UNIVERSIDAD POLITECNICA DE VALENCIA

ESCUELA POLITECNICA SUPERIOR DE GANDIA

Grado en Ing. Sist. de Telecom., Sonido e Imagen

“Desarrollo e implementación práctica de un PENTEST”

TRABAJO FINAL DE GRADO

Autor/a:
Rafael Manuel Martí Talón

Tutor/a:
Jaime Lloret Mauri

GANDIA, 2016

Resumen

Este Trabajo de Fin de Grado pretende proporcionar una visión general de las fases que debe tener una auditoría de seguridad informática profesional, así como enumerar las herramientas más utilizadas en cada fase. Para ello se parte del *Penetration Testing Execution Standard* ampliándolo utilizando fuentes actuales y contrastadas. Además, incluye una sección con lugares donde practicar el pentest de forma gratuita y otra en la que se realiza una pequeña auditoría para poner a prueba la metodología y herramientas nombradas contra máquinas virtuales vulnerables disponibles en la red.

Palabras clave

Hacking Ético, Pentest, Seguridad informática

Abstract

This final degree project aims to provide a general vision of the phases that a professional penetration test must have and enumerate the most used tools for each phase. In order to do that, it is based on the *Penetration Testing Execution Standard* and extends it using current and contrasted sources. In addition, it also includes a section with places where the pentest can be practiced with no cost. Furthermore, it also includes a section with a small pentest to test the methodology and tools against vulnerable virtual machines found on the Internet.

Key words

Ethical Hacking, Pentest, Security

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN

1.1 RELEVANCIA DEL TEMA

Cuando hablamos de auditoría de seguridad, test de penetración o pentest nos referimos al conjunto de actividades emprendidas con el objetivo de identificar y explotar vulnerabilidades en una red o sistema, con el objetivo de poner a prueba los sistemas de seguridad existentes, para así poder subsanar los fallos encontrados antes de que sean aprovechados por un atacante no autorizado.

La seguridad informática es un tema que está de plena actualidad por casos como la filtración de los papeles de Panamá, debida supuestamente a una intromisión no autorizada en el servidor de correo de un bufete de abogados [1], así como por otros ocurridos en años anteriores como las filtraciones de WikiLeaks [2], el caso Snowden [3] o el ataque a Sony Pictures [4].

Según el Instituto Nacional de Ciberseguridad en su recopilación de titulares de ciberseguridad de 2015 [5], el cibercrimen y las fugas de información han sufrido un marcado aumento desde principios de 2014, como puede verse en la figura 1 bajo estas líneas y, además, señala que esta tendencia al alza va a mantenerse. El aumento de titulares relacionados con la seguridad en la prensa generalista indica un aumento en el número y la gravedad de los ciberataques que no puede ser pasada por alto. Por consiguiente, consideramos de gran importancia la realización de este trabajo sobre la auditoría de seguridad informática.

Figura 1: Aumento de titulares sobre ciberataques desde 2014

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 e incluso así tengo mis dudas" es importante dedicar los recursos necesarios para la protección de los datos e infraestructuras de una organización mediante la realización de auditorías de seguridad de forma periódica.

1.2 OBJETIVOS

Este trabajo no pretende de ningún modo ser una guía definitiva sobre pentest, solo pretende esbozar las fases que toda auditoría de seguridad profesional debe tener y algunas de las herramientas y técnicas más conocidas para la consecución de cada fase.

Así pues, en el presente trabajo abordaremos el pentest o auditoría de seguridad desde una perspectiva profesional y lo más cercana posible a como la realizaría una

empresa real usando como base el *Penetration Testing Execution Standard* y ampliándolo usando siempre fuentes de información contrastadas y actuales.

Durante la exposición de las fases del pentest comentaremos, sobre todo, herramientas gratuitas y de código abierto disponibles en la distribución Kali Linux, aunque también dedicaremos una sección a programas disponibles en otros sistemas operativos y otra a lugares donde practicar el pentest de forma gratuita.

Por último, realizaremos una pequeña auditoría en la que aplicaremos los conocimientos adquiridos utilizando la metodología y las herramientas nombradas para comprobar si funcionan efectivamente en la práctica.

1.3 PRECEDENTES DEL PROYECTO

Dada la popularidad del tema -y contrariamente a lo que pudiera parecer- no hemos encontrado muchos TFC o TFG relacionados con la seguridad informática ni con la auditoría de seguridad en nuestra universidad. Los dos únicos trabajos que tratan el tema directamente son “Auditoría de Seguridad Informática” escrito por Antonio Alegría en 2009 y "Auditorías de seguridad informática y la OSSTMM" escrito por Alberto Hervalejo en el mismo año, de los cuales solo hemos podido acceder al segundo.

Este último trata el tema de la auditoría de seguridad centrándose principalmente en el *Open Source Security Test Methodology Manual* de ISECOM y añade comentarios para ayudar al lector a comprenderlo mejor.

Sí ha habido otros trabajos que tratan de forma marginal el tema o se centran en un apartado específico como la seguridad inalámbrica. Un ejemplo sería el de Juan Martí titulado “Seguridad de redes de área local inalámbrica en entornos corporativos” en 2005 o el de Celine Ballester “Seguridad en redes inalámbricas” en 2003, ambos tutorizados por Jaime Lloret, que también dirige el presente trabajo.

Como puede verse, todos los trabajos nombrados tienen una antigüedad superior a los siete años, por lo que consideramos importante la realización de este TFG que pretende proporcionar conocimientos y fuentes actuales en un campo que cambia tan rápidamente como el de la seguridad.

