



Universidad de Valladolid

# Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Mención en Tecnologías de la Información

## Maqueta de Red para el Estudio de Vulnerabilidades

Autor:  
**D. Sergio Sanz Ferrero**





Universidad de Valladolid

# Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Mención en Tecnologías de la Información

## Maqueta de Red para el Estudio de Vulnerabilidades

Autor:

**D. Sergio Sanz Ferrero**

Tutor:

**Dr. Jesús M. Vegas Hernández**



## Agradecimientos

*Dedicado a*

*mis padres, por ser el apoyo más fuerte y haberme enseñado lo que es el esfuerzo.*

*mis amigos, por haberme acompañado y motivado durante estos años.*

*mis compañeros de estudio, por haber pasado tantas horas de biblioteca juntos.*

*Jesús, por la libertad y los consejos que me ha ofrecido para realizar este proyecto.*



## Resumen

Uno de los principales retos a los que se está enfrentando y se deberá enfrentar la informática en los próximos años es la ciberseguridad. En este trabajo se abordarán conceptos básicos sobre diseño de redes seguras, monitorización, vulnerabilidades y análisis forense.

En este proyecto se trata de mostrar de forma didáctica las principales vulnerabilidades que puede haber en una red. Se pretende ilustrar el comportamiento de la red, en qué se basan los principales ataques y qué medidas aplicar para evitar que tengan impacto en nuestros sistemas.

Para ello, se ha procedido a diseñar y construir una maqueta de red. La red consta de dos partes principales, una LAN y una DMZ para poder así simular el comportamiento que puede haber en cualquier red a una escala reducida.

Para mostrar el comportamiento y verlo de forma ilustrativa se ha construido un sistema de monitorización con las herramientas Elasticsearch, Logstash y Kibana. Esto nos permitirá ver a tiempo real el tráfico que pasa por la red. Se incluyen gráficos y datos de interés para poder analizar cada una de las situaciones.

Esto será accesible mediante una interfaz web, desde la cuál podremos ver el estado de la red, lanzar distintos ataques y analizar el tráfico que se genera en cada una de las situaciones. Se incluye una descripción de los ataques y de cuáles son las medidas para detectarlos y mitigarlos.

## Palabras clave

Ciberseguridad, Red, Firewall, Vulnerabilidad, Fuerza Bruta, Denegación de Servicio, Spoofing, Monitorización, Netflow, Elasticsearch, Logstash, Kibana.





## **Abstract**

One of the main challenges that is facing and will face IT in the coming years is cybersecurity. In this paper, basic concepts about safe network design, monitoring, vulnerabilities, and forensic analysis will be discussed.

This project is based on showing in an educative way the main vulnerabilities that can exist in a network. It is intended to illustrate the behavior of the network, what the main attacks are based on and what measures can be applied to prevent them from having impact in our systems.

To do this, we have proceeded to design and assemble a network model. The network consists of two main parts, a LAN and a DMZ, in order to simulate the behavior that can exist in any network at a reduced scale.

To show the behavior and see it in an illustrative way, a monitoring system has been built with the tools Elasticsearch, Logstash and Kibana. This will allow us to see in real time the traffic that goes through the network. Graphs and data are included to be able to analyze each one of the situations.

This will be accessible through a web interface, from which we can see the status of the network, launch various attacks and analyze the impact. It includes a description of the attacks and what are the measures to detect and prevent them.

## **Keywords**

Cibersecurity, Network, Firewall, Vulnerability, Brute-Force, Denial of Service, Spoofing, Monitorization, Netflow, Elasticsearch, Logstash, Kibana.



# Índice

|  |           |
|--|-----------|
| <b>1. Introducción</b>                     | <b>1</b>  |
| 1.1. Motivación . . . . .                  | 1         |
| 1.2. Objetivos . . . . .                   | 2         |
| 1.3. Alcance . . . . .                     | 3         |
| 1.4. Metodología y Planificación . . . . . | 4         |
| 1.4.1. Sprint 1 . . . . .                  | 8         |
| 1.4.2. Sprint 2 . . . . .                  | 10        |
| 1.4.3. Sprint 3 . . . . .                  | 13        |
| 1.4.4. Sprint 4 . . . . .                  | 15        |
| 1.5. Estructura de la Memoria . . . . .    | 17        |
| <br>                                       |           |
| <b>2. Diseño de la red</b>                 | <b>19</b> |
| 2.1. Requisitos de Red . . . . .           | 19        |
| 2.2. Diseño Lógico . . . . .               | 20        |
| 2.3. Diseño Físico . . . . .               | 25        |
| <br>                                       |           |
| <b>3. Monitorización</b>                   | <b>29</b> |
| 3.1. Netflow . . . . .                     | 29        |
| 3.2. Elasticsearch . . . . .               | 30        |
| 3.3. Logstash . . . . .                    | 32        |
| 3.4. Kibana . . . . .                      | 33        |

|  |           |
|--|-----------|
| <b>4. Vulnerabilidades</b>                   | <b>35</b> |
| 4.1. Escaneo de Vulnerabilidades . . . . .   | 35        |
| 4.2. Metasploit . . . . .                    | 40        |
| 4.3. Sniffing . . . . .                      | 41        |
| 4.3.1. Man In The Middle . . . . .           | 41        |
| 4.3.2. MAC Flooding . . . . .                | 41        |
| 4.3.3. ARP Spoofing . . . . .                | 42        |
| 4.4. Escaneo de Puertos . . . . .            | 44        |
| 4.5. Ataque de Fuerza Bruta . . . . .        | 48        |
| 4.5.1. Módulos PAM . . . . .                 | 51        |
| 4.5.2. Port Knocking . . . . .               | 52        |
| 4.6. Ataque Denegación de Servicio . . . . . | 54        |
| <b>5. Interfaz Web</b>                       | <b>60</b> |
| <b>6. Apéndice</b>                           | <b>65</b> |
| <b>7. Conclusiones y líneas futuras</b>      | <b>70</b> |
| <b>8. Bibliografía</b>                       | <b>72</b> |
| <b>9. Anexo I - Configuraciones</b>          | <b>74</b> |
| <b>10. Anexo II - Manual de Usuario</b>      | <b>81</b> |
| <b>11. Anexo III - Manual de Instalación</b> | <b>87</b> |
| <b>12. Anexo IV- Contenido del CD-ROM</b>    | <b>90</b> |

# 1. Introducción

## 1.1. Motivación

La ciberseguridad es la práctica de proteger cualquier sistema computacional de ciberataques y amenazas. En este trabajo se pretende explicar algunos de los principios básicos y un análisis de las vulnerabilidades más comunes. Para ello se ha diseñado y construido una maqueta de red situada en el laboratorio 2L016 de la Escuela Técnica Superior de Ingeniería Informática de la Universidad de Valladolid. Mediante esta maqueta se aprenderá a configurar los distintos dispositivos de red y a tener una visión desde el punto de vista de la administración de redes.

Se abordarán varios aspectos básicos y fundamentales para analizar las principales vulnerabilidades. Se utilizarán las herramientas más usadas para realizar escaneos de puertos, ataques de denegación de servicio o de fuerza bruta. Existen numerosas posibilidades para poder subsanar estos ataques. En cualquiera de los casos se ofrecerán diversas alternativas para poder mitigarlos. Dependiendo la situación o el momento en el que se encuentre será posible evitar el ataque por completo. En otras ocasiones, solamente tendremos la posibilidad de detectarlo y usar las medidas oportunas para que no consiga tener impacto en nuestros sistemas.

Se cubrirán las nociones básicas de pentesting y hacking ético. Se tendrá en cuenta los ataques del exterior de la red hacia el interior y los que ocurren en la propia red, que pueden ser llevados a cabo por cualquiera que tenga acceso físico o utilice una VPN para conectarse a los equipos de la LAN.

Para la detección y monitorización de estos ataques usaremos Elasticsearch como base de datos no relacional, Logstash para el procesamiento de los logs y Kibana para la visualización del tráfico. Estas herramientas nos permitirán analizar el tráfico que pasa por nuestro firewall a tiempo real. El tráfico será enviado mediante el protocolo Netflow, aprovechando que está desarrollado por Cisco y todo el equipo equipamiento que utilizaremos es propietario de la compañía.

Dado que este proyecto tiene una visión meramente didáctica, se desarrollará una aplicación web, a través de la cuál se podrá lanzar los distintos ataques expuestos contra la maqueta de red configurada. Se incluirá una explicación de los mismos con varias medidas de prevención y detección. Además, la herramienta Kibana nos permite la posibilidad de hacerla embebida dentro de nuestra página web. Por lo tanto, se podrá ver el tráfico que se genera al lanzar los ataques desde la propia interfaz desarrollada.

## 1.2. Objetivos

El principal objetivo de este TFG es mostrar, desde un punto de vista didáctico, los principales campos de la ciberseguridad. Se pretenda dar una visión del hacking ético y de la mitigación de amenazas. Por ello, además de conocer las principales herramientas de pentesting, tests de penetración con la intención de encontrar debilidades de seguridad ,que suelen usarse, nos centraremos en el tráfico de red que se genera y en el análisis del mismo. Aprovecharemos nuestra maqueta de red para poner a prueba los conceptos aprendidos.

Dada la gran amplitud del campo y la variedad de vulnerabilidades, nos centraremos en los principales protocolos de la capa de transporte, TCP y UDP. Aunque también analizaremos protocolos necesarios para los servicios de red como ARP e ICMP.

El objetivo general ya comentado, puede ser desglosado en otros objetivos más específicos e individuales. Aunque gracias a la metodología desarrollada, podremos ver posteriormente las tareas e historias de usuario que se seguirán, podremos tener una visión inicial de objetivos:

### Objetivos específicos

- Diseño y construcción de maqueta de red, con una DMZ y una LAN.
- Configuración de Switches y Routers.
- Implementación de VLANs.
- Configuración de firewalls mediante ACL.
- Explotar vulnerabilidades.
- Conocer las principales herramientas de hacking ético.
- Protección y mitigación frente ataques.
- Monitorización del tráfico de red.
- Centralización de logs.
- Impacto de un ataque.
- Construcción de interfaz web para ilustrar los ataques.

### 1.3. Alcance

Un usuario podrá utilizar la interfaz web y moverse entre las distintas pantallas para entender las principales vulnerabilidades. Se presentará una definición, descripción y explicación del lanzamiento del ataque. Además, también se explicarán algunas de las posibilidades para su mitigación o detección.

El usuario podrá lanzar los distintos tipos de ataques y moviéndose a la pestaña donde se incluye la monitorización en tiempo real del tráfico que pasa por el firewall, podrá analizar y entender el comportamiento. Solo se observará el tráfico de la red del último día. Esto nos permitirá que no haya tanto ruido y poder ver así el comportamiento de la red ante situaciones concretas, pero observando también, el tráfico que se genera exponiendo servicios al exterior. Por lo tanto, el usuario será capaz de identificar los distintos ataques analizando los parámetros de las diferentes situaciones.

Una gran parte de este proyecto se basa en la investigación de las vulnerabilidades y de las tecnologías usadas para la monitorización de infraestructuras. Por tanto, el alcance final, es poder comprender estos aspectos básicos y poder ilustrar el comportamiento de manera gráfica para facilitar su comprensión a un usuario final.

## 1.4. Metodología y Planificación

Teniendo en cuenta las características del proyecto se ha optado usar una metodología ágil. Usaremos Scrum[1] como metodología, ya que se basa en un desarrollo incremental y es una de las más usadas actualmente. No se busca una planificación y posterior ejecución completa, por eso hemos usado esta metodología. Queremos un desarrollo adaptativo, pues los objetivos del proyecto no están claramente establecidos previamente. Además, nos interesa estar en contacto directo con el cliente. A medida que se vaya creando funcionalidad nueva, será el propio cliente, que es el tutor, el que nos marque la funcionalidad nueva a desarrollar.

El proyecto se dividirá en varios sprints. En cada uno de ellos incrementaremos la funcionalidad del proyecto. Se irá abordando de manera priorizada y en cada uno de estos sprints podremos encontrar los siguientes eventos:

- Product Backlog: lista total de historias de usuario ordenados por prioridad.
- Sprint Backlog: incremento de funcionalidad, obtenido del Product Backlog, que se llevará a cabo durante la ejecución del sprint.
- Sprint Planning: reunión al principio de cada sprint en la que se fijan los objetivos del sprint. Se identifican las tareas y cada una es estimada.
- Sprint Review: realizada al final del sprint. Participará el product owner y se realizará una demo de la funcionalidad conseguida.
- Sprint Retrospective: reunión al final del sprint en la que se analiza qué fue bien y que se debe cambiar.

Además se incluyen una serie de roles que serán llevados a cabo:

- Scrum Master: es la persona encargada de apoyar al equipo de proyecto y mantener la consistencia del proyecto ágil.
- Product Owner: persona encargada de establecer un puente con los usuarios finales, los responsables del negocio y el equipo de desarrollo. En este proyecto el product owner es el tutor del TFG, por lo que él será el encargado de decidir que funcionalidad o que objetivos quiere.
- Equipo técnico: personas encargadas de crear y desarrollar el producto. El propio alumno asumirá este rol.

Realizaremos alguna variación respecto al enfoque principal de Scrum. No se realizará una reunión diaria (Daily Scrum), dadas las circunstancias del proyecto, no será necesario. Además, intentaremos adaptarlo a nuestras necesidades. Scrum inicialmente está pensado



para construcción software, aunque también es usado en proyectos en los que se incluyen tareas de infraestructura, como es nuestro caso.

Tampoco hay un equipo técnico, el equipo de desarrollo es una sola persona, que a su vez es el Scrum Master. No es lo más adecuado y no es lo que proponen las guías de desarrollo ágil, pero se adaptará en la medida de lo posible.

## Riesgos

En nuestro proyecto se realizarán tareas de construcción software, pero el hardware también es una parte importante. La mayor consecuencia de esto es el aumento del riesgo y la posible mala estimación de tiempo, ya que hay más factores que influyen en el proyecto.

Los riesgos más significativos de este proyecto tienen su origen en el hardware. Tanto en el fallo del mismo, como en la adquisición. Esto puede provocar grandes retrasos, sobre todo en las primeras fases del proyecto, que se intentarán evitar, en el caso de ocurrencia, usando dispositivos alternativos. Lo mismo ocurre con el software que se utilizará. Los riesgos de fallo en hardware y software pueden ocurrir durante todo el proyecto.

También se puede provocar un retraso en el aprendizaje de nuevas tecnologías en las que se intentará protegernos frente al riesgo utilizando tecnologías ya conocidas, o aquellas que sean más utilizadas y mejor documentadas.

## Roles

Durante todo el proyecto se mantendrán los mismos roles, ya que solo hay implicadas dos personas.

| Persona                     | Contacto                           | Rol                                     |
|-----------------------------|------------------------------------|---|
| Jesús María Vegas Hernández | jvegas@infor.uva.es                | Product Owner                           |
| Sergio Sanz Ferrero         | sergio.sanz.ferrero@alumnos.uva.es | Scrum Master/Miembro del equipo técnico |

**Tabla 1:** Roles

## Gestión del proyecto

Estableceremos una guía temporal para tener un control sobre el producto que se va creando así como de las actividades que se realizarán en cada período de tiempo.

Como la metodología elegida es Scrum nos basaremos en sprints. Nuestro proyecto estará definido por el número de sprints que tendrá el mismo.

El proyecto dará comienzo el 18 de febrero de 2019. Realizaremos 4 sprints a lo largo del proyecto basándonos en los objetivos principales. Cada sprint tendrá una duración de 4 semanas. Por lo tanto, la fecha de fin de proyecto estimada será el 15 de junio de 2019.

Cada sprint corresponde con un capítulo diferente de esta memoria, dividiéndose así el proyecto en 4 capítulos básicos: diseño de la red, monitorización, vulnerabilidades e interfaz web.

A la estimación inicial de tiempos de las tareas de cada sprint se añade una estimación de 12 horas por sprint para la redacción de esta memoria. La estimación inicial de los sprints será la siguiente:

- Sprint 1 (de la semana del 18 de febrero a la semana del 18 de marzo)
- Sprint 2 (de la semana del 18 de marzo a la semana del 17 de abril)
- Sprint 3 (de la semana del 17 de abril a la semana del 16 de mayo)
- Sprint 4 (de la semana del 16 de mayo a la semana del 15 de junio)

Para poder realizar el seguimiento del proyecto se utilizan herramientas de gestión de proyectos y centralización como Trello o Github.

El product backlog es el siguiente:

| Rank. | Descripción  |
|-------|--|
| 1     | Como product owner quiero tener instalado los dispositivos     |
| 2     | Como product owner quiero el diseño y análisis de la red       |
| 3     | Como product owner quiero la configuración de la red           |
| 4     | Como product owner quiero la creación de una base de datos     |
| 5     | Como product owner quiero centralizar los logs                 |
| 6     | Como product owner quiero procesar los logs                    |
| 7     | Como product owner quiero visualizar el tráfico del firewall   |
| 8     | Como product owner quiero un análisis de vulnerabilidades      |
| 9     | Como product owner quiero un análisis de mitigaciones          |
| 10    | Como product owner quiero la creación de una interfaz web      |
| 11    | Como product owner quiero lanzar ataques desde la interfaz web |

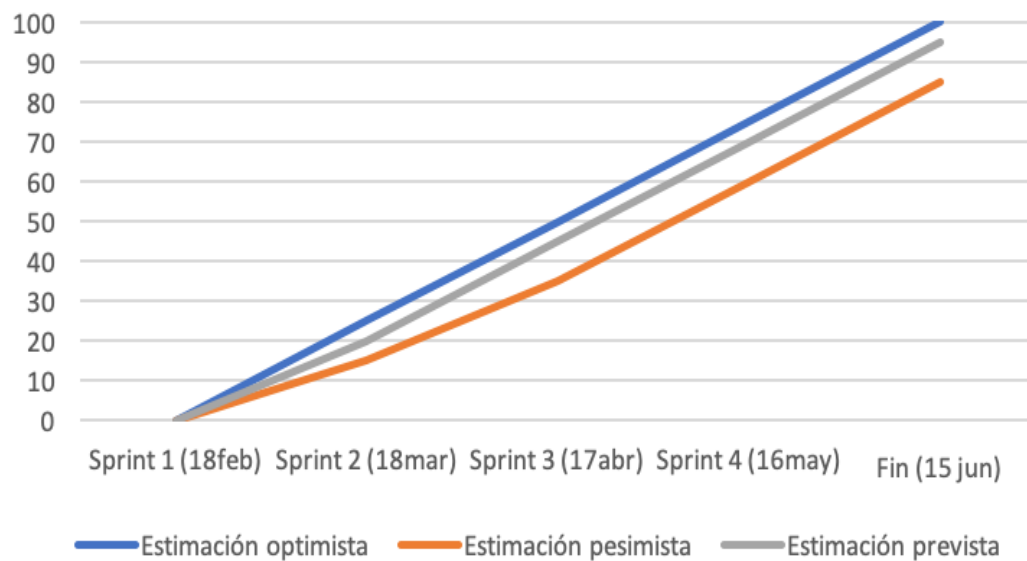
**Tabla 2:** Product backlog

El objetivo principal de la gestión de un proyecto es proveer un control sobre el mismo, establecer una guía temporal de actividades que se deben realizar como de recursos para

facilitar el trabajo en fases sucesivas y así evitar en la medida de lo posible retrasos que afecten a la duración total del proyecto.

Dado que la metodología elegida para realizar este proyecto ha sido SCRUM, las fases del proyecto quedarán definidas por el número de sprints que tendrá el mismo.

La estimación de tiempo general del trabajo queda definida en el Burnup que se muestra a continuación. Esta estimación ha podido sufrir variaciones a lo largo de la ejecución del proyecto.



**Figura 1:** Burnup global

Se han tomado tres estimaciones generales, teniendo en cuenta las circunstancias que puedan ocurrir a lo largo del proyecto.

- Estimación optimista: conseguir todos los objetivos a tiempo, sin retrasos que puedan surgir a lo largo del proyecto.
- Estimación prevista: surge un ligero retraso al comienzo del proyecto que es cuando más riesgos hay presente y provoca un pequeño retraso en la duración total del proyecto.
- Estimación pesimista: circunstancias adversas a la hora del correcto desarrollo del proyecto.

### 1.4.1. Sprint 1

Este sprint corresponde con la realización del capítulo 2 de esta memoria. Se abordarán todas las tareas e historias de usuario relacionadas con la construcción de la maqueta de red sobre la que se analizarán las vulnerabilidades.

|   |
|---|
| Nº 1 Sprint 1   |
| Nombre historia: Instalación de dispositivos  |
| Prioridad: alta   |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 16   |
| Descripción: Se analizarán los dispositivos de red que se utilizarán para comprender su funcionamiento y sus características principales. Se resetearán todos los dispositivos de red y se instalarán los sistemas operativos necesarios. |
| Criterios de aceptación: Poder utilizar cualquier dispositivo de red, teniendo accesibilidad a cualquiera de los dispositivos.  |

**Tabla 3:** Historia de usuario 1

|   |
|---|
| Nº 2 Sprint 1   |
| Nombre historia: Diseño y análisis de red   |
| Prioridad: alta   |
| Riesgo en desarrollo: bajo  |
| Horas estimadas: 20   |
| Descripción: Se diseñará la red en la que se puedan probar diferentes ataques y monitorizar intentando simular la estructura básica de cualquier red. |
| Criterios de aceptación: requisitos, diseño lógico y físico bien definido.  |

**Tabla 4:** Historia de usuario 2

|   |
|---|
| Nº 3 Sprint 1   |
| Nombre historia: Configuración de la red  |
| Prioridad: alta   |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 11   |
| Descripción: Se realizará la configuración y cableado de los dispositivos de red. |
| Criterios de aceptación: La red puede ser usada.                                  |

**Tabla 5:** Historia de usuario 3

| Historia de usuario         | Tareas  | Duración estimada | Duración real |
|-----------------------------|---|-------------------|---------------|
| Instalación de dispositivos | Adquisición de maqueta de red y preparación   | 2h                | 7h            |
|                             | Investigar sobre el hardware que se utilizará | 6h                | 7h            |
|                             | Reseteo de routers y switches                 | 3h                | 4h            |
|                             | Instalación SO servidor y configuración       | 2h                | 2h            |
|                             | Instalación SO Raspberry y configuraciones    | 2h                | 2h            |
|                             | Conexión CLI e instalación de drivers         | 1h                | 3h            |

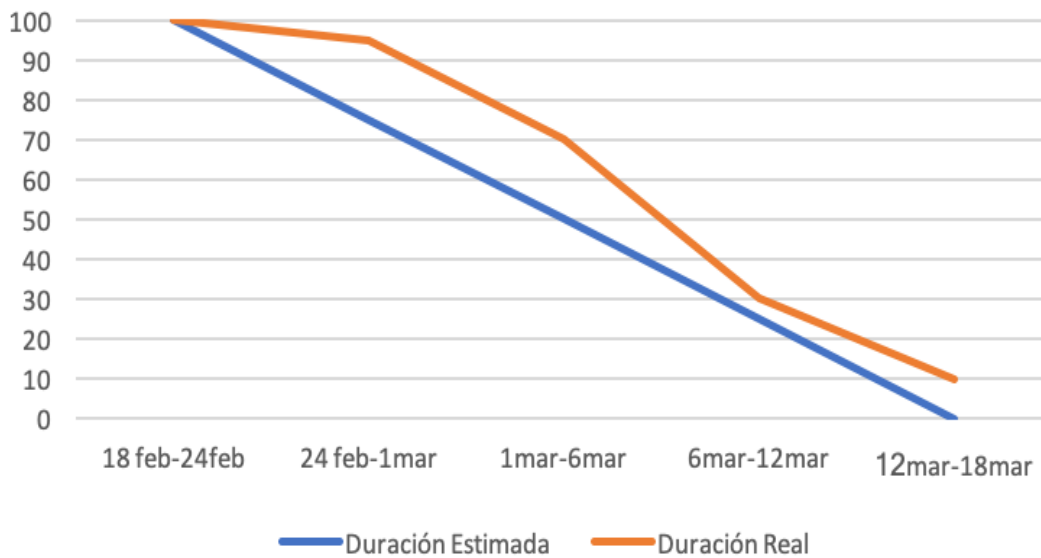
**Tabla 6:** Tareas historia de usuario 1

| Historia de usuario         | Tareas   | Duración estimada | Duración real |
|-----------------------------|--|-------------------|---------------|
| Diseño y análisis de la red | Análisis de redes seguras                          | 4h                | 4h            |
|                             | Diseño topología de red                            | 6h                | 7h            |
|                             | Búsqueda de tecnologías de monitorización de redes | 10h               | 10h           |

**Tabla 7:** Tareas historia de usuario 2

| Historia de usuario     | Tareas  | Duración estimada | Duración real |
|-------------------------|---|-------------------|---------------|
| Configuración de la red | Cableado de los dispositivos                                    | 2h                | 2h            |
|                         | Levantar interfaces y configurar IPs                            | 1h                | 1h            |
|                         | Configuración VLANs   | 1h                | 2h            |
|                         | Configuración protocolo de enrutamiento                         | 1h                | 2h            |
|                         | Configuración rutas estáticas                                   | 1h                | 2h            |
|                         | Configuración NAT y salida a Internet                           | 2h                | 3h            |
|                         | Levantar servicios en el servidor                               | 1h                | 1h            |
|                         | Configuración listas de control de acceso para seguridad básica | 2h                | 2h            |

**Tabla 8:** Tareas historia de usuario 3



**Figura 2:** Burndown Sprint 1

Al tratarse de un sprint relacionado con tareas de infraestructura y no de construcción de software, la estimación de tiempo se convierte en un poco más compleja. Como podemos observar al comienzo del sprint todo empezó bastante más lento de lo estimado. La adquisición de dispositivos retrasó la estimación inicial del proyecto al depender de terceros. El resto de tareas también implicaron un pequeño retraso debido a restricciones de hardware, configuraciones por parte del administrador de la escuela y a problemas con el funcionamiento conjunto de la red.

