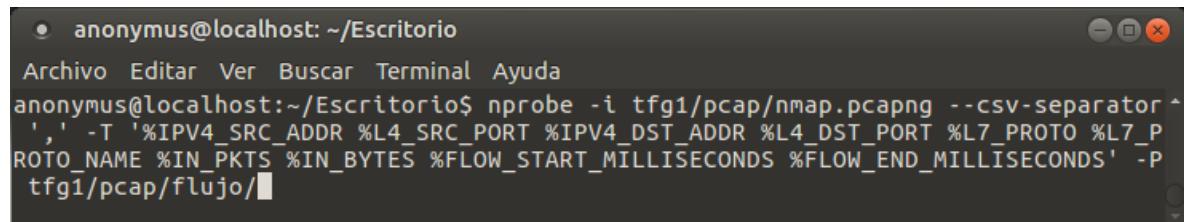
A continuación se muestra una tabla donde se describe la información que se va a exportar:

Campo	Descripción
• IP origen	IP versión 4 que identifica al origen del flujo IP.
• IP destino	IP versión 4 que identifica al destino del flujo IP.
• Puerto origen	Puerto origen de capa 4 usado para el transporte del flujo.
• Puerto Destino	Puerto destino de capa 4 usado para el transporte del flujo.

• Nº Aplicación	Número del identificador de la aplicación que ha sido encontrada en el tratamiento DPI.
• Nº de paquetes	Cantidad de paquetes totales que corresponden al flujo.
• Nº de Bytes	Cantidad de Bytes totales en el flujo.
• Instante inicial	Tiempo de inicio del flujo.
• Instante final	Tiempo final del flujo.

**Tabla 4 Campos exportados con nProbe.**

Como ya se mencionó anteriormente, nProbe tiene muchas funcionalidades y es muy versátil a la hora de exportar la información. Debido a que este estudio solo requiere de los campos mostrados en la *Tabla 4*, es necesario ejecutar nProbe como se muestra en la imagen 4-3.



```
anonymus@localhost: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
anonymus@localhost:~/Escritorio$ nprobe -i tfg1/pcap/nmap.pcapng --csv-separator ',' -T '%IPV4_SRC_ADDR %L4_SRC_PORT %IPV4_DST_ADDR %L4_DST_PORT %L7_PROTO %L7_PROTO_NAME %IN_PKTS %IN_BYTES %FLOW_START_MILLISECONDS %FLOW_END_MILLISECONDS' -P tfg1/pcap/flujo/
```

**Figura 4-3 Ejecución nProbe.**

En la *tabla 5* se detallan los parámetros usados en nProbe.

Opción	Descripción
• -i	Indica la interfaz o fichero PCAP de entrada.
• --csv-separator	Impone que la salida del fichero tenga formato csv.
• -T	Esta opción permite el orden y la información que se desea exportar.
• -P	Selección la carpeta donde se desea guardar la información a exportar.

**Tabla 5 Descripción de las opciones usadas en nProbe.**

Una vez se realiza este procesamiento. En la carpeta destino especificada con el parámetro *-P* aparecerá un fichero de tipo *flows*, el cuál tendrá la información deseada.

## 4.4 Fase 3: Procesamiento de IPFIX.

Esta etapa es la más importante, debido a que resulta ser el peso grande del estudio realizado. Hasta ahora los datos no se han analizado para detectar anomalías en el tráfico de red. Es en esta fase donde se realiza el tratamiento de la información para toma de decisiones en la detección de comportamientos anómalos.

### 4.4.1 Lenguaje de programación y librerías.

El lenguaje de programación utilizado para el desarrollo del trabajo ha sido **Python** [34], puesto que es el lenguaje más extendido y usado en el mundo de la ciberseguridad.



Figura 4-4 Logo de Python [34].

Además, este lenguaje tiene la posibilidad de usar numerosas librerías muy útiles que facilitan el trabajo del programador. En la *tabla 6* podemos observar las principales librerías utilizadas para el desarrollo.

Librería	Descripción
• Pandas [35]	Esta librería permite el análisis de datos de forma eficiente. Ofrece diferentes estructuras de datos como son: <ul style="list-style-type: none"><li>• Data Frame.</li><li>• Series.</li><li>• Estructuras de datos multidimensionales.</li></ul>
• Numpy [36]	Destinada para poder realizar operaciones con matrices y vectores.
• Datetime [37]	Módulo que proporciona clases y objetos para la manipulación de horas y fechas.
• Os [38]	Permite ejecutar comandos propios del sistema operativo.
• Sys [39]	Proporciona variables y funcionalidades con el propio intérprete de Python.
• mysql.connector [40]	Permite la interacción con una base de datos MySQL.
• math [41]	Dota a Python de operaciones matemáticas.
• shutil [42]	Proporciona acceso y manipulación de documentos y directorios alojados en el sistema operativo.
• ConfigParser [43]	Permite el acceso de forma sencilla a ficheros de configuración.
• Ipcalc [44]	Permite realizar cálculos de subredes IP tanto en IPv4 como IPv6.

Tabla 6 Librerías utilizadas.

#### 4.4.2 Diseño del sistema.

En la figura 4-5 se muestra el diseño llevado a cabo para la implementación del sistema desarrollado.

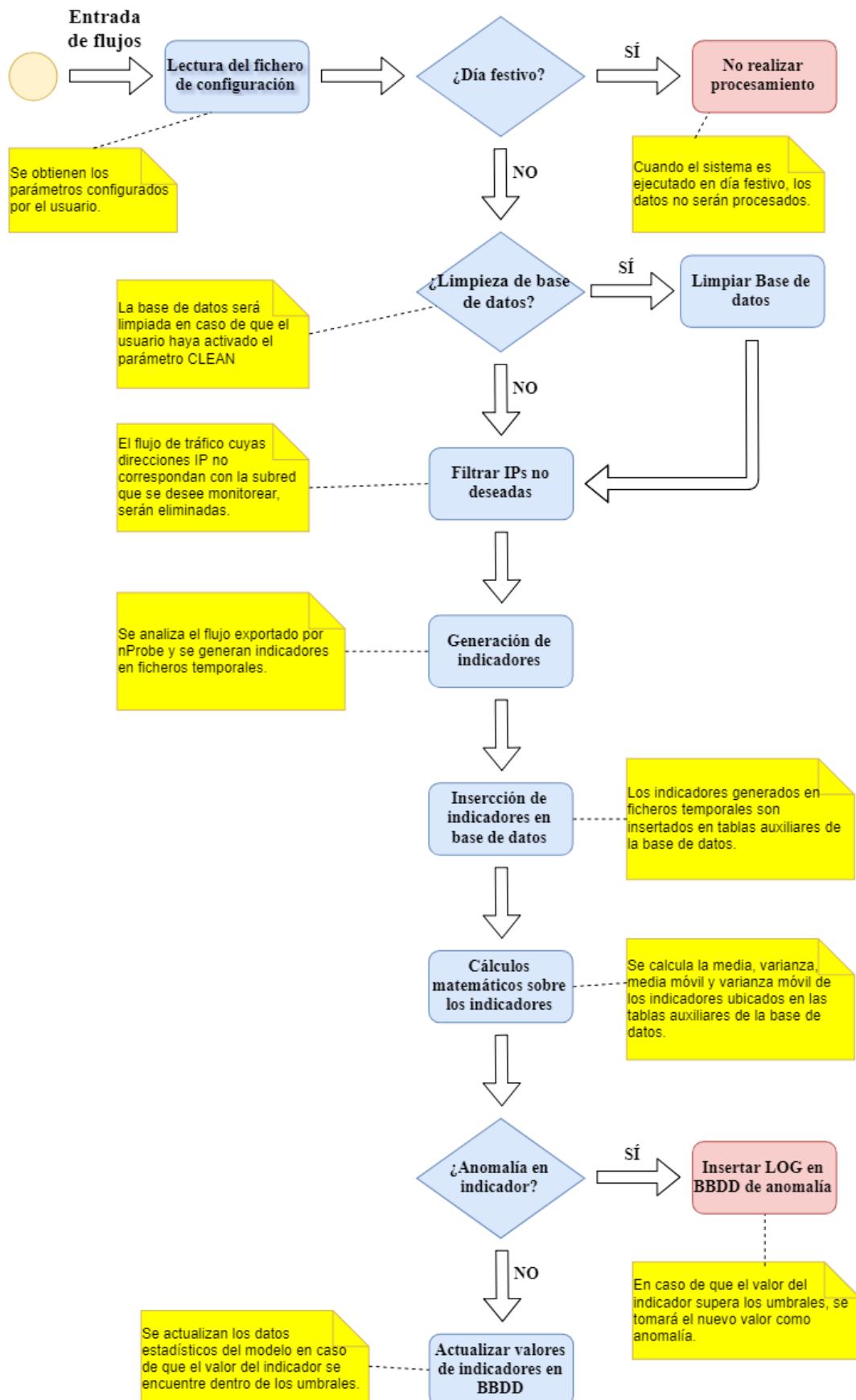


Figura 4-5 Diseño del sistema.

#### 4.4.2.1 Diseño de los principales componentes del sistema.

##### Lectura del fichero de configuración

Obtención de los parámetros definidos por el usuario. Para facilitar al usuario, los parámetros se encuentran divididos en 4 secciones:

- FECHA
- BBDD
- ESTADISTICAS
- TRATAMIENTO

##### Descripción:

Parámetro	Uso y opciones
Fecha	Define la fecha y hora que se desee asignar a los indicadores. Este parámetro por defecto es AUTO, en este valor la fecha y hora se asignan automáticamente.  En caso de estar realizando pruebas o desarrollando, se puede asignar manualmente en el formato YYYYMMDDHH. Un ejemplo sería: 2018092117, que correspondería con el 21 de septiembre de 2018 a las 17:00.
User	Define el usuario miembro de la base de datos.
Password	Define la contraseña de la base de datos.
Host	Define la dirección IP donde se encuentra alojada la base de datos.
Database	Define el usuario miembro de la base de datos.
Clean	Si se desea limpiar la base de datos este valor deberá estar a TRUE, en caso contrario deberá estar a FALSE.
Capacidad_ventana	Define el tamaño de las N últimas muestras que se tienen en cuenta en los cálculos matemáticos realizados.
Coeficiente_error	Coeficiente de error para la toma de decisión. Normalmente este valor estará comprendido entre 2 y 4. Cuanto más pequeño sea este valor el sistema será más restrictivo.
Aprendizaje	Este parámetro es muy importante, ya que cuando su valor es igual a TRUE, el sistema se encuentra en modo de aprendizaje. Es decir, el sistema se encuentra en modo de entrenamiento. Todos los datos de entrada serán clasificados como tráfico limpio y valdrán para realizar una modelización del tráfico.  Cuando el parámetro se encuentra a FALSE el sistema evaluará todos los indicadores, generando alertas en caso de detectar anomalías.

Eliminar_anomalos	Si el parámetro se encuentra a TRUE, los valores que provoquen una alarma no se insertarán en la base de datos. Si por lo contrario, se desea que estos valores se inserten en base de datos, el parámetro deberá estar a FALSE.
Alertas_nuevas	Si se encuentra a TRUE y se detecta un nuevo indicador, el sistema generará una alarma notificando el nuevo indicador. Si no se desea alertar de nuevos indicadores, el valor deberá estar a FALSE.
Alpha	Parámetro de suavizado de los cálculos estadísticos, este valor suele estar comprendido entre 0.1 y 0.5
network	Define el rango de subred de direcciones IP que se desean tener en cuenta para el estudio. Un ejemplo de valor sería: 192.168.0.1/24 dónde se estarían indicando todas las IPs de la subred 192.168.0.1 con máscara 24.

**Tabla 7 Parámetros del fichero de configuración.**

#### Filtrar IPs no deseadas

La información exportada en formato IPFIX puede contener flujos de direcciones IPs que no se desean tener en cuenta para el estudio. Es por ello por lo que las direcciones IP origen correspondientes a la subred definida en el parámetro **network** serán convertidas a la dirección IP 1.1.1.1 y las direcciones IP destino a la 2.2.2.2.

Este tratamiento se realiza para eliminar el tráfico de direcciones irrevelantes para el estudio así como para facilitar y simplificar el procesamiento de los datos.

**Tabla 8 Tratamiento de las direcciones IP del tráfico.**

---

### Generación de indicadores

---

En base al flujo IPFIX filtrado se crean los siguientes indicadores:

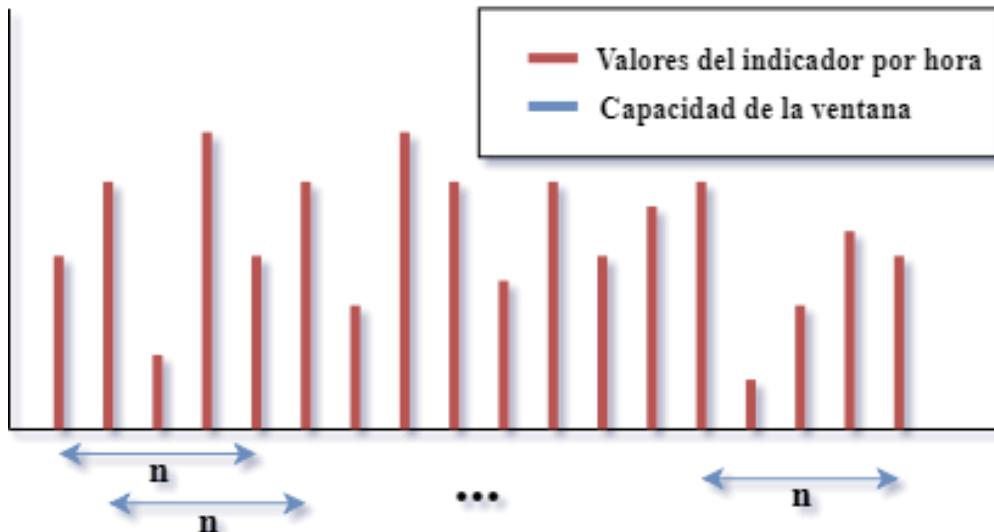
<b>Indicador</b>	<b>Descripción</b>
Aplicaciones	Recoge las diferentes aplicaciones detectadas en los flujos así como el número de veces que intervienen en ellos.
ICMP-destino	Muestra el número de veces que se han generados mensajes ICMP[45]. Este indicador puede alertar de ataques de denegación de servicios.
IPdestino-origen	Informa por cada IP destino el número de veces que se conecta con las diferentes IP origen.
IPorigen-destino	Informa por cada IP origen el número de veces que se conecta con las diferentes IP destino.
Numero-APP	Número de veces que aparece cada aplicación en el tráfico de la red.
Numero-IP	Número de IPs destino distintas.
PuertosDestino	Registro de todos los puertos destino por cada IP origen.
PuertosOrigen	Registro de todos los puertos origen por cada IP origen.

*Tabla 9 Diferentes indicadores.*

#### 4.4.3 Cálculo de los indicadores.

Para crear un modelo estadístico se parten de dos tipos de premisas:

1. Mediante cálculo de la media aritmética móvil[46] y varianza[47]:



*Figura 4-6 Media móvil.*

- Como se puede observar en la *Figura 4-6*, el cálculo realizado corresponde a una media móvil, debido a que se calcula el promedio de las n últimas muestras. Este comportamiento se denomina como ventana deslizante. Se ha implementado de esta forma para conseguir que el sistema pueda responder a lo largo del tiempo. Ya que si no se implementa una ventana deslizante, llegaría un punto en el que la media no varía y su resultado sería prácticamente constante.
- La media aritmética o promedio cuya fórmula es:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n x_i$$

- **n** Número de la capacidad de la ventana.
- $x_i$  Se denota como los valores que toma la variable.

- La varianza es una medida de dispersión y la raíz cuadrada de la varianza nos permite calcular la desviación que posee cada indicador. A continuación se muestra la fórmula de la varianza:

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{X})^2$$

- **n** Número de la capacidad de la ventana.
- $x_i$  Se denota como los valores que toma la variable.