1.4 ESTRUCTURA DEL PROYECTO

En el siguiente capítulo del proyecto explicaremos nuestra elección del PTES para la estructura de la parte teórica de este trabajo y definiremos las siete fases de un pentest y las herramientas más comúnmente utilizadas para cada una de ellas en el SO Linux.

En el tercer capítulo, hablaremos de las herramientas disponibles exclusivamente en Windows y MacOS así como de recopilaciones de herramientas disponibles para que estos sistemas puedan usarse como plataformas de pentest.

En el cuarto, comentaremos algunos sitios web en los que encontrar recursos para el aprendizaje práctico del pentest mediante el uso de máquinas virtuales tanto en línea como fuera de línea.

En el quinto capítulo, realizaremos un pequeño pentest dado que no ha sido posible realizar uno más exhaustivo debido a la propia complejidad del tema y a la limitación de espacio, en él seguiremos las fases y herramientas definidas anteriormente.

En el último capítulo del trabajo, extraeremos conclusiones de los resultados obtenidos en la quinta parte y evaluaremos si se han cumplido los objetivos.

2. FASES DE UN PENTEST Y HERRAMIENTAS EN LINUX

Como hemos comentado anteriormente organizaremos la parte teórica de este TFG basándonos en las siete fases definidas en el *Penetration Testing Execution Standard*. Hemos elegido esta metodología debido a que ha sido redactada por algunos de los mayores expertos del sector. La mayoría de ellos son CEO de su propia empresa y ponentes habituales en las más importantes convenciones de seguridad como BlackHat o DEFCON. Además, creemos que la terminología usada no es tan compleja como la utilizada en otras metodologías por lo que resulta fácil de entender y aplicar. A continuación, comenzaremos con la primera de las fases, la comunicación previa con el cliente.

2.1 COMUNICACIÓN PREVIA

El objetivo de esta sección del PTES es explicar los diferentes aspectos que deben contemplarse para que la comunicación inicial con el cliente sea exitosa.

En primer lugar, previamente a la realización del pentest debemos establecer una comunicación inicial con el cliente y entender en profundidad los objetivos que persigue al realizarlo. Para ello es muy útil tener preparados unos cuestionarios que permitan un primer acercamiento a las necesidades del cliente. Varios ejemplos de dichos cuestionarios se encuentran en Anexos.

Cada cliente tiene unas necesidades específicas y lo que es crítico para un cliente puede no serlo para otro. Por ejemplo, a una tienda online pasar horas fuera de línea le puede significar la pérdida de miles de euros en ventas. Sin embargo, para un banco, sería mucho más grave la pérdida y divulgación de una base de datos de tarjetas de crédito.

Hay varios temas en los que auditor y cliente deben ponerse de acuerdo durante la comunicación inicial los cuales trataremos a continuación.

Normalmente, el primer documento que veremos durante una negociación contractual es el acuerdo de confidencialidad, este es necesario para proteger la privacidad de cualquier información que consigamos durante la auditoría. Al firmarlo, nos comprometemos a mantener la confidencialidad de los datos del cliente, durante y también después de la auditoría, hasta el momento de su destrucción. Este acuerdo incluye cualquier tipo de información relacionada con el cliente, desde capturas de pantalla, pulsaciones de teclas, correos intercambiados, hasta planes de marketing o negocio, información financiera y cualquier dato remotamente relacionada con la auditoría.

Además de comprometernos a no revelar información a terceros, en este documento también se acuerda guardar toda la información obtenida bajo llave, ya que sería desastroso si alguien entrara en nuestros sistemas y descubriera los detalles para acceder a la red del cliente. [6]

Asimismo, definir el alcance es una de las partes más importantes de un test de penetración, y también una de las más obviadas. Descuidar la correcta realización de la comunicación inicial puede provocar al auditor o a su empresa muchos problemas incluyendo el *scope creep* (aumentos no controlados en el alcance del proyecto), clientes insatisfechos o incluso problemas legales. De este modo, el alcance de un proyecto define específicamente qué se va a probar y qué no.

Es común encontrar clientes que no saben exactamente qué necesitan auditar o que no son capaces de comunicar de manera efectiva qué esperan de la auditoría. Por ello,

durante la comunicación inicial es importante que el auditor guie al cliente en los detalles de esta. Por consiguiente, será necesaria una reunión con el cliente para definir el alcance. Otros aspectos, como el coste y el método utilizado, serán tratados en reuniones aparte.

Para empezar, es necesario establecer explícitamente que rangos de direcciones IP están dentro del alcance del test. Por ejemplo, en la reunión debería verificarse que el cliente posee todos los sistemas a auditar como el servidor DNS, el servidor de e-mail, el servidor en el que están alojadas sus páginas web o su sistema de *firewall*/IDS/IPS. Hay muchas compañías que externalizan estos servicios y sería necesario un permiso por escrito de la compañía propietaria del equipo para realizar pruebas en ellos.

Además, se debe tener en cuenta en qué lugar se encuentran los sistemas que se van a auditar, ya que la ley varía de unas zonas geográficas a otras y el test puede cambiar en función de ello. Por ejemplo, en los países pertenecientes a la Unión Europea es común la existencia de leyes muy estrictas respecto a la privacidad de los individuos y esto podría alterar de manera significativa la forma en que se realizaría una prueba de ingeniera social. [7]

Por otro lado, mientras que el alcance define qué probaremos, las reglas del test definen como se va a realizar el proceso. Estos son dos aspectos diferentes que deben tratarse por separado.