#### 1.4.2. Sprint 2

Este sprint corresponde con el capítulo 3 de esta memoria. Se abordarán todos los aspectos relacionados con la monitorización del tráfico que pasa por nuestro firewall externo.

|   |
|---|
| Nº 4 Sprint 2   |
| Nombre historia: Creación base de datos   |
| Prioridad: media  |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 16   |
| Descripción: Elegir la tecnología más adecuada para el almacenamiento de logs y crear base de datos para su centralización. |
| Criterios de aceptación: Creación de base de datos y realización de consultas.  |

**Tabla 9:** Historia de usuario 4

|   |
|---|
| Nº 5 Sprint 2   |
| Nombre historia: Centralización de logs                                 |
| Prioridad: media  |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 8  |
| Descripción: Envío de logs al servidor y configuración del envío.       |
| Criterios de aceptación: Recepción de logs en el servidor a tiempo real |

**Tabla 10:** Historia de usuario 5

|   |
|---|
| Nº 6 Sprint 2   |
| Nombre historia: Procesamiento de logs  |
| Prioridad: media  |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 17   |
| Descripción: Recogida de logs, parseo y almacenamiento en base de datos                               |
| Criterios de aceptación: Consulta de logs, generados por el firewall, en la base de datos por campos. |

**Tabla 11:** Historia de usuario 6

|   |
|---|
| Nº 7 Sprint 2   |
| Nombre historia: Visualización tráfico firewall                                     |
| Prioridad: baja   |
| Riesgo en desarrollo: medio   |
| Horas estimadas: 12   |
| Descripción: visualización gráfica del tráfico de red                               |
| Criterios de aceptación: poder consultar y filtrar el tráfico de una manera visual. |

**Tabla 12:** Historia de usuario 7

| Historia de usuario    | Tareas  | Duración estimada | Duración real |
|------------------------|---|-------------------|---------------|
| Creación base de datos | Investigación bases de datos más usadas para centralización de logs | 2h                | 3h            |
|                        | Comprensión de Elasticsearch  | 3h                | 4h            |
|                        | Instalación y configuración de Elasticsearch                        | 2h                | 4h            |
|                        | Creación de base de datos   | 2h                | 2h            |
|                        | Almacenar logs en la base de datos                                  | 2h                | 2h            |
|                        | Realizar consultas a la base de datos                               | 2h                | 2h            |
|                        | Eliminación de contenido de la base de datos                        | 2h                | 2h            |
|                        | Automatizar eliminación de logs diariamente                         | 1h                | 1h            |

**Tabla 13:** Tareas historia de usuario 4

| Historia de usuario    | Tareas  | Duración estimada | Duración real |
|------------------------|---|-------------------|---------------|
| Centralización de logs | Comprensión de Netflow para el envío de logs        | 4h                | 4h            |
|                        | Configuración router para enviar logs               | 2h                | 2h            |
|                        | Establecer parámetros de configuración de Netflow   | 1h                | 2h            |
|                        | Comprobar recepción en el servidor mediante sniffer | 1h                | 1h            |

**Tabla 14:** Tareas historia de usuario 5

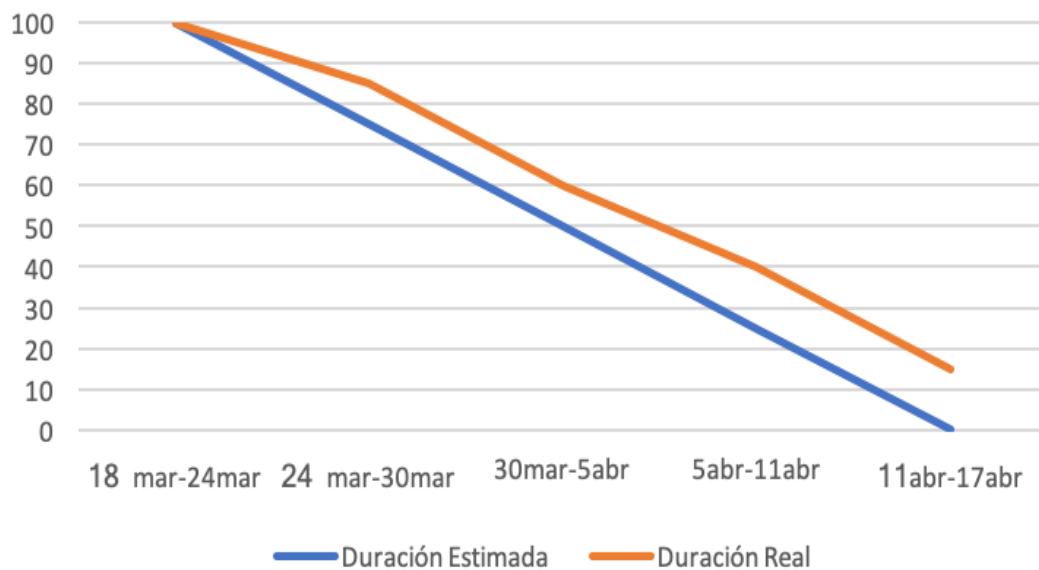
| Historia de usuario   | Tareas   | Duración estimada | Duración real |
|-----------------------|--|-------------------|---------------|
| Procesamiento de logs | Investigar herramientas recepción y parseo de logs | 4h                | 5h            |
|                       | Instalación y configuración de Logstash            | 5h                | 6h            |
|                       | Recepción logs en el servidor                      | 2h                | 3h            |
|                       | Parseo de los logs                                 | 2h                | 2h            |
|                       | Envío de logs a la base de datos                   | 2h                | 2h            |
|                       | Creación de índices                                | 2h                | 2h            |

**Tabla 15:** Tareas historia de usuario 6

| Historia de usuario         | Tareas  | Duración estimada | Duración real |
|-----------------------------|---|-------------------|---------------|
| Visualizar tráfico firewall | Investigar herramientas que muestren el tráfico de red gráficamente | 4h                | 4h            |
|                             | Instalación y configuración Kibana                                  | 2h                | 3h            |
|                             | Diseño de gráficos más relevantes                                   | 2h                | 2h            |
|                             | Recuperación documentos base de datos                               | 2h                | 2h            |
|                             | Creación de dashboards  | 2h                | 3h            |

**Tabla 16:** Tareas historia de usuario 7





**Figura 3:** Burndown Sprint 2

Tras la correcta construcción de la red, que produjo un pequeño retraso en el proyecto, se finalizó las tareas del sprint 1 y se empezó a realizar el sprint 2. En este sprint se produce un retraso constante ya que la investigación y el aprendizaje de las tecnologías requerían un esfuerzo inicial, el cuál se produjo durante todo el sprint.

### 1.4.3. Sprint 3

Este sprint corresponde con el capítulo 4 de esta memoria. Se abordarán todos los aspectos relacionados con la investigación y documentación de las principales vulnerabilidades a nivel de red.

|  |
|--|
| Nº 8 Sprint 3  |
| Nombre historia: Análisis vulnerabilidades                     |
| Prioridad: alta  |
| Riesgo en desarrollo: bajo                                     |
| Horas estimadas: 29  |
| Descripción: investigación de ataques y lanzamiento            |
| Criterios de aceptación: descripción y lanzamiento de ataques. |

**Tabla 17:** Historia de usuario 8

|  |
|--|
| Nº 9 Sprint 3  |
| Nombre historia: Análisis de mitigaciones  |
| Prioridad: alta  |
| Riesgo en desarrollo: bajo   |
| Horas estimadas: 21  |
| Descripción: Investigación de mitigaciones y aplicación de ellas.                                  |
| Criterios de aceptación: Descripción mitigaciones y aplicación de las mismas frente a los ataques. |

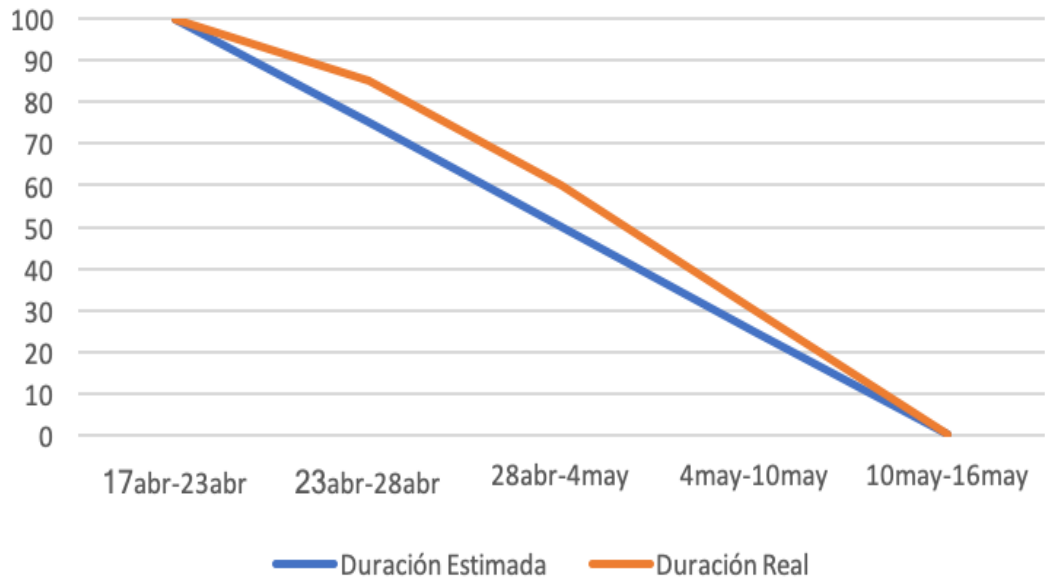
**Tabla 18:** Historia de usuario 9

| Historia de usuario       | Tareas  | Duración estimada | Duración real |
|---------------------------|---|-------------------|---------------|
| Análisis vulnerabilidades | Investigación denegación de servicio            | 3h                | 4h            |
|                           | Investigación fuerza bruta                      | 3h                | 3h            |
|                           | Investigación tests de vulnerabilidades         | 3h                | 4h            |
|                           | Investigación Sniffing                          | 3h                | 3h            |
|                           | Investigación escaneo de puertos                | 3h                | 3h            |
|                           | Investigación Man In The Middle                 | 3h                | 3h            |
|                           | Instalación herramientas lanzamiento de ataques | 3h                | 3h            |
|                           | Ejecución ataques                               | 4h                | 4h            |
|                           | Análisis del tráfico                            | 4h                | 4h            |

**Tabla 19:** Tareas historia de usuario 8

| Historia de usuario      | Tareas   | Duración estimada | Duración real |
|--------------------------|--|-------------------|---------------|
| Análisis de mitigaciones | Investigación Port Knocking  | 3h                | 3h            |
|                          | Investigación Módulos PAM  | 3h                | 3h            |
|                          | Investigación mitigación mediante archivos de configuración servidor | 3h                | 3h            |
|                          | Investigación mitigación en dispositivos Cisco                       | 3h                | 3h            |
|                          | Creación listas de control de acceso en el firewall                  | 1h                | 1h            |
|                          | Ejecución mitigaciones   | 4h                | 5h            |
|                          | Detección de vulnerabilidades mediante análisis del tráfico          | 4h                | 4h            |

**Tabla 20:** Tareas historia de usuario 9



**Figura 4:** Burndown Sprint 3

Durante este sprint se pudo recuperar el retraso que se llevaba. La estimación del aprendizaje de las vulnerabilidades y mitigaciones ha sido bastante realista y se pudo finalizar en la fecha estimada.

#### 1.4.4. Sprint 4

Este sprint corresponde con el capítulo 5 de esta memoria. Se abordarán todos los aspectos relacionados con la construcción de la interfaz web en la que se explicarán las vulnerabilidades, se podrá lanzar ataques y ver el estado que produce en la red.

|  |
|--|
| Nº 10 Sprint 4   |
| Nombre historia: Creación interfaz web   |
| Prioridad: alta  |
| Riesgo en desarrollo: bajo   |
| Horas estimadas: 18  |
| Descripción: Creación de vistas e inclusión de descripción de ataques y mitigaciones.                          |
| Criterios de aceptación: Un alumno debe ser capaz de entender las vulnerabilidades mediante las descripciones. |

**Tabla 21:** Historia de usuario 10

|  |
|--|
| Nº 11 Sprint 4   |
| Nombre historia: Lanzamiento de ataques desde web  |
| Prioridad: alta  |
| Riesgo en desarrollo: medio  |
| Horas estimadas: 17  |
| Descripción: Configuración para lanzar ataques desde la interfaz web contra la maqueta de red. |
| Criterios de aceptación: Visualización del estado de la red tras lanzamiento de ataque.        |

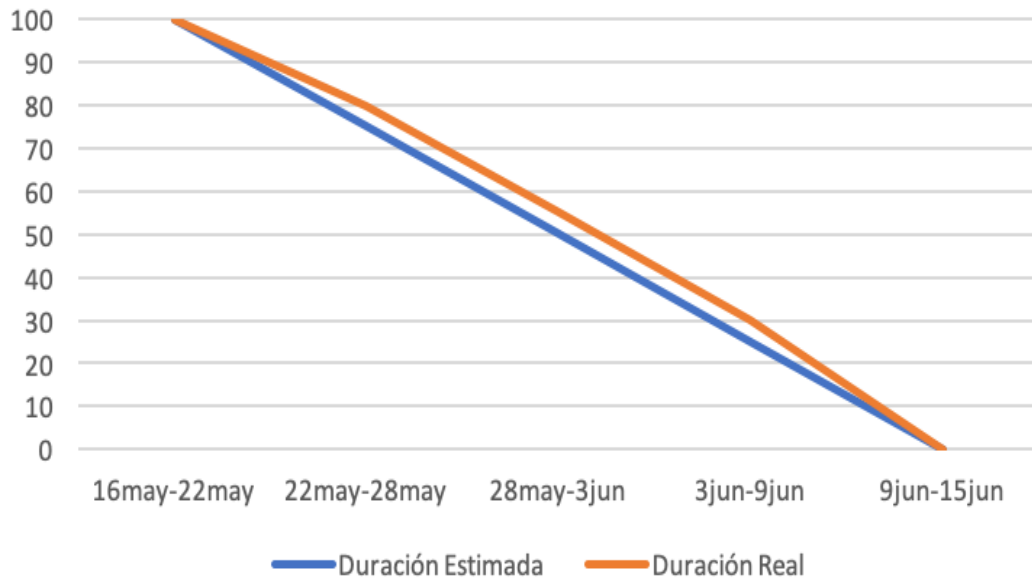
**Tabla 22:** Historia de usuario 11

| Historia de usuario   | Tareas   | Duración estimada | Duración real |
|-----------------------|--|-------------------|---------------|
| Creación interfaz web | Diseño plataforma web                              | 3h                | 3h            |
|                       | Creación de vistas para cada ataque                | 8h                | 8h            |
|                       | Incorporación descripciones ataques y mitigaciones | 10h               | 10h           |
|                       | Creación vista del estado de la red                | 1h                | 1h            |

**Tabla 23:** Tareas historia de usuario 10

| Historia de usuario    | Tareas   | Duración estimada | Duración real |
|------------------------|--|-------------------|---------------|
| Lanzamiento de ataques | Instalación servidor web                                 | 1h                | 1h            |
|                        | Comunicación con atacante                                | 2h                | 3h            |
|                        | Programación scripts que lanzan ataques                  | 3h                | 3h            |
|                        | Programar lanzamiento de scripts remotos                 | 3h                | 3h            |
|                        | Configuración proxy con autenticación                    | 2h                | 2h            |
|                        | Uso de calve pública y privada entre servidor y atacante | 1h                | 1h            |
|                        | Automatización del borrado de logs diario                | 1h                | 1h            |
|                        | Subir aplicación web al servidor                         | 1h                | 1h            |
| Pruebas funcionamiento | 2h   | 2h                |               |

**Tabla 24:** Tareas historia de usuario 11



**Figura 5:** Burndown Sprint 4

Finalmente, en el último Sprint, se realizó una estimación bastante realista ya que no se produjeron problemas y se conocía la tecnología utilizada. De esta manera, el proyecto se pudo acabar en la fecha estimada.

## 1.5. Estructura de la Memoria

Como se ha comentado anteriormente, se han realizado cuatro Sprints, cada uno corresponde con uno de los capítulos en los que se ha dividido el cuerpo del trabajo. Estos son los capítulos principales, en orden cronológico, que se han desarrollado:

- Capítulo 2: Diseño de red. Como primer paso para llevar este proyecto es necesario la construcción de la maqueta de red. En este capítulo se diseña la red a nivel físico y lógico atendiendo los requisitos establecidos. Una vez diseñado se pudo proceder a realizar las configuraciones físicas para tener la red operativa y poder continuar con el resto del proyecto.
- Capítulo 3: Monitorización. Este capítulo contiene una explicación de las herramientas de monitorización utilizadas. Se explicará e introducirá el uso de las tecnologías Elasticsearch, Logstash y Kibana. Además, se documenta el uso de Netflow para el envío de logs al servidor.
- Capítulo 4: Vulnerabilidades. Contiene una descripción de las vulnerabilidades principales que se pueden presentar en cualquier red. Nos centraremos tanto en el lanzamiento de los ataques como en medidas de mitigación y de detección a través de las herramientas de monitorización previamente configuradas.

- Capítulo 5: Interfaz Web. Contiene el análisis y diseño de la aplicación web. Esta aplicación web se desarrolla para ilustrar de forma didáctica el trabajo realizado.

Además de estos capítulos principales, se han llevado a cabo los siguientes capítulos que facilitan la comprensión y añaden datos de interés al proyecto.

- Apéndice: Contiene un caso real de un ataque a nuestra red.
- Configuraciones: Se añaden las configuraciones de los dispositivos, así como los Scripts utilizados para el lanzamiento de los ataques.
- Conclusiones y líneas futuras.
- Bibliografía utilizada.
- Manual de usuario: incluye descripción de uso de la aplicación web desarrollada y de aspectos básicos del uso de Kibana para filtrar el tráfico.
- Manual de instalación: Incluye los pasos necesarios para poder montar la infraestructura completa de este proyecto.
- Contenido del CD-ROM.

## 2. Diseño de la red

### 2.1. Requisitos de Red

La red deberá simular un entorno básico de laboratorio en el que poder probar los distintos ataques. Es por ello que simularemos los componentes básicos de una red. Tiene que tener cierta flexibilidad para poder adaptarse a nuestras necesidades del proyecto. Para ello incorporamos los siguientes requisitos.

| Número | Descripción del requisito  | Crítico |
|--------|--|---------|
| 1      | La red permitirá la conexión entre usuarios  | S       |
| 2      | La red debe ser flexible y permitir el cambio de configuraciones   | S       |
| 3      | La red tendrá una DMZ con un servidor  | S       |
| 4      | La red permitirá la conexión de los usuarios internos con el servidor  | S       |
| 5      | La red solo permitirá conexiones SSH desde la subred de laboratorio 157.88.123.0/24 con el servidor para evitar ataques externos | S       |
| 6      | La red no permitirá la conexión de usuarios externos con usuarios internos   | S       |
| 7      | Los datos de los usuarios internos deben estar protegidos de intrusos  | S       |
| 8      | La red solo permitirá el acceso desde el exterior a ciertos servicios  | S       |
| 9      | La red permitirá acceso web al servidor  | S       |
| 10     | La red deberá ser escalable a más equipos de usuarios internos o más servidores  | N       |
| 11     | La red estará disponible el mayor tiempo posible sin necesidad de alta disponibilidad, ni redundancia                            | N       |

**Tabla 25:** Requisitos de la red

### Grupos de usuarios

Tres roles principales se distinguirán en el uso de la red:

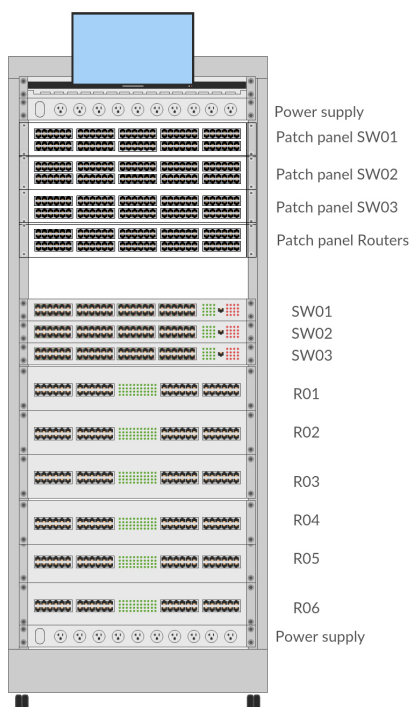
- Usuarios externos: solo podrán usar los servicios de acceso mediante HTTP a los puertos 80 y 8080, que ofrece el servidor de la DMZ. Realizarán ataques desde fuera de la red. Son ataques en los que la IP pública 157.88.123.102 estará involucrada. Consideraremos usuarios externos a los usuarios de la subred 157.88.123.0/24 que

también podrán usar el servicio SSH. De esta forma podremos simular los ataques pero evitaremos ataques externos por parte de terceros.

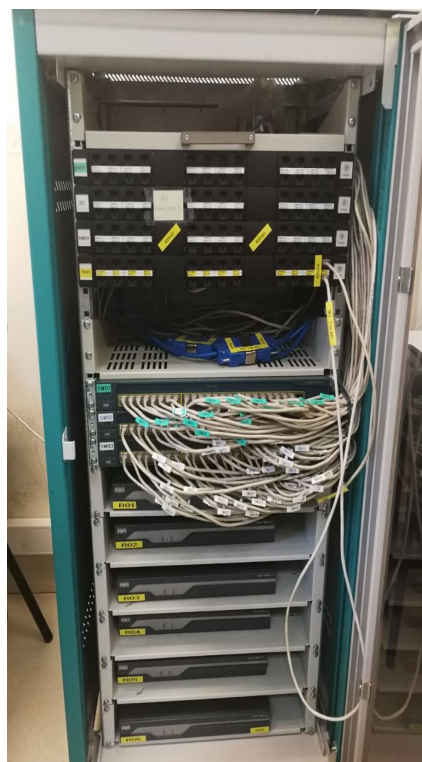
- Usuarios internos: representará a los usuarios de la red que están conectados al switch interior. Podrán realizar las tareas típicas, conectarse a Internet y conectarse al servidor de la DMZ. También se simularán ataques en los que se atacará a otros usuarios internos.
- Administrador: tendrá la capacidad para cambiar configuraciones de red y monitorizar el tráfico. Tendrá acceso físico y virtual a toda la red.

## 2.2. Diseño Lógico

El rack del laboratorio había sido montado físicamente anteriormente con el siguiente etiquetado y la siguiente disposición:



**Figura 6:** Etiquetado maqueta



**Figura 7:** Maqueta de red

Para facilitar la comprensión se hace un renombrado de los dispositivos siguiendo el siguiente esquema:

Tipo de elemento puede ser:



- PC => Ordenador
- R => Router
- SW => Switch
- SVR => Servidor

Lugar en el que se encuentra:

- DMZ => Zona desmilitarizada
- INT => Límite con la LAN
- EXT => Límite con el exterior

Característica (opcional):

- FWL => Firewall
- WEB => Servidor web
- MONIT => Servidor monitorización

El esquema de nombres tendrá el siguiente formato:

"Elemento" + "Número identificativo" + " - " + "Lugar" + " - " + "Característica"

Por lo que la lista con el nombre de los dispositivos es la siguiente:

- R01-EXT-FWL: Router que separa a la red del exterior. Es el último router antes de que se de salida al exterior, por lo tanto se incluyen ACL que controlan las entradas y salidas de la red.
- R02-INT-FWL: Router que separa la LAN del resto de la red. Todo el tráfico del interior de la red está obligado a pasar por él, se incluyen ACL para poder filtrar el tráfico que nos interese, aislándolo del resto.
- SW01-EXT: Switch de conexiones en el exterior.
- SW03-INT-DMZ: Switch central que se encargará de la conmutación de paquetes entre las VLANs.

- SVR01-DMZ-WEB-MONIT: Servidor que contendrá un website y donde estarán instaladas las herramientas de monitorización (Elasticsearch, Logstash y Kibana). Será accesible mediante SSH desde la subred de laboratorio 157.88.123.0/24 para evitar ataques externos. Recibirá los logs mediante Netflow desde R01-EXT-FWL.
- PC02-INT: PC del interior desde el que se realizarán ataques. No se automatizará ningún ataque, pero se aprovechará para documentar ataques desde el interior.
- PC03-EXT: PC del exterior desde el que se realizarán ataques. Contendrá scripts para lanzar ataques contra la IP pública de nuestra red.