- $\bar{X}$  Media de la variable.
- Cuando se almacena un valor en un indicador, este es evaluado a través de la siguiente ecuación y si no cumple la condición el valor será clasificado como anómalo:

$$\epsilon \leq k \times \sqrt{\sigma^2}$$

- $\epsilon$  Se define como el error y se calcula de la siguiente manera:

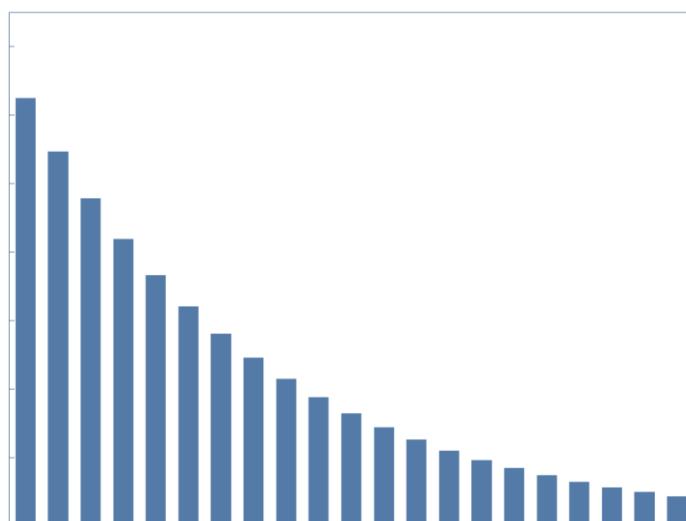
$$\epsilon = |x_i - \bar{X}|$$

- $k$  Corresponde con el coeficiente de error. (Este parámetro es configurable por el usuario a través del fichero de configuración).

## 2. Mediante cálculo de la media exponencial móvil y varianza exponencial móvil:

- La media móvil exponencial[48] cuya fórmula es:

$$EMA(t) = \begin{cases} x_1, & t = 1 \\ \alpha \times x_i + (1 - \alpha) \times EMA(t - 1), & t > 1 \end{cases}$$



*Figura 4-7 Forma de media exponencial móvil.*

- $\alpha$  Corresponde al parámetro de suavizado que puede valer entre 0 y 1.
- $x_i$  Valor en el instante t.
- La EMA es usado comúnmente para las tendencias de los valores de mercado. Se ha optado por implementar este modelo estadístico debido a los buenos resultados que se obtienen cuando se evalúan indicadores.
- La varianza exponencial cuya fórmula es:

$$\sigma^2_{exp} = \left( \frac{\alpha}{2 - \alpha} \right) \times \sigma^2$$

- $\alpha$  Parámetro de suavizado.
- $\sigma^2$  Valor de la varianza de todo el conjunto de muestras.

- Al igual que en el caso anterior de media aritmética, en este también se toma decisiones en base a la siguiente fórmula:

$$\epsilon \leq k \times \sqrt{\sigma^2_{exp}}$$

- **$\epsilon$**  Se define como el error y se calcula de la siguiente manera:

$$\epsilon = |x_i - EMA(t)|$$

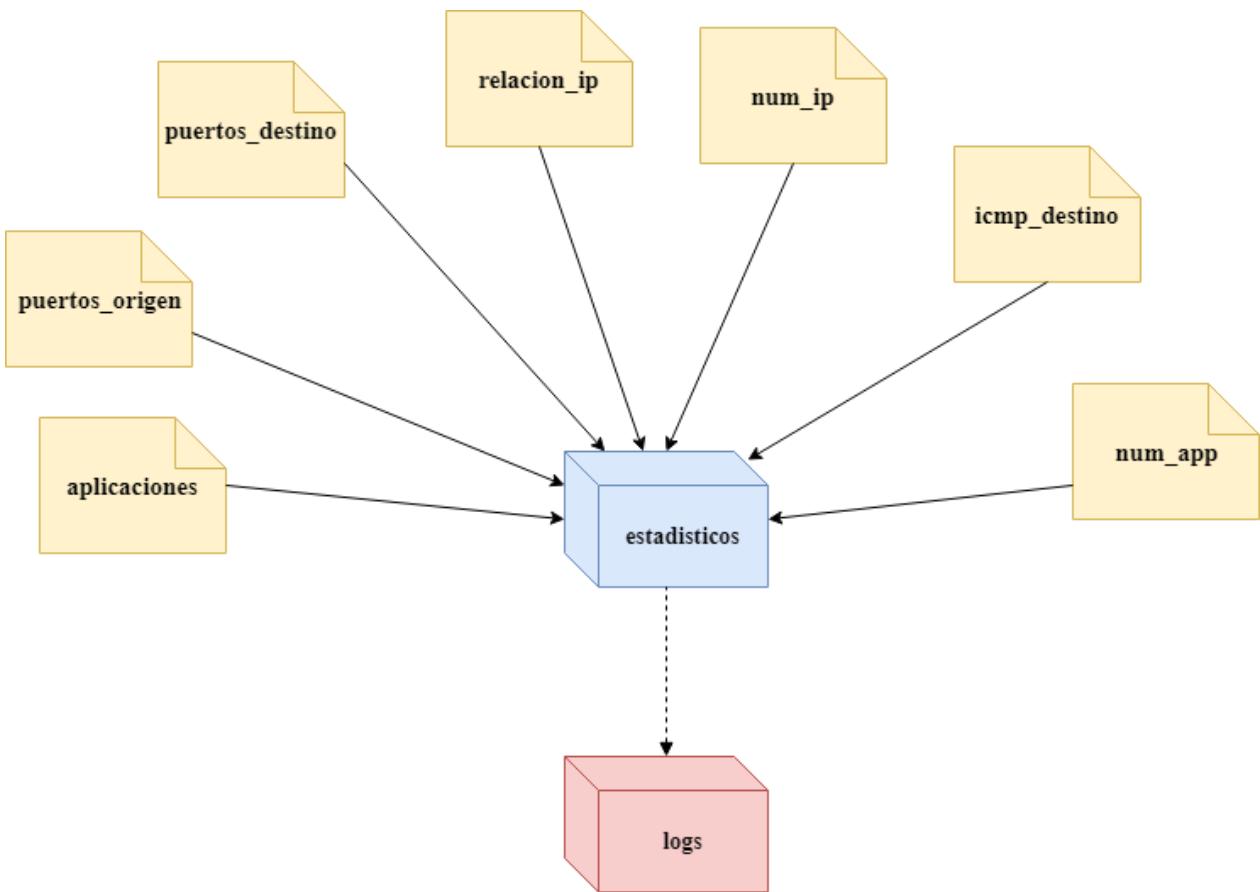
- **$k$**  Corresponde con el coeficiente de error. (Este parámetro es configurable por el usuario a través del fichero de configuración).

## 4.5 Fase 4: Base de datos.

Esta última etapa del sistema corresponde con la exportación a una base de datos toda la información recolectada y procesada.

Se ha utilizado MySQL[49] como sistema de gestor para la base de datos.

### 4.5.1 Diseño de la base de datos.



*Figura 4-8 Diagrama funcional de la base de datos.*

La base de datos esta compuesta por tablas auxiliares, cada una de estas tablas contiene la información de un indicador. Por cada uno de ellos se realiza un cálculo matemático y el resultado se inserta en la tabla estadísticos.

mysql> show tables;
+-----+-----+
Tables_in_tfg
+-----+-----+
aplicaciones
estadisticos
icmp_destino
logs
num_app
num_ip
puertos_destino
puertos_origen
relacion_ip
+-----+-----+
9 rows in set (0,00 sec)

*Figura 4-9 Tablas que componen la base da datos.*

Cada vez que se realiza un cálculo se comprueba que el resultado está dentro del margen de confianza. En caso de no cumplir dicho margen el nuevo valor del indicador no será insertado en la tabla **estadísticos**, pero sí se insertará un registro en la tabla de logs.

Si por lo contrario el valor si se encuentra dentro del margen de confianza, dicho valor si será insertado en la

tabla de estadísticos.

#### 4.5.1.1 Tablas auxiliares.

A continuación se detalla el diseño de cada una de las tablas auxiliares que alimentan a la tabla principal que contienen las estadísticas de cada indicador.

Aplicaciones	
Descripción:	Tabla que contiene el indicador de aplicaciones.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• aplicacion</li> <li>• num_veces</li> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
aplicacion	Nombre de la aplicación identificada en un flujo.
num_veces	Número de flujos en los que se ha identificado la aplicación.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

*Tabla 10 Indicador de aplicaciones.*

Icmp_destino	
Atributo	Descripción
<b>Descripción:</b>	Tabla que contiene el indicador de mensajes ICMP originados en una máquina.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• ip_destino</li> <li>• num_veces</li> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
ip_destino	Ip destino a la es destinado el mensaje ICMP.

---

num_veces	Número de veces que se ha generado el mensaje ICMP con estas direcciones ip.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

---

**Tabla 11 Indicador de ICMP.**

---

<b>Puertos_destino</b>	
<b>Descripción:</b>	Tabla que contiene el indicador de los puertos destino.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• puerto</li> <li>• num_veces</li> <li>• fecha</li> <li>• hora</li> </ul>
<b>Atributo</b>	
ip	Ip versión 4 origen del indicador.
puerto	Puerto destino identificado en el fujo.
num_veces	Número de veces que se identifica el puerto destino con la misma ip origen.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

---

**Tabla 12 Indicador Puertos destino.**

---

<b>Puertos_origen</b>	
<b>Descripción:</b>	Tabla que contiene el indicador de los puertos origen.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• puerto</li> <li>• num_veces</li> <li>• fecha</li> <li>• hora</li> </ul>

---

Atributo	Descripción
ip	Ip versión 4 origen del indicador.
puerto	Puerto origen identificado en el flujo.
num_veces	Número de veces que se identifica el puerto origen con la misma ip origen.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

**Tabla 13 Indicador Puertos destino.**

Relación_ip	
Descripción:	Tabla que contiene el indicador de las diferentes ip.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• puerto</li> <li>• ip_destino</li> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
ip_destino	Ip versión 4 destino en el flujo.
num_veces	Número de veces que ha aparecido la ip destino para la misma ip origen.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

**Tabla 14 Indicador relación de las diferentes ip.**

Num_ip	
Descripción:	Tabla que contiene el indicador de las diferentes ip.
	<ul style="list-style-type: none"> <li>• ip</li> <li>• ip_aux</li> </ul>

<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
ip_aux	Campo que siempre contiene el valor ‘DISTINTAS’ para facilitar la visualización y tratamiento realizado posteriormente.
num_veces	Número total de IP destino distintas.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

**Tabla 15 Número total de ip destinos.**

Num_app	
Descripción:	Tabla que contiene el indicador de las diferentes ip.
<b>Clave primaria:</b>	<ul style="list-style-type: none"> <li>• ip</li> <li>• aplicación</li> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
aplicación	Campo que siempre contiene el valor ‘APLICACIONES’ para facilitar la visualización y tratamiento realizado posteriormente.
num_veces	Número de distintas aplicaciones identificadas.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

**Tabla 16 Número total de las diferentes aplicaciones.**

#### 4.5.1.2 Tabla principal.

A continuación se detalla el diseño de la tabla más importante. La importancia de esta tabla es debido a que contiene todas las estadísticas de los cálculos realizados a partir de los datos de las tablas auxiliares.

Estadísticos	
Descripción:	Contiene todos los indicadores así como los datos estadísticos para cada uno de ellos.
Clave primaria:	<ul style="list-style-type: none"> <li>• ip</li> <li>• indicador</li> <li>• fecha</li> <li>• hora</li> </ul>
Atributo	Descripción
ip	Ip versión 4 origen del indicador.
indicador	<p>Identificador del indicador. Para poder comprender la información estos son los posibles valores que se pueden encontrar:</p> <ul style="list-style-type: none"> <li>• Nombre de una aplicación.</li> <li>• O ‘número de puerto’ (Indicador de puerto origen).</li> <li>• D ‘número de puerto’ (Indicador de puerto destino).</li> <li>• I ‘IP’ (Indicador de icmp).</li> <li>• DISTINTAS (Indicador de distintas ip).</li> <li>• APLICACIONES (Indicador de distintas aplicaciones).</li> <li>• IP (Indicador de relación-ip).</li> </ul>
media	Número que corresponde con el cálculo de la media del indicador.
varianza	Número que corresponde con el cálculo de la varianza del indicador.
media_exp	Número que corresponde con el cálculo de la media móvil exponencial del indicador.
varianza_exp	Número que corresponde con el cálculo de la varianza exponencial móvil del indicador.
muestas	Número de muestras del indicador.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

*Tabla 17 Tabla principal de indicadores.*

#### 4.5.1.3 Tabla de alarmas.

Si algún valor registrado en las tablas auxiliares no se encuentra dentro de los umbrales, se generará una alarma en la tabla logs registrando tal evento.

---

## Logs

<b>Descripción:</b>	Contiene todos los eventos generados cuando un indicador ha sido clasificado como anómalo.
---------------------	--

- Clave primaria:**
- ip
  - indicador
  - tipo
  - fecha
  - hora

Atributo	Descripción
ip	Ip versión 4 origen del indicador.
indicador	Nombre del indicador que ha sido registrado como anómalo.
tipo	Los datos son evaluados a través de dos formas. Por lo que se pueden generar dos tipos de alarmas: <ul style="list-style-type: none"><li>• ERROR: Evento que identifica la media.</li><li>• ERROR_EXP: Evento que identifica la media móvil.</li></ul>
valor	Valor que tiene el indicador cuando se ha clasificado como anómalo.
fecha	Fecha en formato YYYY-MM-DD al que corresponde el indicador.
hora	Hora en formato HH al que corresponde el indicador.

*Tabla 18 Logs.*

# 5 PRUEBAS Y RESULTADOS

---

*El ordenador nació para resolver problemas que antes no existían.*

*- Bill Gates -*

Este capítulo se dedica a la descripción de las pruebas realizadas, en el cuál podemos encontrar casos reales de detecciones. También se exponen los resultados obtenidos. Así como el software usado para la realización de las pruebas.

## 5.1 Descripción de software usado para la realización de las pruebas

Para la realización de las pruebas se ha utilizado vmware para virtualizar una máquina con las siguientes características:

Máquina virtual	
• Memoria RAM	3 GB
• Procesadores	8
• Sistema Operativo	Ubuntu 16.04.4 LTS

*Tabla 19 Máquina virtualizada con vmware.*

## 5.2 Pruebas realizadas

A continuación se muestran los pasos llevados a cabo para la realización de las pruebas. Para una mejor comprensión del lector se pueden dividir de la siguiente forma:

1. Captación de los datos y exportación IPFIX.
2. Preparación del archivo de configuración.
3. Ejecución del script **leer.py**
4. Consultas tablas BBDD.
5. Errores.

Para finalizar, nos centraremos en varios indicadores y se mostrará la evolución temporal para ver de una manera gráfica el comportamiento del sistema.