Es esencial establecer una línea de tiempos clara, definir fechas para el inicio y finalización del test y también planificar en que se invertirá exactamente el tiempo. Tener una línea de tiempo bien definida al principio del test ayudará a todos los involucrados a identificar qué se debe hacer y quién será responsable de cada parte. Además, si la organización tiene una gran cantidad de ubicaciones a auditar, es necesario determinar las ubicaciones a las que deberá desplazarse el auditor durante el proceso. [7]

Otro aspecto importante de la comunicación previa es el tiempo estimado de realización de la auditoría. Este está directamente relacionado con la experiencia del auditor, pues si tiene experiencia le será más fácil determinar cuánto llevará realizarlo. En caso contrario, una buena opción sería releer correos electrónicos y registros de test similares que haya realizado la empresa para poder hacer una aproximación. Una vez la estimación haya sido calculada es prudente añadir un 20% a ese tiempo.

Este tiempo extra -llamado *padding*- es absolutamente necesario en cualquier auditoría. Proporciona un margen de seguridad en caso de que ocurra alguna interrupción durante el proceso, bien sea esta una caída de red o el descubrimiento de una vulnerabilidad que requiera una solución inmediata. [7]

Además, el auditor debe adaptar la realización del test al horario establecido por el cliente (ya que puede querer realizarlo en determinada franja horaria o ciertos días de la semana, por ejemplo, fuera de horas de trabajo o en fin de semana). [8] [7]

Otro aspecto muy importante a tratar es el contacto con el cliente. En este sentido, debe quedar claro cada cuanto tiempo se informará de los avances, qué tipo de comunicación y de encriptado se utilizará, etc. También es imprescindible un contacto de emergencia para informar al cliente de cualquier vulnerabilidad crítica encontrada. [7]

En otro orden de cosas, los ataques de ingeniería social son ampliamente usados por los atacantes aprovechando pretextos de lo más variopintos. El auditor debe asegurarse de que todos aquellos pretextos que pretenda utilizar sean aceptables en un

entorno corporativo. Por ello, debe contar con la aprobación por escrito de la empresa para usarlos. [7]

Asimismo, la realización de un ataque de denegación de servicio (DoS) o de denegación de servicio distribuido (DDoS) debe ser tratada con el cliente antes del principio de la auditoría debido a que es un tema que preocupa a las organizaciones por su peligro intrínseco.

Si la organización está preocupada por la disponibilidad de sus servicios el ataque deberá ser realizado en un entorno de pruebas que sea idéntico al entorno de producción. Sin embargo, no es necesario realizar la prueba si la organización sólo está preocupada por la confidencialidad e integridad de sus datos. [7]

Otro aspecto importante a tener en cuenta durante una auditoría es la protección de los datos, ya que durante ella se consigue información de naturaleza delicada. Por este motivo es importante que los datos obtenidos estén bien protegidos. La encriptación de los datos es una de los primeros pasos que debemos tomar para su protección, así, debemos encriptar siempre los discos duros que usemos y también usar la contraseña del BIOS para proteger los datos en caso de robo o acceso no autorizado.

Otra buena práctica es guardar los discos duros en una caja fuerte siempre que sea posible, ya que además de protegerlos contra robo también resultaran protegidos contra incendios y otros incidentes, estas medidas también pueden aplicarse a portátiles. Además, un laboratorio de auditorías debería estar en una habitación separada con controles de seguridad para restringir el acceso a personal no autorizado. [6]

En otro orden de cosas, es muy importante que el pentest no empiece hasta que no haya sido firmado el permiso para realizarlo. Este documento determina el alcance del proyecto y contiene las firmas que demuestran que el cliente conoce y aprueba las actividades del auditor. Además, debe especificar que el proceso puede generar inestabilidad en los sistemas, pero que el auditor pondrá el máximo cuidado posible en no derribar ningún sistema durante el test. Sin embargo, en caso de error, el cliente no hará responsable de ello al auditor.

Otro aspecto que debe ser tratado antes de iniciar el test es el pago. Debe quedar claro y recogido por escrito en qué fechas y condiciones se realizará. [7]

Una vez tratados todos los puntos anteriores y firmados los documentos pertinentes podremos empezar la siguiente fase dedicada a la recogida de información.

2.2 RECOGIDA DE INFORMACIÓN

La primera fase de cualquier test de penetración es la recogida de información, en ella se pretende obtener la máxima cantidad de información posible sobre el objetivo, para utilizarla en las fases posteriores ya que cuanta más información se consiga en esta fase más posibles vectores de ataque tendremos disponibles. [7]

Podemos ampliar esta definición añadiendo que el objetivo del *Footprinting* es pasar de un objetivo en el mundo real (una empresa, corporación, gobierno u otra organización) a un objetivo en el mundo virtual, es decir, una lista de direcciones IP relevantes y accesibles. [9]

En el presente apartado nos basaremos en la estructura proporcionada por [10] que a su vez se basa en [7].

Normalmente, la recogida de información se divide en dos partes, en la primera se recoge información de forma externa a la organización por lo que es denominada *External Footprinting*, la segunda parte será realizada cuando se haya obtenido acceso a la red de la organización y nuestro objetivo será obtener la máxima cantidad de información posible para así poder continuar nuestro ataque, esta parte que trataremos en la fase de post-explotación es llamada *Internal Footprinting.* [10]

La parte de *External Footprinting* a su vez puede dividirse en dos partes según la agresividad de los métodos usados para obtener la información, si no hay contacto directo con la empresa auditada se considerara que el método es pasivo y si hay contacto directo que es activo.