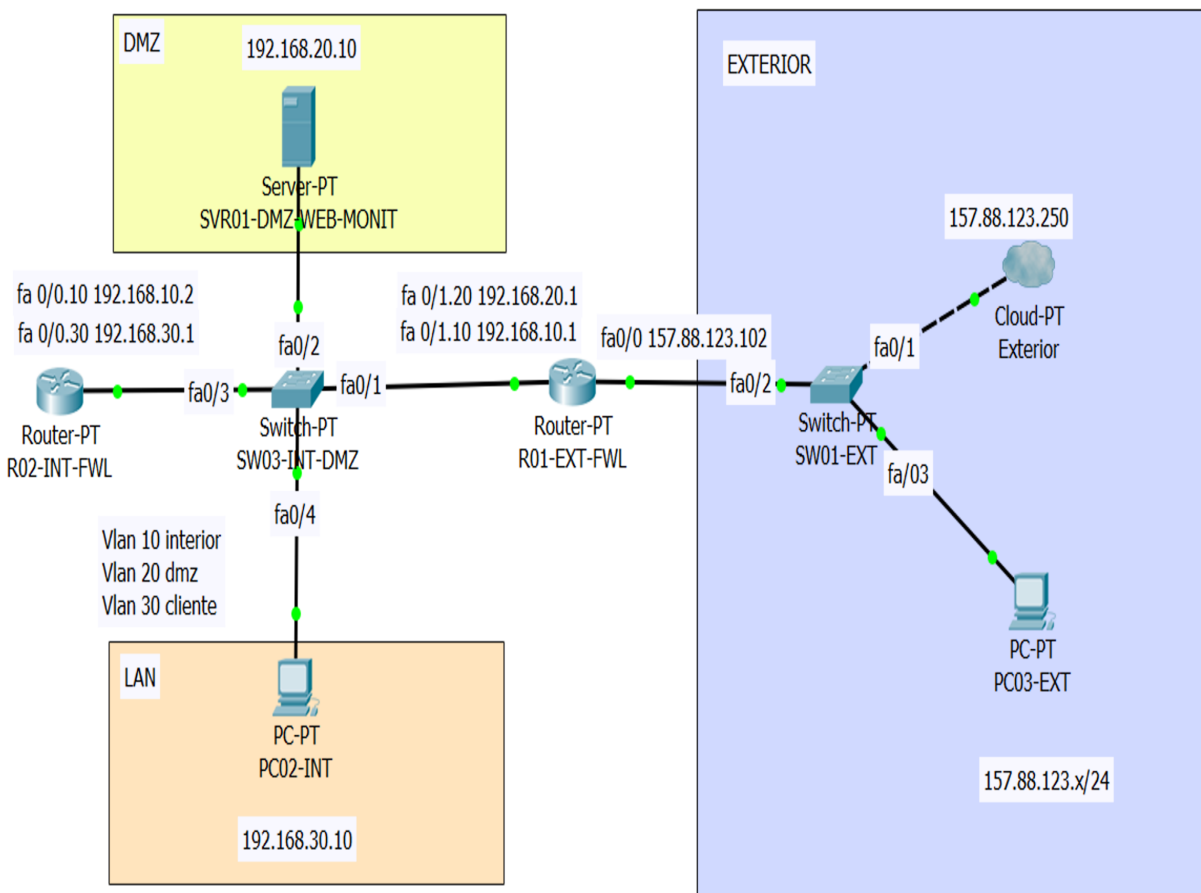
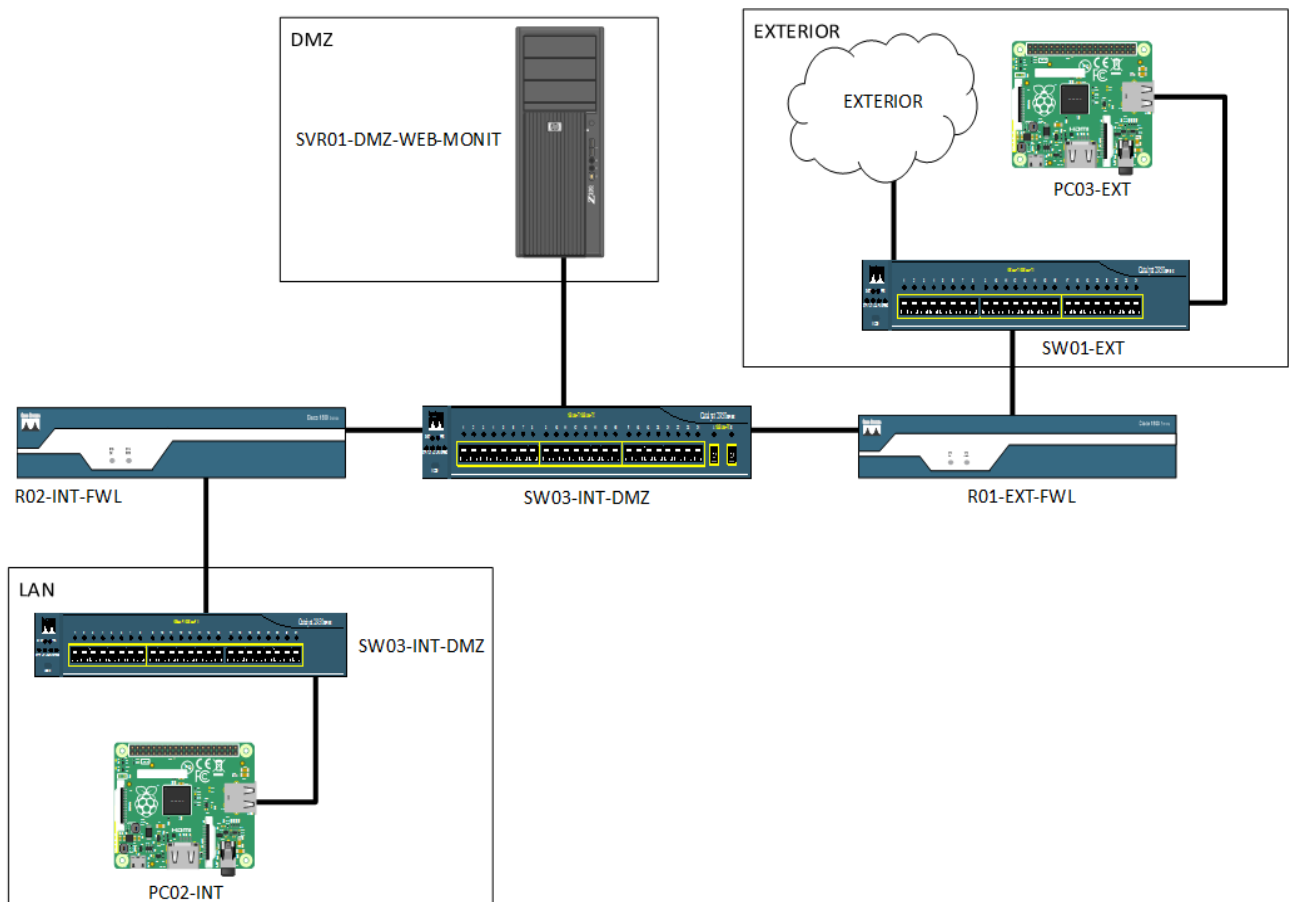


Figura 8: Conexiones de red



**Figura 9:** Diseño lógico de red

## Protocolos de red

No se utilizará DHCP para la asignación de direcciones IP. Evidentemente el servidor de la DMZ dispondrá de una dirección IP estática. También se ha optado por esa decisión para las máquinas de los usuarios del interior de la red, ya que en todo momento nos interesa tenerlos identificados para la correcta comprensión de los ataques.

El protocolo de enrutamiento que se usará es OSPF. La razón principal por la que se ha escogido es la posible escalabilidad. Es un protocolo que ofrece rápida convergencia y se adapta muy bien a escalabilidad en redes mayores. Esta escalabilidad tendrá menos limitaciones que si usáramos, por ejemplo, EIGRP, ya que es un estándar abierto que admite dispositivos de todos los fabricantes. EIGRP solo soporta dispositivos de Cisco, que en nuestro caso lo podríamos utilizar, pero si deseáramos incorporar un nuevo dispositivo, supondría un mayor esfuerzo. La desventaja principal es el alto consumo de recursos, pero es algo que no nos preocupa, debido a que la red no tendrá demasiado tráfico.

Para la administración y configuración remota se usará SNMP. Mediante este protocolo podremos ver el estado de la red y cambiar configuraciones. Es posible que nos interese

cambiar los startup-config y running-config remotamente en un futuro.

Aunque en próximos capítulos se explicará más en profundidad, se utilizará Netflow para la recolección de tráfico IP y la centralización de los logs en el servidor.

La ISO con la que cuentan actualmente los dispositivos de red está desactualizada y no admite el protocolo SSH. Si que admite Telnet, pero al ser un protocolo que no está cifrado, evitaremos su uso en la medida de lo posible. Por tanto, las configuraciones de los dispositivos se llevarán a cabo conectando el dispositivo a una máquina mediante un cable de consola y utilizando la CLI (Interfaz de línea de comandos).

## VLANS

Se configurarán redes lógicas independientes dentro de la red, es decir en el SW03-DMZ:

- VLAN 10: VLAN para el intercambio de datos con los usuarios internos. Obliga, mediante una ruta estática, a que todo el tráfico que vaya a la subred 192.168.30.0/24 pase a través del R02-INT-FWL, para así poder filtrar el tráfico. Se utiliza simplemente para tareas de conexión.
- VLAN 20: correspondiente a la DMZ, los dispositivos y servidores pertenecerán a la subred 192.168.20.0/24.
- VLAN 30: correspondiente a las máquinas de los usuarios internos, pertenecerán a la subred 192.168.30.0/24.

## Aplicaciones en red

El servidor de la DMZ será el que contenga todas las aplicaciones disponibles en red. Será accesible por usuarios externos, internos y por el administrador. Este servidor nos servirá para dos aspectos fundamentales:

En primer lugar, para tener servicios activos y como servidor web. Simularemos la existencia de servicios activos para recrear la realidad de cualquier red. Esto nos permitirá la posibilidad de realizar escaneos u otro tipo de ataques desde el exterior. Además, aprovecharemos al servidor para alojar la aplicación web que ilustrará el trabajo realizado.

En segundo lugar, este servidor también será un centralizador de logs. Recibirá los logs del firewall externo y mediante las herramientas Elasticsearch, Logstash y Kibana podremos hacer una visualización a tiempo real del estado de la red. Estas tres herramientas estarán instaladas en el servidor.

Será usado por todos los usuarios, tanto de la red interna, como de la red externa. Bien sea para monitorizar, o para usar alguno de sus servicios. La dirección IP de este servidor será la 192.168.20.10.

## Seguridad

La seguridad básica se lleva a cabo en los dispositivos de red mediante ACL (Listas de control de acceso) en los firewall. Esta seguridad se realiza a nivel de capa 3. Aún así, en los próximos capítulos se discutirá acerca de la actuación ante los distintos tipos de ataques, incluyendo seguridad en otras capas.

Básicamente, solo se permitirá tráfico a los servicios proveniente del exterior cuyo origen sea la subred de laboratorio 157.88.123.0/24. Por tanto, los servicios que ofrezca el servidor no pueden ser explotados por usuarios externos, excepto las aplicaciones que corren en los puertos 80 y 8080, que serán accesibles por todo el mundo. El resto del tráfico será todo bloqueado.

También se protegerá a los usuarios de la LAN de cualquier paquete que se reciba del exterior, excepto las respuestas ICMP o conexiones TCP. Además evitaremos spoofing de usuarios externos.

Utilizaremos un proxy Nginx [2] para redirigir el tráfico a nuestro servidor web y que sea necesario una autenticación para añadir una capa extra de seguridad. Por tanto, la visualización del tráfico de red no será accesible si no se dispone de la autenticación y evitaremos que cualquier persona del exterior pueda conocer la red interna.

No disponemos de un WAF (Firewall de aplicaciones webs) lo que nos limita para poder filtrar el tráfico basándose en el contenido de los paquetes de las aplicaciones web y detectar ataques típicos como Cross-Site Scripting, SQL injections o inclusión de archivos.

### 2.3. Diseño Físico

Además de las configuraciones básicas de los dispositivos principales, que se incluye en los anexos del proyecto, en esta sección se incluye un inventario de los dispositivos de red usados, con características principales y sus conexiones.

Cabe destacar que todas las conexiones se han realizado con cables Ethernet de categoría 5e. Actualmente son los más usados, soportando velocidades Gigabit Ethernet de 1.000 Mbps y diseñado para transmisión a frecuencias de hasta 100 MHz.

Como hosts se utilizarán dos Raspberry Pi con Raspbian como SO, tanto para el host de la LAN como para el host del exterior. En cambio, como servidor se utilizará un ordenador personal con las siguientes características principales:


- Fabricante: Hewlett-Packard
- Modelo: Compaq dx7500 Microtower
- Sistema Operativo: Linux Ubuntu Desktop 18.04.2
- Procesador: Intel(R) Core(TM)2 Quad CPU Q8200 2.33GHz
- Memoria RAM: 4 GB
- Disco duro: HDD 512 GB

Los dispositivos de red se enumeran en la tabla 26.


| ID          | Tipo   | Modelo              |
|-------------|--------|---------------------|
| R01-EXT-FWL | Router | Cisco 1841          |
| R02-INT-FWL | Router | Cisco 1841          |
| SW01-EXT    | Switch | Cisco Catalyst 2960 |
| SW03-DMZ    | Switch | Cisco Catalyst 2960 |

**Tabla 26:** Dispositivos de red

Las características de los dispositivos de red se enumeran en las tablas 27 y 28.

| Componente físico x 2  |
|--|
| Producto: Router Cisco 1841 [3]  |
|  |
| Fabricante: Cisco  |
| Gama de producto: 1800   |
| Protocolos: TCP/IP SNMP  |
| Frecuencia: 50 Hz/60 Hz  |
| Memoria RAM: 128 MB  |
| Memoria Flash: 32 MB   |
| Protocolo de gestión remota: HTTP o SNMP   |
| Protocolo de interconexión de datos: Ethernet o FastEthernet                         |

**Tabla 27:** Características Router

| Componente físico x 2  |
|--|
| Producto: Switch Cisco Catalyst 2950 [4]   |
|  |
| Fabricante: Cisco  |
| Frecuencia: 50 Hz/60 Hz  |
| Memoria RAM: 16MB  |
| Memoria Flash: 8MB   |
| Cantidad de puertos FastEthernet: 24 10/100 MB                                     |
| Cantidad de puertos SFP: 2 10/100/1000 MB  |
| Consumo energético: 22W  |
| Protocolos de gestión: SNMP, RMON, Telnet  |

**Tabla 28:** Características Switch

De las conexiones y puertos posibles en ambos dispositivos usaremos el cable de alimentación para conectarlo a la corriente, los puertos FastEthernet y el puerto de consola para conectarlo a un ordenador a través del cable de consola y poder realizar las configuraciones mediante la CLI. Se puede conectar a cualquier ordenador mediante un adaptador de DB9 a USB.

Las conexiones de los switches se presentan en la tabla 29.

| Switch   | Puerto | Dirección IP    | VLAN  |
|----------|--------|-----------------|-------|
| SW01-EXT | Fa0/1  | 157.88.123.0/24 | 1     |
|          | Fa0/2  | 157.88.123.0/24 |       |
|          | Fa0/3  | 157.88.123.0/24 |       |
| SW03-DMZ | Fa0/1  | Trunk           | Trunk |
|          | Fa0/2  | 192.168.20.0/24 | 20    |
|          | Fa0/3  | Trunk           | Trunk |
|          | Fa0/4  | 192.168.30.0/24 | 30    |

**Tabla 29:** Puertos Switches

Las conexiones de los routers se presentan en la tabla 30.

| <b>Router</b> | <b>Interfaz</b> | <b>Dirección IP</b> | <b>Dot IQ</b> | <b>ACL</b>      |
|---------------|-----------------|---------------------|---------------|-----------------|
| R01-EXT-FWL   | Fa0/0           | 157.88.123.102      | NO            | 112 in          |
|               | Fa0/1           | 192.168.10.1        | 10            | 110 in, 113 out |
|               | Fa0/1           | 192.168.20.1        | 20            | 110 in, 111 out |
| R02-INT-FWL   | Fa0/0           | 192.168.10.2        | 10            | 122 in          |
|               | Fa0/0           | 192.168.30.1        | 30            | 121 in          |

**Tabla 30:** Interfaces Routers



## 3. Monitorización

### 3.1. Netflow

Netflow [5] es un instrumento integrado dentro del software de Cisco IOS que sirve para hacer una caracterización del funcionamiento de la red. Dentro del campo de la ciberseguridad y en especial de los analistas, la monitorización es uno de los principales aspectos en los que se trabaja.

Netflow nos servirá de utilidad para monitorizar el tráfico que pasa por nuestro firewall. Entre otras cosas sirve para medir el uso de la red, la productividad, la utilización de los recursos o vulnerabilidades. Ayuda donde aplicar QoS, optimizar y detectar ataques.

Todos los paquetes que pasen, en este caso, por nuestro firewall, se examinan en busca de atributos de los paquetes IP. Gracias a estos atributos se determina si el paquete es único o si hay otros similares para crear un flujo. Los paquetes con la misma dirección IP, puertos de origen y destino, interfaz de protocolo y clase de servicio se agrupan en un flujo y, a continuación, se agrupan los paquetes y los bytes.

Un flujo será exportado a nuestro servidor cuando esté inactivo durante un tiempo, es decir, no se reciben paquetes del mismo flujo, o si el flujo está activo durante un período largo de tiempo y dura más que el temporizador activo. Utiliza temporizadores para determinar si el flujo está activo o inactivo, que por defecto son de 15 segundos. Todos estos parámetros son configurables.

Los principales campos de la versión 5 de Netflow, que es la que utilizaremos, se muestra a continuación. Hay que tener en cuenta que en cada paquete que se envía al servidor, puede contener varios flujos.

- Cabecera
  - Versión Netflow
  - Número de flujos exportados en el paquete
  - Timestamp
  - Número de secuencia
  - ID
- Registro
  - Dirección IP origen y destino, con máscaras de subred
  - Puerto de origen y destino
  - Dirección IP del siguiente salto
  - Índice SNMP de interfaz de entrada y salida

- Paquetes y bytes del flujo
- Comienzo y fin del flujo
- Tipo de protocolo
- Tipo de servicio
- Interfaz de enrutador o switch
- Flags

Para exportar los datos de Netflow de nuestro router lo configuraremos mediante línea de comandos (CLI), teniendo acceso por consola al mismo. Configuraremos el router para que cada vez que haya tráfico que pase por la interfaz interna del router exterior (interface fastEthernet 0/1.20), lo envíe a nuestro servidor de monitorización (192.168.20.10). También recogeremos el tráfico que llegue por la interfaz externa (fastEthernet 0/0), ya que habrá tráfico que será bloqueado por el firewall pero que nos interesa monitorizar. Además cambiaremos el temporizador de inactividad de flujos para que se pueda tener una visión de la red prácticamente a tiempo real.

Netflow, por defecto, envía los datos mediante el protocolo UDP al puerto 2055. Los comandos utilizados para activar y exportar Netflow son los siguientes:

```
#ip flow ingress
#ip flow egress
```

Para poder exportarlos a nuestro servidor:

```
#ip flow-export destination 192.168.20.10 2055
#ip flow-export version 5
```

Exportaremos tanto el tráfico de entrada proveniente del exterior del firewall, como el de entrada hacia el firewall proveniente del interior para ver las respuestas del servidor.

Una vez tenemos preparado el envío, necesitamos un recolector de este tráfico que esté escuchando en el puerto 2055 de UDP. Una de las soluciones más utilizadas recientemente para la recolección, creación de índices y de dashboards para mostrar el tráfico es ELK (Elasticsearch, Logstash y Kibana). Nos ofrece un servicio optimizado para recoger datos de múltiples aplicaciones distribuidas y poder realizar consultas agregadas a una base de datos muy grande de forma muy eficiente. Además, la interfaz de usuario de Kibana presenta de manera muy usable los datos recogidos.

## 3.2. Elasticsearch

Motor de análisis y búsqueda de texto completo. Permite almacenar, buscar y analizar grandes volúmenes de datos de forma rápida y en tiempo casi real. Usaremos Elasticsearch

[6] para poder almacenar los datos que recibiremos mediante Netflow.

Algunos de los conceptos básicos que necesitaremos conocer para el desarrollo de este trabajo son los siguientes:

- Índice: colección de documentos que poseen características similares. Cada índice es identificado con un nombre. Este nombre es usado para referirse al índice a la hora de actualizar, obtener o eliminar documentos incluidos en él.
- Documento: unidad básica de información que puede ser indexada. Este documento está en formato JSON. Se pueden almacenar todos los documentos que queramos, estos residen físicamente en un índice.
- Tipo: es utilizado para almacenar distintos tipos de documentos en el mismo índice.
- Cluster: colección de uno o más nodos que conjuntamente almacenan toda la información.
- Nodo: servidor único, que es parte de un cluster. Almacena datos y participa en el indexado y búsqueda.

Básicamente se trata de una base de datos no relacional, a través de la cuál se pueden almacenar, indexar y buscar eventos. Permite hacerlo de manera distribuida, formando clusters, aunque nosotros solo tendremos un único nodo.

Se configurará Elasticsearch mediante el archivo de configuración `elasticsearch.yml` para que corra en el host `192.168.20.10` y que sea accesible mediante el puerto `9200`. Se ejecutará únicamente en un nodo, el volumen de datos no será tan alto y no será necesario que sea distribuido. Además estableceremos las rutas de los logs del propio Elasticsearch.

El formato que se establece para almacenar el tráfico que recopilaremos del router es el especificado en el apéndice 9.

Esto nos servirá para poder filtrar en Kibana por alguno de los campos y para poder crear los dashboards con los diferentes parámetros. Para poder acceder a los documentos se sigue la estructura `ruta_a_elasticsearch/índice/tipo/id_documento` Por tanto, para poder ver el formato de los documentos que hemos creado podemos usar el siguiente comando:

```
$curl -X GET http://localhost:9200/netflow?pretty
```

También nos interesará eliminar documentos. Esto será necesario debido a que solo queremos mostrar el tráfico generado en un período de tiempo. Si no borramos el resto de documentos, habría mucho ruido, y necesitaríamos almacenar muchos logs, lo que se puede convertir en un problema.

```
$curl -X DELETE http://localhost:9200/netflow*
```

Por lo tanto, automatizaremos la tarea de eliminación de documentos mediante el uso de crontab. Añadiremos la tarea para que se ejecute todos los días a última hora para que no interceda en el uso normal de un usuario. Añadimos en el crontab:

```
59 23 * * * ~/home/sergio/borrado.sh
```

### 3.3. Logstash

Sirve para el procesamiento, se realiza en uno o más pipelines. Toda la información que se recibe pasará por Logstash [7] para después poder almacenarlo en Elasticsearch. Tiene la capacidad de unificar datos desde diferentes orígenes y normalizar los datos en el destino elegido.

Tiene tres partes fundamentales:

- **Input:** obtención de datos. Recogida de tráfico Netflow.
- **Output:** salida de datos. Utilizaremos Elasticsearch para almacenar los datos recibidos de Netflow.
- **Filter:** procesamiento intermedio. Se suele utilizar Grok para poder parsear y estructurar texto arbitrario.

Se suelen utilizar codecs que pueden operar como parte del output o del input para separar el transporte del mensaje del proceso de serialización. No es necesario añadir esto desde un archivo de configuración `.yml`, se puede lanzar también desde la línea de comandos.

Utilizaremos el módulo de Netflow [8] que simplifica la normalización, colección y su visualización. Esto permite parsear directamente el tráfico, indexarlo en Elasticsearch y lo muestra en una suite de dashboards de Kibana. No necesitaremos hacer el parseo nosotros mediante un archivo de configuración. Si fuese necesario uno de los plugins más utilizados para poder llevar esta tarea es Grok.

Aún así, necesitaremos configurar el módulo. La configuración del módulo se realiza en `logstash.yml`. Incluiremos lo siguiente:

```
modules:  
- name: netflow  
  var.input.udp.port: 2055  
  var.elasticsearch.hosts: http://127.0.0.1:9200  
  var.elasticsearch.ssl.enabled: false  
  var.kibana.host: 0.0.0.0:443  
  var.kibana.scheme: http  
  var.kibana.ssl.enabled: false  
  var.kibana.ssl.verification_mode: disable
```

En la configuración añadiremos el puerto por el que va a escuchar, para recoger el netflow del router, y las rutas de elasticsearch y kibana. Además desactivaremos el SSL para que pueda ser accesible sin necesidad de certificado.

Una vez cambiados los parámetros de la configuración, ejecutaremos el siguiente comando:

```
$bin/logstash --modules netflow --setup
```

Es decir, ejecutamos Logstash con la configuración del módulo y con la opción de setup para que nos añada los dashboards por defecto, los cuales después podemos modificar y añadir otros nuevos que se ajusten a nuestras necesidades.

### 3.4. Kibana

Plataforma de visualización diseñada para operar con Elasticsearch. Se puede visualizar los datos que almacena la base de datos en forma de gráficos o mapas. Podremos construir nuestros propios Dashboards para poder mostrar nuestros datos [9].

Para poder visualizar los logs de la base de datos, tendremos que seleccionar en Kibana los índices almacenados en Elasticsearch de los que queremos recuperar los datos. Los índices se almacenan por fecha, por lo que seleccionaremos todos los índices para poder ver todo el tráfico.

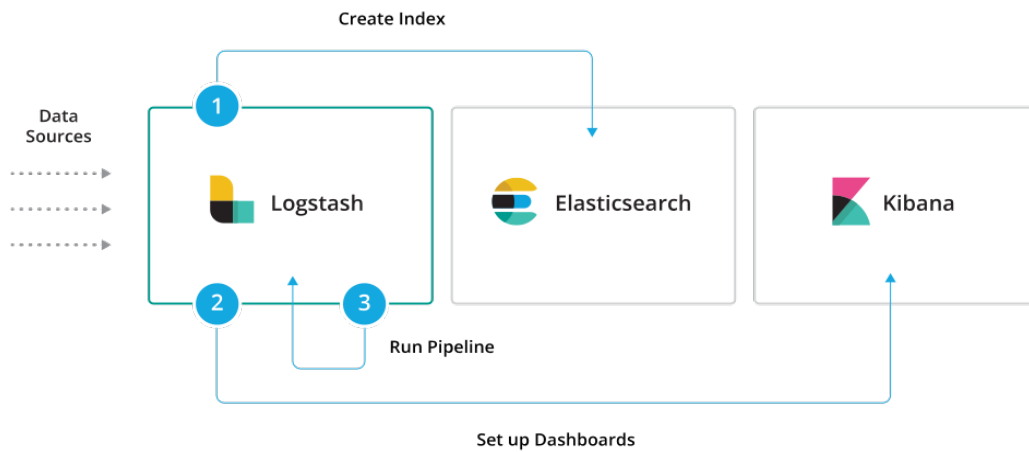
En el Discover se puede explorar todos los datos, desde donde se puede acceder a todos los documentos de todos los índices que coincidan con el índice seleccionado. Además podemos seleccionar filtros de tiempo, o de cualquiera de los campos de los documentos.

Los Dashboards de Kibana muestran una colección de visualizaciones y búsquedas. Se puede editar el contenido de los mismos y configurarlos en base a nuestras necesidades.

## Funcionamiento de la pila ELK

El funcionamiento cuando se ejecuta un módulo es el siguiente:

1. Se crea el índice en Elasticsearch.
2. Se establecen los Dashboards de Kibana, incluyendo patrones de índices, búsquedas y visualizaciones requeridas para visualizar los datos.
3. Se arranca la pipeline de Logstash con la configuración para leer y parsear los datos.