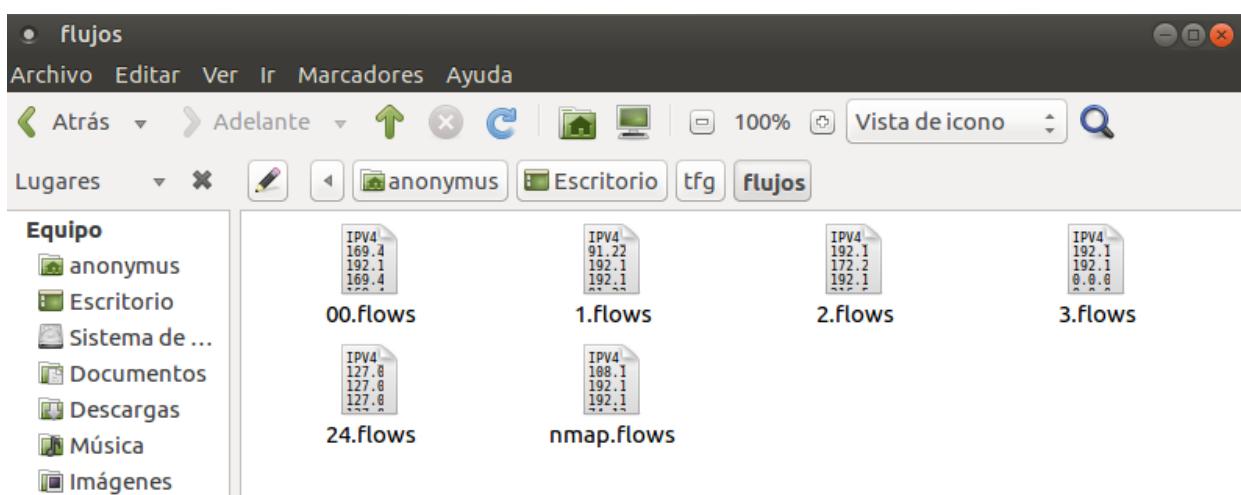
### 5.2.1 Captación de los datos y exportación IPFIX

```
anonymus@localhost:~/Escritorio$ nprobe -i tfg1 pcap/01.pcapng --csv-separator ',' -T '%IPV4_SRC_ADDR %L4_SRC_PORT %IPV4_DST_ADDR %L4_DST_PORT %L7_PROTO %L7_PROTO_NAME %IN_PKTS %IN_BYTES %FLOW_START_MILLISECONDS %FLOW_END_MILLISECONDS' -P tfg1 pcap/flujo/
```

En este paso ejecutamos nProbe pasándole por parámetro el archivo de captura de tráfico.

*Figura 5-1 Ejecución nprobe para la prueba.*

Como se puede observar en la *Figura 5-1*. Le pasamos a nProbe el archivo de captura de tráfico *01.pcapng*. Nprobe creará un fichero en formato IPFIX con la información de los flujos IP en la carpeta indicada de nombre *flujo*.



*Figura 5-2 Carpeta con los ficheros de flujos IP.*

A cada hora nuestro sistema capturara tráfico y realizará este paso, creando un fichero de flujos IP en formato IPFIX a cada hora.

En la *Figura 5-2* se puede observar como el sistema ha creado diferentes ficheros.

### 5.2.2 Preparación del archivo de configuración

Antes de la ejecución del sistema será necesario definir las configuraciones globales, tal y como se puede ilustrar en la *Figura 5-3*.

Primero se debe definir el rango de la subred sobre la que se va a realizar el estudio a través del parámetro **network**. En este ejemplo la subred sería **192.168.0.0/24**

Como es la primera vez que va a ser ejecutado, es muy importante que se defina la variable aprendizaje a valor **TRUE**. Esto indica al sistema que se encuentra en modo aprendizaje y que debe tomar los datos que va a procesar para crear un modelo estadístico de **tráfico limpio**.

En este paso es muy importante alimentar al sistema con un buen conjunto de datos, debido a que si no es así, el modelo estadístico será muy deficiente y no tomará buenas decisiones en base al tráfico.

Para este ejemplo, se ha definido una ventana de **tamaño 3**, por lo que no resultaría coherente desactivar el modo aprendizaje hasta tener al menos **3 exportaciones IPFIX**.

Cuando el sistema ha sido alimentado y se tienen suficientes datos se desactivará el modo aprendizaje definiendo la variable aprendizaje a valor ***FALSE***. Una vez desactivado el modo aprendizaje, activaremos la generación de alertas nuevas. Esto permite la reportación de un log cuando se identificado un indicador que no se encuentra en el modelo estadístico.

Cabe destacar, que mientras el sistema no se encuentra en modo aprendizaje. Este sigue aprendiendo con el tiempo. Esto es debido a que el modelo estadístico se crea a través de una ventana deslizante. Por lo que los valores de los diferentes indicadores se actualizan con el tiempo.

```
# Fichero de configuraciones globales del programa

# Configuraciones de la fecha y hora
# en la que se realiza el estudio.
# Si se desea que la fecha y hora corresponda
# con el archivo de captura realizado, se deberá
# poner el valor AUTO, en cambio si se desea asignar una hora
# y fecha, debe ser en el formato: yyyyymmddhh por ejemplo,
# 10 de Marzo de 2018 a las 14:00 -> 2018031014
[FECHA]
fecha = AUTO

# Configuraciones sobre la Base de datos
[BBDD]
user = wala
password = wala
host = localhost
database = tfg
clean = FALSE

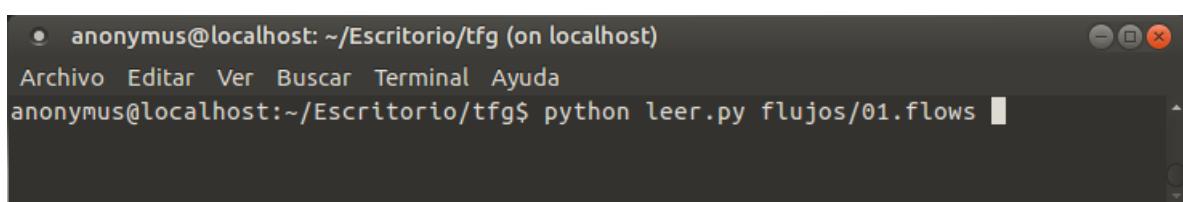
# Configuraciones llevadas a cabo en las estadísticas
# y comportamiento del sistema
[ESTADISTICAS]
capacidad_ventana = 3
coeficiente_error = 3
aprendizaje = TRUE
eliminar_anomalous = FALSE
alertas_nuevas = FALSE
alpha = 0.3

# Red sobre la que se desea monitorizar
[TRATAMIENTO]
network = 192.168.0.0/24
```

Figura 5-3 Configuración inicial.

### 5.2.3 Ejecución del script

Después de haber finalizado con la configuración, se ejecuta el script del sistema programado. Como se muestra en la **Figura 5-4** al script hay que indicarle el fichero de flujo que debe procesar. En este ejemplo se le indica que procese el fichero **01.flows** ubicado en la carpeta **flujos**.



The screenshot shows a terminal window with the following content:

```
● anonymous@localhost: ~/Escritorio/tfg (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda
anonymous@localhost:~/Escritorio/tfg$ python leer.py flujos/01.flows
```

Figura 5-4 Ejecución del sistema programado.

Después de la ejecución del script se habrán creado ficheros en texto plano en la carpeta **indicadores** con los

diferentes indicadores tal y como se muestra en la *Figura 5-5*.

Estos ficheros son temporales y son creados dinámicamente en cada ejecución del script.



*Figura 5-5 Indicadores temporales.*

#### 5.2.4 Consultas tablas BBDD

Después de la ejecución del script deberían existir registros en las tablas de base de datos, para ellos realizamos una consulta en la tabla del indicador aplicaciones y otra en tabla de estadísticos:

```

anonymus@localhost: ~/Escritorio (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda

mysql> select * from aplicaciones;
+-----+-----+-----+-----+-----+
| ip   | aplicacion      | num_veces | fecha    | hora   |
+-----+-----+-----+-----+-----+
| 1.1.1.1 | DNS           | 3          | 2018-10-03 | 13     |
| 1.1.1.1 | DNS.GMail     | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | DNS.Google    | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | DNS.GoogleDrive | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | DNS.GoogleServices | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | DNS.Office365 | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | HTTP          | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | LLMNR         | 4          | 2018-10-03 | 13     |
| 1.1.1.1 | MDNS          | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | NetBIOS       | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | QUIC          | 19         | 2018-10-03 | 13     |
| 1.1.1.1 | QUIC.Google    | 3          | 2018-10-03 | 13     |
| 1.1.1.1 | QUIC.GoogleDocs | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | QUIC.GoogleDrive | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | QUIC.GoogleServices | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | SSDP          | 1          | 2018-10-03 | 13     |
| 1.1.1.1 | SSL            | 12         | 2018-10-03 | 13     |
| 1.1.1.1 | SSL.GMail     | 2          | 2018-10-03 | 13     |
| 1.1.1.1 | SSL.Google    | 2          | 2018-10-03 | 13     |
| 1.1.1.1 | SSL.Office365 | 2          | 2018-10-03 | 13     |
| 1.1.1.1 | SSL.WhatsApp  | 2          | 2018-10-03 | 13     |
| 1.1.1.1 | SSL_No_Cert   | 39         | 2018-10-03 | 13     |
| 1.1.1.1 | Unknown        | 4          | 2018-10-03 | 13     |
| 1.1.1.1 | VMware         | 7          | 2018-10-03 | 13     |
+-----+-----+-----+-----+-----+
24 rows in set (0,39 sec)

```

*Figura 5-6 Consulta de tabla aplicaciones.*

Como se puede observar en la *Figura 5-6* en el tráfico circulante en la red se han identificado 23 aplicaciones distintas y 4 de los flujos han sido categorizados como **desconocido**. Esto resulta ser interesante, ya que si se obtiene numerosas muestras de aplicación desconocida, podríamos estar ante una detección de tráfico malicioso.

A continuación en la *Figura 5-7* se muestra la consulta (\*) de la tabla de estadísticos:

```

anonymus@localhost: ~/Escritorio (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda
mysql> mysql> select * from estadisticos;
+-----+-----+-----+-----+-----+-----+-----+-----+
| ip   | indicador      | hora | media | varianza | fecha    | muestras | media_exp | varianza_exp |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1.1.1.1 | 104.199.64.136 | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 108.177.15.189 | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 13.107.42.11  | 13   | 3     | 0       | 2018-10-03 | 1        | 3         | 0          |
| 1.1.1.1 | 151.101.132.246 | 13   | 2     | 0       | 2018-10-03 | 1        | 2         | 0          |
| 1.1.1.1 | 168.63.42.114  | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 169.44.82.101  | 13   | 2     | 0       | 2018-10-03 | 1        | 2         | 0          |
| 1.1.1.1 | 169.60.79.31   | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 172.217.16.227 | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 172.217.16.234 | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
| 1.1.1.1 | 172.217.16.238 | 13   | 1     | 0       | 2018-10-03 | 1        | 1         | 0          |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

*Figura 5-7 Consulta tabla estadísticos.*

(\*) NOTA: Debido a la gran dimensión de la tabla se ha tenido que mostrar sólo una parte.

Como se observa en la *Figura 5-7* los indicadores sólo tienen una muestra por lo que la varianza tiene valor cero. A continuación en la *Figura 5-8* se muestra la misma consulta realizada en la *Figura 5-7* pero ahora el sistema ha sido alimentado con una segunda captura de tráfico.

IP	Indicador	n	M	S <sup>2</sup>	S	SE
1.1.1.1	'O_902'	13	7	0	2018-04-10	1
1.1.1.1	'O_912'	13	2	0	2018-04-10	1
1.1.1.1	'QUIC'	13	17	4	2018-04-10	2
1.1.1.1	'QUIC.Google'	13	43	1600	2018-04-10	2
1.1.1.1	'QUIC.GoogleDocs'	13	18.5	306.25	2018-04-10	2
1.1.1.1	'QUIC.GoogleDrive'	13	1	0	2018-04-10	1
1.1.1.1	'QUIC.GoogleServices'	13	5.5	20.25	2018-04-10	2
1.1.1.1	'SSDP'	13	9.5	72.25	2018-04-10	2
1.1.1.1	'SSL'	13	133	14641	2018-04-10	2
1.1.1.1	'SSL.GMail'	13	2	0	2018-04-10	1
1.1.1.1	'SSL.Google'	13	6.5	20.25	2018-04-10	2
1.1.1.1	'SSL.Office365'	13	1.5	0.25	2018-04-10	2
1.1.1.1	'SSL.WhatsApp'	13	3.5	2.25	2018-04-10	2
1.1.1.1	'SSL_No_Cert'	13	39	0	2018-04-10	1

*Figura 5-8 Segunda consulta a tabla estadísticos.*

Como podemos observar ahora las columnas 5 y 9 que corresponden respectivamente con la varianza y la varianza exponencial poseen valores diferentes a cero.

Los valores de esta tabla serán más cercanos a la realidad cuanto mayor sea el número de muestras con la que se alimente al sistema.

Si tenemos muy pocos datos, los valores de los indicadores presentarán mucha dispersión, por lo que el sistema no obtendrá buenas decisiones.

## 5.2.5 Errores

Cuando se tienen varias capturas y el sistema posee datos suficientes, desactivamos la variable aprendizaje del archivo configuración y ejecutamos el script pasándole tráfico malicioso para verificar que el sistema funciona.

```
anonymus@localhost:~/Escritorio/tfg (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda
anonymus@localhost:~/Escritorio/tfg$ python leer.py flujos/nmap.flows
```

*Figura 5-9 Ejecución del script con tráfico malicioso.*

A continuación como se muestra en la *Figura 5-10* realizamos una consulta a la tabla de logs para verificar si el sistema ha generado alguna detección en los indicadores.

```
mysql> select* from logs;
+-----+-----+-----+-----+-----+
| ip    | fecha   | hora  | tipo   | indicador | valor |
+-----+-----+-----+-----+-----+
| 1.1.1.1 | 2018-10-05 | 13 | ERROR   | Unknown   | 30120 |
| 1.1.1.1 | 2018-10-05 | 13 | ERROR_EXP | Unknown   | 30120 |
+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

*Figura 5-10 Consulta de logs.*

Como se puede observar, el indicador **Unknown** ha sido clasificado como tráfico malicioso tanto en el test de la media móvil como el de la media exponencial móvil. En esta prueba el indicador ha obtenido un valor **30120**. Esto significa que ha habido 30120 flujos IP que nProbe lo ha clasificado como desconocido. En base a este resultado podemos afirmar que el sistema funciona correctamente.

En ocasiones tendremos casos de indicadores nuevos que no se han tenido en cuenta en la fase de aprendizaje del modelo, por lo que resulta interesante la generación de un log que muestre cuando ocurren estos casos. Para ello activaremos la variable ***alertas\_nuevas*** a valor ***TRUE*** del fichero de configuración para dotar al sistema de este comportamiento.

Para verificar el funcionamiento de esta característica, no tendremos en cuenta los siguientes indicadores en la fase de aprendizaje:

- Facebook.
- Youtube.
- Dropbox.
- Skype.

Accederemos a Facebook, Youtube, Dropbox y Skype mientras capturamos el tráfico para poner a prueba el modelo y comprobar si es capaz de realizar la detección y categorizarlos como nuevos.

mysql> select * from logs;						
ip	fecha	hora	tipo	indicador	valor	
1.1.1.1	2018-10-05	13	ERROR	Unknown	30120	
1.1.1.1	2018-10-05	13	ERROR_EXP	Unknown	30120	
1.1.1.1	2018-10-06	13	NUEVO	DNS.Facebook	40	
1.1.1.1	2018-10-06	13	NUEVO	Skype	238	
1.1.1.1	2018-10-06	13	NUEVO	SSL.Dropbox	34	
1.1.1.1	2018-10-06	13	NUEVO	SSL.Facebook	66	
1.1.1.1	2018-10-06	13	NUEVO	SSL.YouTube	47	

7 rows in set (0,00 sec)

Figura 5-11 Consulta en logs de nuevos indicadores.

Como se muestra en la *Figura 5-11* se puede verificar como efectivamente el sistema ha detectado y reportado los nuevos indicadores marcándolos como ***NUEVO***.

## 5.2.6 Indicador Unknown

Este indicador corresponde con el número de flujos marcados como aplicación desconocida, como vimos anteriormente, este indicador es clave para la detección de anomalías. Es por ello por lo que es seleccionado para realizar las pruebas sobre el Sistema y verificar el funcionamiento del mismo.