Una de las técnicas más ampliamente utilizadas para el *Passive Footprinting* es el método OSINT (Open Source Intelligence) que se basa en la obtención de conocimiento desde fuentes de acceso público. [11]

Los autores de [7] amplían y concretan esta definición añadiendo que el método OSINT es usado para encontrar los puntos de entrada a una organización. Estos puntos de acceso pueden ser físicos, electrónicos o humanos ya que muchas organizaciones o sus empleados no son conscientes de la información que hacen pública ni de cómo esta puede ser usada por un atacante.

No obstante, aunque sea el método más usado, no está exento de inconvenientes porque, como señala [7], el principal problema de OSINT es que la información obtenida puede no ser exacta; las fuentes de información pueden ser manipuladas deliberada o accidentalmente para proporcionar información errónea; la información puede volverse obsoleta con el paso del tiempo o, simplemente, estar incompleta.

Del mismo modo, [11] coincide en la importancia de valorar las fuentes que utilizamos ya que una elección equivocada puede tener como consecuencia resultados erróneos y desinformación. Y, además, señala otro posible problema, este surge a raíz de la cantidad de información disponible en la red ya que, si disponemos de una gran cantidad de datos, pero no de los medios para examinarlos se produciría una ralentización del proceso, por lo cual es vital seleccionar las fuentes de información correctas.

Para la realización del *Passive Footprinting* contamos con infinidad de herramientas. A continuación, hablaremos someramente de algunas de las más usadas.

Normalmente, el primer sitio al que se debe acudir para recoger información es la web de la propia empresa a auditar. En algunos casos, puede ser útil usar la herramienta HTTrack recomendada por [9] y [12] para crear una copia idéntica pero fuera de línea de dicha página web. La copia incluirá todas las páginas, enlaces, imágenes y código de la web original. Esto nos permite explorar y recoger información fuera de línea sin perder tiempo esperando las respuestas del servidor web.

También, es importante saber que cuanto más tiempo se pase navegando por el sitio web más probable es que la actividad sea registrada o monitorizada. Además, al clonar un sitio web es difícil pasar desapercibido debido a la cantidad de peticiones realizadas al servidor.

Bien realicemos una copia de la web o simplemente estemos navegando por ella, es importante prestar atención a los detalles. Con frecuencia, se pueden encontrar, sin mucho esfuerzo, números de teléfono, direcciones de correo electrónico, horario de trabajo, empresas asociadas, nombres de empleados, redes sociales, etc.

Asimismo, debemos fijarnos en apartados como “Noticias” o “Anuncios” ya que las compañías suelen filtrar información de forma no intencionada. De este modo, debemos estar atentos a fusiones de compañías o compras, ya que incluso la fusión más fluida crea cambio y desorden que puede ser aprovechado para, por ejemplo, un ataque de ingeniería social.

Por último, también es importante fijarnos en los puestos de trabajo técnico ofertados por la compañía, ya que suelen revelar información detallada sobre las tecnologías y equipos usados por la organización.

En muchos casos, una vez que hayamos examinado concienzudamente la web de la empresa auditada, tendremos la suficiente información sobre ella (quiénes son, qué hacen, dónde está situada, que tecnologías usa) para llevar a cabo un *Footprinting* más en profundidad. [12]

Otra de las formas clásicas de obtener información sobre nuestro objetivo es el servicio Whois. Este nos da acceso a datos como la dirección IP; el *host name* de los DNS de la compañía o información de contacto (tales como la dirección o el número de teléfono). [7] [9] [12] [10]

Una vez hemos investigado en profundidad la web de la empresa, podemos pasar a aplicar el Google Hacking, que consiste en usar algunas funciones del famoso buscador y sus opciones de búsqueda para conseguir información más precisa o detallada de la que conseguiríamos con una búsqueda normal. El número de comandos o *dorks* disponibles para la búsqueda avanzada es muy grande y dependerá del criterio del auditor cuales usar. [12] [9] [10]

Una de las más útiles es “allintitle” que provoca que las páginas que tengan las palabras buscadas en su título sean devueltas. Una forma de usarla sería usar la búsqueda: “allintitle: index of”, que nos proporcionaría una lista de páginas web que tienen indexados los directorios disponibles en su servidor web.

Otra directiva muy utilizada es “inurl”, que busca sitios web que contienen las palabras buscadas en su URL. Por ejemplo, podríamos usar la búsqueda: “inurl: admin” para localizar la página para acceso administrativo de un determinado sitio web.

Otro posible uso de Google es navegar por una web usando la cache en vez de la propia web del sitio. Este proceso no solo reduce las posibles huellas que podamos dejar en el servidor web, sino que a veces permite ver páginas web o archivos que han sido retirados del sitio web original. [6] [12]

También puede ser interesante utilizar otros buscadores como Yahoo! o Bing ya que podrían proporcionar diferentes resultados incluso con los mismos criterios de búsqueda. [12] [9] [10]

Por otro lado, el gran crecimiento de las redes sociales como Facebook o Twitter provee un nuevo medio para obtener información de toda índole. En este sentido, es sorprendente la cantidad de datos que cualquier empleado puede proporcionar inintencionadamente. Con esto nos referimos a información como nombres de familiares, o fechas que podrían utilizarse en contraseñas, entre otras muchas posibilidades. Incluso datos tan relevantes y comprometidos como “se nos ha roto el firewall…” [9] [12] [7]

Asimismo, es importante conseguir la máxima cantidad posible de direcciones de correo electrónico y nombres de dominio relacionados con la empresa auditada. Una

herramienta simple pero muy efectiva es TheHarvester. Esta puede ser usada en Google, Bing y servidores PGP en busca de direcciones de correo y dominios. Además, también puede ser usada en LinkedIn para buscar nombres. [9] [12] [7]

En este sentido, dado que muchas empresas usan los mismos nombres para los usuarios y las direcciones de correo, podremos usar las direcciones recopiladas por TheHarvester (y variaciones de ellas) para intentar autenticarnos en servicios como SSH, VPN o FTP.