**Figura 10:** Funcionamiento ELK [10]

El recorrido que seguirá los logs generados por nuestro router es el siguiente:



**Figura 11:** Sistema completo de monitorización

## 4. Vulnerabilidades

En un tests de penetración de red, se probará el entorno de red frente a vulnerabilidades y amenazas. Estos test se dividen en test de penetración internas y externas.

- Test de penetración interna: ser parte de la red interna y realizar tests desde dentro. Bien con acceso mediante una VPN o físicamente.
- Test de penetración externo: incluye realizar tests de la IP pública.

### 4.1. Escaneo de Vulnerabilidades

Existen herramientas como OpenVas o Nessus que nos permiten hacer un escaneo automatizado de vulnerabilidades. En este caso nos centraremos en Nessus, aunque ambos son muy similares en cuanto a configuración y uso[11].

Consideraremos este tipo de herramientas como una primera aproximación a explotar un sistema o a auditar nuestra propia seguridad. Aunque en el resultado de este escaneo no se detecten vulnerabilidades, no quiere decir que no existan. Debemos realizar análisis más complejos.

Una vez descargado el software y configurado mediante su interfaz web (localhost:8834), procedemos a personalizar el escaneo que queremos llevar a cabo [12].

## Políticas

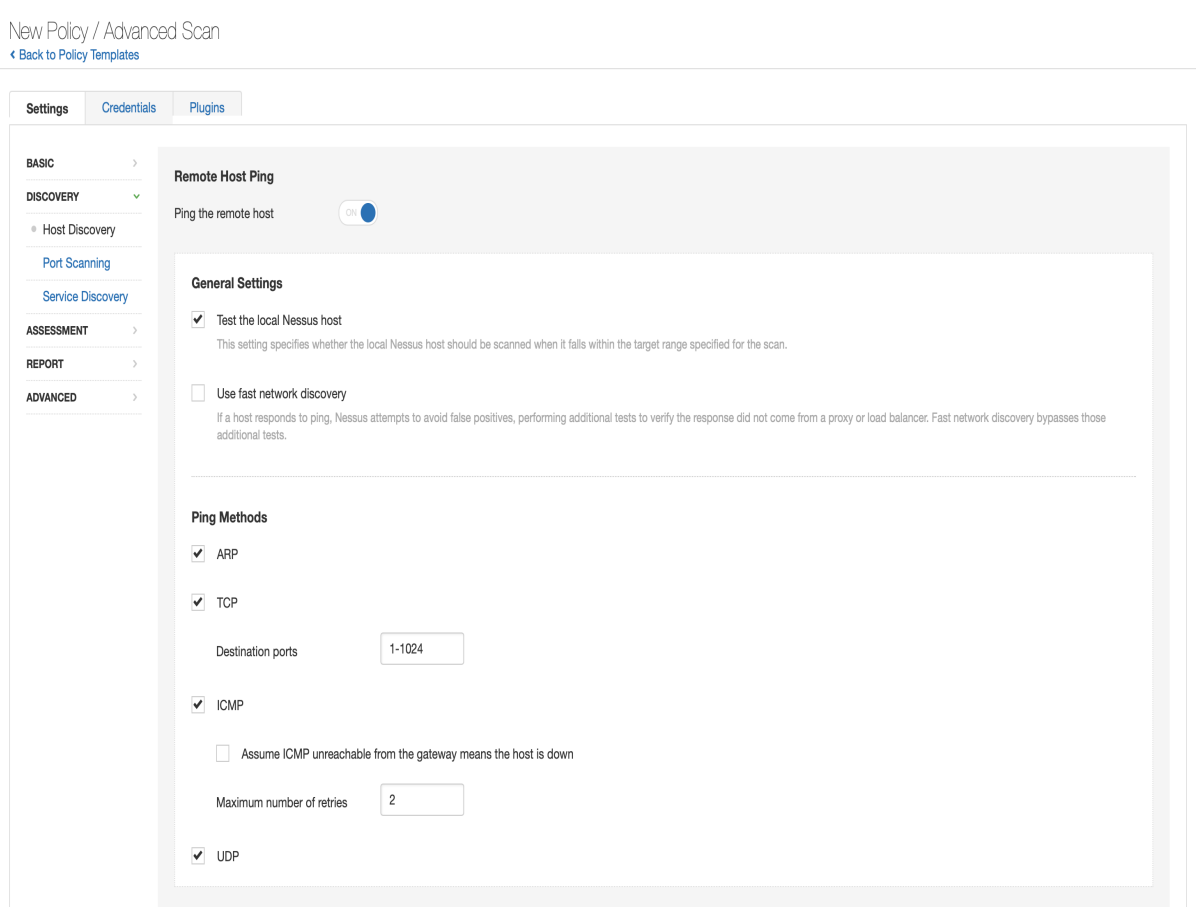
Crearemos una nueva política. Éstas nos permiten crear templates definiendo las acciones que se llevarán a cabo durante el escaneo. Ejecutaremos un escaneo avanzado donde podremos configurar los plugins y los parámetros a nuestra medida.

## Ajustes

La ajustes avanzados incluyen:

- Basic: para especificar aspectos básicos organizativos, incluyendo nombre y descripción del escaneo.
- Discovery: para establecer el descubrimiento y la exploración de puertos, incluyendo los rangos y los métodos.

- **Assessment:** para identificar malware, vulnerabilidades de fuerza bruta, y la susceptibilidad de un sistema web.
- **Report:** el procesamiento y la salida del escaneo.
- **Advanced:** otros parámetros para hacer más eficiente un escaneo.



**Figura 12:** Discovery

## Credenciales

Nos permite especificar credenciales de FTP, HTTP, SMB, de bases de datos...Esto permitirá realizar un análisis más en profundidad para determinar exposiciones al riesgo locales o infracciones de cumplimiento. Se suele utilizar para redes corporativas muy grandes. Nessus realizará un a variedad más amplia de verificaciones que darán resultado en un escaneo más preciso.



# Plugins

En este apartado tendremos una lista de vulnerabilidades. Podremos habilitar familias de plugins o plugins individuales para llevar a cabo.

New Policy / Advanced Scan Disable All Enable All

[Back to Policy Templates](#) Show Enabled | Show All

Settings Credentials **Plugins**

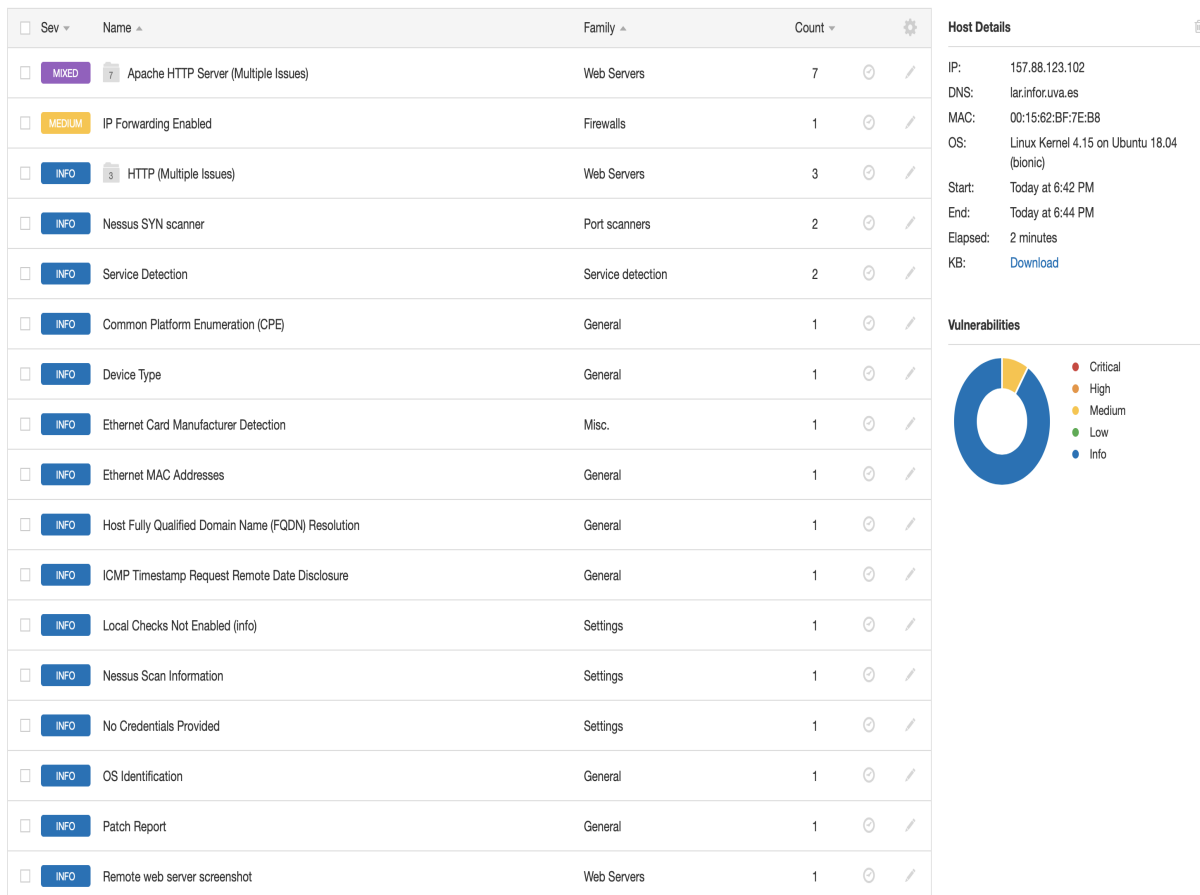
| STATUS  | PLUGIN FAMILY                      | TOTAL | STATUS  | PLUGIN NAME  | PLUGIN ID |
|---------|------------------------------------|-------|---------|--|-----------|
| ENABLED | AIX Local Security Checks          | 11363 | ENABLED | A Vulnerability in IOS Firewall Feature Set - Cisco Systems                          | 48962     |
| ENABLED | Amazon Linux Local Security Checks | 1270  | ENABLED | Access Point Web-browser Interface Vulnerability                                     | 48993     |
| ENABLED | Backdoors                          | 121   | ENABLED | Cable Modem Termination System Authentication Bypass - Cisco Systems                 | 48966     |
| ENABLED | CentOS Local Security Checks       | 2761  | ENABLED | Cisco 10000 Series Denial of Service Vulnerability (cisco-sa-20110928-c10k)          | 56313     |
| ENABLED | CGI abuses                         | 4067  | ENABLED | Cisco 10000, uBR10012, uBR7200 Series Devices IPC Vulnerability - Cisco Systems      | 49020     |
| ENABLED | CGI abuses : XSS                   | 671   | ENABLED | Cisco 12000 Series Router ICMP Unreachable DoS                                       | 10971     |
| ENABLED | CISCO                              | 974   | ENABLED | Cisco 12000 Series Routers Multiple Vulnerabilities (DoS, ACL Bypass)                | 10970     |
| ENABLED | Databases                          | 609   | ENABLED | Cisco 600 Series Router HTTP GET DoS (cisco-sa-20001204-cbos)                        | 10561     |
| ENABLED | Debian Local Security Checks       | 6110  | ENABLED | Cisco 6000/6500/7600 Crafted Layer 2 Frame Vulnerability - Cisco Systems             | 48973     |
| ENABLED | Default Unix Accounts              | 170   | ENABLED | Cisco 6400 NRP2 Unauthenticated Telnet Access (CSCot65960)                           | 10981     |
| ENABLED | Denial of Service                  | 110   | ENABLED | Cisco 675 Router Default Unpassworded Account  | 10045     |
| ENABLED | DNS                                | 181   | ENABLED | Cisco 7600 Series Route Switch Processor 720 with 10 Gigabit Ethernet Uplinks D...   | 73269     |
| ENABLED | F5 Networks Local Security Checks  | 759   | ENABLED | Cisco 9900 Series IP Phone Crafted Header Unregister Vulnerability                   | 72725     |
| ENABLED | Fedora Local Security Checks       | 13966 | ENABLED | Cisco ACE 4710 Appliance / ACE30 Module Multiple Vulnerabilities (Logjam)            | 91427     |
| ENABLED | Firewalls                          | 259   | ENABLED | Cisco ACE 4710 Device Manager GUI Remote Command Injection Vulnerability (ci...      | 89690     |
| ENABLED | FreeBSD Local Security Checks      | 4015  | ENABLED | Cisco ACE30 and ACE4710 OpenSSL 'ChangeCipherSpec' MITM Vulnerability                | 76127     |
| ENABLED | FTP                                | 255   | ENABLED | Cisco Adaptive Security Appliance Authenticated Cross-Site Scripting Vulnerabilit... | 102497    |
| ENABLED | Gain a shell remotely              | 281   | ENABLED | Cisco Adaptive Security Appliance ICMP Echo Request ACL Bypass (cisco-sa-20...       | 92630     |
| ENABLED | General                            | 268   | ENABLED | Cisco Adaptive Security Appliance Software DHCPv6 Packet Handling DoS (cisco...      | 90714     |

Save Cancel

Figura 13: Nessus plugins

## Lanzamiento

Una vez configurado todos los parámetros del escaneo, procedemos a definir cuál será nuestro objetivo y a lanzarlo. Una vez haya finalizado el escaneo, podremos encontrar en reports las principales vulnerabilidades del sistema objetivo que hemos incluido. Podremos exportarlo en varios formatos. Si estamos realizando un test de penetración, es recomendable exportarlo en formato .nessus para después poder importarlo, por ejemplo, en Metasploit.



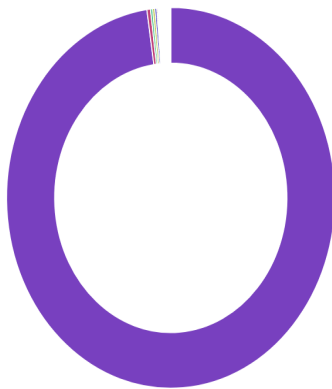
**Figura 14:** Resultados test de vulnerabilidades

Los resultados del test de vulnerabilidades aparecen ordenadas por el riesgo que implican en el sistema, de mayor a menor. Estas vulnerabilidades también pueden aparecer agrupadas si se trata del mismo servicio o dispositivo afectado. Debemos atender con especial urgencia aquellas que son críticas o suponen un riesgo elevado para nuestro sistema.

Podremos acceder a cada una de las vulnerabilidades. Resulta muy interesante para el usuario, pues incluye una descripción de la vulnerabilidad, con referencias a documentación que nos ayudará a entender la brecha de seguridad. En muchos casos se incluyen en el propio reporte medidas de mitigación inmediatas.

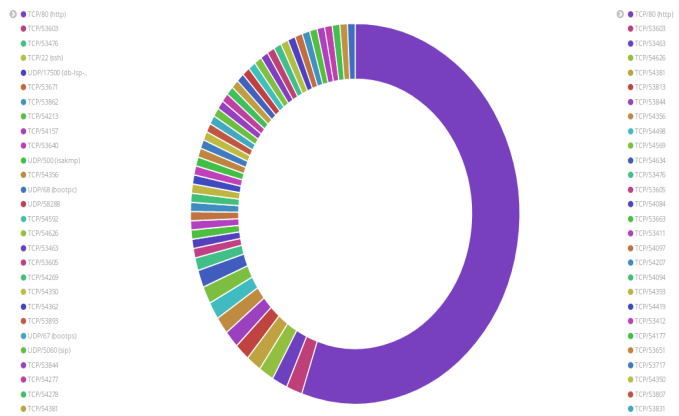
Es recomendable lanzar este tipo de escaneos cuando se haya diseñado y levantado una nueva red. Esto servirá tanto para ataque como para conseguir información de las vulnerabilidades de nuestra red y poder protegernos ante ellas. Será necesario lanzarlo para auditar la seguridad de cualquier red, nos permite obtener mucha información acerca de nuestros sistemas en poco tiempo. Por tanto, debería ser una obligación para cualquier administrador de redes utilizar estas herramientas.

Netflow: Source Ports (Bytes)



**Figura 15:** Puertos origen

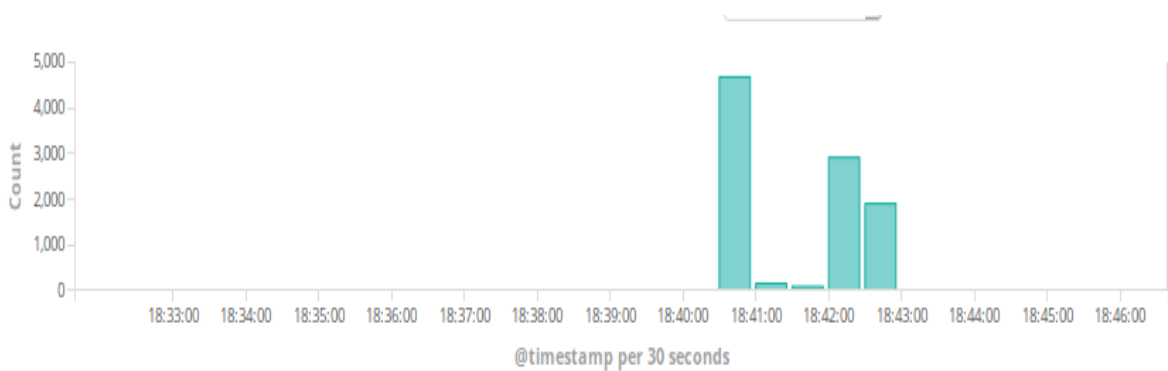
Netflow: Destination Ports (Bytes)



**Figura 16:** Puertos destino

Como podemos observar todo el tráfico tiene su origen en el puerto 80. Esto es porque herramientas como Nessus u Openvas son usadas mediante la interfaz web. No quiere decir que siempre sea así, si lo lanzásemos mediante Metasploit el origen de los puertos no sería el 80. Aún así es un indicio de que se trata de un escaneo de vulnerabilidades.

Además, si nos fijamos en los puertos destino, observamos que son variadas, aunque el puerto 80 recibe más tráfico que otros. Esto es debido a que el puerto 80 está abierto en el servidor y además presenta vulnerabilidades por usar una versión no actualizada de Apache.



**Figura 17:** Tráfico generado por Nessus

El tráfico de un escaneo de vulnerabilidades no es tan alto como ataques de denegación de servicio que veremos posteriormente. Aún así, todo dependerá de los parámetros de configuración del escaneo. Dependerá del número de puertos que serán escaneados, de los plugins que usemos o de los servicios que ofrezca el servidor.

## 4.2. Metasploit

Metasploit es un software gratuito y open-source que puede ser usado para automatizar tareas complejas. Metasploit es un gran framework, por lo que en este trabajo tan solo se presenta una pequeña introducción al mismo.

MSFConsole es la interfaz más popular de este framework y será con la que interactuemos para lanzar exploits aprovechando las vulnerabilidades.

Otras de las utilidades más usadas de este framework es MSFPayload que es usado para generar payloads, código shell u otros ejecutables, que serán introducidos en la máquina de la víctima después de que el exploit haya sido completado.

Tras haber usado Nessus para realizar un escaneo de vulnerabilidades, Metasploit nos ofrece la posibilidad de lanzar exploits para explotar esas vulnerabilidades. Para ello primero importaremos el reporte de Nessus:

```
msf > db_import ruta/.nessus
```

Una vez importado el escaneo, procederemos a lanzar los exploits:

```
msf > db_autopwn -x -p
```

Tras lanzarlo, podremos ver las sesiones obtenidas mediante :

```
msf > sessions -l
```

El problema de lanzar todos estos exploits es que se generará mucho ruido y se podrá detectar fácilmente. Por ello, este tipo de ataques se suelen realizar cuando se dispone de poco tiempo o simplemente se quiere auditar la seguridad.

Sin embargo, si después del escaneo, queremos explotar una vulnerabilidad concreta, podemos buscar el exploit concreto y usarlo. A continuación se muestra un ejemplo en el que buscamos un exploit, utilizamos el que deseamos, miramos los parámetros del exploit, establecemos los parámetros y lo ejecutamos:

```
msf > search ssh
msf > use auxiliary/scanner/ssh/ssh_login
msf auxiliary(ssh_login) > show options
msf auxiliary(ssh_login)> set RHOSTS 157.88.123.102
msf auxiliary(ssh_login)> set USERPASS_FILE
/usr/share/metasploit-framework/data/wordlists/root_userpass.txt
msf auxiliary(ssh_login) > run
```

Lanzando solo un exploit conseguiremos hacer menos ruido. A pesar de eso, en este ejemplo se lanza un ataque por fuerza bruta, lo que también generará bastante tráfico. Por tanto, deberemos analizar los exploits que queramos ejecutar, partiendo de las vulnerabilidades ofrecidas por herramientas como Nessus y predecir las consecuencias y su fácil detección o no.

### 4.3. Sniffing

Utilizado tanto para ataques como para defensas. Consiste en capturar los paquetes que pasan por un enlace de red. El objetivo principal para los atacantes es capturar credenciales que no están cifradas. Se suele capturar tráfico HTTP, SMTP o FTP. La mejor manera de protegerse frente a esto es usar protocolos cuya comunicación sea cifrada, aunque estas comunicaciones también pueden ser descifradas, pero requiere un mayor esfuerzo. En cambio, hoy en día, también es muy usada para analizar tráfico de red, cuellos de botella, interacciones entre sistemas...

Existen dos tipos de categorías:

- Activo: se interactúa con el objetivo, enviando paquetes y peticiones. Como por ejemplo, ARP Spoofing.
- Pasivo: el atacante no interactúa con el objetivo. Simplemente captura los paquetes enviados y recibidos por la red.

Nuestras tarjetas de red deben estar en modo promiscuo para poder capturar el tráfico que no es destinada a nuestra computadora.

#### 4.3.1. Man In The Middle

El atacante se situará en el medio de una comunicación entre cliente y servidor. Por lo tanto, toda comunicación entre ambos extremos será capturada por el atacante. Una vez situado en el medio, podrá llevar a cabo multitud de ataques. A continuación veremos ejemplos en los que su objetivo final es este.

#### 4.3.2. MAC Flooding

Como ya sabemos, ARP es un protocolo de la capa de enlace. Su propósito general es resolver una IP en una dirección MAC. Un switch dispone de una memoria interna, conocida como CAM (Content-Addressable Memory) donde asigna puertos a direcciones MAC[13]. Cuando llega una trama a un switch y se desconoce el destino, porque es la

primera vez que llega, o ha expirado, se enviará por todos los puertos de la VLAN excepto por el que fue recibido. Gracias a esto podrá enviar el paquete por un puerto único.

La idea de este ataque consiste en enviar una gran cantidad de réplicas de ARP para sobrecargarlo, llenando la tabla CAM del switch de asignaciones, y que pase a modo hub. Esto quiere decir que todo el tráfico que le llegue, lo mandará a todos los hosts de la red. Un atacante podría ahora escuchar todo el tráfico de la red mediante un sniffer.

### 4.3.3. ARP Spoofing

LLevaremos a cabo un ataque MITM mediante ARP Spoofing dentro de nuestra red local. Los equipos involucrados serán de la VLAN 30, es decir los de la subred 192.168.30.0/24. Estas serán las IPs de los equipos involucrados:

- Atacante: 192.168.30.10
- Víctima: 192.168.30.20
- Default Gateway: 192.168.30.1

Lo primero que debemos hacer es activar el reenvío IP en el equipo del atacante mediante:

```
echo 1 >/proc/sys/net/ipv4/ip_forward
```

La tabla arp del atacante es:

```
$arp -a
? (192.168.30.1) at 00:15:62:bf:7e:74 [ether] on eth0
? (192.168.30.20) at 00:23:7d:c7:1b:4d [ether] on eth0
```

La tabla arp de la víctima es:

```
$arp -a
? (192.168.30.1) at 00:15:62:bf:7e:74 [ether] on eth0
? (192.168.30.10) at 00:1c:42:60:bc:a0 [ether] on eth0
```

Para realizar el spoofing usaremos la herramienta Arpspoof del paquete Dsniff. Inundará la red con tramas ARP para que todas las maquinas actualicen su tabla. El objetivo es que la víctima en vez de enviar los paquetes al Default Gateway, los envíe a la máquina del atacante. Para ello cambiaremos la dirección MAC asociada a la IP del Default Gateway por la dirección MAC del atacante:

```
arpspoof -I eth0 -t 192.168.30.20 192.168.30.1
arpspoof -I eth0 -t 192.168.30.1 192.168.30.20
```

Una vez hecho el envenenamiento de su tabla ARP, podremos observar que ahora cada vez que la víctima envía los datos, pasarán por el atacante. Si consultamos la tabla arp de la victima obtenemos lo siguiente:

```
? (192.168.30.1) at 00:1c:42:60:bc:a0 [ether] on eth0
? (192.168.30.10) at 00:1c:42:60:bc:a0 [ether] on eth0
```

Como vemos, cada vez que se envíe algo por la red, pasará por el gateway, que en realidad es el atacante. Para detectar si ha ocurrido este ataque, consultar la tabla arp y ver si se repiten dos direcciones MAC iguales para IPs distintas. El tráfico que ha sido generado, inundando la red de paquetes ARP es el siguiente:

|     |            |                   |                   |     |    |   |       |                   |
|-----|------------|-------------------|-------------------|-----|----|---|-------|-------------------|
| 442 | 83.876665  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 470 | 85.877886  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 474 | 87.878213  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 482 | 89.879382  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 487 | 91.880894  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 492 | 93.881822  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 496 | 95.882164  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 501 | 97.883432  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 505 | 98.067907  | Parallel_60:bc:a0 | Broadcast         | ARP | 42 | Who has 192.168.30.1? Tell 192.168.30.10  |       |                   |
| 506 | 98.068511  | Cisco_bf:7e:74    | Parallel_60:bc:a0 | ARP | 60 | 192.168.30.1                              | is at | 00:15:62:bf:7e:74 |
| 511 | 99.883940  | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 514 | 101.884836 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 522 | 103.084027 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | Who has 192.168.30.20? Tell 192.168.30.10 |       |                   |
| 523 | 103.084392 | HewlettP_c7:1b:4d | Parallel_60:bc:a0 | ARP | 60 | 192.168.30.20                             | is at | 00:23:7d:c7:1b:4d |
| 524 | 103.885267 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 528 | 105.886510 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 534 | 107.887757 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 550 | 109.889039 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 557 | 111.889332 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 570 | 113.890579 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 604 | 115.891174 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 614 | 117.891514 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 631 | 119.891793 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 664 | 121.892985 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 676 | 123.894266 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 688 | 125.895535 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 699 | 127.895965 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 713 | 129.896178 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 723 | 131.897651 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 738 | 133.898058 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 748 | 135.899400 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |
| 762 | 137.900485 | Parallel_60:bc:a0 | HewlettP_c7:1b:4d | ARP | 42 | 192.168.30.1                              | is at | 00:1c:42:60:bc:a0 |

**Figura 18:** Tráfico generado por arpspoof

Una vez ejecutado podremos realizar otro tipo de ataques como es el de denegación de servicio, DNS spoofing... La herramienta que se suele usar para realizar el ARP spoofing y posteriormente llevar a cabo los otros ataques es Ettercap. No la hemos usado, pues el proceso que hemos llevado a cabo es más ilustrativo y nos enseña cómo funciona este ataque.