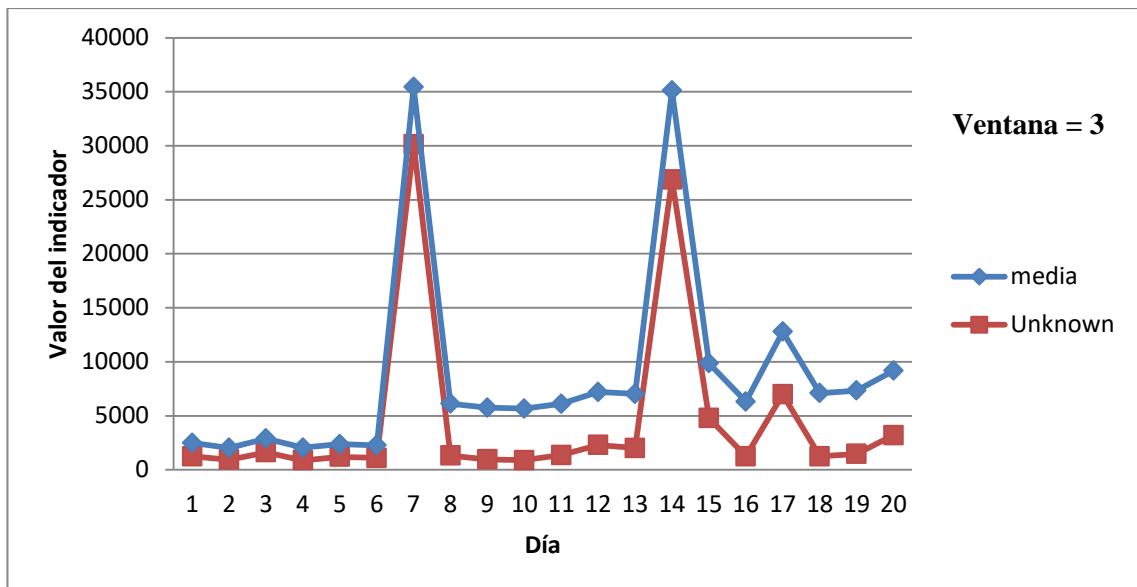


Figura 5-12 Evolución temporal del indicador Unknown.

Como se puede observar en la *Figura 5-12* los días 7 y 14 obtenemos un incremento considerable del indicador, por lo que el sistema lo detecta y reporta una anomalía en tal indicador. También resulta interesante fijarse en el comportamiento del sistema definiendo la capacidad de la ventana a 3. Se observa como el sistema tiene en cuenta las últimas 3 muestras.

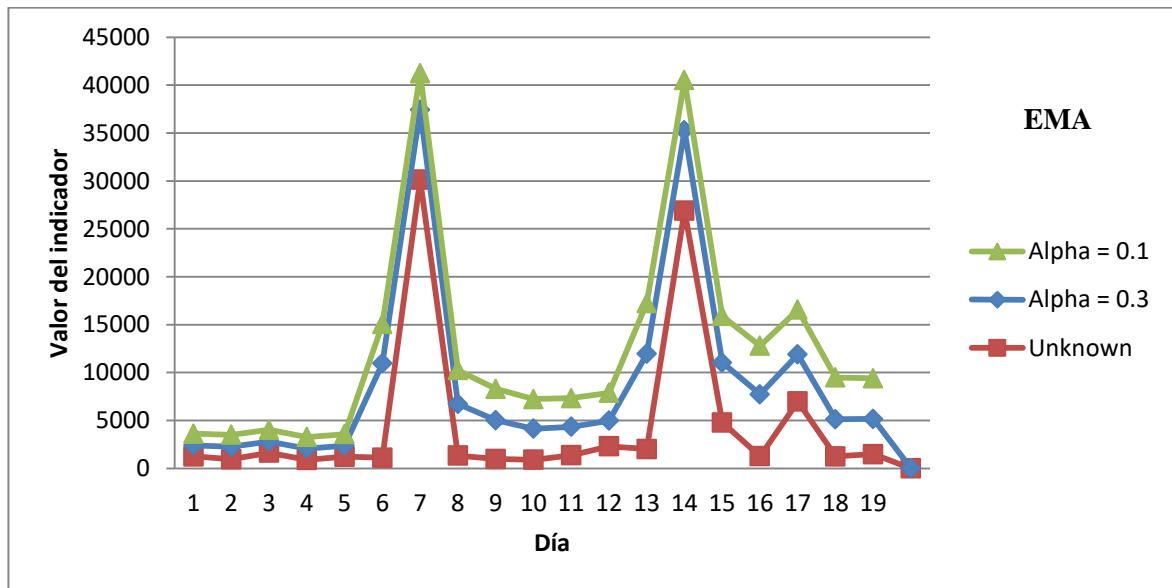


Figura 5-13 Evolución temporal indicador EMA.

En la *Figura 5-13* se muestra la evolución temporal del cálculo de la media exponencial móvil del indicador *Unknown*. A través de este gráfico podemos apreciar la importancia del parámetro *Alpha*, cuanto mayor sea el valor de este parámetro, más se acercará el valor de la *EMA* a la última muestra del indicador.

En este caso, hemos obtenido detección de anomalías en los días 6, 7, 8, 13 y 14 cuando *Alpha* tiene valor **0.1**, en cambio, cuando *Alpha* tiene valor **0.3**. Se obtiene correctamente la detección en los días 7 y 14.





# 6 CONCLUSIONES Y LÍNEAS FUTURAS

---

*Estudia el pasado si quieres definir el futuro.*

*- Confucio -*

**D**espués de la investigación llevada a cabo para el estudio realizado, se procede a la exposición de las conclusiones obtenidas en base a los resultados. También se aportarán ideas para las posibles mejoras futuras que podrían complementar la presente investigación.

## 6.1 Conclusiones

Tras la investigación y desarrollo llevado a cabo en el presente trabajo, se puede afirmar que se han cumplido los siguientes objetivos fijados:

- Se ha investigado y probado en profundidad las diferentes tecnologías, aplicaciones y métodos útiles para la detección de anomalías en el tráfico de red.
- Investigación de distintas amenazas posibles de vulnerabilidades en la red.
- Comparativa de las diferentes tecnologías.
- Creación de indicadores que permitan modelar el tráfico de red.
- Generación de logs.
- Desarrollo de un sistema con capacidad de aprendizaje de patrones de comportamiento en una red, para la detección de vulnerabilidades.
- El software desarrollado es completamente configurable para el usuario, permitiendo ser escalable y poder adaptarse a diferentes volúmenes de tráfico.
- Profundización en el lenguaje más usado comúnmente en el mundo de la ciberseguridad Python.
- Instalación y configuración de un escenario real para la validación de pruebas.

En base a las pruebas y resultados se puede llegar a la conclusión de que el sistema desarrollado funciona correctamente. Se obtienen logs cuando se alimenta al sistema con tráfico malicioso.

Cabe destacar, que el sistema desarrollado es un mecanismo de defensa, el cual permite la detección de posibles nuevas vulnerabilidades que puedan comprometer el buen funcionamiento de una red. Esto es muy importante en la actualidad, pues como hemos visto en la investigación del estado del arte, cada año se obtienen registros de nuevas vulnerabilidades que sistemas como el desarrollado en este estudio, pueden detectar y conseguir así la mitigación de riesgos.

## 6.2 Líneas futuras

Debido a que el presente trabajo se trata de una primera investigación en la detección de vulnerabilidades. Existen posibles líneas de mejora que permitan el avance de la investigación y desarrollo:

- Unificar indicadores similares para evitar tener numerosos indicadores y así reducir falsos positivos en la detección.
- Realizar un estudio similar haciendo uso de redes neuronales.
- Realizar comparativa con sistemas que hagan uso de redes neuronales.
- Trabajar directamente con la base de datos para reducir el tiempo computacional que implica la generación de ficheros de forma dinámica.
- Creación de un dashboard tipo web con la posibilidad de poder configurar el sistema de forma fácil e intuitiva para el usuario.
- Combinar el sistema desarrollado con firewall para conseguir un sistema que detecta y corta las posibles nuevas vulnerabilidades.
- Añadir nuevas aplicaciones a la librería nDPI para aumentar el abanico de detecciones de aplicaciones.





# REFERENCIAS

---

- [1] incibe (Instituto Nacional de Ciberseguridad), “Amenaza vs Vulnerabilidad, ¿sabes en qué se diferencian? | INCIBE,” 2017. [Online]. Available: <https://www.incibe.es/protege-tu-empresa/blog/amenaza-vs-vulnerabilidad-sabes-se-diferencian>. [Accessed: 29-Oct-2018].
- [2] P. K. Pateriya and S. S. Kumar, “Analysis on Man in the Middle Attack on SSL,” *Int. J. Comput.* ..., vol. 45, no. 23, pp. 43–46, 2012.
- [3] R. G. Wiliam, “Trujillo\_2016,” p. 177, 2016.
- [4] INCIBE, “Ransomware: una guía de aproximación para el empresario,” p. 25, 2017.
- [5] M. Barrionuevo, M. Lopresti, N. Miranda, and F. Píccoli, “red usando imágenes y técnicas de Computación de Alto Desempeño .,” pp. 1166–1175.
- [6] N. Stivet, T. Álvarez, and L. F. Pedraza, “Redes neuronales y predicción de tráfico Neural networks and prediction of traffic,” *Edición Espec.*, vol. 15, no. 29, pp. 90–97, 2011.
- [7] Anonymous, “Detecting Anomalies in Communication Packet Streams Based on Generative Adversarial Networks,” *Neuropsychology*, vol. 58, no. 6, pp. 1151–1161, 2006.
- [8] P. Larrañaga, I. Inza, and A. Moujahid, “Tema 8. Redes Neuronales,” *Researchgate*, p. 19, 2015.
- [9] R&S, “R & S ® PACE 2 Solution Guide Contents.”
- [10] T. Firewalls, “Firewalls industriales DPI ¿ Qué es un firewall industrial DPI ? Firewalls industriales DPI,” pp. 1–5.
- [11] J. J. Dougherty, “Interested in learning more ? In sti tu Au th re ns f rig,” *Style (DeKalb, IL)*, no. Security 401, 2011.
- [12] E. De La Hoz, E. M. De La Hoz, A. Ortiz, and J. Ortega, “Modelo de detección de intrusiones en sistemas de red, realizando selección de características con FDR y entrenamiento y clasificación con SOM,” *Inge Cuc*, vol. 8, no. 1, pp. 85–116, 2012.
- [13] E. Arias, “Instituto Politécnico Nacional,” *Cic.Ipn.Mx*, pp. 1–80, 2010.
- [14] L. R. M, “Snort como herramienta administrativa,” no. 5, pp. 74–78.
- [15] cvedetails, “CVE security vulnerability database. Security vulnerabilities, exploits, references and more,” 2018. [Online]. Available: <https://www.cvedetails.com/>. [Accessed: 01-Nov-2018].
- [16] “Vulnerability distribution of cve security vulnerabilities by types.” [Online]. Available: <https://www.cvedetails.com/vulnerabilities-by-types.php>. [Accessed: 01-Nov-2018].
- [17] L. Deri, M. Martinelli, and A. Cardigliano, “nDPI: Open-Source High-Speed Deep Packet Inspection.”
- [18] sourceforge, “L7-filter Supported Protocols.” [Online]. Available: <http://l7-filter.sourceforge.net/protocols>. [Accessed: 01-Nov-2018].
- [19] P. Bulletin, “NBAR2 Protocol Library,” pp. 1–47, 2013.
- [20] R. Hofstede *et al.*, “Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.
- [21] M. De, “Monitorización de una red académica mediante Netflow 1.”
- [22] R. Antonello *et al.*, “Deep packet inspection tools and techniques in commodity platforms: Challenges and trends,” *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 1863–1878, 2012.
- [23] “Túnel (informática) - Wikipedia, la enciclopedia libre.” [Online]. Available:

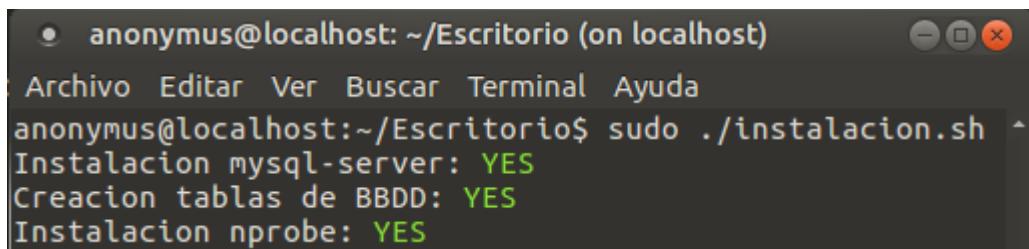
- [https://es.wikipedia.org/wiki/T%C3%BAnel\\_\(inform%C3%A1tica\)](https://es.wikipedia.org/wiki/T%C3%BAnel_(inform%C3%A1tica)). [Accessed: 04-Nov-2018].
- [24] concepto.de, “¿Qué es ISP?” [Online]. Available: <https://concepto.de/isp/>. [Accessed: 05-Nov-2018].
- [25] E. Lgplv, D. Packet, and I. Library, “nDPI - Quick Start Guide,” no. October, pp. 1–15, 2016.
- [26] P. Kocher, “Internet Engineering Task Force (IETF) A. Freier Request for Comments: 6101 P. Karlton Category: Historic Netscape Communications,” 2011.
- [27] A. V. Aho and M. J. Corasick, “Efficient string matching: an aid to bibliographic search,” *Commun. ACM*, vol. 18, no. 6, pp. 333–340, Jun. 1975.
- [28] L. Deri SharkFest, “#sf17eu • Estoril, Portugal How to rule the world... by looking at packets! Turning Wireshark into a Traffic Monitoring Tool: Moving from packet details to the big picture.”
- [29] ntop.org, “nProbe documentation — nProbe 8.5 documentation.” [Online]. Available: <https://www.ntop.org/guides/nProbe/>. [Accessed: 07-Nov-2018].
- [30] <http://www.tcpdump.org>, “Manpage of TCPDUMP.” [Online]. Available: <http://www.tcpdump.org/manpages/tcpdump.1.html>. [Accessed: 07-Nov-2018].
- [31] “Secure Copy - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Secure\\_Copy](https://es.wikipedia.org/wiki/Secure_Copy). [Accessed: 07-Nov-2018].
- [32] web.mit.edu, “Protocolo SSH.” [Online]. Available: <https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rges-4/ch-ssh.html>. [Accessed: 07-Nov-2018].
- [33] wikipedia, “Valores separados por comas - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Valores\\_separados\\_por\\_comas](https://es.wikipedia.org/wiki/Valores_separados_por_comas). [Accessed: 07-Nov-2018].
- [34] Elvis Pranskevichus, “What’s New In Python 3.7 — Python 3.7.1 documentation.” [Online]. Available: <https://docs.python.org/3/whatsnew/3.7.html>. [Accessed: 08-Nov-2018].
- [35] “pandas: powerful Python data analysis toolkit Release 0.23.4 Wes McKinney & PyData Development Team,” 2018.
- [36] “Guide to Numpy : Travis E. Oliphant : Free Download, Borrow, and Streaming : Internet Archive.” [Online]. Available: <https://archive.org/details/NumPyBook/page/n25>. [Accessed: 08-Nov-2018].
- [37] “8.1. datetime — Basic date and time types — Python 2.7.15 documentation.” [Online]. Available: <https://docs.python.org/2/library/datetime.html>. [Accessed: 08-Nov-2018].
- [38] “os — Miscellaneous operating system interfaces — Python 3.7.1 documentation.” [Online]. Available: <https://docs.python.org/3/library/os.html>. [Accessed: 08-Nov-2018].
- [39] “28.1. sys — System-specific parameters and functions — Python 2.7.15 documentation.” [Online]. Available: <https://docs.python.org/2/library/sys.html>. [Accessed: 08-Nov-2018].
- [40] “MySQL Connector/Python Developer Guide.”
- [41] “math — Mathematical functions — Python 3.7.1 documentation.” [Online]. Available: <https://docs.python.org/3/library/math.html>. [Accessed: 08-Nov-2018].
- [42] “10.10. shutil — High-level file operations — Python 2.7.15 documentation.” [Online]. Available: <https://docs.python.org/2/library/shutil.html>. [Accessed: 08-Nov-2018].
- [43] “configparser — Configuration file parser — Python 3.7.1 documentation.” [Online]. Available: <https://docs.python.org/3/library/configparser.html>. [Accessed: 08-Nov-2018].
- [44] “Wijnand Modderman-Lenstra,” 2017.
- [45] M. U. Y. Valiosa, “CURSO DE TCP / IP : ICMP ( Protocolo de Mensajes de Control de Internet ).”
- [46] E. W. Weisstein and E. W. Weisstein, “Media aritmética,” *MathWorld*.
- [47] wikipedia, “Varianza - Wikipedia, la enciclopedia libre.” [Online]. Available: <https://es.wikipedia.org/wiki/Varianza>. [Accessed: 15-Nov-2018].