Otra excelente fuente de información son los metadatos, que, en nuestro caso, se corresponden con los datos que pueden extraerse de un archivo y que no han sido puestos intencionadamente por el usuario, sino que se producen de forma automática. Un ejemplo de esto podría ser un archivo Word o PDF, que puede incluir entre otras cosas: nombre del archivo, tamaño, autor o la ubicación donde fue guardado. Esto puede permitir, entre otras cosas, el descubrimiento de nombres de usuario, nombres de PC y servidores, rutas de red o ficheros.

Así pues, una herramienta muy efectiva en la recopilación de metadatos es MetaGooFil. Su funcionamiento consiste en buscar en internet archivos pertenecientes a la empresa auditada, descargarlos y extraer metadatos útiles. [12] [9] [7]

Una herramienta muy útil para el *Passive Footprinting* y recomendada por [12] es la página web Robtex ya que como señala [10], se la considera "La Navaja Suiza de Internet" a causa de su capacidad de realizar partes del *Active Footprinting* de forma pasiva como, por ejemplo, averiguar información sobre dominios, subdominios o servidores DNS sin interactuar directamente con la organización auditada

Para empezar con el *Active Footprinting*, la herramienta que debemos utilizar y en la que coinciden todos los libros consultados es Nmap. Según [13] Nmap es una herramienta para descubrimiento de redes y auditoría de seguridad que usa paquetes IP no comunes para determinar que equipos hay disponibles en una red, que servicios tienen en funcionamiento, que sistema operativo usan…

Para la realización del *Active Footprinting* nos centraremos en las funciones más básicas de esta herramienta, entre las que destacan los barridos ping y el escaneo de puertos. El escaneo de puertos puede realizarse de varias maneras: la más básica es el escaneo TCP. Este consiste en realizar el *three-way handshake* en cada puerto especificado y luego cerrar la conexión. Si no especificamos ningún puerto, la herramienta realizará el escaneo de los mil puertos más comunes. Es recomendable escanear todos los puertos, ya que cambiar los puertos que utiliza un servicio es una medida de seguridad fácil de implementar por un administrador.

Otra forma de realizar el escaneo de puertos es el escaneo SYN, que es el que Nmap usa por defecto y es más rápido que el escaneo TCP, puesto que en lugar de completar el *three-way handshake* solo completa los dos primeros pasos del proceso. Además, proporciona un cierto grado de sigilo, ya que al no completar el *handshake* no suele quedar registrado en la maquina objetivo.

También es recomendable realizar un escaneo de puertos UDP. Dado que UDP es un protocolo no orientado a la conexión es difícil para la herramienta saber si un puerto está abierto, pues el comportamiento por defecto, si está abierto, suele ser absorber el paquete. Una forma de conseguir más información es usar la opción "-sV" de la aplicación que mandara paquetes especiales para provocar una respuesta de los puertos UDP y así averiguar qué servicios están a la escucha.

Nmap tiene muchas otras funciones como el *Xmas Scan* o el *Null Scan*. El primero de ellos consiste en activar los *flags* FIN, PSH y URG de un paquete y desactivar SYN y ACK. Con esto pretendemos obtener un paquete de RST si el puerto está cerrado o silencio si está abierto, tal y como indica el TCP RFC. Por otra parte, el *Null Scan* consiste en paquetes que no tienen activado ningún *flag* y la respuesta que esperamos conseguir es la misma que con el *Xmas Scan*. Esto solo funcionara si el sistema operativo atacado cumple completamente el TCP RFC, por ejemplo, en Windows no es así. Es importante destacar que ninguno de estos dos escaneos intenta establecer ningún tipo de conexión, sino que su objetivo es determinar si un puerto está abierto o cerrado.

Finalmente, debemos hablar del NSE (*Nmap Scripting Engine*), que nos permite ampliar Nmap mas allá de las funciones de escaneo tradicionales. El NSE nos permite realizar tareas avanzadas como escaneo de vulnerabilidades, detección de puertas traseras, entre muchas otras que ampliaremos en la fase de análisis de vulnerabilidades. [12]

El segundo lugar donde buscar información sobre una compañía es en sus servidores DNS. Una de las primeras cosas que debemos intentar debe ser una "transferencia de zona", es decir, hacer que el servidor DNS crea que nosotros somos otro servidor DNS y nos transfiera toda la información de su tabla de traducción de nombres de dominio a IP. [7] [12] [9]

La herramienta Dig recomendada en [7], [12] y [9] nos permite realizar la transferencia de forma simple. Puesto que los servidores, si están bien configurados, no permiten realizar la transferencia de zona excepto a otras máquinas específicamente autorizadas, podemos usar la herramienta Fierce recomendada en [7], [12] y [10] que además de intentar la transferencia de zona también permite hacer un ataque por fuerza bruta al servidor DNS. Este ataque consiste en hacer consultas para ver que nombres de dominio existen realmente. Por ejemplo, si sabemos que la compañía usa el dominio "trustedsec.com" es posible que también posea el dominio "support.trustedsec.com" y otros similares. Fierce nos permite realizar estas pruebas de forma automatizada.