## Hijacking

Una vez realizado ARP Spoofing y estar en el medio de una comunicación, podremos realizar varios tipos de ataques. Como ya hemos comentado podríamos sniffar todo el tráfico para ver contraseñas en texto plano, pero también podemos realizar un secuestro de sesión.

El hijacking consiste en robar los tokens o cookies de sesión para autenticar a un usuario en un sitio web. Aunque es un método que está cada vez más en desuso, pues solo funciona con HTTP, cuando la comunicación no es cifrada. A pesar de ello, hay herramientas como SSL Strip que convierten todos los enlaces HTTPS en HTTP. De esta manera, ya podríamos captar todas las contraseñas en texto plano.

## Mitigación

La mejor forma de detectar este tipo de ataques es mediante un IDS o un IPS. También es conveniente revisar las tablas ARP para ver si a dos IPs distintas le corresponde la misma dirección MAC y así detectarlo.

Existen mecanismos de autenticación para prevenirlo y así tener a un usuario siempre identificado. También puede ser usado ARP estático, de tal forma que sea imposible ninguna alteración, pero no es recomendable para redes grandes. Sin duda, la técnica más usada en empresas u organizaciones con redes grandes es usar una VPN.

En algunos switches modernos de Cisco también existe el DAI (Dynamic ARP Inspection)[14] para prevenir ataques relacionados con el protocolo ARP. Al activarlo, intercepta todas las peticiones y respuestas ARP y verifica la autenticidad antes de actualizar la tabla o de reenviar los paquetes a los destinos. Nosotros no podremos implementar este método ya que el que hemos utilizado se trata de uno más antiguo, sin esta característica.

### 4.4. Escaneo de Puertos

Nmap [15] es una herramienta open-source cuyas utilidades principales son el descubrimiento de redes y auditorías de seguridad. Es útil también para administradores de redes y sistemas para mantener el inventario de los dispositivos, o monitorización de servicios y actividades de los hosts. Lo interesante de esta herramienta es la tabla de puertos que nos muestra la salida. Esta tabla muestra el estado, el número de puerto, el protocolo, el nombre de servicio y el estado. El estado puede ser:

- Open: una aplicación está escuchando en ese puerto



- Filtered: un firewall, filtro u otro obstáculo de red está bloqueando ese puerto, por tanto Nmap no puede indicarnos si está abierto o está cerrado
- Unfiltered: cuando responde a Nmap, pero no puede determinar si el puerto está abierto o cerrado
- Closed: no hay aplicaciones escuchando en el puerto

Esta técnicas de escaneo de puertos es fácilmente detectable por sistemas de detección de intrusos (IDS), como es el caso de Snort.

La sintaxis de Nmap es:

```
nmap [Scan Type...] [Options] {target specification}
```

Antes de realizar cualquier ataque es muy importante hacer un análisis. Para ello una de las opciones más importantes que podemos utilizar es la siguiente: Utilizaremos la opción -A para que nos de información de la versión del servicio y del sistema operativo que está ejecutando el host remoto.

A continuación comentaremos alguno de los rastreos más comunes y el tráfico que genera, dependiendo de lo que nos interese atacar, usaremos uno u otro.

## TCP SYN Scan

Es la técnica más utilizada. Se activa con la opción -sS. Escanea miles de puertos por segundo y es relativamente discreto, ya que nunca completa las conexiones TCP. Envía un paquete SYN, como si fuera a abrir una conexión real y luego espera la respuesta. Un SYN o ACK indica que el puerto está abierto, mientras que un RST(reinicio) indica que no se escucha. Si no se ha recibido nada o se ha recibido un paquete ICMP, indica que el puerto está filtrado. El resultado de lanzar este comando es el siguiente:

```
$nmap 157.88.123.102 -sS
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-29 18:07 CEST
Nmap scan report for lar.infor.uva.es (157.88.123.102)
Host is up (0.0013s latency).
Not shown: 977 closed ports
PORT      STATE      SERVICE
7/tcp    filtered  echo
9/tcp    filtered  discard
13/tcp   filtered  daytime
19/tcp   filtered  chargen
21/tcp   filtered  ftp
23/tcp   filtered  telnet
79/tcp   filtered  finger
80/tcp   open      http
113/tcp  filtered  ident
179/tcp  filtered  bgp
512/tcp  filtered  exec
513/tcp  filtered  login
```

```
514/tcp    filtered shell
515/tcp    filtered printer
544/tcp    filtered kshell
1718/tcp   filtered h323gatedisc
1719/tcp   filtered h323gatestat
1720/tcp   filtered h323q931
1723/tcp   filtered pptp
1984/tcp   filtered bigbrother
2065/tcp   filtered dlsrpn
5060/tcp   filtered sip
10000/tcp  filtered snet-sensor-mgmt
MAC Address: 00:15:62:BF:7E:B8 (Cisco Systems)
```

Nmap done: 1 IP address (1 host up) scanned in 18.48 seconds

## Escaneos UDP

La mayoría de los servicios de Internet se ejecutan a través del protocolo TCP. Pero existen servicios muy importantes y vulnerables como DNS, SNMP o DHCP que utilizan el protocolo UDP. Este escaneo se activa con la opción `-sU` y se puede combinar con el escaneo TCP.

Se envía un paquete UDP, normalmente con contenido vacío, a menos que se especifique. Si devuelve un código ICMP (tipo 3, código 3) el puerto está cerrado. Otros errores ICMP indican el puerto como filtrado. Si no se recibe nada, significa que el puerto está abierto o que se ha bloqueado la comunicación.

La dificultad de este escaneo es hacerlo rápidamente, ya que los puertos abiertos o filtrados no mandan respuesta y deja a Nmap en espera.

## Mitigación

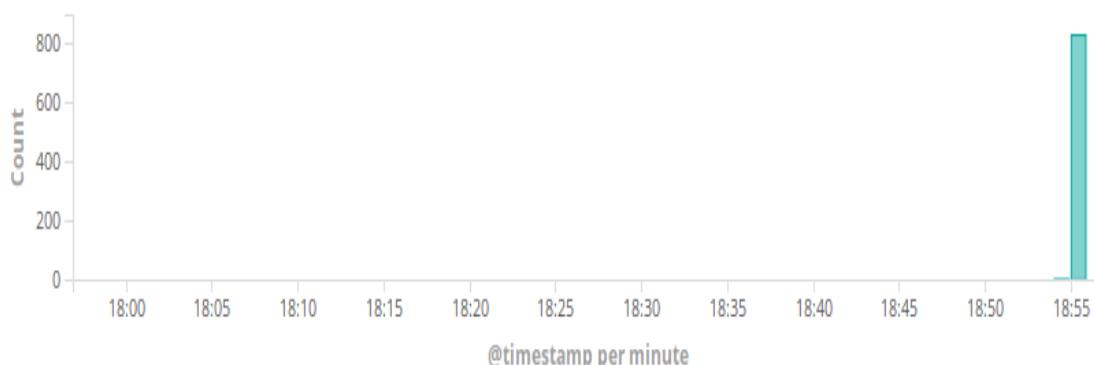
La única forma de evitar un escaneo de puertos es bloquear todo el tráfico que llega a la red desde el exterior. En nuestro caso, no podemos bloquear todo el tráfico, estamos ofreciendo servicios al exterior y si lo bloqueamos no podrán ser usados. En cualquier entorno de laboratorio, esto es una situación que no se va a dar.

Por lo tanto, estamos expuestos a recibir constantemente escaneos de puertos. No nos tendremos que centrar en evitar estos, nos centraremos en proteger el acceso a los servicios.

Hay una solución para evitar escaneos de puertos masivos. Existe la posibilidad de monitorizar continuamente el estado de la red y tener un IDS para detectar este tipo de ataques. Cuando observemos actividad malicioso y continuada por parte de una o varias IPs, podremos bloquear esta IP en el firewall y evitar que puedan continuar haciéndolo.

Nosotros podríamos simularlo en el firewall, analizando a tiempo real el tráfico y bloqueando la IP, mediante el uso de listas de control de acceso, que realiza los ataques.

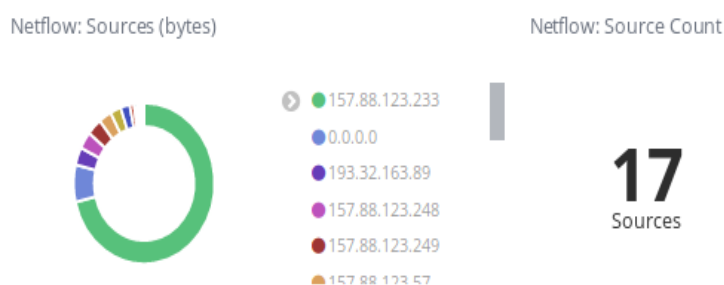
Usaremos Kibana para detectar un escaneo de este tipo, previamente lanzado.



**Figura 19:** Monitorización durante un período corto de tiempo

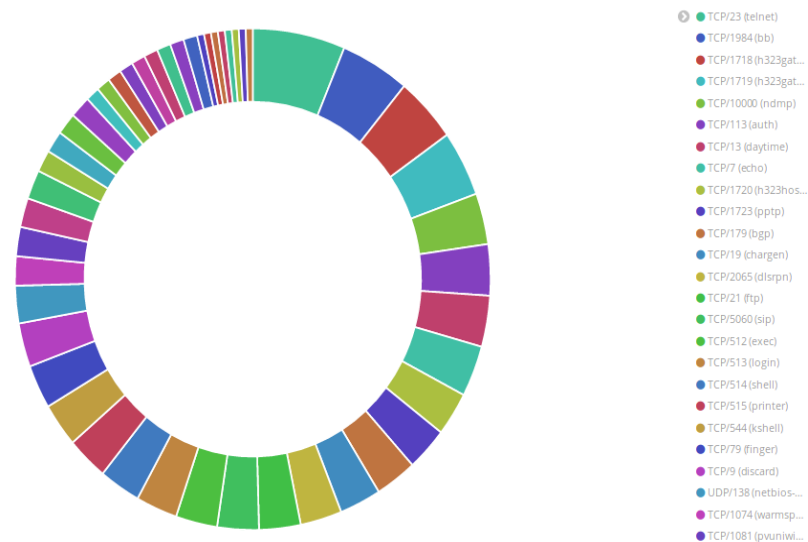
Como observamos en la figura, durante el período de tiempo que hemos monitorizado el tráfico, podemos observar claramente un pico. Ante cualquier tipo de ataque, uno de los primeros pasos es fijarse en el volumen de tráfico que se genera y si se genera por la misma IP durante un corto período de tiempo. Nos servirá para identificar si estos ataques han generado un tráfico tan alto que pueda haber provocado un ataque de denegación de servicio.

Se puede observar que la IP de origen que más tráfico ha generado es la del atacante. Filtrando el tráfico por la IP del atacante, obtendremos el destino de las peticiones y los puertos.



**Figura 20:** Origen de las peticiones

En este tipo de ataques se observará que las peticiones se realizarán a múltiples puertos, en vez de a un puerto concreto, como puede ocurrir en un ataque de fuerza bruta. Es una distribución uniforme y no se ha dirigido a ningún puerto en concreto.



**Figura 21:** Puertos de destino

Es evidente que al haberse generado tantas peticiones desde una IP contra otra IP a multitud de puertos, nos encontramos ante un escaneo de puertos.

Podremos analizar los logs para ver que puertos ha detectado como abiertos. Para ello, nos fijaremos en los flags de los paquetes TCP. Como hemos visto en la descripción del ataque, según el tipo de flag visto en los logs, podremos detectar si un puerto está abierto, cerrado, filtrado o sin filtrar.

## 4.5. Ataque de Fuerza Bruta

Un ataque de fuerza bruta[16] consiste en que un atacante configure valores predeterminados, realice solicitudes a un servidor, utilizando esos valores y finalmente analice la respuesta. Hydra junto con Medusa son las dos herramientas que más se utilizan para realizar este tipo de ataques, ambos son crackeadores de contraseñas. Para realizar este ataque se usará un diccionario de contraseñas que se generará mediante la herramienta Crunch. La sintaxis de Crunch es:

```
crunch <min-len> <max-len> [<charset string>] [options]
```

Para poder realizar un ataque de estas características, lo primero que supondremos es el formato de la contraseña. Existen multitud de diccionarios con las contraseñas por defecto de muchos sistemas y las más usadas por la gente, que en ciertos casos nos puede ahorrar mucho tiempo a la hora de llevar este tipo de ataques.

No solo existen diccionarios de contraseñas, también podremos encontrar diccionarios de usuarios o puertos. Cuanto menos específicos seamos, más tiempo llevará el ataque e incluso se podría no llegar a completar en un período de tiempo aceptable.

Mediante la herramienta Crunch generaremos diccionarios contemplando todas las posibilidades. Si no usa una contraseña muy robusta, será efectivo en un tiempo más o menos razonable.

Una vez que tenemos el diccionario de posibles contraseñas, podemos proceder a lanzar el ataque de fuerza bruta hacia un servicio. Supondremos también que el usuario será root, aunque también se podría hacer un diccionario de usuarios. Para lanzar el ataque, usaremos la herramienta Hydra.

Ejecutaremos el comando:

```
hydra -l root -P diccionario.txt 157.88.123.102 ssh
```

## Mitigación

Para bloquear un ataque de fuerza bruta a uno de nuestros servicios tenemos varias opciones. Dependiendo el caso en el que nos encontremos y las circunstancias realizaremos unas u otras. La posibilidad de que un ataque de fuerza bruta tenga éxito o no dependerá de de estas medidas. En este caso nos centraremos en uno de los servicios más afectados, SSH.

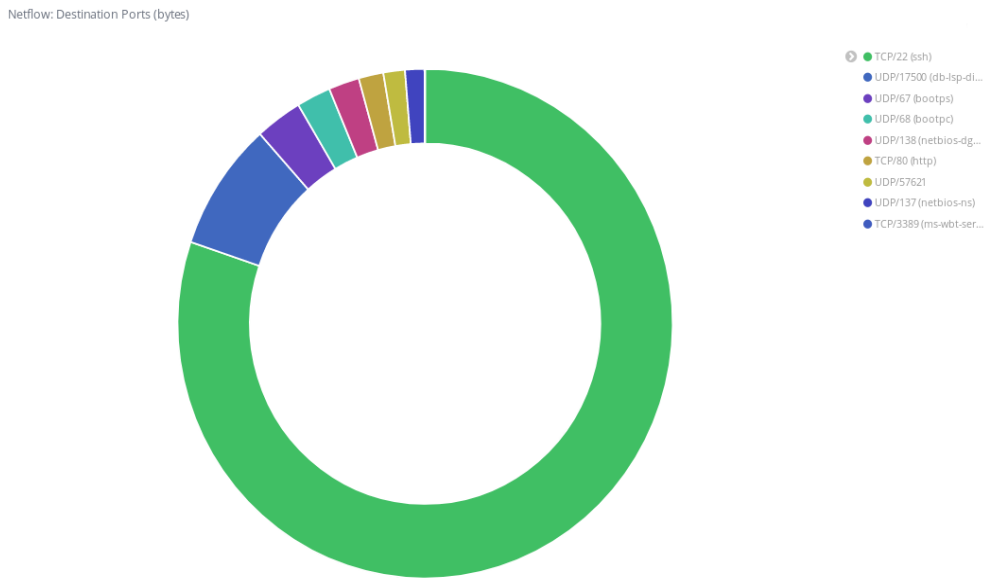
Lo primero que debemos tener en cuenta es que estos ataques se realizan mediante diccionarios de usuarios y contraseñas. Muchos de estos diccionarios suelen ser muy completos y se incluyen contraseñas por defecto de dispositivos y contraseñas habituales.

Por lo tanto un primer paso es tener una contraseña que no sea tan débil ante estos ataques. Es recomendable usar contraseñas largas, con letras, números y otros caracteres. Llevar a cabo un ataque por fuerza bruta con una contraseña así, llevaría varios días y sería fácilmente detectable.

Otra de las medidas que ayudan a mitigar este tipo de ataques, es cambiar el puerto defecto. Así pues ciertos servicios como SSH, que por defecto corren en el 22, son usualmente atacados por ataques automatizados. Aunque con un escaneo de puertos, se podría ver cuáles son los puertos que están abiertos y los servicios que corren, muchos de estos ataques son ejecutados mediante scripts, que a menos que sean dirigidos, no utilizarán otro tipo de herramientas. Cambiar el puerto, por lo tanto, aumentaría el tiempo para que tuviera éxito el ataque.

Al igual que con el resto de ataques, los IDS/IPS nos ayudarán en la detección de estos ataques. La monitorización de la red nos permitirá identificar los ataques, viendo picos donde se registran más peticiones a un puerto determinado. Al contrario que en un escaneo de puertos se puede observar que el ataque va dirigido a un puerto en concreto y

que muchas de las peticiones van dirigidas.



**Figura 22:** Puertos destino

Tras detectar que se ha producido el ataque de fuerza bruta, será necesario ver si alguna conexión ha llegado a tener éxito. Esto se puede realizar mirando los logs del servidor o analizando el tráfico. Si analizamos el tráfico y vemos que en alguna de esas conexiones se ha transmitido un número alto de bytes, probablemente haya conseguido tener éxito. Finalmente, habrá que mirar en los logs del servidor para confirmar el ataque.

## Configuraciones básicas

Detectar un ataque por fuerza bruta si está la red monitorizada suele ser bastante fácil, además si disponemos de un IDS/IPS, nos saltará también una alerta y podremos bloquearlo. Para protegernos de estos ataques tenemos varias opciones [17]:

- Archivo de configuración para el demonio

```
/etc/ssh/sshd_config
```

- No permitir el acceso mediante root. Esto nos permitirá que cuando usen un diccionario para los usuarios, no puedan usar root. Para ello escribiremos en el fichero:

```
PERMITROOTLOGIN NO
```

- Usar versiones seguras de los servicios. En el caso de SSH, la versión 1 del protocolo tiene muchas vulnerabilidades conocidas, por lo que debemos usar la 2. Escribimos en el fichero

PROTOCOL 2

- Modificar el número de segundos que la pantalla de login está activa, para que pasado el tiempo se cierre. Escribimos en el fichero

```
LOGINGRACETIME 30
```

- Modificar el número de intentos máximos que podemos fallar el login, después de estos intentos fallidos se cerrará la conexión. Escribimos en el fichero

```
MAXAUTHTRIES 3
```

- Desactivar contraseñas, uso de RSA, clave pública y privada.
- Si mediante la monitorización observamos que alguien está intentando acceder al sistema mediante fuerza bruta, podemos bloquear su IP en el firewall mediante una ACL. Otra opción que tenemos, es bloquear a todos excepto a las IPs que demos permiso.
- Portknocking
- Módulos PAM

#### 4.5.1. Módulos PAM

Los módulos PAM[18] son una suite de librerías compartidas que permiten administrar la autenticación de los usuarios. Podremos aplicar mecanismos de seguridad sin tener que aplicar los cambios y modificar cada aplicación. Las configuraciones se pueden añadir a los ficheros del directorio `/etc/pam.d`. En ese directorio encontramos ficheros para los diferentes servicios.

La sintaxis de las reglas dentro de esos ficheros es la siguiente:

```
tipo control modulo argumentos
```

- Tipo: se usa para especificar con qué grupos está asociada. Los valores pueden ser:
  - account: típicamente usada para restringir o permitir el acceso a servicios basándose en el tiempo, recursos del sistema, como máximo número de usuarios.
  - auth: usado para comprobar la identificación del usuario o para comprobar privilegios de grupos de usuarios.
  - password: para comprobar el token asociado con cada usuario.
  - session: asociado con lo que debe hacer el usuario antes o después de recibir un servicio.
- Control: indica el comportamiento

- **required**: los módulos requeridos son obligatorios. En caso de error, PAM devuelve el error, pero solo después de ejecutar el resto de los módulos en la misma pila.
  - **requisite**: Los módulos necesitan tener éxito. Pero si fallan, PAM devuelve inmediatamente el error sin ejecutar otro módulo.
  - **sufficient**: cuando tienen éxito, hacen que PAM devuelva inmediatamente éxito sin ejecutar ningún otro módulo.
  - **optional**: pueden tener éxito o fallar, devuelve éxito o fracaso según si el módulo tiene éxito o no.
- **Modulo**: el nombre completo de PAM usado por la aplicación. Se encuentran en `/lib64/security`.
  - **Argumentos**: lista de tokens usados para modificar el comportamiento del módulo.

Por ejemplo, en el caso de SSH, editaremos el fichero `/etc/pam.d/sshd` y añadiremos los módulos necesarios para una correcta autenticación. Algunos de estos módulos son `pam_access.so`, `pam_nologin.so`...

#### 4.5.2. Port Knocking

Consiste en esconder un puerto hasta que una secuencia de puertos ocurra. En Linux, el software más utilizado es `knockd`. Un servidor `knockd` escucha todo el tráfico que pasa por una interfaz Ethernet, buscando secuencias de puertos[19].

El cliente enviará paquetes UDP o TCP a unos puertos específicos del servidor. No necesariamente tienen que estar esos puertos abiertos, ya que `knockd` escucha en la capa de enlace y ve todo el tráfico. Cuando se detecta que la secuencia de puertos ocurre, `knockd` abrirá el puerto del servicio que deseemos. Además, en `knockd.conf` se puede configurar para que pasado un cierto tiempo, se vuelva a cerrar el puerto o que para una secuencia incorrecta también se cierre. Un ejemplo de port knocking sería el siguiente:

```
1923:tcp, 8902:udp, 5558:tcp, 1099:udp, 68543:udp
```

Tras realizar esta secuencia de puertos, el puerto deseado se abriría. Si esta secuencia no cambia, cualquiera que utilice un sniffer, puede descubrirla. Es recomendable utilizar un generador pseudoaleatorio de números.

Simulamos el comportamiento para mantener oculto el puerto que utilizaremos para realizar conexiones SSH. Tendremos que configurar `knockd` en el servidor, en el fichero `knockd.conf`.

[openSSH]



```

sequence    = 1923:tcp,8902:udp,5558:tcp,1099:udp,68543:udp
seq_timeout = 5
command     = /sbin/iptables -A INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
tcpflags    = syn

[closeSSH]
sequence    = 68543:udp,1099:udp,5558:tcp,8902:udp,1923:tcp
seq_timeout = 5
command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
tcpflags    = syn

```

En este caso, al producirse la secuencia de puertos en el intervalo de 5 segundos, se abrirá el puerto 22. Nuestro servidor, por tanto tiene los puertos 22 y 80 abiertos, aunque con nuestra configuración de knockd, solo se mostrará el 80.

```

$ nmap 157.88.123.102

Starting Nmap 7.70 ( http://nmap.org )
Nmap scan report for lar.infor.uva.es(157.88.123.102)
Host is up (0.0028s latency).
Not shown:999 filtered ports
PORT      STATE  OPEN
80/tcp    open   http

Nmap done: 1 IP address (1 host up) scanned in 5.82 seconds

```

Tras ejecutar desde el cliente la secuencia de puertos:

```
$knock 1923:tcp,8902:udp,5558:tcp,1099:udp,68543:udp
```

3 Observamos que el puerto 22 aparece abierto:

```

$ nmap 157.88.123.102

Starting Nmap 7.70 ( http://nmap.org )
Nmap scan report for lar.infor.uva.es(157.88.123.102)
Host is up (0.0028s latency).
Not shown:998 filtered ports
PORT      STATE  OPEN
22/tcp    open   ssh
80/tcp    open   http

Nmap done: 1 IP address (1 host up) scanned in 5.82 seconds

```

Lo mismo ocurre si deseamos cerrarlo al ejecutar la secuencia adecuada. Esto nos permitirá mantener oculto el servicio, pero sigue siendo accesible. La necesidad de cambiar el puerto por defecto es real, aunque no puedan ver cuáles son los puertos que realmente están abiertos, podrían acceder al servicio.

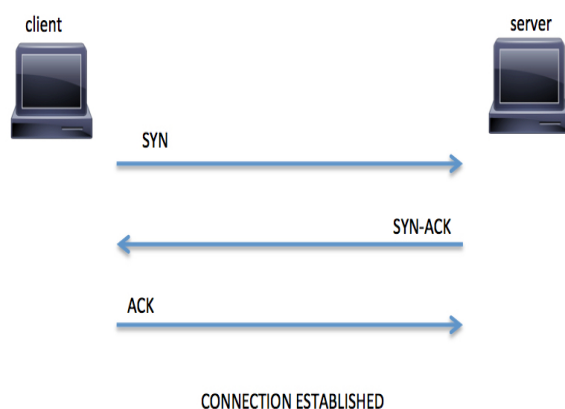
## 4.6. Ataque Denegación de Servicio

### TCP SYN Flood

Es un método de denegación de servicio que afecta a hosts que corren procesos TCP. El ataque aprovecha el tiempo de espera después de recibir un segmento SYN en un puerto que está en estado de escucha[20].