- [48] R. G. Brown, *Exponential Smoothing for Predicting Demand*. Cambridge, Massachusetts: Arthur D. Little Inc., 1956.
- [49] “MySQL 8.0 Reference Manual - Including MySQL NDB Cluster 8.0.”

# ANEXO A: INSTALACIÓN DEL PROYECTO

- Script de instalación del proyecto:

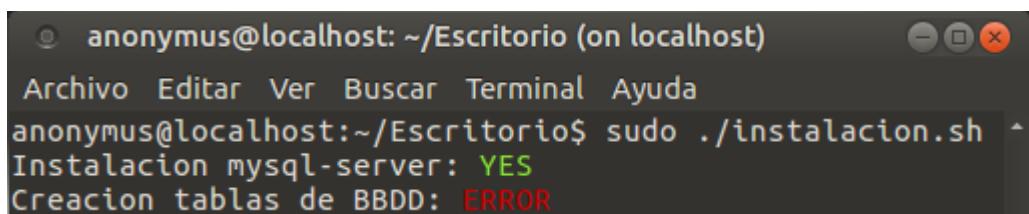
Ejecutamos el script de instalación del proyecto, si no ha habido ningún error el script deberá mostrar **YES** en todas los pasos realizados.



```
anonymus@localhost: ~/Escritorio (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda
anonymus@localhost:~/Escritorio$ sudo ./instalacion.sh
Instalacion mysql-server: YES
Creacion tablas de BBDD: YES
Instalacion nprobe: YES
```

Figura A-1 instalación del proyecto.

En caso de que ocurra algún error en un paso del script, la instalación se para y muestra **ERROR** en dicho paso.



```
anonymus@localhost: ~/Escritorio (on localhost)
Archivo Editar Ver Buscar Terminal Ayuda
anonymus@localhost:~/Escritorio$ sudo ./instalacion.sh
Instalacion mysql-server: YES
Creacion tablas de BBDD: ERROR
```

Figura A-2 Error en la instalación.

```
1.#!/bin/bash
2.#
3.#@Autor: Agustín Walabonso Lara Romero
4.#@Descripción: Instalación del proyecto
5.#
6.
7.#####
8.## Actualización de los paquetes
9.#####
10.sudo apt-get update -y
11.
12.#####
13.## Instalación de MySQL
14.#####
15.sudo apt-get install mysql-server -y
16.if [ $? = 0 ]
17.then
18.echo -e "Instalacion mysql-server: \e[92mYES\e[0m"
19.else
20.echo -e "Instalacion mysql-server: \e[0;31mERROR\e[0m"
21.exit 0
22.fi
23.
24.#####
25.## Creación de las tablas de la base de datos
```

```

26.      ##### USUARIO_BASE_DATOS='wala'
27.      USUARIO_BASE_DATOS='wala'
28.      BBDD=tfg
29.      CREAR_TABLAS=crear_tablas.sql
30.      mysql -u wala -
   p$USUARIO_BASE_DATOS $BBDD < $CREAR_TABLAS 2>/dev/null
31.      if [ $? = 0 ]
32.      then
33.          echo -e "Creacion tablas de BBDD: \e[92mYES\e[0m"
34.      else
35.          echo -e "Creacion tablas de BBDD: \e[0;31mERROR\e[0m"
36.          exit 0
37.      fi
38.
39.      ##### Instalación de nprobe
40.      # Instalación de nprobe
41.      #####
42.      sudo apt-get install nprobe -y
43.      if [ $? = 0 ]
44.      then
45.          echo -e "Instalacion nprobe: \e[92mYES\e[0m"
46.      else
47.          echo -e "Instalacion nprobe: \e[0;31mERROR\e[0m"
48.          exit 0
49.      fi

```

- Comandos necesarios para la creación del usuario y de la base de datos:

```

1. # Creación del usuario wala con contraseña wala
2. CREATE USER 'wala'@'localhost' IDENTIFIED BY 'wala';
3. # Creación de base datos con nombre tfg
4. CREATE DATABASE tfg;
5. # Conceder al usuario wala permisos con la base de datos
6. GRANT ALL ON tfg.* TO 'wala'@'localhost';

```

- Script de creación de las tablas de la base de datos:

```

1. --
2. -- @Autor: Agustín Walabonso Lara Romero
3. -- @Descripción: Script de creación de las tablas de la base de datos
4. --
5.
6. -- Creación de la tabla de aplicaciones
7. CREATE TABLE IF NOT EXISTS aplicaciones (
8.     ip VARCHAR(15) NOT NULL,
9.     aplicacion VARCHAR(50) NOT NULL,
10.    num_veces INT(11) NOT NULL,
11.    fecha DATE NOT NULL,
12.    hora VARCHAR(2) NOT NULL,
13.    PRIMARY KEY (ip, aplicacion, num_veces, fecha, hora)
14. );
15.
16. -- Creación de la tabla estadísticos
17. CREATE TABLE IF NOT EXISTS estadisticos (
18.     ip VARCHAR(15) NOT NULL,
19.     indicador VARCHAR(40) NOT NULL,
20.     hora INT(11) NOT NULL,
21.     media FLOAT DEFAULT NULL,

```

```

22.      varianza FLOAT DEFAULT NULL,
23.      fecha DATE NOT NULL,
24.      muestras INT(11) DEFAULT NULL,
25.      media_exp FLOAT DEFAULT NULL,
26.      varianza_exp FLOAT DEFAULT NULL,
27.      PRIMARY KEY (ip, indicador, hora, fecha)
28.  );
29.
30.  -- Creación de la tabla icmp_destino
31.  CREATE TABLE IF NOT EXISTS icmp_destino (
32.      ip VARCHAR(15) NOT NULL,
33.      ip_destino VARCHAR(50) NOT NULL,
34.      num_veces INT(11) NOT NULL,
35.      fecha DATE NOT NULL,
36.      hora VARCHAR(2) NOT NULL,
37.      PRIMARY KEY (ip, ip_destino, num_veces, fecha, hora)
38.  );
39.
40.  -- Creación de la tabla logs
41.  CREATE TABLE IF NOT EXISTS logs (
42.      ip VARCHAR(15) NOT NULL,
43.      fecha DATE NOT NULL,
44.      hora INT(11) NOT NULL,
45.      tipo VARCHAR(20) NOT NULL,
46.      indicador VARCHAR(50) NOT NULL,
47.      valor INT(11) DEFAULT NULL,
48.      PRIMARY KEY (ip, fecha, hora, indicador, tipo)
49.  );
50.
51.  -- Creación de la tabla num_app
52.  CREATE TABLE IF NOT EXISTS num_app (
53.      ip VARCHAR(15) NOT NULL,
54.      aplicacion VARCHAR(50) NOT NULL,
55.      num_veces INT(11) DEFAULT NULL,
56.      fecha DATE NOT NULL,
57.      hora VARCHAR(2) NOT NULL,
58.      PRIMARY KEY (ip, aplicacion, fecha, hora)
59.  );
60.
61.  -- Creación de la tabla num_ip
62.  CREATE TABLE IF NOT EXISTS num_ip (
63.      ip VARCHAR(15) NOT NULL,
64.      ip_aux VARCHAR(15) DEFAULT NULL,
65.      num_veces INT(11) DEFAULT NULL,
66.      fecha DATE NOT NULL,
67.      hora VARCHAR(2) NOT NULL,
68.      PRIMARY KEY (ip, fecha, hora)
69.  );
70.
71.  -- Creación de la tabla puertos_destino
72.  CREATE TABLE IF NOT EXISTS puertos_destino (
73.      ip VARCHAR(15) NOT NULL,
74.      puerto VARCHAR(50) NOT NULL,
75.      num_veces INT(11) NOT NULL,
76.      fecha DATE NOT NULL,
77.      hora VARCHAR(2) NOT NULL,
78.      PRIMARY KEY (ip, puerto, num_veces, fecha, hora)
79.  );
80.
81.  -- Creación de la tabla puertos_origen
82.  CREATE TABLE IF NOT EXISTS puertos_origen (
83.      ip VARCHAR(15) NOT NULL,

```

```
84.      puerto VARCHAR(50) NOT NULL,  
85.      num_veces INT(11) NOT NULL,  
86.      fecha DATE NOT NULL,  
87.      hora VARCHAR(2) NOT NULL,  
88.      PRIMARY KEY (ip,puerto,num_veces,fecha,hora)  
89. );  
90.  
91. -- Creación de la tabla relacion_ip  
92. CREATE TABLE IF NOT EXISTS relacion_ip (  
93.     ip VARCHAR(15) NOT NULL,  
94.     ip_destino VARCHAR(50) NOT NULL,  
95.     num_veces INT(11) NOT NULL,  
96.     fecha DATE NOT NULL,  
97.     hora VARCHAR(2) NOT NULL,  
98.     PRIMARY KEY (ip,ip_destino,num_veces,fecha,hora)  
99. );
```

# ANEXO B: ESTRUCTURA DEL PROYECTO

A continuación se muestra a modo de capturas la jerarquía de ficheros hasta la ubicación de los indicadores:

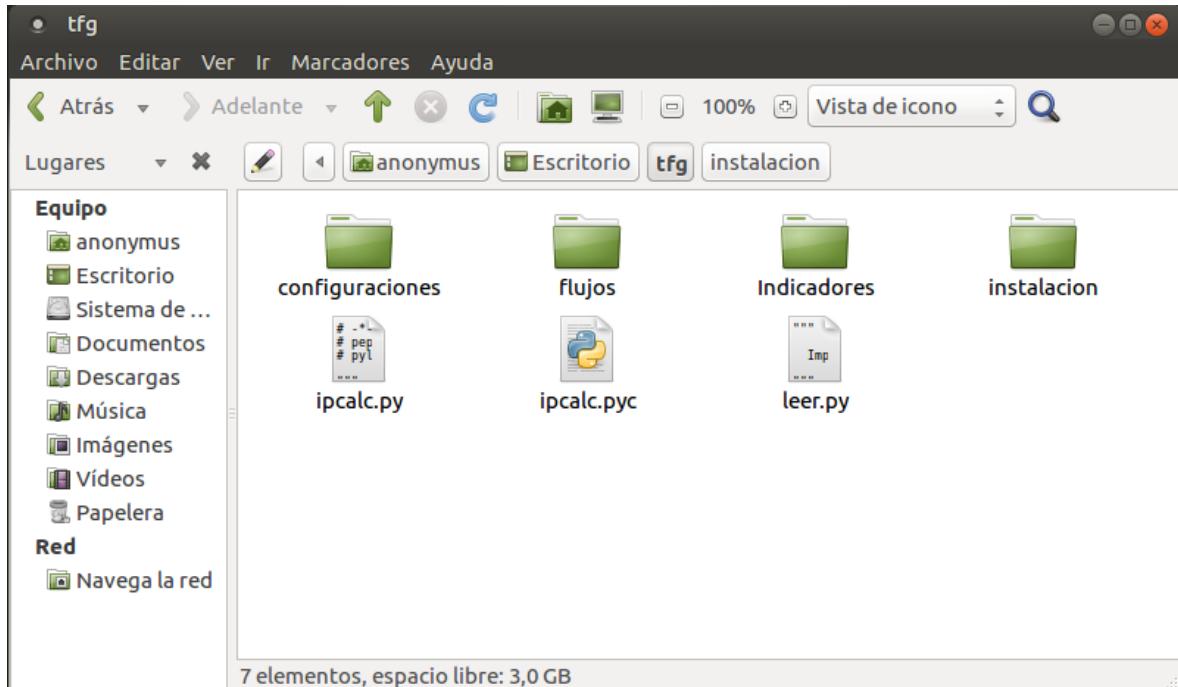


Figura B-1 Carpeta tfg.

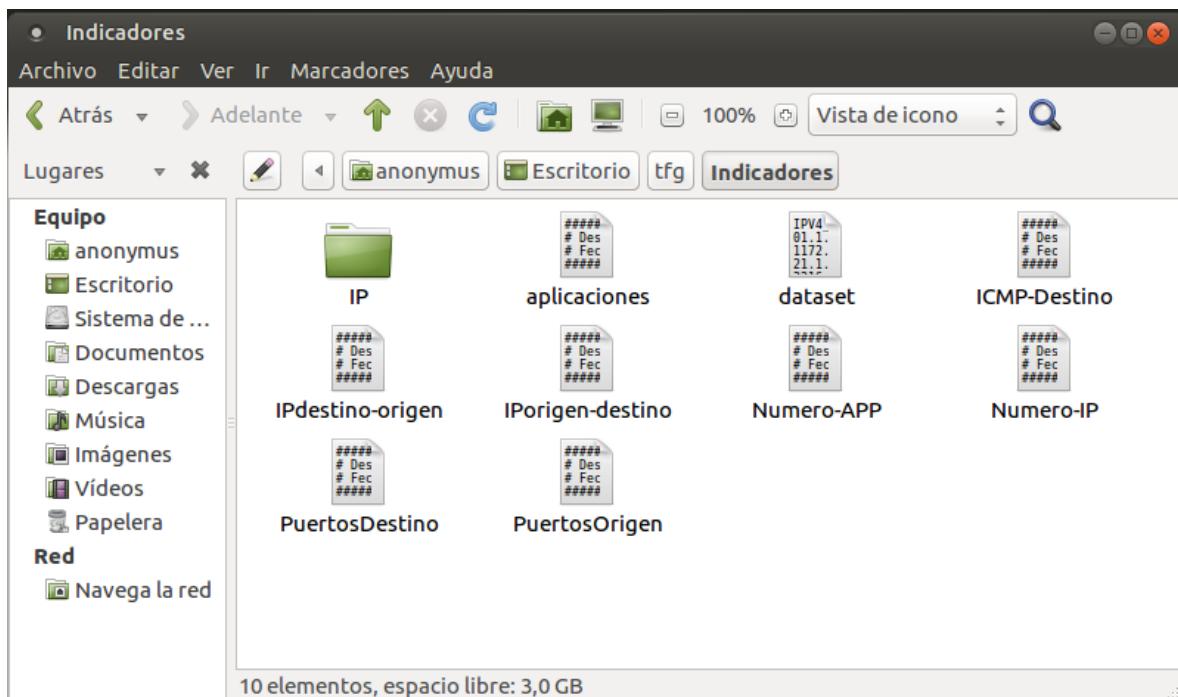


Figura B-2 Carpeta indicadores.

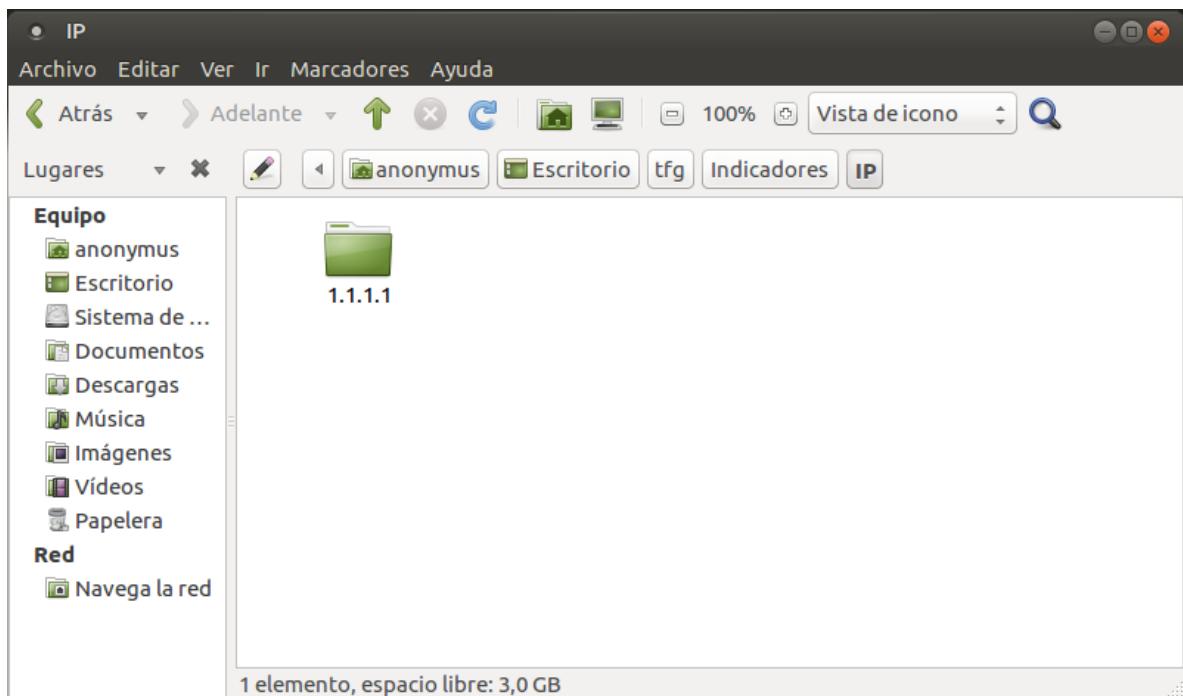


Figura B-3 Carpeta IP.