Otra aplicación que debemos tener en cuenta es Maltego, esta herramienta permite recoger información de manera gráfica. Su funcionamiento se basa en entidades que pueden ser nombres, direcciones de correo, nombres de dominio… Sobre ellas se pueden aplicar acciones conocidas como transformadas que pretenden proporcionar información relacionada con la entidad. Un ejemplo de estas transformadas es la capacidad de Maltego de obtener los servidores de correo asociados a un dominio o de relacionar un nombre con una dirección de correo.

Además, Maltego permite enlazar entidades y sus transformadas para obtener un mapa que muestre la estructura de un dominio y los usuarios relacionados con él. Para ello puede utilizar redes sociales como Facebook, Twitter o LinkedIn. [8] [7]

Los servidores de correo también permiten obtener información importante sobre una organización, siempre que ésta posea un servidor propio. Por ejemplo, podemos mandar un correo con un archivo ejecutable vacío con la intención de que el servidor lo revise y lo rechace. Normalmente, al rechazar el correo este volverá a nosotros e incluirá un mensaje predefinido explicando que el servidor no acepta correos con extensiones peligrosas. A menudo este mensaje también indicará la marca y versión del antivirus usado para la inspección del correo, dato muy importante para un atacante.

Del mismo modo, el hecho de que el mensaje nos sea devuelto también nos permitirá examinar las cabeceras para obtener información básica del servidor de correo como su dirección IP y la marca y versión del software de correo usado. [9] [7] [6]

A continuación, trataremos la recogida de información sobre los servicios web siguiendo la metodología usada en [10] que a su vez se basa en [14]. Esta, consiste en:

- Identificación del servidor web
- Identificación del *Content Management System* (CMS)
- Identificación de vulnerabilidades y *plugins* de los CMS

En primer lugar, es muy importante saber el tipo de servidor web que está ejecutándose, pues esto nos ayudará a buscar vulnerabilidades y *exploits* conocidos. La forma más fácil de conocer el tipo y versión del servidor web usado es analizar el *Banner* del servicio. Hay una gran cantidad de herramientas que nos permiten detectar que servicios se están ejecutando, entre ellas las famosas Ncat, Nmap o su versión grafica Zenmap. Por defecto, Nmap solo indica la versión del servicio si está completamente seguro, no obstante, existe un comando para forzar a la herramienta a que realice una serie de pruebas para tratar de conocer que versión usan los servicios. Sin embargo, aunque nos devuelva la versión del servidor podría tratarse de un falso positivo, por lo que sería conveniente contrastar la información usando otra herramienta como WhatWeb que trataremos más adelante.

Dado que muchos sitios web usan gestores de contenido (Wordpress, Joomla…) resulta fundamental identificar cual se utiliza. La anteriormente mencionada WhatWeb permite esta identificación y además permite análisis estadístico de paquetes, librerías JavaScript y servidores web.

Además, los CMS tienen un diseño modular, es decir, permiten el uso de *plugins* para personalizar el servicio web. Puesto que muchos de estos *plugins* no han sido creados por los programadores del CMS, es frecuente encontrar vulnerabilidades que podrían ser usadas para comprometer el servicio web. En relación con lo anterior, encontramos aplicaciones cuyo propósito es analizar, para un CMS determinado, su versión y *plugins* asociados. Algunas de estas herramientas son BlindElephant y Nikto.

Asimismo, Nikto es una de las herramientas más completas para la realización de una auditoría web. Por defecto, realiza un escaneo de todo el sitio web, con lo que consigue detectar la versión del servidor, el CMS usado, si hay algún *plugin* vulnerable e incluso informa si ha detectado indicios del uso de balanceadores de carga, etcétera. Sin embargo, no siempre es necesario realizar un escaneo completo ya que la cantidad de "ruido" generado podría provocar que seamos descubiertos. Para evitarlo, existe una función llamada "Scan Tuning" que permite elegir uno por uno los test que deseamos realizar. En esta línea, también tiene disponibles distintas técnicas que permiten evadir IDS, IPS y WAF.

Otro protocolo del que podemos extraer información útil es VoIP, ya que es posible interceptar el tráfico y escuchar conversaciones o suplantar a un usuario si se cuenta con las herramientas adecuadas. Algunas de las que podemos usar para la recogida de información en VoIP son ACE VOIP y enumIAX. [10]

La primera es una herramienta de enumeración de directorios VoIP corporativos usando DHCP, TFTP y HTTP, su funcionamiento consiste en hacerse pasar por un

teléfono IP y descargar las entradas de nombre y extensión que un determinado teléfono podría mostrar en la pantalla de su interfaz. Actualmente solo soporta la tecnología usada en los teléfonos IP *Cisco Unified*. Por su parte, enumIAX, realiza la enumeración de usuarios a través de fuerza bruta usando el protocolo IAX2. Puede usarse o bien con diccionario o bien de manera secuencial.

Un aspecto que no debemos olvidar al realizar la recogida de información es la detección de IDS y IPS. Para ello, contamos con varias herramientas como WAFW00F *y* Fragroute.