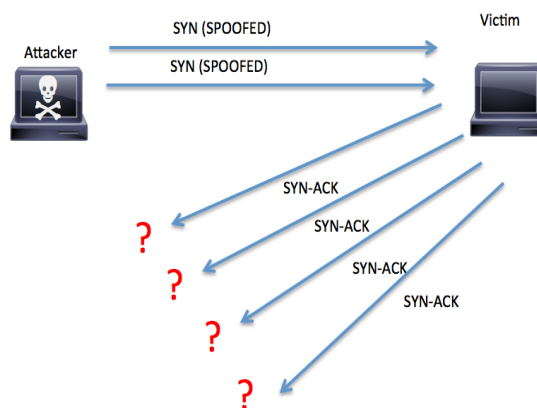
Anteriormente se ha mencionado la importancia de los paquetes SYN en un escáner de puertos. Para entender cómo funciona un ataque de este tipo, necesitamos entender cómo se establece una conexión TCP. Una conexión TCP se establece con lo que comúnmente se conoce como un "three-way handshake":

1. El cliente envía un paquete SYN para establecer una conexión TCP.
2. El servidor recibe el paquete SYN, pasa a estado de SYN-RCVD y responde con un SYN+ACK.
3. Finalmente, el cliente responde con un ACK.



**Figura 23:** TCP three-way handshake [20]

Sin embargo, si no se realiza el tercer paso, el servidor seguirá esperando un ACK en estado de SYN-RCVD. Existe la posibilidad de no responder al SYN+ACK o de hacer "spoofing" de la dirección de origen como se muestra en la siguiente figura.



**Figura 24:** Inundación SYN con spoofing [20]

El ataque por inundación SYN explota este comportamiento. El propósito de este ataque es enviar muchos paquetes SYN al servidor e ignorar los paquetes SYN+ACK devueltos. Esto provoca que el servidor se quede esperando por múltiples peticiones durante el tiempo configurado de espera. Esto provoca una sobrecarga en el servidor, ya que hay un número limitado de conexiones TCP concurrentes. Si el servidor llega al límite, no se podrán aceptar nuevas conexiones.

Para poder probar este tipo de ataques utilizaremos "hping3". Consiste en una herramienta de red capaz de enviar paquetes TCP/IP personalizados y de mostrar las réplicas del objetivo. Es útil para poder probar firewalls, escáneos de puertos, fragmentación...

Lanzaremos el ataque mediante:

```
$hping3 -S --flood -V -p 80 157.88.123.102
```

## UDP/ICMP Flood

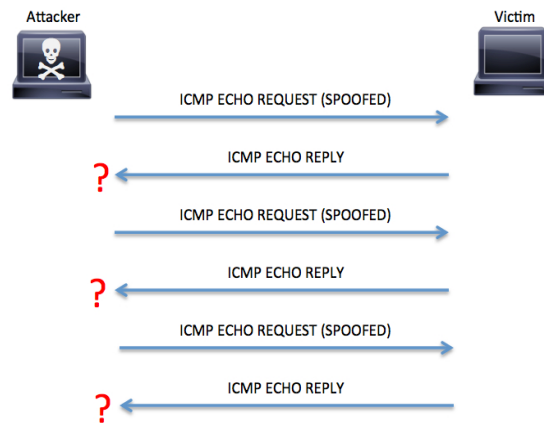
UDP, al contrario que TCP, no necesita crear una sesión entre cliente y servidor, es decir, no hay ningún handshake. ICMP es otro caso en el que no se necesita crear una sesión. Suele ser usado para mandar mensajes de error e información operacional desde los dispositivos de red.

Ambos protocolos no explotan ninguna vulnerabilidad. Este ataque consiste simplemente en mandar paquetes continuamente a puertos aleatorios hasta que el servidor se sobrecargue intentando procesar todas las peticiones. También se puede usar en combinación con "spoofing".

Cuando se envía un paquete UDP en el cuál no hay aplicaciones escuchando, como hemos visto anteriormente, el servidor tendrá que responder con numerosos paquetes

ICMP.

En un ataque de inundación ICMP, se intercambiarán dos tipos de mensajes entre el cliente y el servidor, un ICMP Echo Request y un ICMP Echo Reply.



**Figura 25:** ICMP flood con spoofing [20]

Muchos de los sistemas modernos limitan el número de respuestas ICMP para minimizar el impacto de estos ataques e intentar mitigarlo.

Lanzaremos el ataque mediante:

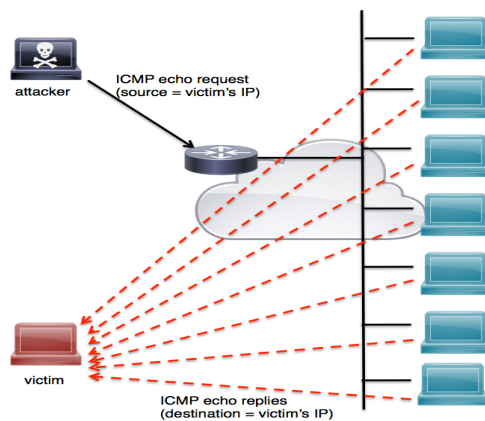
```
$hping3 --flood --rand-source -1 -p 157.88.123.102
$hping3 --flood --rand-source --udp -p 157.88.123.102
```

## Ataques de amplificación

Los ataques de denegación de servicio vistos pueden ser combinados con “spoofing”. Esto se puede aprovechar para sobrecargar otro equipo.

### Ataque SMURF

Un atacante elige un intermediario como amplificador. Envía una gran cantidad de paquetes ICMP a la dirección IP de broadcast de esos sitios intermediarios. La dirección sobre la que haremos spoofing es la de la víctima. Esto provocará que todos los dispositivos de la red manden ICMP Echo Replies a la víctima para sobrecargarla.

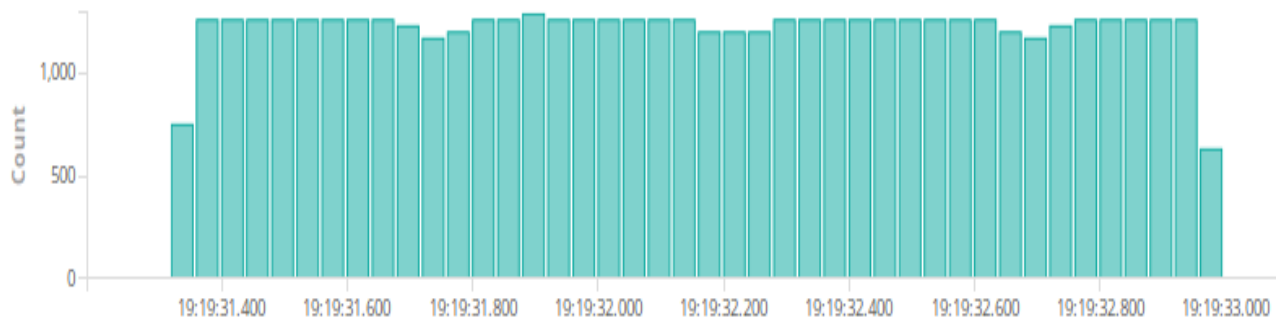


**Figura 26:** Smurf attack [20]

## Mitigación

Hay muchas estrategias para la mitigación de este tipo de ataques, aunque ninguna es totalmente efectiva. Por supuesto, la mejor manera de ver estos ataques, que pueden ser identificados fácilmente, son utilizando IDS/ISP o monitorizando la red y añadiendo reglas de filtrado en el firewall, como por ejemplo evitando paquetes ICMP a nuestro servidor para evitar ICMP Flood.

Tras analizar el tráfico que pasa por nuestra red al realizar cualquiera de los ataques de denegación de servicio, observamos que el tráfico generado es mucho mayor que cualquiera de los ataques.



**Figura 27:** Tráfico generado por TCP SYN Flood

Observamos que en apenas 1 segundo se generan 40.000 peticiones. Esto provocará la saturación del servidor transcurrido un cierto tiempo. Además, estos ataques se suelen realizar distribuidos, multiplicando el número de peticiones. Todas las peticiones van dirigidas al mismo puerto, al igual que en un ataque de fuerza bruta, pero a un volumen mucho mayor.

Aún así, existen otras técnicas para intentar evitar este tipo de ataques. En estos casos es recomendable usar, proxies inversos o balanceadores de carga. El objetivo es, que aunque algún dispositivo de red o servidor se sobrecargue, se pueda seguir utilizando. Para ello también será efectivo añadir redundancia en nuestra red y usar protocolos como HSRP o RSTP para alta disponibilidad. También es importante la correcta configuración de los servicios que ofrezca el servidor.

Existen ciertas configuraciones del kernel de Linux que nos permitirá proteger nuestro servidor. Para ello modificaremos parámetros del fichero `/etc/sysctl.conf`, desde el cuál sirve para pasar configuraciones al kernel en tiempo de ejecución. Realizaremos estas modificaciones:

- **tcp\_syncookies:** para evitar ataques TCP SYN Flood. Cuando la cola de peticiones SYN se completa, el servidor responderá con un paquete SYN-ACK como hace normalmente, pero creando un número de secuencia codificado con la IP de origen, la IP de destino, el puerto y un timestamp. Así pues, la cola ya no será necesaria ya que podrá reconstruirse mediante el número de secuencia. Se activará esto mediante:

```
sysctl -w net.ipv4.tcp_syncookies=1
```

- **ignore\_broadcasts:** para evitar ataques SMURF. Se desactivará la respuesta a las peticiones broadcast de tipo echo ICMP. Se activa:

```
sysctl -w net.ipv4.icmp_echo_ignore_broadcasts=1
```

- **rp\_filter:** permite detectar el IP Spoofing comprobando que los paquetes que entran por una interfaz son alcanzables basándose en la dirección de origen. Se activa con:

```
sysctl -w net.ipv4.conf.all.rp_filter=1
```

Análogamente, se puede seguir la misma estrategia, realizando las configuraciones correspondientes en los routers de Cisco, basándose en los mismos principios que nos hemos basado antes.

- Para evitar ataques ataques TCP SYN FLOOD usaremos TCP intercept[21]. Es utilizado para interceptar y validar las solicitudes de conexión TCP. Se establece una conexión con el cliente en nombre del servidor de destino, si tiene éxito se establece una conexión en nombre del cliente y se unen las dos conexiones. Así, se evitan los intentos de conexión de hosts inalcanzables. Se utilizarán además umbrales de tiempo de espera para detectar conexiones ilegítimas.
- Para evitar ataques SMURF, seleccionamos la interfaz del router por la que pueden llegar estos ataques y escribiremos el siguiente comando:

no ip directed-broadcast

- Reverse Path Forwarding [22] para verificar que el origen de las peticiones es accesible. Evita las direcciones falsificadas, si la dirección de origen es falsa, el paquete se descarta.

## 5. Interfaz Web

### Análisis

El único actor que intervendrá en el sistema es el actor usuario. El objetivo principal de la construcción de la interfaz web es que un usuario pueda comprender las principales vulnerabilidades que existen en una red. De esta manera el usuario podrá navegar entre las distintas pestañas comprendiendo los ataques y analizando el tráfico de red que se genera. Por tanto, el mero objetivo es la enseñanza de la repercusión de estos ataques a un usuario.

Esta interfaz estará dirigida a un usuario con conocimientos sobre redes ya adquiridos y que sea capaz de comprender los principales protocolos y capas de red que existen. También, está dirigido para que un usuario con estos conocimientos adquiridos pueda interactuar con el sistema y mostrar, de una manera didáctica, la repercusión a otros usuarios que no poseen de los conocimientos básicos.

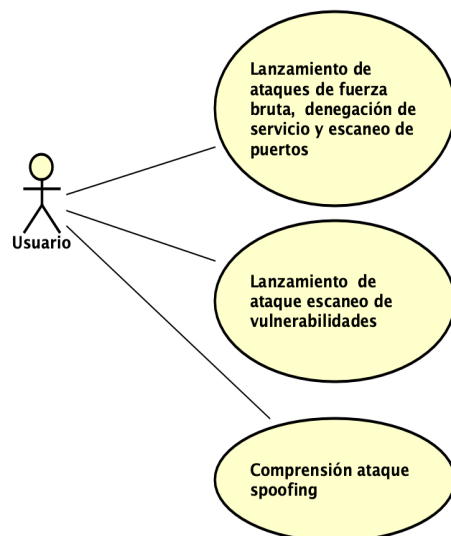


Figura 28: Diagrama casos de uso



| Caso de uso 1           |   |   |
|-------------------------|---|---|
| <b>Nombre</b>           | Lanzamiento de ataques de fuerza bruta, denegación de servicio y escaneo de puertos   |   |
| <b>Autor</b>            | Sergio Sanz Ferrero   |   |
| <b>Descripción</b>      | El sistema deberá comportarse tal y como se describe en la secuencia de acciones del caso de uso. Permite que un usuario pueda lanzar ataques y comprobar la repercusión que tiene sobre la red 157.88.123.102. |   |
| <b>Precondición</b>     | -   |   |
| <b>Secuencia Normal</b> | <b>Paso</b>   | <b>Acción</b>   |
|                         | 1   | El actor usuario selecciona una de las vistas de los ataques.   |
|                         | 2   | El sistema muestra una descripción de los ataques, cómo se lanzan los ataques y medidas de mitigación. Además, nos ofrece la posibilidad de lanzar el ataque desde la interfaz. |
|                         | 3   | El actor usuario selecciona la opción de lanzar ataque.   |
|                         | 4   | El sistema ejecuta remotamente un ataque que se lanza sobre la red y muestra el estado en el que se encuentra.  |
|                         | 5   | El actor usuario selecciona la vista del estado de la red y analiza el tráfico.   |

**Tabla 31:** Caso de uso 1

| Caso de uso 2           |  |  |
|-------------------------|--|--|
| <b>Nombre</b>           | Lanzamiento de ataque escaneo de vulnerabilidades  |  |
| <b>Autor</b>            | Sergio Sanz Ferrero  |  |
| <b>Descripción</b>      | El sistema deberá comportarse tal y como se describe en la secuencia de acciones del caso de uso. Permite que un usuario pueda lanzar un escaneo de vulnerabilidades y comprobar la repercusión que tiene sobre la red 157.88.123.102. |  |
| <b>Precondición</b>     | -  |  |
| <b>Secuencia Normal</b> | <b>Paso</b>  | <b>Acción</b>  |
|                         | 1  | El actor usuario selecciona la vista del escaneo de vulnerabilidades.  |
|                         | 2  | El sistema muestra una descripción del ataques, de cómo se lanzan los ataques y medidas de mitigación.   |
|                         | 3  | El actor usuario descarga el software de Nessus para lanzar un escaneo de vulnerabilidades.  |
|                         | 4  | El actor usuario lanza el escaneo de vulnerabilidades con los ajustes que desee contra la red 157.88.123.10 y obtiene los resultados del test. |
|                         | 5  | El actor usuario selecciona del estado de la red y analiza el tráfico.   |

**Tabla 32:** Caso de uso 2

| Caso de uso 3           |   |  |
|-------------------------|---|--|
| <b>Nombre</b>           | Comprensión de ataque spoofing  |  |
| <b>Autor</b>            | Sergio Sanz Ferrero   |  |
| <b>Descripción</b>      | El sistema deberá comportarse tal y como se describe en la secuencia de acciones del caso de uso. Permite a un usuario comprender en qué consiste los principales ataques en el interior de la red. |  |
| <b>Precondición</b>     | -   |  |
| <b>Secuencia Normal</b> | <b>Paso</b>   | <b>Acción</b>  |
|                         | 1   | El actor usuario selecciona la vista del spoofing.   |
|                         | 2   | El sistema muestra una descripción del ataques, de cómo se lanzan los ataques y medidas de mitigación. |
|                         | 3   | El actor usuario analiza comprende y analiza el tráfico generado por el ataque.                        |

**Tabla 33:** Caso de uso 3

La base de datos Elasticsearch contiene almacenado el tráfico de red que pasa por el firewall y por tanto, el tráfico generado por los ataques. El contenido de la base de datos se muestra en la figura 29.

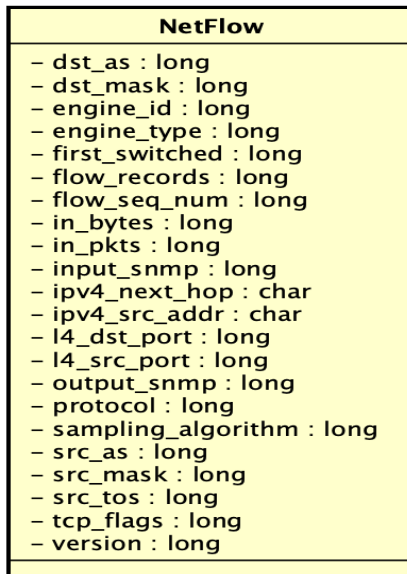


Figura 29: Base de datos

El diagrama de secuencia de análisis de lanzar cualquiera de los eventos y generar ataques queda representado en la figura 30.

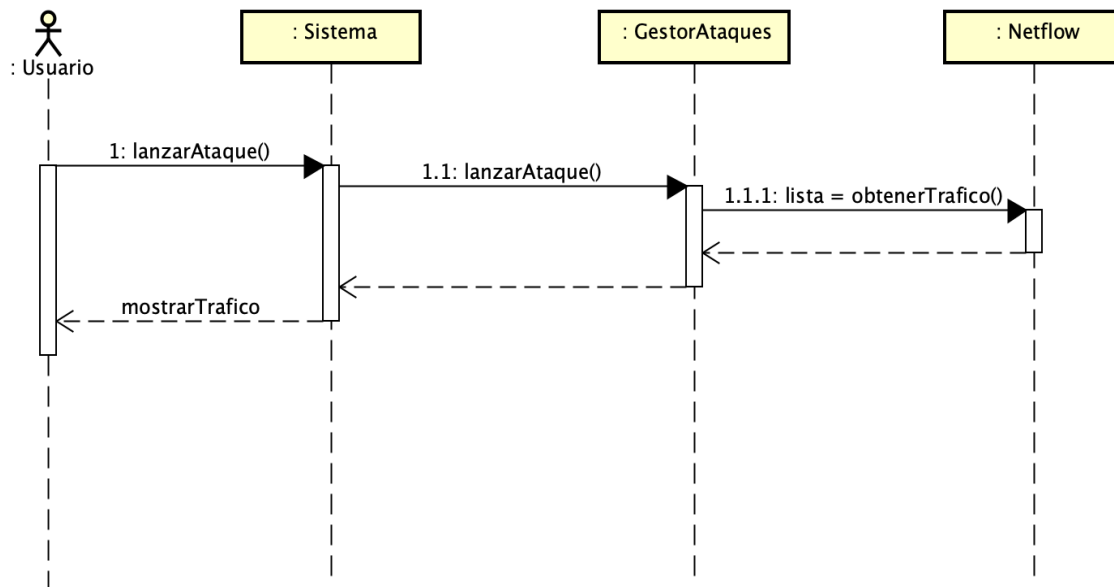
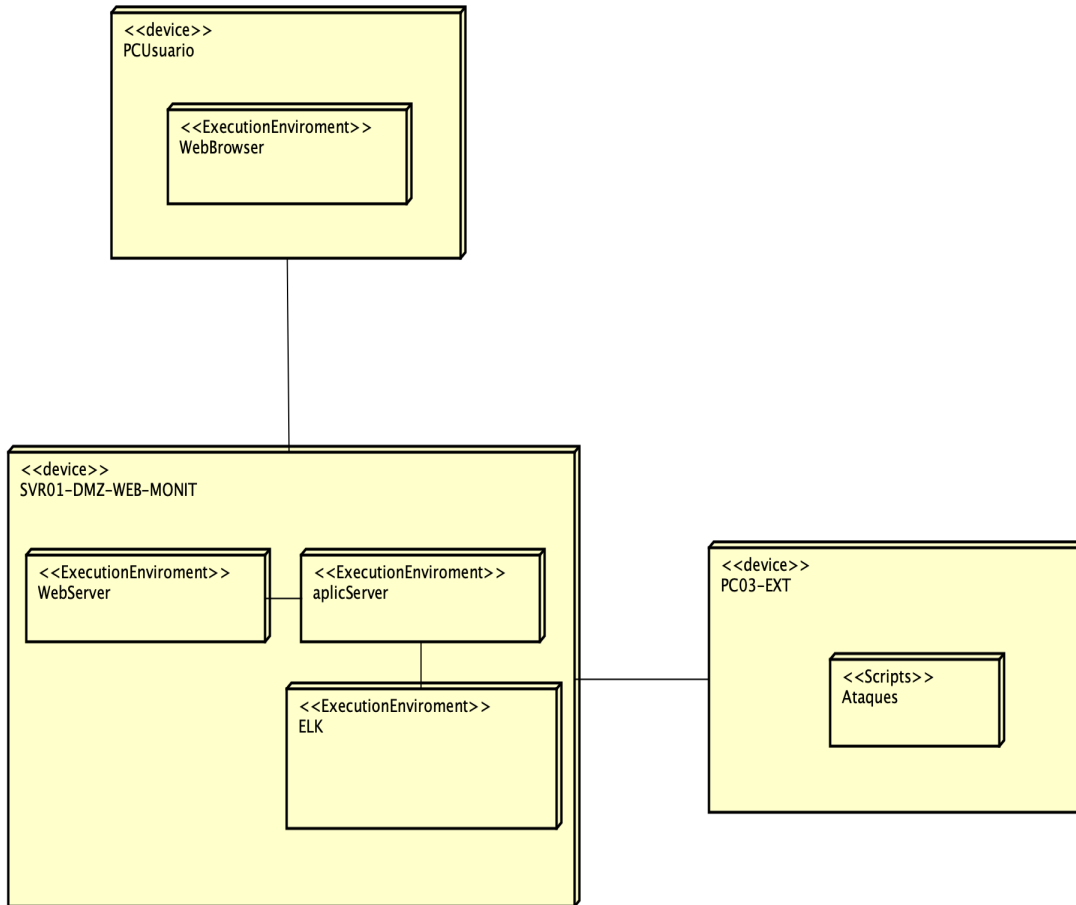


Figura 30: Diagrama de secuencia análisis

Como un análisis previo de la aplicación, nos interesa capturar el tráfico de los ataques que generemos. Es por ello que tendremos un gestor a través del cuál lanzaremos los ataques desde una máquina remota y recuperaremos el tráfico que se genera con su lanzamiento. Netflow contiene todos los parámetros que deseamos analizar.

## Diseño



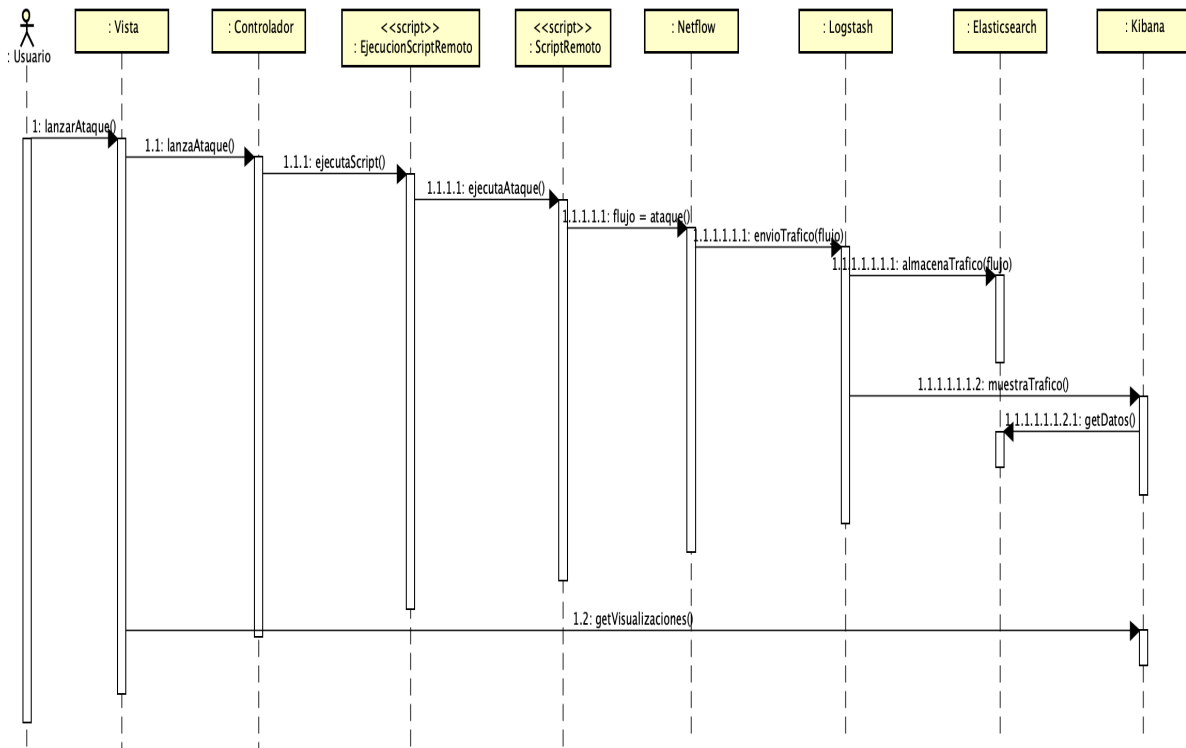
**Figura 31:** Diagrama de despliegue

El comportamiento general del sistema es el siguiente. El usuario utilizará un navegador para acceder a la interfaz web. Éste, a través de las descripciones, será capaz de comprender las principales vulnerabilidades. Cuando haya entendido en que consiste un ataque podrá lanzar el ataque desde la interfaz o descargándose el software que lo permite.

Si lo hace a través de la interfaz, llamará al controlador y ejecutará un script del sistema que ejecutará scripts remotos que lanzarán los ataques. Estos ataques se generarán contra la red 157.88.123.102 y todo el tráfico que pase por nuestro firewall perimetral externo será enviado al servidor de la DMZ mediante Netflow.

Logstash se encarga de hacer un preprocesamiento del tráfico y lo almacena en la base de datos Elasticsearch. Una vez almacenado, Kibana recogerá los datos de la base de datos y generará visualizaciones. Estas visualizaciones estarán embebidas en la interfaz web, donde el usuario puede acceder a ellas consultando la pestaña Estado de la red:

El diagrama de secuencia que representa el comportamiento de nuestro sistema y la infraestructura utilizada se muestra en la figura 32.