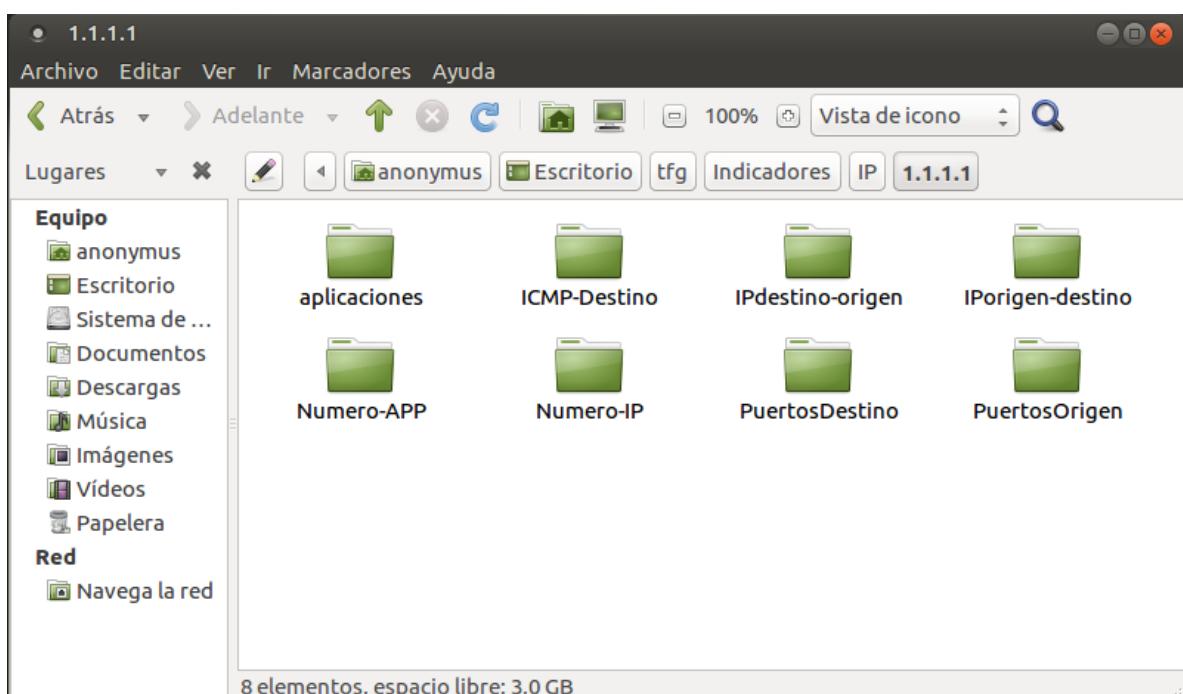
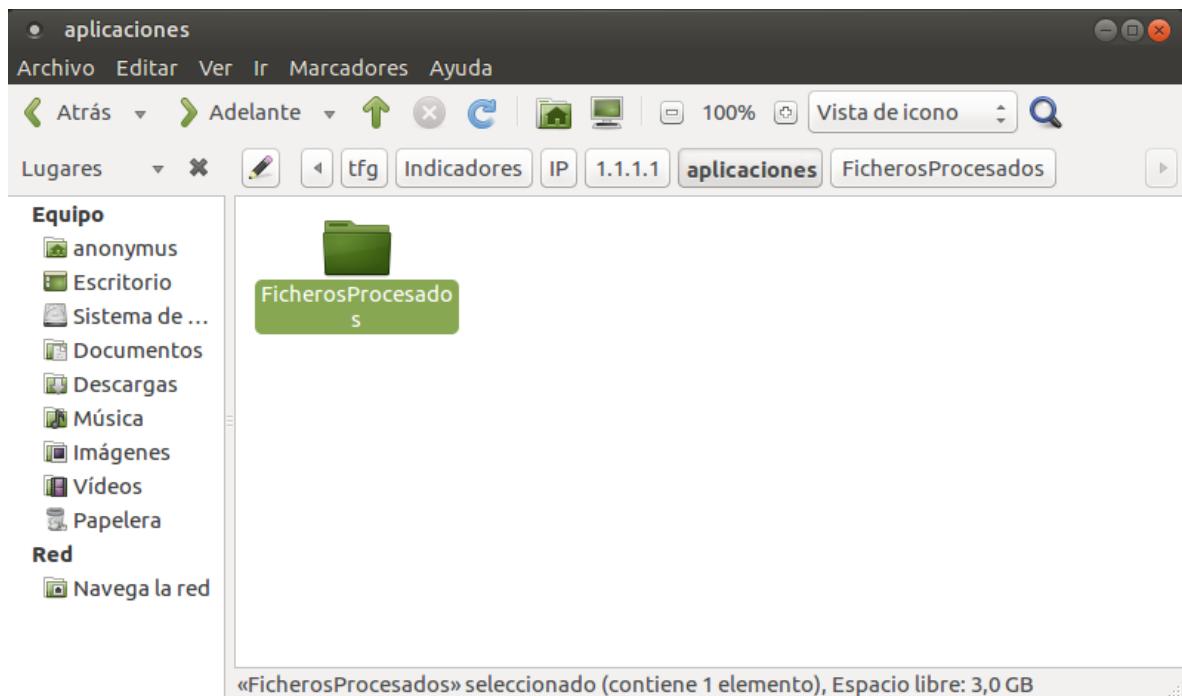


Figura B-4 Carpeta 1.1.1.1



*Figura B-5 Carpeta indicador aplicaciones.*

## ANEXO D: FICHERO DE CONFIGURACIÓN

---

```
1. 
2. # Fichero de configuraciones globales del programa
3. 
4. # Configuraciones de la fecha y hora
5. # en la que se realiza el estudio.
6. # Si se desea que la fecha y hora corresponda
7. # con el archivo de captura realizado, se deberá
8. # poner el valor AUTO, en cambio si se desea asignar una hora
9. # y fecha, debe ser en el formato: yyyyymmddhh por ejemplo,
10. # 10 de Marzo de 2018 a las 14:00 -> 2018031014
11. [FECHA]
12. fecha = 2018041013
13. 
14. # Configuraciones sobre la Base de datos
15. [BBDD]
16. user = wala
17. password = wala
18. host = localhost
19. database = tfg
20. clean = TRUE
21. 
22. # Configuraciones llevadas a cabo en las estadísticas
23. # y comportamiento del sistema
24. [ESTADISTICAS]
25. capacidad_ventana = 3
26. coeficiente_error = 3
27. aprendizaje = TRUE
28. eliminar_anomalous = FALSE
29. alertas_nuevas = TRUE
30. alpha = 0.3
31. 
32. # Red sobre la que se desea monitorizar
33. [TRATAMIENTO]
34. network = 192.168.0.0/24
```

## ANEXO D: CÓDIGO SCRIPT PYTHON

---

```
1. """
2.
3. Importaciones de Librerias necesarias
4.
5. """
6. import pandas as pd
7. import numpy as np
8. from datetime import datetime, date, time, timedelta
9. import os
10.    import sys
11.    import json
12.    import mysql.connector
13.    import math
14.    import shutil
15.    import ConfigParser
16.    import ipcalc as ipc
17.
18. """
19.
20.     Parametros obtenidos por el fichero de configuracion
21.     ubicado en configuraciones/configuraciones.conf
22.
23. """
24. pd.options.mode.chained_assignment = None # default='warn'
25. config = ConfigParser.ConfigParser()
26. config.read('configuraciones/configuraciones.conf')
27. fecha = str(config.get('FECHA', 'fecha'))
28. usuario = str(config.get('BBDD', 'user'))
29. password = str(config.get('BBDD', 'password'))
30. host = str(config.get('BBDD', 'host'))
31. database = str(config.get('BBDD', 'database'))
32. clean = str(config.get('BBDD', 'clean'))
33. capacidad_ventana = int(config.get('ESTADISTICAS', 'capacidad_ventana'))
34. coeficiente_error = int(config.get('ESTADISTICAS', 'coeficiente_error'))
35. alpha = float(config.get('ESTADISTICAS', 'alpha'))
36. aprendizaje = str(config.get('ESTADISTICAS', 'aprendizaje'))
37. eliminar_anomalous = str(config.get('ESTADISTICAS', 'eliminar_anomalous'))
38. alertas_nuevas = str(config.get('ESTADISTICAS', 'alertas_nuevas'))
```

```

39.     network = str(config.get('TRATAMIENTO', 'network'))
40.
41.     def calculaFecha(flow_start):
42.         """
43.
44.             Funcion que calcula y devuelve la fecha y hora a la que
45.             se ha realizado una captura de flujo IP
46.
47.             Parametros:
48.
49.                 flow_start -- milisegundos transcurridos desde
50.                 la fecha 1 de julio de 1970
51.                                         a las 01:02:03.123456.
52.                                         (Este parametro es
53.                                         sumado a la fecha mencionada anteriormente
54.                                         y asi poder calcular la
55.                                         fecha y hora de la captura realizada)
56.                                         """
57.                                         dia_wireshark = datetime.strptime('1970-01-01
58.                                         01:02:03.123456', '%Y-%m-%d %H:%M:%S.%f')
59.                                         dia_wireshark = dia_wireshark +
60.                                         timedelta(milliseconds=flow_start)
61.                                         return str(dia_wireshark.strftime('%Y%m%d%H'))
62.
63.     def calcula_var_exp(alpha, varianza):
64.         """
65.
66.             Funcion que calcula y devuelve la varianza exponencial
67.             Parametros:
68.                 alpha      -- Factor de suavizado para el
69.                 calculo de la varianza
70.                                         varianza -- Valor de la varianza sobre la que
71.                                         se realiza el
72.                                         siguiente calculo:
73.                                         (varianza_exp = (alpha / (2 - alpha)) * varianza)
74.
75.                                         """
76.                                         varianza_exp = (alpha / (2 - alpha)) * varianza
77.                                         return varianza_exp
78.
79.     def calcula_ema(alpha, lista):
80.         """
81.
82.             Funcion que calcula y devuelve la esperanza exponencial
83.             Parametros:
84.                 alpha -- Factor de suavizado para el calculo
85.                 de
86.                                         la esperanza exponencial
87.                                         lista -- Conjunto de valores sobre la que se
88.                                         realiza
89.                                         el siguiente calculo de la esperanza
90.                                         (alpha * lista[i] + (1 - alpha) *
91.                                         EXP[i - 1])

```

```

88.
89.
90.        """
91.        ema = []
92.        if(len(lista) > 0):
93.            ema.append(lista[0])
94.            for i in range(1, len(lista)):
95.                ema.append(alpha * lista[i] + (1 - alpha) * ema[i - 1])
96.        return ema[len(ema) - 1]
97.
98.    def insertarBaseDatosIndicadores(carpeta, tabla, colum_name):
99.        """
100.           Funcion que inserta valores en base de datos
101.           de los distintos indicadores
102.
103.           Parametros:
104.
105.               carpeta --     Ubicacion de la carpeta donde se
106.               encuentra el
107.               indicador que se desea insertar
108.               en base de datos
109.               tabla --      Nombre de la tabla donde se
110.               desea insertar
111.               los valores del
112.               indicador
113.
114.               """
115.
116.               if len(os.listdir(carpeta)) > 1:
117.                   cnx = mysql.connector.connect(user = usuario, pass
118. word = password, host = host, database = database)
119.                   cursor = cnx.cursor()
120.                   ficheroIP_aux = carpeta
121.                   + '/' + os.listdir(carpeta)[0]
122.
123.                   if 'FicherosProcesados' in ficheroIP_aux:
124.                       ficheroIP_aux = carpeta
125.                       + '/' + os.listdir(carpeta)[1]
126.                   ficheroIP = open(ficheroIP_aux, 'r')
127.                   lines = ficheroIP.readlines()
128.
129.                   if(lines[0] != 'NO SE ENCUENTRA LA IP'):
130.                       fecha_aux = lines[0]
131.                       fecha = fecha_aux[0:8]
132.                       hora = fecha_aux[8:]
133.                       ip = lines[1].rstrip('\n')
134.                       i=0
135.                       ayuda_indicador = ''
136.
137.                   if tabla == 'puertos_destino':
138.                       ayuda_indicador = 'D '
139.                   elif tabla == 'puertos_origen':
140.                       ayuda_indicador = 'O '

```

```

138.                               elif tabla == 'icmp_destino':
139.                                   ayuda_indicador = 'I '
140.                                   for linea in lines:
141.                                       if(i >= 2):
142.                                           linea_aux = linea.split(
143.                                               ')
144.                                           r + linea_aux[0]
145.                                           num_veces = linea_aux[len(
146.                                               linea_aux) - 1]
147.                                           cursor.execute('INSERT
INTO ' + tabla + ' VALUES ('' + ip + "", "' + indicador + '", ' +
num_veces + ', ' + fecha + ', ' + hora + ')')
148.                                           cnx.commit()
149.                                           i += 1
150.                                           calcularIndicadores(ip, tabla, colum_name,
hora, fecha)
151.                                           ficheroIP.close()
152.                                           os.system('cat ' + ficheroIP_aux + ' >> ' +
carpeta + '/FicherosProcesados/Historico')
153.                                           os.system('rm ' + ficheroIP_aux)
154.                                           cursor.close()
155.                                           cnx.close()
156.               def generarAlerta(ip, nuevoIndicador, indicador, fecha, hora, medi
a, varianza, media_anterior, varianza_anterior, nuevo_valor, media_exp,
media_exp_anterior, varianza_exp, varianza_exp_anterior):
157.                   """
158.                       Funcion que evalua el valor de un indicador a traves de
dos metodos, la media movil exponencial
159.                           y la media movil.
160.                           En caso de que el sistema no este en modo aprendizaje y
el valor exceda los estandares predefinido
161.                           se insertara una alerta en la base de datos de la tabla
de nombre logs.
162.
163.               Parametros:
164.
165.                   ip --      Direccion IP correspondiente al
indicador
166.
167.                   nuevoIndicador -- Bandera que vale uno
cuando aun no existe ningun resgistro para el indicador
168.
169.                   indicador --     Nombre del indicador
170.
171.                   fecha --      Fecha en formato YYYYMMDD que
corresponde al indicador
172.
173.                   hora --       Hora en formato HH que corresponde
al indicador
174.
175.                   media -- Calculo de la media del indicador
176.                               incluyendo la ultima muestra
177.
178.                   varianza -- Calculo de la varianza del
indicador
179.                               incluyendo la ultima
muestra
180.