WAFW00F, es un detector de *Web Application Firewall* (WAF), que son un tipo especial de firewalls diseñados para funcionar con aplicaciones web. Esta herramienta recibe como argumento una o varias URL y realiza una serie de pruebas para determinar si la web auditada cuenta con un WAF. Actualmente puede detectar hasta 22 tipos distintos. [9] [10]

Asimismo, Fragroute, es una herramienta diseñada para interceptar, modificar y reescribir trafico saliente hacia un host determinado. Cuenta con funciones que simplifican el retrasar, duplicar, fragmentar, reordenar, segmentar y en definitiva modificar cualquier paquete saliente. Lo que nos permite poner a prueba los parámetros que tiene configurados un IDS o IPS para así poder evitarlo. [15]

Además de los anteriores métodos, podemos usar la ingeniería social para obtener información sobre una empresa, como a continuación expondremos.

Así pues, podemos definir la ingeniería social como el proceso de explotar las debilidades “humanas” inherentes a cualquier organización. Cuando usamos la ingeniera social, el objetivo del atacante es conseguir que un empleado divulgue información que debería haber seguido confidencial.

Por ejemplo, podemos usarla para mandar un correo a un comercial de la compañía auditada con el objetivo de que nos responda y así poder observar las cabeceras del correo de respuesta.

Llevándolo un paso más allá, supongamos que el comercial está de viaje y el correo que recibimos es una respuesta automática que dice: “Ausente de la oficina dos semanas por vacaciones, con acceso limitado al correo”. Un ejemplo clásico de ingeniería social sería hacerse pasar por el comercial y llamar al servicio técnico de la compañía pidiendo ayuda para recuperar nuestra contraseña ya que estamos de viaje y no conseguimos acceder a nuestro correo electrónico. Si todo sale bien y conseguimos la contraseña tendríamos acceso al correo del comercial y, si usa la misma contraseña para otros servicios, puede que también acceso al VPN o al FTP de la compañía.

Otra forma de ingeniería social sería dejar un *pendrive* USB o un CD dentro o cerca de la empresa objetivo para que lo encuentre un empleado, por ejemplo, en el parking, los baños o la recepción. El empleado normalmente introducirá el *pendrive* o el CD en su PC para ver que contiene, lanzando automáticamente el troyano que habríamos introducido previamente, dejando así expuesta una vía de acceso a la organización.

Esta es solo una de las muchas aplicaciones posibles en la fase de recogida de información, pero la ingeniera social puede ser usada en varias fases de la auditoría con diferentes objetivos. [9] [7]

Mención aparte merecen las herramientas dedicadas a la recogida de información en redes Wi-Fi como Kismet o Airodump-ng que nos permiten detectar tanto las redes que transmiten su SSID como las que no lo hacen, además estas herramientas nos proporcionan otra información como el tipo de encriptación usada (WEP, WPA, WPA2), la dirección MAC del emisor y el canal en el que se está transmitiendo. En síntesis, toda la información necesaria para atacar la red.

También debemos fijarnos en las direcciones MAC de clientes conectados, puesto que si la red tuviera filtrado por MAC con esa información podríamos suplantar a un usuario autorizado. Para este propósito es muy útil la aplicación Macchanger que nos permite cambiar la dirección MAC de nuestro adaptador Wi-Fi libremente. [9]

Una vez usadas todas las herramientas anteriores, debemos revisar la enorme cantidad de información recogida. Es una buena práctica crear una lista en la que se encuentren todas las direcciones IP y crear otras listas para direcciones de correo, nombres de dominio y URLs.

Cuando tengamos recopilados y ordenados todos los datos, es vital que recordemos el alcance que nos ha sido autorizado previamente, ya que no todas las IP conseguidas estarán dentro de dicho alcance. En ese momento, debemos contactar con la empresa contratante para incrementar el alcance de ser necesario o quitar las IP conseguidas que estén fuera de él. [12] [9] [7]

No debemos olvidar que, aunque la recogida de información es la fase menos técnica de un test de penetración, no debe ser pasada por alto, ya que, cuanta más información consigamos, mayor posibilidad de éxito tendremos en fases posteriores como la de modelado de amenazas al poder determinar con mayor exactitud los riesgos a los que está expuesta la organización. [12]

2.3 MODELADO DE AMENAZAS

"El análisis de modelo de amenazas (TMA) es un análisis que ayuda a determinar los riesgos de seguridad que pueden acaecer en un producto, aplicación, red o entorno, así como la forma en la que aparecen los ataques. El objetivo consiste en determinar cuáles son las amenazas que requieren mitigación y los modos de hacerlo." [16]

Para tratar estas cuestiones nos basamos de nuevo en el PTES. Sin embargo, dado que este no especifica ninguna metodología de modelado de amenazas (pues solo indica que debe ser consistente y repetible en test sucesivos), no nos detendremos sino en sus aspectos más generales, dado que excede el propósito del presente trabajo. No obstante, podemos encontrar metodologías mucho más específicas como la usada por Microsoft en su SDL (*Security Development Lifecycle*) o el OSSTMM (ambas disponibles en la red).

El PTES se centra en dos elementos del modelado de amenazas tradicional, activos y atacantes. Los activos además se subdividen en activos del negocio (empleados, pagina web…) y procesos del negocio.

Cuando modelamos la parte del atacante debemos tener en cuenta el valor de los activos disponibles y cuánto le costaría al atacante obtenerlos. Además, debemos realizar un análisis de impacto para conseguir una visión más clara de las consecuencias de perder un activo.

La fase de modelado de amenazas es fundamental tanto para el auditor como para la organización, ya que permite clarificar qué activos y grupos de riesgo son más

importantes, por ejemplo, ¿qué posibilidades hay de que un grupo de activistas (de cualquier índole) decida tumbar la página web de la organización? Esto nos permite centrarnos en diseñar un ataque que sea lo más parecido al perfil y conocimientos esperados del atacante.