**Figura 32:** Diagrama de secuencia diseño

## 6. Apéndice

Un honeypot es una herramienta de seguridad informática cuya finalidad es hacer de señuelo para poder detectar ataques informáticos. Es útil para poder obtener estadísticas, y ver las consecuencias que se pueden sufrir.

En nuestro caso abriremos el puerto por defecto de SSH, el 22 y detrás tendremos una Raspberry Pi con usuario y contraseña por defecto. Analizaremos el número de intentos de conexión, las conexiones realizadas y las consecuencias.

Cuando queremos saber si nuestro equipo ha sido infectado, o si han logrado entrar en él, alguna de las recomendaciones para seguir son las siguientes:

- Consultar el fichero del directorio home `.bash_history` para ver los comandos que se han ejecutado como usuario
- Consultar el fichero `/root/.bash_history` para ver los comandos que ha ejecutado como superusuario
- Usar el comando `$last`, para poder ver los accesos al servidor, también es recomendable consultar el fichero `/var/log/auth.log` para poder comprobarlo.
- Usar el comando `$lastb`, para poder ver los accesos fallidos al servidor
- Usar el comando `$lastlog`, para poder ver las últimas conexiones de red
- Comprobar si se han creado cuentas de usuarios nuevas en `/etc/passwd`
- Ver las conexiones actuales usando `$netstat -tap`
- Ver puertos abiertos con `$ iptables -l`
- Ver `/var/log/cron.log` para comprobar si hay scripts o tareas que se ejecutan automáticamente cada cierto tiempo
- Comprobar `/etc/rc.local` para ver si se ejecuta algún script o tarea al iniciar el sistema
- Comprobar ficheros con bit SUID activado
- Comprobar `syslog`

Estos son algunos de los pasos básicos que debemos comprobar en cualquier sistema antes de utilizar otras herramientas que ayuden a realizar un análisis forense. En muchas ocasiones nos encontraremos con que estos logs han sido borrados y deberemos usar procedimientos más avanzados.

## Análisis

Tras realizar alguna de las sugerencias indicadas anteriormente, no encontramos nada relevante, no se habían creado usuarios, ni se había ejecutado ningún comando desde Bash. Así pues, primero buscaremos en las conexiones realizadas al equipo. Con el puerto 22 abierto y con acceso por SSH a nuestra Raspberry PI con usuario y contraseña por defecto analizaremos el tráfico concurrido.

Tras abrir el puerto, los intentos de conexión comenzaron prácticamente al instante. Podemos observar algunos de los intentos fallidos:

```
raspberrypi sshd[884]: Failed password for invalid user admin from 193.201.224.199 port 60927 ssh2
raspberrypi sshd[891]: Failed password for invalid user support from 193.201.224.199 port 22683 ssh2
raspberrypi sshd[904]: Failed password for invalid user user from 193.201.224.199 port 52874 ssh2
raspberrypi sshd[911]: Failed password for invalid user admin from 193.201.224.199 port 4187 ssh2
raspberrypi sshd[921]: Failed password for invalid user admin from 141.98.80.31 port 33776 ssh2
raspberrypi sshd[918]: Failed password for root from 193.201.224.199 port 28999 ssh2
raspberrypi sshd[938]: Failed password for root from 193.201.224.199 port 18137 ssh2
raspberrypi sshd[943]: Failed password for invalid user 0 from 193.201.224.199 port 57219 ssh2
raspberrypi sshd[957]: Failed password for invalid user admin from 193.201.224.199 port 25987 ssh2
```

Como vemos, se realiza un ataque de fuerza bruta a distintos puertos, con distintos usuarios. Los intentos comenzaron el 25 de Marzo a las 18:36:09 y acabaron el 26 de Marzo a las 10:14:42, momento en el que se desconectó de la red. En total se recibieron durante el tiempo que el puerto estaba abierto 1.768 peticiones.

Para ver los usuarios que han sido aceptados y que han logrado conectarse a nuestro equipo realizamos un filtrado de auth.log:

```
$cat auth.log|grep -a Accepted
```

```
Feb 18 13:03:12 raspberrypi sshd[688]: Accepted password for pi from 192.168.1.104 port 63442 ssh2
Feb 18 13:19:11 raspberrypi sshd[642]: Accepted password for pi from 192.168.20.10 port 64378 ssh2
Feb 18 16:21:47 raspberrypi sshd[602]: Accepted password for pi from 192.168.20.10 port 61009 ssh2
Feb 20 11:01:21 raspberrypi sshd[788]: Accepted password for pi from 192.168.1.120 port 58828 ssh2
Feb 20 11:26:18 raspberrypi sshd[736]: Accepted password for pi from 192.168.1.120 port 58943 ssh2
Feb 21 00:47:39 raspberrypi sshd[702]: Accepted password for pi from 192.168.30.10 port 52585 ssh2
Feb 21 00:53:46 raspberrypi sshd[758]: Accepted password for pi from 192.168.30.10 port 52692 ssh2
Feb 21 00:53:57 raspberrypi sshd[775]: Accepted password for pi from 192.168.30.10 port 52693 ssh2
Mar 12 19:12:44 raspberrypi sshd[1021]: Accepted password for pi from 157.88.123.214 port 58896 ssh2
Mar 19 16:15:29 raspberrypi sshd[863]: Accepted password for pi from 157.88.123.224 port 62925 ssh2
Mar 19 17:37:24 raspberrypi sshd[716]: Accepted password for pi from 192.168.30.10 port 51134 ssh2
Mar 19 17:52:04 raspberrypi sshd[769]: Accepted password for pi from 157.88.123.224 port 52876 ssh2
Mar 19 17:58:14 raspberrypi sshd[810]: Accepted password for pi from 157.88.123.224 port 54209 ssh2
Mar 25 16:27:33 raspberrypi sshd[669]: Accepted password for pi from 157.88.139.133 port 64006 ssh2
Mar 25 17:32:21 raspberrypi sshd[894]: Accepted password for pi from 157.88.139.133 port 49379 ssh2
Mar 25 18:32:35 raspberrypi sshd[725]: Accepted password for pi from 157.88.139.133 port 52053 ssh2
Mar 25 18:45:54 raspberrypi sshd[1255]: Accepted password for pi from 193.201.224.199 port 18100 ssh2
Mar 25 18:51:08 raspberrypi sshd[690]: Accepted password for pi from 157.88.139.133 port 52886 ssh2
Mar 25 18:53:39 raspberrypi sshd[677]: Accepted password for pi from 157.88.139.133 port 52968 ssh2
Mar 25 20:28:01 raspberrypi sshd[991]: Accepted password for pi from 82.64.37.252 port 51106 ssh2
Mar 25 20:28:04 raspberrypi sshd[1009]: Accepted password for pi from 82.64.37.252 port 51114 ssh2
```

Comprobamos que las IPs son conocidas, excepto dos. La primera es la 193.201.224.199 y

la segunda es la IP 82.64.37.252. Ambas se han conectado mediante el usuario pi. Ahora ya tenemos la certeza que han conseguido entrar en el sistema y que los ataques de fuerza bruta por SSH han tenido éxito.

Comprobamos la duración de cada conexión.

```
$cat auth.log|grep -a "sshd\[1255\]"
```

```
Mar 25 18:45:54 raspberrypi sshd[1255]: Accepted password for pi from 193.201.224.199 port 18100 ssh2
Mar 25 18:45:54 raspberrypi sshd[1255]: pam_unix(sshd:session): session opened for user pi by (uid=0)
Mar 25 18:45:55 raspberrypi sshd[1255]: pam_unix(sshd:session): session closed for user pi
```

La conexión tiene lugar el 25 de Marzo con una duración total de 1 segundo. Entre las 18:45:44 y las 18:45:45.

```
$cat auth.log|grep -a "sshd\[1009\]"
```

```
Mar 25 20:28:04 raspberrypi sshd[1009]: Accepted password for pi from 82.64.37.252 port 51114 ssh2
Mar 25 20:28:04 raspberrypi sshd[1009]: pam_unix(sshd:session): session opened for user pi by (uid=0)
Mar 25 20:28:07 raspberrypi sshd[1009]: pam_unix(sshd:session): session closed for user pi
```

La conexión tiene lugar el 25 de Marzo con una duración total de 3 segundos. Entre las 20:28:04 y las 20:28:07.



**Figura 33:** Número de intentos de conexión fallidos

El mayor número de peticiones provienen de la IP 112.85.42.237, y tienen lugar a partir de las 5:00:00 con una media de 320 peticiones por hora. A pesar de ello, esa dirección no ha logrado entrar en el sistema.

Sin embargo, la IP 193.201.224.199 que consiguió un conexión exitosa realizó únicamente 10 intentos fallidos. Además, la otra IP que consiguió entrar, no realizó ningún intento fallido de conexión.

## DetECCIÓN

Tras ver las intrusiones del sistema, era hora de ver que repercusiones había en el sistema. Se analizó el sistema en busca de archivos que se estuviesen ejecutando periódicamente mediante el demonio Cron, archivos con el bit SUID, y se miró en /etc/rc.local.

En las dos primeras tareas no encontramos nada, pero al analizar /etc/rc.local, observamos que se ejecutaba un script cada vez que se iniciaba el sistema.

El script ejecutaba un troyano que realizaba ataques de fuerza bruta a otros usuarios y finalmente enviaba los datos obtenidos a un servidor. Es un ataque típico que se realiza a Raspberrys con usuario y contraseña por defecto.[23]

Dado que había dos intrusos en el sistema, el siguiente paso era saber quién de los dos había insertado el troyano. Para ello analizamos las fechas de modificación de rc.local.

```
$stat /etc/rc.local
```

```
File: rc.local
Size: 34          Blocks:8          IO Block:4096      regular file
Device: b302h/45826d  Inode:761        Links: 1
Access: (0755/-rwxr-xr-x)  Uid: ( 0/      root)  Gid: ( 0/      root)
Access: 2019-11-13 14:21:16.861342118 +0000
Modify: 2019-03-25 20:28:05.689576955 +0000
Change: 2019-03-25 20:28:05.689576955 +0000
Birth: -
```

Por lo tanto, nuestra investigación finalizaría aquí. Comparando la fecha de modificación del fichero rc.local y la fecha de los ataques exitosos, comprobamos que a las 20:28:05 del día 25 de Marzo, la IP 82.64.37.252, estaba dentro del sistema y es la que ha introducido el malware.

## CONCLUSIÓN

Hemos visto la importancia de cambiar las contraseñas por defecto de los dispositivos que utilicemos, bien sean routers, equipos...

Con herramientas como Nmap podemos ver el sistema operativo que corre detrás de cada puerto. Por tanto, si nos hacen un ataque dirigido y contiene la contraseña por defecto, la intrusión es inmediata.



En este caso, lo más probable es que se trate de botnets que lancen ataques probando con los usuarios y las contraseñas por defecto de estos dispositivos y solo ha necesitado un intento para acceder al sistema.

Sin embargo, hemos visto también un ejemplo de fuerza bruta, en el que se usaba un diccionario de usuarios y contraseñas. No ha sido efectivo, por lo que muchas veces no se trata de lanzar indiscriminadamente peticiones, sino de poder analizar lo que puede haber detrás y de recopilar información acerca de la víctima.

## 7. Conclusiones y líneas futuras

### Conclusiones

En este trabajo se han abordado los principios básicos de la ciberseguridad. Se ha ocupado una gran parte de este proyecto aprendiendo nuevas tecnologías, herramientas y conceptos que no han sido tratados a lo largo del proceso formativo universitario.

Gracias a este proyecto he podido aprender conceptos nuevos, pero también he podido aplicar los conocimientos adquiridos en asignaturas como “Fundamentos de Redes de Computadoras”, “Diseño, Administración y seguridad de redes”, “Administración de Sistemas Operativos”, “Administración de Bases de Datos”, “Garantía y Seguridad de la Información”, “Diseño, Integración y Adaptación de Software”, “Servicios y sistemas web”, “Planificación y Gestión de Plataformas Informáticas”...

Desde el punto de vista de la administración de redes, se ha realizado un diseño de red ajustándose a las necesidades y al material disponible. Muchos de estos dispositivos poseen limitaciones que se tendrán que asumir en cualquiera de los proyectos que se realicen. Gracias a la construcción de la maqueta de red, se han podido realizar las conexiones físicas necesarias para el funcionamiento y se ha procedido a configurar cada uno de los dispositivos. Aunque no es una red de gran tamaño, los elementos básicos han quedado reflejados y se han podido cumplir los requisitos establecidos. Además, se han tenido en cuenta aspectos de diseño de redes seguras, como el uso de listas de control de acceso o el uso de un proxy, que se han podido aplicar para protegernos de amenazas externas.

Uno de los aspectos básicos a la hora de diseñar redes seguras es la monitorización de la infraestructura. Es por ello que se ha investigado acerca de cuál son las tecnologías más utilizadas en la actualidad. Una de las más utilizadas, mejor documentadas y más eficientes a la hora de almacenar grandes cantidades de datos es la pila ELK. Aunque la curva de aprendizaje es costosa al principio, nos ha permitido mostrar visualizaciones a tiempo real de nuestra red, por lo que el objetivo queda cumplido. Gracias a esta tecnología se han almacenado los logs, enviados a nuestro servidor mediante el protocolo Netflow, en una base de datos documental con un formato procesado que permite filtrar el tráfico por múltiples parámetros. De esta manera, hemos podido ver el tráfico que se genera en el servidor, los ataques externos que se reciben y el tráfico generado por los ataques que lanzamos mediante la interfaz web.

También, se ha realizado un trabajo exhaustivo investigando acerca de las principales vulnerabilidades y herramientas utilizadas en los tests de penetración a sistemas. Se ha realizado un análisis de los ataques más típicos que se pueden llevar a cabo, tanto los que provienen del exterior, como del interior de la red. Además de una descripción de los mismo, se han documentado y aplicado soluciones para mitigar estos ataques. Se ha analizado el tráfico que se genera al realizar estos ataques mediante la herramienta de monitorización implementada y así tener un enfoque práctico de lo estudiado.

Para ilustrar lo aprendido, se ha desarrollado una interfaz web para que un usuario pueda comprender la importancia de este campo dentro de las tecnologías de la información. Por tanto, los objetivos principales del proyecto han sido abordados.

Como conclusión final, realizar un análisis previo de la infraestructura que se desea atacar o defender para obtener información es fundamental en cualquier tarea de pentesting o mitigación. Como hemos visto realizar un ataque puede llevar muy poco tiempo y cada vez que ofrezcamos un servicio al exterior se empezarán a recibir ataques casi al instante. Podemos recibir ataques dirigidos, normalmente cuando se dispone de información crítica o confidencial, o ataques no dirigidos mediante el uso de botnets. Los botnets son equipos infectados por atacantes remotos que infectan otros equipos hasta tener cientos de equipos infectados. El tráfico generado en cualquier red por estos últimos es muy alto, por lo que debemos estar protegidos. Por eso, la concienciación de la ciberseguridad es una tarea que todos debemos llevar a cabo.

## Líneas futuras

Se proponen tres ramas diferentes para tener una visión más completa de la infraestructura y que permitirán comprender y analizar vulnerabilidades que no se han abordado en este proyecto.

Gran parte de la seguridad de nuestra red depende del firewall. En muchos de los casos se ha propuesto como mitigación el filtrado de paquetes a nivel de capa 3. Existen una gran cantidad de ataques que no hemos podido analizar, ya que se basan en el contenido de los paquetes HTTP hacia aplicaciones web y no podíamos filtrarlo con nuestros routers. Por tanto, como trabajo futuro y posible escalabilidad de nuestra red, se podría incorporar un WAF (Web application firewall). Permitirá analizar ataques tan comunes en la actualidad como SQL injections o XSS (Cross-site-scripting).

Durante todo el trabajo se ha mencionado el uso de sistemas de detección de intrusos (IDS) o sistemas de prevención de intrusos (IPS) para poder detectar y prevenir muchos de los ataques. Sería de utilidad la implementación, ya que es otra de las herramientas indispensables para la protección de redes. Esto nos permitirá detectar firmas de ataques conocidos y bloquear, en el caso de los IPS, basándose en el contenido de los paquetes de red y no solamente en el origen o destino de las comunicaciones.

Para la monitorización de la infraestructura se han recogido los logs del tráfico que pasaba por el firewall externo. Como trabajo futuro, se pueden recoger logs de diversas fuentes, centralizarlos y mostrar visualizaciones de estos logs. Para ello será necesario configurar Logstash para que admite diversos tipos de logs. Se podrán recoger logs generados por el servidor web, base de datos o cualquier otro servicio que esté activo, procesarlos y almacenarlo en la base de datos.

## 8. Bibliografía

- [1] Tridibesh Satpathy. “Cuerpo de conocimiento sobre Scrum (Guía Sbok)”. En: SCRUMS-study, 2016. Cap. 3, Organización, págs. 37-63.
- [2] Zachary burnham. *How to Install and Configure NGINX for Kibana*. 2019. Último acceso: 16-06-2019. URL: <https://burnhamforensics.com/2019/02/06/how-to-install-and-configure-nginx-for-kibana/>.
- [3] Cisco. *Cisco 1800 Series Integrated Services Routers: Cisco 1841 Router (Modular)*. 2019. Último acceso: 16-06-2019. URL: [https://www.cisco.com/c/en/us/products/collateral/routers/1800-series-integrated-services-routers-isr/product\\_data\\_sheet0900aecd8016a59b.html](https://www.cisco.com/c/en/us/products/collateral/routers/1800-series-integrated-services-routers-isr/product_data_sheet0900aecd8016a59b.html).
- [4] Cisco. *Cisco Catalyst 2950 Series Switches with Cisco Standard Image and Enhanced Image*. 2006. Último acceso: 16-06-2019. URL: [https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-2950-series-switches/prod\\_qas09186a008009258e.html](https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-2950-series-switches/prod_qas09186a008009258e.html).
- [5] Cisco. *Introduction to Cisco IOS NetFlow - A Technical Overview*. 2012. Último acceso: 14-05-2019. URL: [https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod\\_white\\_paper0900aecd80406232.html](https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html).
- [6] *Elasticsearch reference*. 2019. Último acceso: 14-05-2019. URL: <https://www.elastic.co/guide/en/elasticsearch/reference/6.7/index.html>.
- [7] *Logstash reference*. 2019. Último acceso: 14-05-2019. URL: <https://www.elastic.co/guide/en/logstash/6.7/index.html>.
- [8] *Logstash Netflow Module*. Último acceso: 16-06-2019. URL: <https://www.elastic.co/guide/en/logstash/6.7/netflow-module.html>.
- [9] *Kibana reference*. 2019. Último acceso: 14-05-2019. URL: <https://www.elastic.co/guide/en/elasticsearch/reference/6.7/index.html>.
- [10] *Working with Logstash Modules*. Último acceso: 16-06-2019. URL: <https://www.cisco.com/c/en/us/about/security-center/guide-ddos-defense.html#9>.
- [11] Rafay Baloch. “Ethical Hacking and Penetration Testing Guide”. En: CRC Press, 2015. Cap. 5, Vulnerability Assessment, págs. 121-133.
- [12] *Nessus reference*. 2019. Último acceso: 14-05-2019. URL: [https://docs.tenable.com/nessus/8\\_3/Content/ScanAndPolicyTemplates.html](https://docs.tenable.com/nessus/8_3/Content/ScanAndPolicyTemplates.html).
- [13] Borja Merino Febrero. “Análisis de tráfico con Wireshark”. En: Inteco-Cert, 2011. Cap. 4, Ataques en Redes de Área Local, págs. 12-33.
- [14] Cisco. *ARP Poisoning Attack and Mitigation Techniques*. 2016. Último acceso: 14-05-2019. URL: [https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-switches/white\\_paper\\_c11\\_603839.html](https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-switches/white_paper_c11_603839.html).

- [15] Nmap.org. *Nmap reference guide*. Último acceso: 14-05-2019. URL: <https://nmap.org/book/man.html>.
- [16] Owasp.org. *Brute force attack*. 2016. Último acceso: 14-05-2019. URL: [https://www.owasp.org/index.php/Brute\\_force\\_attack](https://www.owasp.org/index.php/Brute_force_attack).
- [17] Chris Binnie. “Practical Linux Topics”. En: Apress, 2016. Cap. 6, Securing SSH with PAM, págs. 51-61.
- [18] Andrew G. Morgan. *The Linux-PAM System Administrators’ Guide*. Último acceso: 14-05-2019. URL: [http://www.linux-pam.org/Linux-PAM-html/Linux-PAM\\_SAG.html](http://www.linux-pam.org/Linux-PAM-html/Linux-PAM_SAG.html).
- [19] Antonio S. Andreatos. “Hiding the SSH port via smart Port Knocking”. En: *Noryh Atlantic University Union, International journal of computers* 11 (2017).
- [20] Cisco. *A Cisco Guide to Defending Against Distributed Denial of Service Attacks*. Último acceso: 14-05-2019. URL: <https://www.elastic.co/guide/en/logstash/6.7/logstash-modules.html>.
- [21] Cisco. *Chapter: Configuring TCP Intercept (Preventing Denial-of-Service Attacks)*. 2014. Último acceso: 14-05-2019. URL: [https://www.cisco.com/c/en/us/td/docs/ios/12\\_2/security/configuration/guide/fsecur\\_c/scfdenl.html](https://www.cisco.com/c/en/us/td/docs/ios/12_2/security/configuration/guide/fsecur_c/scfdenl.html).
- [22] Cisco. *Understanding Unicast Reverse Path Forwarding*. Último acceso: 14-05-2019. URL: <https://www.cisco.com/c/en/us/about/security-center/unicast-reverse-path-forwarding.html>.
- [23] Tobias Olausson. *Raspberry Pi Trojan*. 2017. Último acceso: 14-05-2019. URL: <https://www.tobsan.se/update/2017/11/06/rpi-trojan.html>.
- [24] Raspberry. *Passwordless SSH access*. Último acceso: 16-06-2019. URL: <https://www.raspberrypi.org/documentation/remote-access/ssh/passwordless.md>.

## 9. Anexo I - Configuraciones

### R01-EXT-FWL

```
hostname router
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
mmi polling-interval 60
no mmi auto-configure
no mmi pvc
mmi snmp-timeout 180
ip subnet-zero
ip cef
no ip dhcp use vrf connected

interface FastEthernet0/0
 ip address 157.88.123.102 255.255.0.0
 ip flow ingress
 ip nat outside
 ip access-group 112 in
 duplex auto
 speed auto

interface FastEthernet0/1
 no ip address
 ip nat inside
 duplex auto
 speed auto

interface FastEthernet0/1.10
 encapsulation dot1Q 10
 ip address 192.168.10.1 255.255.255.0
 ip nat inside
 ip access-group 110 in
 ip access-group 113 out
 no snmp trap link-status

interface FastEthernet0/1.20
 encapsulation dot1Q 20
 ip address 192.168.20.1 255.255.255.0
 ip flow ingress
 ip nat inside
 ip access-group 110 in
 ip access-group 111 out
 no snmp trap link-status

interface Serial0/0/0
 no ip address
 shutdown
 no fair-queue
 clockrate 125000

interface Serial0/0/1
 no ip address
 shutdown

router ospf 1
 log-adjacency-changes
 network 157.88.0.0 0.0.255.255 area 1
 network 192.168.10.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 1
 default-information originate

ip classless
```

```

ip route 0.0.0.0 0.0.0.0 157.88.123.250
ip route 192.168.30.0 255.255.255.0 192.168.10.2
ip flow-export version 5
ip flow-export destination 192.168.20.10 2055

ip http server
ip nat inside source list 101 interface FastEthernet0/0 overload
ip nat inside source static tcp 192.168.20.10 22 157.88.123.102 22 extendable
ip nat inside source static tcp 192.168.20.10 80 157.88.123.102 80 extendable
ip nat inside source static tcp 192.168.20.10 8080 157.88.123.102 8080 extendable

access list 101 permit ip 192.168.10.0 0.0.0.255 any
access list 101 permit ip 192.168.20.0 0.0.0.255 any
access list 101 permit ip 192.168.30.0 0.0.0.255 any
access-list 101 deny ip any any

access-list 111 permit tcp any host 192.168.20.10 eq www
access-list 111 permit tcp 157.88.123.0 host 192.168.20.10 eq ssh
access-list 111 permit tcp any host 192.168.20.10 eq 8080

access-list 111 permit icmp 192.168.10.0 0.0.0.255 host 192.168.20.10
access-list 111 permit icmp 192.168.20.0 0.0.0.255 host 192.168.20.10
access-list 111 permit icmp 192.168.30.0 0.0.0.255 host 192.168.20.10
access-list 111 deny ip any any

access-list 112 deny ip 192.168.10.0 0.0.0.255 any
access-list 112 deny ip 192.168.20.0 0.0.0.255 any
access-list 112 deny ip 192.16830.0 0.0.0.255 any
access-list 112 deny ip 127.0.0.0 0.255.255.255 any
access-list 112 permit ip any any

access-list 113 permit tcp any any established
access-list 113 permit icmp any any echo-reply
access-list 113 permit icmp any any unreachable
access-list 113 permit udp any host 192.168.20.10 eq domain
access-list 113 deny ip any any

end

```

## R02-INT-FWL

```
hostname Router
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
mmi polling-interval 60
no mmi auto-configure
no mmi pvc
mmi snmp-timeout 180
ip subnet-zero
ip cef
no ip dhcp use vrf connected

interface FastEthernet0/0
no ip address
duplex auto
speed auto

interface FastEthernet0/0.10
encapsulation dot1Q 10
ip address 192.168.10.2 255.255.255.0
ip access-group 122 in
no snmp trap link-status

interface FastEthernet0/0.30
encapsulation dot1Q 30
ip address 192.168.30.1 255.255.255.
ip access-group 121 in
no snmp trap link-status

interface FastEthernet0/1
no ip address
duplex auto
speed auto

interface Serial0/0/0
no ip address
shutdown
no fair-queue
clockrate 125000

interface Serial0/0/1
no ip address
shutdown

router ospf 1
log-adjacency-changes
network 157.88.0.0 0.0.255.255 area 1
network 192.168.10.0 0.0.0.255 area 1
network 192.168.20.0 0.0.0.255 area 1

ip classless
ip http server

access list 121 permit ip 192.168.30.0 0.0.0.255 any
access-list 121 deny ip any any

access-list 122 deny ip 192.168.30.0 0.0.0.255 any
access-list 122 deny ip 127.0.0.0 0.255.255.255 any
access-list 122 permit ip any any

end
```