```

```

181.                     media_anterior --      Calculo de la media del
   indicador
182.                                         NO
   incluyendo la ultima muestra
183.
184.                     varianza_anterior --      Calculo de la
   varianza del indicador
185.                                         NO
   incluyendo la ultima muestra
186.
187.                     nuevo_valor --      Valor de la ultima muestra
   del indicador
188.
189.                     media_exp --    Calculo de la media exponencial
   movil del indicador
190.                                         incluyendo la ultima
   muestra
191.
192.                     media_exp_anterior -- Calculo de la media
   exponencial movil del indicador
193.                                         NO
   incluyendo la ultima muestra
194.
195.                     varianza_exp --      Calculo de la varianza
   exponencial movil del indicador
196.                                         incluyendo la
   ultima muestra
197.
198.                     varianza_exp_anterior -- Calculo de la
   varianza exponencial movil del indicador
199.                                         N
   O incluyendo la ultima muestra
200.

201.         """
202.
203.         rellenar = True
204.         cnx = mysql.connector.connect(user = usuario, password = p
   assword, host = host, database = database)
205.         cursor = cnx.cursor()
206.
207.         if media_anterior == None:
208.             media_anterior = media
209.             varianza_anterior = varianza
210.             media_exp_anterior = media_exp
211.             varianza_exp_anterior = varianza_exp
212.
213.             if nuevoIndicador and aprendizaje == 'TRUE' and alertas_nu
   evas == 'TRUE':
214.                 cursor.execute('INSERT INTO logs VALUES (' + ip
   + '", ' + str(fecha) + ', ' + str(hora) + ', "NUEVO", "' + indicador
   + '", ' + str(nuevo_valor) + ')')
215.             else:
216.                 error = abs(nuevo_valor - media_anterior)
217.                 error_exp = abs(nuevo_valor - media_exp_anterior)
218.                 if error > coeficiente_error
   * math.sqrt(varianza_anterior) and aprendizaje == 'FALSE':
219.                     cursor.execute('INSERT INTO logs VALUES
   ("' + ip + '", ' + str(fecha) + ', ' + str(hora) + ', "ERROR", "' +
   indicador + '", ' + str(nuevo_valor) + ')')
220.                     rellenar = False

```

```

221.             if error_exp > coeficiente_error
222.                 * math.sqrt(varianza_exp_anterior) and aprendizaje == 'FALSE':
223.                     cursor.execute('INSERT INTO logs VALUES
224.                         (' + ip + '", ' + str(fecha) + ', ' + str(hora) + ', "ERROR_EXP", "' +
225.                         indicador + '", ' + str(nuevo_valor) + ')')
226.                     rellenar = False
227.
228.     def rellenarBaseDatosEstadisticos(ip, indicador, hora, media, vari-
229.         anza, fecha, nuevo_valor, num_muestras, media_exp, varianza_exp):
230.             """
231.                 Funcion que rellena la tabla Estadisticos, la cual
232.                 contiene todos los indicadores con
233.                     los calculos estadisticos realizados.
234.                 Parametros:
235.
236.                     ip --           Direccion IP correspondiente al
237.                     indicador
238.                     indicador --      Nombre del indicador
239.
240.                     hora --          Hora en formato HH que corresponde
241.                     al indicador
242.                     media --          Calculo de la media del indicador
243.                                     incluyendo la ultima muestra
244.
245.                     varianza --       Calculo de la varianza del
246.                     indicador
247.                                     incluyendo la ultima
248.                     muestra
249.                     fecha --          Fecha en formato YYYYMMDD que
250.                     corresponde al indicador
251.                     nuevo_valor --    Valor de la ultima muestra
252.                     del indicador
253.                     indicador
254.                     num_muestras --   Numero de muestras del
255.                     media_exp --      Calculo de la media exponencial
256.                                     movil del indicador
257.                                     incluyendo la ultima
258.                     muestra
259.                     varianza_exp --   Calculo de la varianza
260.                     exponencial movil del indicador
261.                                     incluyendo la ultima muestra
262.             """
263.             rellenar = True
264.             cnx = mysql.connector.connect(user = usuario, password = p-
265.                 assword, host = host, database = database)
266.             cursor = cnx.cursor()

```

```

264.         cursor.execute('SELECT media FROM estadisticos WHERE ip=
    "' + ip + '" AND hora=' + str(hora) + ' AND indicador = "' + indicador
    + '"' + ' order by fecha desc limit 1')
265.         media_anterior = cursor.fetchone()
266.         cursor.execute('SELECT varianza FROM estadisticos WHERE
    ip= "' + ip + '" AND hora=' + str(hora) + ' AND indicador = "' +
    indicador + '"' + ' order by fecha desc limit 1')
267.         varianza_anterior = cursor.fetchone()
268.         cursor.execute('SELECT media_exp FROM estadisticos WHERE
    ip= "' + ip + '" AND hora=' + str(hora) + ' AND indicador = "' +
    indicador + '"' + ' order by fecha desc limit 1')
269.         media_exp_anterior = cursor.fetchone()
270.         cursor.execute('SELECT varianza_exp FROM estadisticos
    WHERE ip= "' + ip + '" AND hora=' + str(hora) + ' AND indicador = "' +
    indicador + '"' + ' order by fecha desc limit 1')
271.         varianza_exp_anterior = cursor.fetchone()
272.         if media_anterior is None or varianza_anterior is None:
273.             generarAlerta(ip, True, indicador, fecha, hora, me
    dia, varianza, None, None, nuevo_valor, media_exp, media_exp_anterior,
    varianza_exp, varianza_exp_anterior)
274.             cursor.execute('INSERT INTO estadisticos VALUES
    ("' + ip + '", "' + indicador + '", ' + str(hora) + ',
    ' + str(media) + ', ' + str(varianza) + ', ' + str(fecha) + '
    , ' + str(num_muestras) + ', ' + str(media_exp) + '
    , ' + str(varianza_exp) + ')')
275.         else:
276.             rellenar = generarAlerta(ip, False, indicador, fec
    ha, hora, media, varianza, media_anterior[0], varianza_anterior[0], nue
    vo_valor, media_exp, media_exp_anterior[0], varianza_exp, varianza_exp_
    anterior[0])
277.             if rellenar :
278.                 cursor.execute('INSERT INTO estadisticos
    VALUES (' + ip + '", "' + indicador + '", ' + str(hora) + ',
    ' + str(media) + ', ' + str(varianza) + ', ' + str(fecha) + '
    , ' + str(num_muestras) + ', ' + str(media_exp) + '
    , ' + str(varianza_exp) + ')')
279.             cnx.commit()
280.             cursor.close()
281.             cnx.close()
282.             return rellenar
283.
284.     def calcularIndicadores(ip, tabla, indicador, hora, fecha):
285.         """
286.
287.             Funcion que extrae los datos de las tablas auxiliares
    para
288.             llevar a cabo el calculo de los indicadores.
289.
290.             Parametros:
291.
292.                 ip --      Direccion IP correspondiente al
    indicador
293.
294.                 tabla --      Nombre de la tabla donde del
    indicador
295.
296.                 hora --      Hora en formato HH que corresponde
    al indicador
297.
298.                 fecha --      Fecha en formato YYYYMMDD que
    corresponde al indicador
299.

```

```

300.      """
301.          eliminar_registro = False
302.          cnx = mysql.connector.connect(user = usuario, password = p
    assword, host = host, database = database)
303.          cursor = cnx.cursor()
304.          cursor.execute('SELECT DISTINCT ' + indicador + ' FROM ' +
    tabla + ' WHERE ip=' + ip + "'")
305.          Indicadores=cursor.fetchall()
306.          for indicador_aux in Indicadores:
307.              cursor.execute('SELECT num_veces FROM ' + tabla
    + ' WHERE ip= "' + ip + '" AND hora=' + str(hora) + ' AND ' + indicador
    + ' = "' + indicador_aux[0] + '"' order by fecha desc limit
    '+ str(capacidad_ventana) +'')
308.              num_veces = cursor.fetchall()
309.              flat_list = [item for sublist in num_veces for ite
    m in sublist]
310.              cursor.execute('SELECT num_veces FROM ' + tabla
    + ' WHERE ip= "' + ip + '" AND hora=' + str(hora) + ' AND ' + indicador
    + ' = "' + indicador_aux[0] + '"' order by fecha desc ')
311.              num_veces_exp = cursor.fetchall()
312.              flat_list_exp = [item for sublist in num_veces_exp
    for item in sublist]
313.              flat_list_exp.reverse()
314.              ema = calcula_ema(alpha, flat_list_exp)
315.              varianza_exp = calcula_var_exp(alpha, np.var(flat_
    list))
316.              cursor.execute('SELECT num_veces FROM ' + tabla
    + ' WHERE ip= "' + ip + '" AND hora=' + str(hora) + ' AND ' + indicador
    + ' = "' + indicador_aux[0] + '"' order by fecha desc limit 1')
317.              nuevo_valor = cursor.fetchone()[0]
318.              eliminar_registro = rellenarBaseDatosEstadisticos(
    ip, indicador_aux[0], hora, np.mean(flat_list), np.var(flat_list), fech
    a, nuevo_valor, len(flat_list), ema, varianza_exp)
319.              if eliminar_registro == False and aprendizaje == 'F
    ALSE' and eliminar_anomalous == 'TRUE':
320.                  cursor.execute('DELETE FROM ' + tabla + '
    WHERE ' + indicador + '=' + "''" + indicador_aux[0] + "''" + AND
    hora=' + str(hora) + ' AND fecha=' + str(fecha) + ' AND ip=' + ip
    + "'")
321.
322.                  cursor.close()
323.                  cnx.close()
324.
325.      def crearCarpetaIp(carpeta, ipFiltrado, indicador):
326.          """
327.
328.          Funcion que crea dinamicamente una carpeta
329.          que contiene la informacion de un indicador
330.
331.          Parametros:
332.
333.              carpeta -- Nombre de la carpeta
334.
335.              ipFiltrado -- Direccion IP asociado al
    indicador
336.
337.              indicador -- Nombre del indicador
338.
339.          """
340.          if (os.path.isdir( carpeta ) == False):
341.              os.makedirs('Indicadores/IP/' + ipFiltrado + '/' +
    indicador + '/FicherosProcesados')

```

```

342.                 os.system('touch Indicadores/IP/' + ipFiltrado
343.                 + '/' + indicador + '/FicherosProcesados/Historico')
344.         def buscaLinea(palabra, fichero):
345.             """
346.
347.                 Funcion que busca una palabra en un fichero
348.                 y en caso de existir, devuelve el numero de linea
349.                 en el que se encuentra
350.
351.                 Parametros:
352.
353.                     palabra -- Palabra que se desea buscar
354.
355.                     fichero -- Nombre del fichero en el que se
356.                         busca la coincidencia
357.                     """
358.                     linea = 0
359.                     for line in fichero :
360.                         if 'Destino:\t' + palabra in line or 'Origen:\t' +
361.                             palabra in line:
362.                             return linea
363.                         linea += 1
364.         def tratamientoIP(ipFiltrado, fichero, ficheroIP):
365.             """
366.
367.                 Funcion auxiliar para la generacion de los ficheros
368.                 que contienen la informacion de los indicadores
369.
370.                 Parametros:
371.
372.                     ipFiltrado -- Direccion IP asociada al
373.                         indicador
374.                     carpeta -- Nombre de la carpeta
375.                     indicador -- Nombre del indicador
376.
377.                     """
378.                     file1 = open(fichero, 'r')
379.                     lines = file1.readlines()
380.                     linea = buscaLinea(ipFiltrado, lines)
381.                     if linea == None:
382.                         return 0
383.                     else:
384.                         fileIP = open(ficheroIP, 'wr')
385.                         fileIP.write(fecha + '\n')
386.                         fileIP.write(ipFiltrado + '\n')
387.                         linea += 1
388.                         bandera = 0
389.                         while bandera == 0:
390.                             if '---' not in lines[linea]:
391.                                 fileIP.write(lines[linea])
392.                                 linea += 1
393.                             else:
394.                                 bandera = 1
395.                         fileIP.close()
396.                         file1.close()
397.
398.         def mostrarAplicaciones(df, ipFiltrado):

```

```

400.      """
401.
402.          Aplicaciones
403.              Funcion que genera el fichero del indicador de
404.                  y llama a las diferentes funciones para la generacion
405.                      en la base de datos
406.
407.          Parametros:
408.              df -- DataFrame del indicador de
409.                  aplicaciones
410.              ipFiltrado -- Direccion IP asociada al
411.                  indicador
412.      """
413.      if(ipFiltrado == 'all'):
414.          Aplicaciones = df.L7_PROTO_NAME.unique()
415.          file = open('/home/anonymus/Escritorio/tfg/Indicad
416.          ores/aplicaciones','wr')
417.          file.write('#####\n')
418.          file.write('# Descripcion: Aplicaciones detectadas
419.          en el analisis #\n')
420.          file.write('# Fecha: '+ fecha
421.          +'#\n')
422.          file.write('#####\n\n')
423.          file.write('--> Total de
424.          aplicaciones:\t'+ str(len(Aplicaciones)) +'\n\n')
425.          i=1
426.          for app in Aplicaciones:
427.              file.write(str(i) + '\t' + app + '\n')
428.              i+=1
429.          file.write('\n--> Analisis de aplicaciones con IP
430.          origen:\n')
431.          for ip in df.IPV4_SRC_ADDR.unique():
432.              file.write('-----\n')
433.              file.write('IP Origen:\t' + ip + '\n')
434.              file.write(str(df[df['IPV4_SRC_ADDR'] == i
435.              p]['L7_PROTO_NAME'].value_counts()))
436.              file.write('-----\n')
437.              file.close()
438.              os.system('sed -i "/dtype:/d"
439.              Indicadores/aplicaciones')
440.              os.system('sed -i "/.\.\./d"
441.              Indicadores/aplicaciones')
442.          else:
443.              fichero = '/home/anonymus/Escritorio/tfg/Indicador
444.              es/aplicaciones'
445.              ficheroIpAplicaciones = '/home/anonymus/Escritorio
446.              /tfgr/Indicadores/IP/' + ipFiltrado + '/' + 'aplicaciones' + '/' + fecha
447.              crearCarpetaIp('Indicadores/IP/' + ipFiltrado
448.              + '/' + 'aplicaciones', ipFiltrado, 'aplicaciones')
449.              tratamientoIP(ipFiltrado,fichero,ficheroIpAplicaci
450.              ones)
451.              insertarBaseDatosIndicadores('/home/anonymus/Escri
452.              torio/tfg/Indicadores/IP/' + ipFiltrado
453.              + '/' + 'aplicaciones', 'aplicaciones', 'aplicacion')

```

```

440.
441.     def mostrarDataset(df, ipFiltrado):
442.         df.to_csv('/home/anonymus/Escritorio/tfg/Indicadores/datas
443. et', sep='\t', encoding='utf-8')
444.     def mostrarIpPuertosDestino(df, ipFiltrado):
445.         """
446.
447.             Funcion que genera el fichero del indicador de Puertos
448. destino
449.             y llama a las diferentes funciones para la generacion
450. del indicador
451.             en la base de datos
452.
453.             Parametros:
454.                 df -- DataFrame del indicador de
455. aplicaciones
456.
457.                 ipFiltrado -- Direccion IP asociada al
458. indicador
459.             """
460.             if (ipFiltrado == 'all'):
461.                 file = open('/home/anonymus/Escritorio/tfg/Indicad
ores/PuertosDestino', 'wr')
462.                 file.write('#####
#####\n')
463.                 file.write('# Descripcion: Puertos destinos
detectados en el analisis #\n')
464.                 file.write('# Fecha: '+ fecha
+'
#')
465.                 file.write('##> Puertos destino con el numero de
veces abiertos: \n\n')
466.                 file.write(str(df.L4_DST_PORT.value_counts()) + '\
n\n --> IP origen y puerto destino con el numero de veces abierto')
467.                 for ip in df.IPV4_SRC_ADDR.unique():
468.                     file.write('\n-----
--')
469.                     file.write('nIP Origen:\t' + ip + '\n')
470.                     file.write(str(df[df['IPV4_SRC_ADDR'] == i
p]['L4_DST_PORT'].value_counts()))
471.                     file.write('\n-----
--')
472.                     file.close()
473.                     os.system('sed -i "/dtype:/d"
 Indicadores/PuertosDestino')
474.                     os.system('sed -i "/\.\./d"
 Indicadores/PuertosDestino')
475.             else:
476.                 fichero = '/home/anonymus/Escritorio/tfg/Indicador
es/PuertosDestino'
477.                 ficheroIpPuertosOrigen = '/home/anonymus/Escritori
o/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'PuertosDestino' + '/' +
fecha
478.                 crearCarpetaIp('Indicadores/IP/' + ipFiltrado
+ '/' + 'PuertosDestino', ipFiltrado, 'PuertosDestino')
479.                 tratamientoIP(ipFiltrado, fichero, ficheroIpPuertosO
rigen)

```

```

479.                 insertarBaseDatosIndicadores('/home/anonymus/Escri
    torio/tfg/Indicadores/IP/' + ipFiltrado
    + '/' + 'PuertosDestino', 'puertos_destino', 'puerto')
480.
481.     def mostrarIpPuertosOrigen(df, ipFiltrado):
482.         """
483.
484.             Funcion que genera el fichero del indicador de puertos
        origen
485.             y llama a las diferentes funciones para la generacion
        del indicador
486.             en la base de datos
487.
488.             Parametros:
489.
490.                 df -- DataFrame del indicador de
        aplicaciones
491.
492.                 ipFiltrado -- Direccion IP asociada al
        indicador
493.
494.             """
495.             if(ipFiltrado == 'all'):
496.                 file = open('/home/anonymus/Escritorio/tfg/Indicad
    ores/PuertosOrigen', 'wr')
497.                 file.write('#####\n')
498.                 file.write('# Descripcion: Puertos origen
    detectados en el analisis #\n')
499.                 file.write('# Fecha: ' + fecha
    +
500.                 file.write('#####\n\n')
501.                 file.write('--> Puertos origen con el numero de
    veces abiertos: \n\n')
502.                 file.write(str(df.L4_SRC_PORT.value_counts()) + '\
    n\n --> IP origen y puerto origen con el numero de veces abierto')
503.                 for ip in df.IPV4_SRC_ADDR.unique():
504.                     file.write('\n-----
    --')
505.                     file.write('\nIP Origen:\t' + ip + '\n')
506.                     file.write(str(df[df['IPV4_SRC_ADDR'] == i
    p]['L4_SRC_PORT'].value_counts()))
507.                     file.write('\n-----
    --')
508.                     file.close()
509.                     os.system('sed -i "/dtype:/d"
    Indicadores/PuertosOrigen')
510.                     os.system('sed -i "/\.\./d"
    Indicadores/PuertosOrigen')
511.             else:
512.                 fichero = '/home/anonymus/Escritorio/tfg/Indicador
    es/PuertosOrigen'
513.                 ficheroIpPuertosOrigen = '/home/anonymus/Escritori
    o/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'PuertosOrigen' + '/' +
        fecha
514.                 crearCarpetaIp('Indicadores/IP/' + ipFiltrado
    + '/' + 'PuertosOrigen', ipFiltrado, 'PuertosOrigen')
515.                 tratamientoIP(ipFiltrado, fichero, ficheroIpPuertosO
    rigen)

```

```

516.                 insertarBaseDatosIndicadores('/home/anonymus/Escri
      torio/tfg/Indicadores/IP/' + ipFiltrado
      + '/' + 'PuertosOrigen', 'puertos_origen', 'puerto')
517.
518.     def mostrarIPoIPd(df, ipFiltrado):
519.         """
520.
521.             Funcion que genera el fichero del indicador de relacion
      ip
522.             y llama a las diferentes funciones para la generacion
      del indicador
523.             en la base de datos
524.
525.             Parametros:
526.
527.                 df -- DataFrame del indicador de
      aplicaciones
528.
529.                 ipFiltrado -- Direccion IP asociada al
      indicador
530.
531.             """
532.             if(ipFiltrado == 'all'):
533.                 file = open('/home/anonymus/Escritorio/tfg/Indicad
      ores/IPorigen-destino', 'w')
534.                 file.write('#####\n')
535.                 file.write('# Descripcion: Relacion de IP origen
      con IP destino \n')
536.                 file.write('# Fecha: ' + fecha
      + '\n')
537.                 file.write('#####\n')
538.                 for ip in df.IPV4_SRC_ADDR.unique():
539.                     file.write('\n-----
      --')
540.                     file.write('\nIP Origen:\t' + ip + '\n')
541.                     file.write(str(df[df['IPV4_SRC_ADDR'] == i
      p]['IPV4_DST_ADDR'].value_counts()))
542.                     file.write('\n-----
      --')
543.                     file.close()
544.                     os.system('sed -i "/dtype:/d"
      Indicadores/IPorigen-destino')
545.                     os.system('sed -i "/\.\./d" Indicadores/IPorigen-
      destino')
546.             else:
547.                 fichero = '/home/anonymus/Escritorio/tfg/Indicador
      es/IPorigen-destino'
548.                 ficheroIpPuertosOrigen = '/home/anonymus/Escritori
      o/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'IPorigen-destino' + '/' +
      fecha
549.                 crearCarpetaIp('Indicadores/IP/' + ipFiltrado
      + '/' + 'IPorigen-destino', ipFiltrado, 'IPorigen-destino')
550.                 tratamientoIP(ipFiltrado, fichero, ficheroIpPuertosO
      rigen)
551.                 insertarBaseDatosIndicadores('/home/anonymus/Escri
      torio/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'IPorigen-
      destino', 'relacion_ip', 'ip_destino')
552.
553.     def mostrarIPdIPo(df, ipFiltrado):
554.         """

```

```

555.                               Funcion que genera el fichero del indicador de relacion
556.           ip
557.           y llama a las diferentes funciones para la generacion
558.           del indicador
559.           en la base de datos
560.           Parametros:
561.           df --          DataFrame del indicador de
562.           aplicaciones
563.           ipFiltrado --      Direccion IP asociada al
564.           indicador
565.           """
566.           if(ipFiltrado == 'all'):
567.               file = open('/home/anonymus/Escritorio/tfg/Indicad
ores/IPdestino-origen','wr')
568.               file.write('#####
#####\n')
569.               file.write('# Descripcion: Relacion de IP destino
con IP origen    #\n')
570.               file.write('# Fecha: '+ fecha
+'#\n')
571.               file.write('#####
#####\n')
572.               file.write('#####
#####\n')
573.               for ip in df.IPV4_DST_ADDR.unique():
574.                   file.write('\n-----
--')
575.                   file.write('\nIP Destino:\t' + ip + '\n')
576.                   file.write(str(df[df['IPV4_DST_ADDR'] == i
p]['IPV4_SRC_ADDR'].value_counts()))
577.                   file.write('\n-----
--')
578.                   file.close()
579.                   os.system('sed -i "/dtype:/d"
Indicadores/IPdestino-origen')
580.                   os.system('sed -i "/.\./d" Indicadores/IPdestino-
origen')
581.           else:
582.               fichero = '/home/anonymus/Escritorio/tfg/Indicador
es/IPdestino-origen'
583.               ficheroIpPuertosOrigen = '/home/anonymus/Escritori
o/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'IPdestino-origen' + '/' +
fecha
584.               crearCarpetaIp('Indicadores/IP/' + ipFiltrado
+ '/' + 'IPdestino-origen', ipFiltrado, 'IPdestino-origen')
585.               tratamientoIP(ipFiltrado,fichero,ficheroIpPuertosO
rigen)
586.
587.       def mostrarDestinoICMP(df,ipFiltrado):
588.           """
589.
590.           Funcion que genera el fichero del indicador de mensajes
591.           icmp
592.           y llama a las diferentes funciones para la generacion
593.           del indicador
594.           en la base de datos
595.           Parametros:

```

```

596.                               df --      DataFrame del indicador de
   aplicaciones
597.                               ipFiltrado --      Direccion IP asociada al
   indicador
599.
600.                               """
601.                               if(ipFiltrado == 'all'):
602.                                   file = open('/home/anonymus/Escritorio/tfg/Indicad
ores/ICMP-Destino','wr')
603.                                   file.write('#####
#####\n')
604.                                   file.write('# Descripcion: ICMPs con IPs origen
detectados en el analisis #\n')
605.                                   file.write('# Fecha: '+ fecha
   +'\n')
606.                                   file.write('#####
#####\n\n')
607.                                   file.write('\n--> IP origen con ICMP:\n')
608.                                   file.write('\n-----\n')
609.                                   file.write(str(df['IPV4_SRC_ADDR'][df['L7_PROTO_NA
ME'] =='ICMP'].value_counts()))
610.                                   file.write('\n-----')
611.                                   file.close()
612.                               os.system('sed -i "dtype:/d" Indicadores/ICMP-
Destino')
613.                               os.system('sed -i "/.\./d" Indicadores/ICMP-
Destino')
614.                           else:
615.                               fichero = '/home/anonymus/Escritorio/tfg/Indicador
es/ICMP-Destino'
616.                               ficheroIpPuertosOrigen = '/home/anonymus/Escritori
o/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'ICMP-Destino' + '/' + fecha
617.                               crearCarpetaIp('Indicadores/IP/' + ipFiltrado
   + '/' + 'ICMP-Destino', ipFiltrado, 'ICMP-Destino')
618.                               tratamientoIP(ipFiltrado,fichero,ficheroIpPuertosO
rigen)
619.                               insertarBaseDatosIndicadores('/home/anonymus/Escri
torio/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'ICMP-
Destino', 'icmp_destino', 'ip_destino')
620.
621.     def mostrarNumIpDistintas(df, ipFiltrado):
622.         """
623.
624.             Funcion que genera el fichero del indicador de
Aplicaciones
625.             y llama a las diferentes funciones para la generacion de
numero
626.             IP distintas en la base de datos
627.
628.             Parametros:
629.
630.                 df --      DataFrame del indicador de
aplicaciones
631.
632.                 ipFiltrado --      Direccion IP asociada al
indicador
633.
634.                 """
635.                 if(ipFiltrado == 'all'):
636.                     file = open('/home/anonymus/Escritorio/tfg/Indicad
ores/Numero-IP', 'wr')

```

```

637.                                     file.write('#####\n')
638.                                     file.write('# Descripcion: Numero de IP distintas\n')
639.                                     file.write('# Fecha: '+ fecha + '\n')
640.                                     file.write('#####\n#####\n')
641.                                     for ip in df.IPV4_SRC_ADDR.unique():
642.                                         df2 = df[df.IPV4_SRC_ADDR == ip]
643.                                         IPs_destino = df2.IPV4_DST_ADDR.unique()
644.                                         num_IPs = len(IPs_destino)
645.                                         file.write('\n-----\n')
646.                                         file.write('IP Origen:\t' + ip + '\n')
647.                                         file.write('DISTINTAS' + ' ' + str(num_IPs))
648.                                         file.write('\n-----\n')
649.                                         file.write('-----\n')
650.                                     file.close()
651.                                     os.system('sed -i "/dtype:/d" Indicadores/Numero-IP')
652.                                     os.system('sed -i "/\.\./d" Indicadores/Numero-IP')
653.                                     else:
654.                                         fichero = '/home/anonymus/Escritorio/tfg/Indicadores/Numero-IP'
655.                                         ficheroNumIP = '/home/anonymus/Escritorio/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'Numero-IP' + '/' + fecha
656.                                         crearCarpetaIp('Indicadores/IP/' + ipFiltrado + '/' + 'Numero-IP', ipFiltrado, 'Numero-IP')
657.                                         tratamientoIP(ipFiltrado, fichero, ficheroNumIP)
658.                                         insertarBaseDatosIndicadores('/home/anonymus/Escritorio/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'Numero-IP', 'num_ip', 'ip_aux')
659.                                     df -- DataFrame del indicador de aplicaciones
660.                                     def mostrarNumApp(df, ipFiltrado):
661.                                         """
662.                                         Funcion que genera el fichero del indicador de
663.                                         Aplicaciones
664.                                         numero
665.                                         y llama a las diferentes funciones para la generacion de
666.                                         aplicaciones distintas en la base de datos
667.                                         Parametros:
668.                                         df -- DataFrame del indicador de
669.                                         aplicaciones
670.                                         ipFiltrado -- Direccion IP asociada al
671.                                         indicador
672.                                         """
673.                                         if(ipFiltrado == 'all'):
674.                                             file = open('/home/anonymus/Escritorio/tfg/Indicadores/Numero-APP', 'wr')
675.                                             file.write('#####\n')

```

```

678.                 file.write('# Descripcion: Numero de APPLICACIONES
679.                 distintas \n')
680.                 +
681.                 file.write('#####\n#####\n')
682.                 for ip in df.IPV4_SRC_ADDR.unique():
683.                     df2 = df[df.IPV4_SRC_ADDR == ip]
684.                     aplicaciones = df2.L7_PROTO_NAME.unique()
685.                     num_app = len(aplicaciones)
686.                     file.write('\n-----\n')
687.                     file.write('IP Origen:\t' + ip + '\n')
688.                     file.write('APPLICACIONES' + ' ' + str(
689.                         num_app))
690.                     file.write('\n-----\n')
691.                 file.close()
692.                 os.system('sed -i "/dtype:/d" Indicadores/Numero-
693. APP')
694.                 os.system('sed -i "/\.\./d" Indicadores/Numero-
695. APP')
696.             else:
697.                 fichero = '/home/anonymus/Escritorio/tfg/Indicador
698. es/Numero-APP'
699.                 ficheroNumApp = '/home/anonymus/Escritorio/tfg/Ind
700. icadores/IP/' + ipFiltrado + '/' + 'Numero-APP' + '/' + fecha
701.                 crearCarpetaIp('Indicadores/IP/' + ipFiltrado
702. + '/' + 'Numero-APP', ipFiltrado, 'Numero-APP')
703.                 tratamientoIP(ipFiltrado, fichero, ficheroNumApp)
704.                 insertarBaseDatosIndicadores('/home/anonymus/Escri
705. torio/tfg/Indicadores/IP/' + ipFiltrado + '/' + 'Numero-
706. APP', 'num_app', 'aplicacion')
707.             def limpiarbaseDatos():
708.                 """
709.                     Funcion que limpia la base de datos
710.                     """
711.                     cnx = mysql.connector.connect(user = usuario, password = p
712. assword, host = host, database = database)
713.                     cursor = cnx.cursor()
714.                     cursor.execute('DELETE FROM aplicaciones;')
715.                     cursor.execute('DELETE FROM puertos_origen;')
716.                     cursor.execute('DELETE FROM puertos_destino;')
717.                     cursor.execute('DELETE FROM relacion_ip;')
718.                     cursor.execute('DELETE FROM num_ip;')
719.                     cursor.execute('DELETE FROM num_app;')
720.                     cursor.execute('DELETE FROM icmp_destino;')
721.                     cursor.execute('DELETE FROM estadisticos;')
722.                     cursor.execute('DELETE FROM logs;')
723.                     cnx.commit()
724.                     cnx.close()
725.             ipFiltrado = ['all', '1.1.1.1']
726.             # Lectura del fichero en bruto de IPFIX
727.             ficheroFlujos = sys.argv[1]
728.             df = pd.read_csv(ficheroFlujos)

```

```

726.
727.      # Conversion de todas las IP internas a la IP 1.1.1.1
728.      df['IPV4_SRC_ADDR'][df['IPV4_SRC_ADDR'].isin(ipc.Network(network))]
729.      ] = '1.1.1.1'
730.
731.      # Conversion de todas las IP externas a la IP 2.2.2.2
732.      df['IPV4_DST_ADDR'][df['IPV4_DST_ADDR'].isin(ipc.Network(network))]
733.      ] = '2.2.2.2'
734.      df = df[(df.IPV4_SRC_ADDR == '1.1.1.1') | (df.IPV4_DST_ADDR == '2.
735.          2.2.2.2')]
736.
737.      # Obtencion de la fecha inicio de IPFIX
738.      vacio = True
739.      if len(df) > 0:
740.          inicio = df.FLOW_START_MILLISECONDS[0]
741.          vacio = False
742.      if fecha == 'AUTO':
743.          fecha = calculaFecha(inicio)
744.      dia = datetime.strptime(fecha, '%Y%m%d%H')

```