## SW03-INT-DMZ

```
version 12.1
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
hostname Switch
ip subnet-zero
spanning-tree mode pvst
no spanning-tree optimize bpdu transmission
spanning-tree extend system-id

interface FastEthernet0/1
  switchport mode trunk

interface FastEthernet0/2
  switchport access vlan 20
  switchport mode access

interface FastEthernet0/3
  switchport mode trunk

interface FastEthernet0/4
  switchport access vlan 30
  switchport mode access

interface FastEthernet0/5
  switchport access vlan 30
  switchport mode access

interface FastEthernet0/6
interface FastEthernet0/7
interface FastEthernet0/8
interface FastEthernet0/9
interface FastEthernet0/10
interface FastEthernet0/11
interface FastEthernet0/12
interface FastEthernet0/13
interface FastEthernet0/14
interface FastEthernet0/15
interface FastEthernet0/16
interface FastEthernet0/17
interface FastEthernet0/18
interface FastEthernet0/19
interface FastEthernet0/20
interface FastEthernet0/21
interface FastEthernet0/22
interface FastEthernet0/23
interface FastEthernet0/24

interface Vlan1
  ip address 192.168.125.33 255.255.255.0
  no ip route-cache
  ip http server

line con 0
line vty 0 4
  login
line vty 5 15
  login

end
```

# Índices Elasticsearch

```
"netflow" : {
  "properties" : {
    "dst_as" : {
      "type" : "long"
    },
    "dst_mask" : {
      "type" : "long"
    },
    "engine_id" : {
      "type" : "long"
    },
    "engine_type" : {
      "type" : "long"
    },
    "first_switched" : {
      "type" : "date"
    },
    "flow_records" : {
      "type" : "long"
    },
    "flow_seq_num" : {
      "type" : "long"
    },
    "in_bytes" : {
      "type" : "long"
    },
    "in_pkts" : {
      "type" : "long"
    },
    "input_snmp" : {
      "type" : "long"
    },
    "ipv4_dst_addr" : {
      "type" : "text",
      "fields" : {
        "keyword" : {
          "type" : "keyword",
          "ignore_above" : 256
        }
      }
    },
    "ipv4_next_hop" : {
      "type" : "text",
      "fields" : {
        "keyword" : {
          "type" : "keyword",
          "ignore_above" : 256
        }
      }
    },
    "ipv4_src_addr" : {
      "type" : "text",
      "fields" : {
        "keyword" : {
          "type" : "keyword",
          "ignore_above" : 256
        }
      }
    },
    "l4_dst_port" : {
      "type" : "long"
    },
    "l4_src_port" : {
      "type" : "long"
    },
    "last_switched" : {
      "type" : "date"
    }
  }
}
```

```

    },
    "output_snmp" : {
        "type" : "long"
    },
    "protocol" : {
        "type" : "long"
    },
    "sampling_algorithm" : {
        "type" : "long"
    },
    "sampling_interval" : {
        "type" : "long"
    },
    "src_as" : {
        "type" : "long"
    },
    "src_mask" : {
        "type" : "long"
    },
    "src_tos" : {
        "type" : "long"
    },
    "tcp_flags" : {
        "type" : "long"
    },
    "version" : {
        "type" : "long"
    }
}
},

```

## Proxy Nginx

```

server {
    listen 80;

    server_name 192.168.20.10;

    auth_basic "Restricted Access";
    auth_basic_user_file /etc/nginx/htpasswd.users;

    location / {
        proxy_pass http://localhost:443;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}

```

## Ejecución Remota de Scripts

```

#!/bin/bash
#Borrado de logs de la base de datos
curl -X DELETE http://localhost:9200/netflow*
#Ejecución de scripts remotos
case $1 in
    ("fuerzabruta.sh") /usr/bin/ssh -t pi@157.88.123.118 'bash /home/pi/fuerzabruta.sh';;
    ("dos.sh") /usr/bin/ssh -t pi@157.88.123.118 'sudo bash /home/pi/dos.sh';;&

```

```
        ("nmap.sh") /usr/bin/ssh -t pi@157.88.123.118 'sudo bash /home/pi/nmap.sh';;
        (*) echo "$1";;
esac
```

## Lanzamiento Fuerza Bruta

```
#!/bin/bash
rm hydra.restore
hydra -t 16 -l root -P diccionario.txt 157.88.123.102 ssh&
sleep 300
pkill hydra
```

## Lanzamiento Denegación de Servicio

```
#!/bin/bash
hping3 -S --flood -V -p 80 157.88.123.102
sleep 10
pkill hping
```

## Lanzamiento Escaneo de Puertos

```
#!/bin/bash
nmap -sS -A 157.88.123.102
```

## 10. Anexo II - Manual de Usuario

La plataforma desarrollada es un portal que permite ilustrar de manera didáctica las principales vulnerabilidades que presenta una red. Lo que pretende esta plataforma es que un usuario pueda comprender la repercusión que tienen estos ataques. Se incluye una descripción y la posibilidad de lanzar algunos ataques a la red para después ver el tráfico que se genera y entender su funcionamiento.

Las vulnerabilidades o métodos incluidos en cualquier penetración a un sistema que se explicarán son los siguientes:

- Escaneo de vulnerabilidades. Podremos encontrar una descripción de Nessus, uno de los software más utilizados para realizar este tipo de escaneos. También se incluye una pequeña descripción de Metasploit para aprovechar los resultados obtenidos. Se anima al usuario a que use la aplicación para descubrir las vulnerabilidades que presenta la red 157.88.123.102, descargándose el software y comprobando el tráfico que se genera en la vista que nos presenta el estado de la red.
- Fuerza bruta. Incluye una descripción del ataque a través de la herramienta Hydra. En este caso nos centraremos en realizar un ataque de fuerza bruta contra el servicio ssh.
- Denegación de servicio. El contenido se desarrolla en base a la herramienta Hping3 para realizar escaneos TCP SYN.
- Escaneo de puertos. Nos centraremos en la utilización de Nmap para realizar escaneos de puertos TCP.
- Spoofing. Al tratarse de un ataque que generalmente se realiza en el interior de la red, para posteriormente realizar un Man In The Middle, no podremos lanzarlo desde el exterior y no será monitorizado. Se incluye una explicación del ataque con una captura del tráfico generado recogido mediante la herramienta Whireshark.

### Vistas vulnerabilidades

Se utiliza una pestaña para explicar cada vulnerabilidad. En cada pestaña se incluyen tres apartados principales:

- Definición: se propone una descripción del ataque, en el que un usuario puede comprender el funcionamiento del mismo. Dentro de estos ataques generales, existen muchos tipos de ataques. En el caso de que haya más de un ataque, se explicarán los ataques en los que los intervengan los protocolos TCP y UDP.

# Vulnerabilidades

Estado de la red

Escaneo de vulnerabilidades

Fuerza Bruta

Denegación de servicio

Escaneo de puertos

Spoofing

## Escaneo de vulnerabilidades

Existen herramientas muy similares como OpenVas o Nessus que nos permiten hacer un escaneo automatizado de vulnerabilidades. En este caso nos centraremos en Nessus, aunque ambos son muy parecidos de utilizar.

Consideraremos este tipo de herramientas como una primera aproximación a explotar un sistema o a auditar nuestra propia seguridad. Aunque en el resultado de este escaneo no se detecten vulnerabilidades, no quiere decir que no existan. Debemos realizar análisis más complejos.

Los parámetros de configuración de un escaneo Nessus son los siguientes

- Basic: para especificar aspectos básicos organizativos, incluyendo nombre y descripción del escaneo.
- Discovery: para establecer el descubrimiento y la exploración de puertos, incluyendo los rangos y los métodos.
- Assessment: para identificar malware, vulnerabilidades de fuerza bruta, y la susceptibilidad de un sistema web.
- Report: el procesamiento y la salida del escaneo.
- Advanced: otros parámetros para hacer más eficiente un escaneo.

### Metasploit

Metasploit es un software gratuito y open-source que puede ser usado para automatizar tareas complejas. MSFConsole es la interfaz más popular de este framework y será con la que interactuemos para lanzar exploits aprovechando las vulnerabilidades.

Tras haber usado Nessus para realizar un escaneo de vulnerabilidades, Metasploit nos ofrece la posibilidad de lanzar exploits para explotar esas vulnerabilidades. El problema de lanzar todos estos exploits es que se generará mucho ruido y se podrá detectar fácilmente. Por ello, este tipo de ataques se suelen realizar cuando se dispone de poco tiempo o simplemente se quiere auditar la seguridad.

## Figura 34: Descripción

- Lanzamiento: se explica la herramienta utilizada para llevar a cabo el ataque. Se incluyen las opciones que hemos elegido para ser ejecutado contra nuestra red.

### Lanzamiento

Para lanzar el escaneo, lo primero que necesitaremos será disponer del software, el cuál se puede descargar desde la [página oficial](#).

Una vez instalado, seleccionaremos los plugings que deseemos. En este apartado tendremos una lista de vulnerabilidades, podremos habilitar familias de plugings o plugings individuales para llevar a cabo.

Una vez configurado todos los parámetros del escaneo, procedemos a definir cuál será nuestro objetivo, en este caso la IP 157.88.123.102 y a lanzarlo. Cuando haya finalizado el escaneo, podremos encontrar en reports las principales vulnerabilidades del sistema objetivo que hemos incluido. Podremos exportarlo en varios formatos. Si estamos realizando un test de penetración, es recomendable exportarlo en formato .nessus para después poder importarlo, por ejemplo, en Metasploit.

Para importar los resultados del escaneo a Metasploit usaremos MSFConsole, que es la interfaz más popular de Metasploit y escribiremos:

```
msf > db_import ruta/.nessus #importa los resultados de Nessus
msf > db_autopwn -x -p #lanzamiento de los exploits
msf > sessions -l #consultar sesiones obtenidas
```

De esta forma se lanzarán todos los exploits disponibles para explotar las vulnerabilidades encontradas, aunque esto generará mucho ruido. Si deseamos usar un exploit para explotar una vulnerabilidad concreta, a continuación mostraremos un ejemplo de como lanzarlo:

```
msf > search ssh #búsqueda de un exploit
msf > use auxiliary/scanner/ssh/ssh_login #selección de un exploit
msf auxiliary(ssh_login) > show options #muestra los parámetros
msf auxiliary(ssh_login)> set RHOSTS 157.88.123.102 #establecimiento de parámetros
msf auxiliary(ssh_login)> set USERPASS_FILE /usr/share/metasploit-framework/data/wordlists/root_userpass.txt # establecimiento de parámetros
msf auxiliary(ssh_login) > run #ejecuta exploit
```

## Figura 35: Lanzamiento

- Detección y defensa: se incluye una descripción de como detectar el ataque comprobando el estado de la red y viendo el tráfico que se genera al lanzarlo. También se

Se ofrecen varias soluciones para defenderse ante el ataque, pues no hay una manera completamente efectiva, en muchos casos deberemos combinar varias medidas o elegiremos una u otra dependiendo de las necesidades y de la situación en la que se encuentre una red.

#### DetECCIÓN Y DEFENSA

Podremos acceder a cada una de las vulnerabilidades, clasificadas según el impacto que puedan tener en el sistema. Se incluye una descripción, referencias a otros documentos de interés y soluciones para mitigarlas.

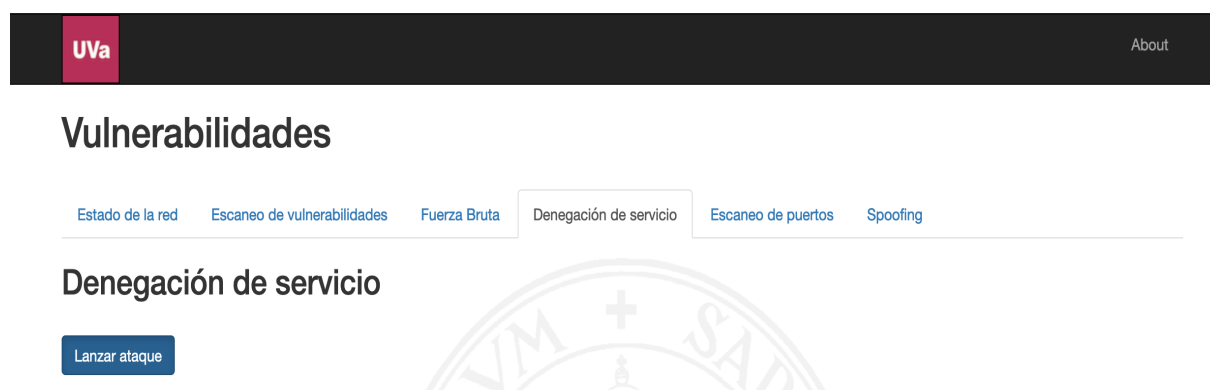
Es recomendable lanzar este tipo de escaneos cuando se haya diseñado y levantado una nueva red. Esto servirá tanto para ataque como para conseguir información de las vulnerabilidades de nuestra red y poder protegernos ante ellas.

Todo el tráfico generado por este tipo de escaneos tiene su origen en el puerto 80. Esto es porque herramientas como Nessus u Openvas son usadas mediante la interfaz web. No quiere decir que siempre sea así, si lo lanzásemos mediante Metasploit el origen de los puertos no sería el 80. Aún así es un indicio de que se trata de un escaneo de vulnerabilidades. Las peticiones que se envían son a múltiples puertos, aunque los puertos que se encuentran abiertos y presentan vulnerabilidades recibirán más tráfico que otros.

**Figura 36:** Detección y defensa

Los ataques en los que recibamos del exterior de la red y que pasen por nuestro firewall externo podrán ser lanzados a través del botón "Lanzar ataque", que se encuentra debajo del título de la vulnerabilidad, dentro de cada pestaña.

Esta función estará disponible para fuerza bruta, denegación de servicio y escaneo de puertos. No estará disponible para spoofing, ya que es un ataque que se produce en el interior de una red. Tampoco lo estará para el escaneo de vulnerabilidades, se anima al usuario a que se descargue el software para poder probar el ataque contra la red y poder familiarizarse con este tipo de herramientas.



**Figura 37:** Botón lanzamiento ataque

## Vista estado de la red

Pestaña en la que se puede comprobar el estado de la red y el tráfico que se genera al lanzar cada ataque. Se presenta de forma embebida la herramienta que hemos utilizado para la visualización de los logs almacenados en nuestra base de datos mediante Elasticsearch y que se generan cuando pasa tráfico por nuestro firewall externo.

Podremos encontrar dos elementos principales que nos permitirán visualizar el tráfico de una forma sencilla:

En el Discovery encontraremos todos los logs y el tráfico que se ha generado. Es útil para una visión global del estado de la red. Dentro de la vista podremos filtrar el tráfico por la IP que genera los ataques, que es la 157.88.123.102 y por cada uno de los campos que se encuentran en la columna de la izquierda.

Los Dashboards contienen visualizaciones y gráficos de diferentes campos del tráfico de red. Los Dashboards que podemos encontrar son los siguientes:

- **Overview:** contiene un resumen y una visión general del tráfico de red. Visualiza el origen de las peticiones además de los parámetros principales. También, podemos encontrar los logs del tráfico generado.
- **Conversation Partners:** visualización de los orígenes y destinos, además de un resumen de bytes y paquetes transmitidos.
- **Traffic Analysis:** permite identificar el volumen de bytes de las comunicaciones.
- **Top-N:** entre otros muchos parámetros muestra los puertos, orígenes y destinos o protocolos más usados.
- **Geo-Location:** visualización en un mapa del origen y destino. Muestra los países y ciudades que han generado el tráfico.
- **Autonomous System:** muestra las comunicaciones con sistemas autónomos.
- **Flow exporters:** bytes transmitidos por las interfaces de entrada y salida del firewall.
- **Raw Flow Records:** cantidad de flujos de Netflow transmitidos.





Figura 38: Estado de la red

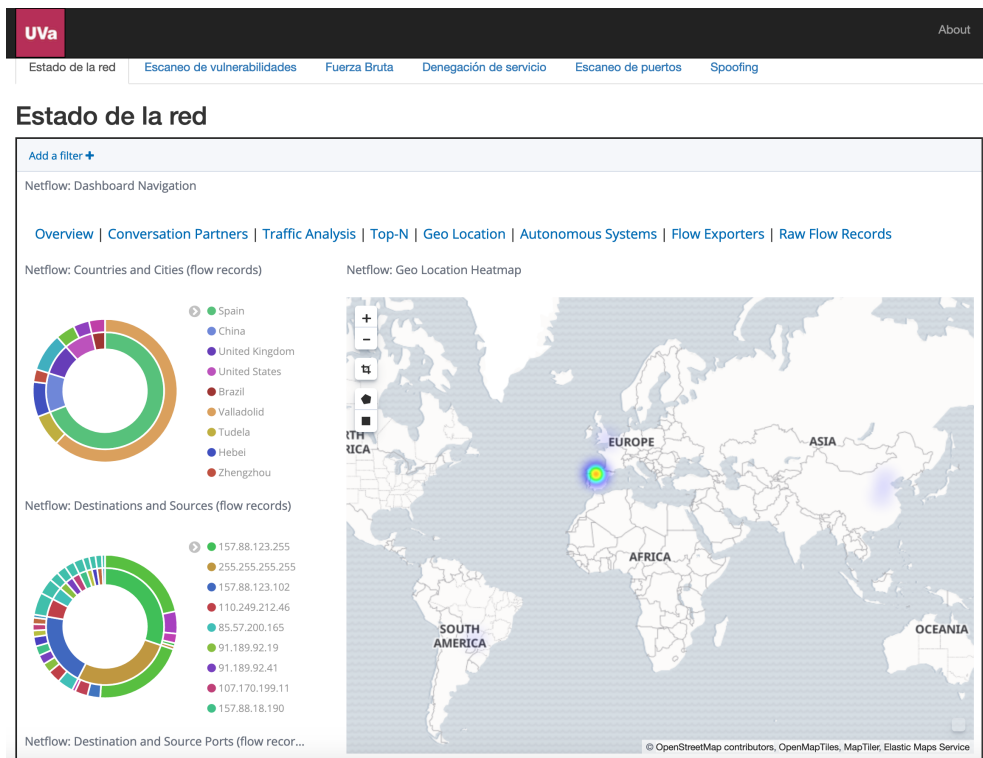
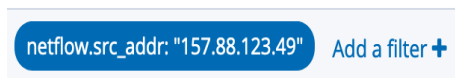


Figura 39: Estado de la red II

Para comprobar el estado de la red y poder aislar el tráfico que nos interesa del resto de tráfico, podemos añadir filtros por cualquiera de los parámetros. Esto nos permitirá analizar los ataques por separado y eliminar el ruido que nos entorpece el análisis. Podemos añadir el filtro pulsando directamente en cualquiera de los campos que nos encontremos o pulsando sobre la lupa con símbolo de suma que nos encontremos al lado de los parámetros. También podemos añadir el filtro directamente en el apartado Add a filter.



**Figura 40:** Filtro

Cuando colocamos el cursor sobre un filtro podemos realizar múltiples acciones. Siguiendo el orden el que aparecen en la figura 41, explicaremos su función:

- Deshabilitar el filtro. El filtro seguirá apareciendo pero no estará habilitado.
- Anclar un filtro para búsquedas posteriores.
- Excluir resultados. Kibana mostrará los resultados de todo menos del campo por el que estamos excluyendo.
- Eliminar el filtro. El filtro desaparece inmediatamente.
- Edición de filtro. Podremos modificar el filtro y la query para poder realizar otras opciones de filtrado como puede ser el uso de wildcards.



**Figura 41:** Edición de filtro

## 11. Anexo III - Manual de Instalación

En esta sección se explicará la estructura que de ficheros y los pasos a seguir para poder desarrollar este proyecto.

El proyecto se ha desarrollado en varias fases. A través de los ficheros disponibles en el repositorio <https://github.com/sergiosanzferrero/TFG>, podremos montar toda la infraestructura. Dentro del repositorio podemos encontrar dos directorios, el directorio que contiene la aplicación web y otro directorio con las configuraciones necesarias.

A continuación describiremos los pasos seguidos para conseguir el objetivo de este proyecto.

### 1. Dispositivos de red

En primer lugar necesitaremos configurar los dispositivos de red para poder ejecutar los ataques contra la misma y disponer de un servidor en el que se alojará la web y las herramientas de monitorización.

Podremos importar la configuración de los routers y de los switches. Dentro de los dispositivos, existen dos tipos de configuraciones, los `running-config` y los `startup-config`. Los `running-config` contienen la configuración que está corriendo en el dispositivo, mientras que los `startup-config` nos permitirán hacer esta configuración persistente, en caso de pérdida de alimentación o de un reinicio.

Para ello será necesario haber realizado las conexiones necesarias. El diagrama físico está disponible en la figura 8. Una vez realizado las conexiones podremos importar las configuraciones, aunque lo más sencillo es copiar y pegar la configuración disponible en los ficheros `running-config-dispositivo.txt` directamente al CLI y ejecutar el comando “`copy running-config startup-config`” para guardar la configuración.

### 2. Monitorización

Necesitamos tener un SO disponible. En el caso de este proyecto se ha utilizado Linux Ubuntu Desktop, ya que facilita la instalación y configuración de las herramientas.

Las herramientas de monitorización que usaremos serán Elasticsearch, Logstash y Kibana. Las podremos descargar de la página oficial, en la que se incluye documentación del proceso de instalación y de su uso <https://www.elastic.co/es/>.

Cada herramienta usada tiene un archivo de configuración. Estos archivos de configuración se encuentran en el directorio `config` de cada aplicación. Utilizaremos la configuración

guardada en los ficheros “elasticsearch.yml”, “logstash.yml” y “kibana.yml”.

Una vez tengamos importada la configuración, tendremos que ejecutar cada uno de ellos en el siguiente orden:

1. Elasticsearch, ejecutaremos `elasticserch` en el directorio bin de la aplicación.
2. Kibana, ejecutaremos `kibana` en el directorio bin de la aplicación.
3. Logstash, ejecutaremos `logstash` en el directorio bin de la aplicación con los parámetros “`-modules netflow -setup`”.

La eliminación de índices la llevamos a cabo mediante `crontab`, para ello configuraremos `crontab` para que se ejecute cada día el fichero “`borrado.sh`”.

Activamos el servicio `ssh`, abriendo el puerto 22 para poder realizar ataques de fuerza bruta contra la red y poder acceder remotamente al servidor.

Además, se utilizará `Nginx` [2] como proxy para poder acceder con autenticación a Kibana. Podremos añadir nuevos usuarios en `/etc/nginx/htpasswd.users` y editar la configuración, cuyo fichero está disponible en el repositorio, en `/etc/nginx/sites-available/default`.

### 3. Interfaz web

La estructura de la interfaz web es la siguiente:

- Front-end: dispondremos de un `index.html`, el cuál incluye una descripción de los ataques, una explicación de cómo se lanzará el ataque y medidas de mitigación ante diversas situaciones. Existe una pestaña llamada “estado de la red” que contiene la aplicación Kibana embebida para que tras pulsar uno de los botones de “Lanzar ataque” podamos comprobar el tráfico que se genera. Utiliza `HTML5`, `CSS` y la versión de `Bootstrap 3.3.7`.
- Back-end: dispondremos de un `Servlet`, desarrollado en `Java`, que será el Controlador de nuestra aplicación. Cuando se pulsa uno de los botones de “Lanzar ataque” se generará un ataque contra nuestro red.

A continuación se describirá el desarrollo del lanzamiento del ataque y los ficheros necesarios para su despliegue:

Tras pulsar el botón “Lanzar ataque” se inicializa el controlador que ejecutará un script almacenado en el servidor. Este script se llama “`EjecucionScriptRemotos.sh`”, el cuál necesita permiso para poder ser ejecutado. Este script ejecutará remotamente los scripts

almacenados en la máquina que realizará los ataques, que es PC03-EXT. Esta máquina tiene instalado el sistema operativo Raspbian y tiene instaladas las herramientas Hydra, Hping3 y Nmap. Los scripts almacenados ejecutarán los ataques sobre la red 157.88.123.102. Los scripts que disponemos para lanzar los ataques son “fuerzaBruta.sh”, “dos.sh”, “nmap.sh”.

Para que la comunicación entre el servidor web y la máquina que realiza los ataques sea efectiva necesitaremos configurar el servicio ssh. Para poder ejecutar los scripts remotamente tendremos que configurar la conexión entre ambas máquinas para que se realice sin necesidad de contraseña, utilizando RSA [24], es decir, clave pública y clave privada.

Una vez que hayamos realizado la configuración, el ataque se hará sobre la red y será el firewall externo el que enviará los logs mediante Netflow al servidor de monitorización que estará ejecutando instancias de Elasticsearch, Logstash y Kibana. Kibana estará accesible en `lar.infor.uva.es` y podremos acceder solo mediante autenticación.

La página web esta almacenada en el mismo servidor de monitorización. Para alojarla se utiliza el servidor Apache Tomcat8.5.42. Una vez instalado el servidor se construye la aplicación en formato “.war” que es el que se situará en el directorio webapps de Tomcat. Tras haber realizado todos los pasos, la aplicación estará disponible en: `lar.infor.uva.es:8080/TFG`

## 12. Anexo IV- Contenido del CD-ROM

El contenido del CD es el siguiente:

- Código fuente: el directorio TFG contiene los ficheros fuente de la aplicación web desarrollada.
- Versión de instalación de la aplicación: TFG.war
- Configuraciones: directorio que contiene los archivos de configuración de los dispositivos de red, de las herramientas de monitorización utilizadas y de otros archivos de configuración necesarios para la instalación de la infraestructura. Además, se incluyen los scripts utilizados para los lanzamientos de ataques remotos. Estos ficheros también se adjuntan en los anexos de esta memoria.
- Memoria: contiene una copia en formato digital de esta memoria. En los anexos de esta memoria se incluyen tanto el manual de usuario, como el manual de instalación.