El modelo debe construirse cooperativamente con la organización auditada siempre que sea posible y aun en un entorno de caja negra el auditor debe crear un modelo basado en el punto de vista del atacante con la información obtenida mediante OSINT.

El proceso de modelado de amenazas, en líneas generales, sería:

1. Recopilar información relevante
2. Identificar y categorizar en activos principales y secundarios
3. Identificar y categorizar amenazas y grupos de riesgo
4. Emparejar los grupos de riesgo con los activos

A continuación, explicaremos brevemente a qué aludimos al hablar de activos y procesos del negocio.

Durante el análisis de activos del negocio, debemos revisar toda la documentación recopilada y reunirnos con los administradores de la organización para que, así, el auditor sea capaz de identificar que activos es más probable que elija un atacante, qué valor tienen y qué impacto tendría su pérdida parcial o total.

Algunos de los activos que deberían tenerse en cuenta son:

- Información de productos: patentes, planes futuros, código fuente y cualquier otra información que la organización considere importante para el éxito de un producto.
- Estrategias de marketing: en este ámbito, los planes sobre promociones, lanzamientos o asociación con otra empresa son objetivos muy buscados.
- Datos financieros: puede incluir información sobre cuentas, tarjetas de crédito, o inversiones, entre otras.

Otros datos que también hay que tener en cuenta son los de los empleados y los clientes, ya que podrían ser objeto de investigación y la pérdida de sus datos podría comprometer la seguridad de la compañía. Algunos de los datos considerados sensibles son:

- Datos identificativos (DNI, fotografía, teléfono, lugar de residencia…)
- Información médica
- Información financiera (cuentas bancarias, nómina…)
- Información sobre proveedores

Asimismo, no hay que olvidar el componente humano de la organización. Los recursos humanos que deben ser identificados como activos del negocio son todos aquellos que: puedan ser engañados para divulgar información, manipulados para tomar decisiones que afectarían adversamente a la compañía o que puedan permitir al atacante acceso a la organización. Hay que puntualizar que estos activos humanos no tienen por qué pertenecer necesariamente a la dirección de la empresa, sino simplemente estar

relacionados con otros activos encontrados previamente o en posición de permitir acceso a estos.

Sin duda, los activos que más interesan al auditor son los relacionados con la parte técnica de la organización, puesto que la información sobre estos puede ser muy valiosa en la fase de recogida de información. Así pues, algunos de los activos en los que debemos fijarnos son:

- Diseño de la infraestructura. Se trata de las tecnologías y/o equipamiento que se usan en la organización como sucede con los planos, diagramas de cableado o conectividad, listas de equipamiento informático, mapas de red o aplicaciones usadas a nivel de usuario que forman parte del diseño de la infraestructura.
- Configuración de los sistemas. Esta información permite descubrir vulnerabilidades a través de errores de configuración o versiones antiguas de software. Esto incluye, documentación sobre la configuración base de los equipos, listas de comprobación de la configuración, procedimientos de securización y políticas de grupo e inventarios de software.
- Credenciales de usuario. Las credenciales para diferentes servicios (VPN, portal web…) permiten acceso a información sobre la organización bien sea con privilegios de usuario o administrador, en cuyo caso, podría significar el acceso total a la organización.

En cuanto a los procesos de negocio, podemos señalar que son importantes ya que un negocio lo es si genera ganancias. Estas últimas se consiguen al hacer que materias primas o conocimientos pasen a través de una serie de procesos para mejorarlos y crear valor añadido -en forma de producto o servicio-. De este modo, los procesos de negocio y los activos del negocio (personas, tecnología…) forman cadenas de valor. Sabiendo cuáles son estas cadenas y encontrando sus imperfecciones, entenderemos cómo funciona el negocio y cómo podrían los diferentes grupos de riesgo hacer que la compañía pierda dinero.

Según el PTES hay tres categorías de procesos de negocio:

- Relacionados con la infraestructura técnica
- Relacionados con la gestión de la información
- Relacionados con los recursos humanos

Cuando definimos los grupos de riesgo, debemos tener claro si la amenaza es externa o interna, a qué grupo o grupos pertenece y cualquier información adicional que nos ayude a establecer las capacidades o motivaciones de ese grupo. Algunos ejemplos de grupos de riesgo pueden ser vistos en la tabla 1 bajo estas líneas:

Internos	**Externos**
Empleados	Socios empresariales
Dirección o Gerencia	Competidores
Administradores (de red, sistemas, servidores)	Contratistas
Desarrolladores	Proveedores
Técnicos	Crimen organizado
	Activistas

Tabla 1: Grupos de riesgo según su ubicación

Una vez hemos identificado un grupo de riesgo, debemos analizar las capacidades de dicho grupo para crear un modelo de amenazas que refleje la probabilidad de que un ataque por su parte resulte exitoso. Para ello debemos tener en cuenta varios aspectos:

- Herramientas en uso. Esto incluye cualquier herramienta que sepamos que es usada por ese grupo. Además, cualquier herramienta que sea libremente accesible debe ser tenida en cuenta dependiendo de la habilidad necesaria para utilizarla.
- Acceso a *exploits/payloads*. Debemos analizar la capacidad del grupo de riesgo para obtener o desarrollar exploits que puedan ser peligrosos para la organización. Asimismo, el acceso a dichos exploits/payloads a través de terceras partes también debe contemplarse.
- Mecanismos de comunicación usados. Estos varían desde tecnologías comúnmente usadas como encriptación hasta mecanismos más complejos como *bulletproof hosting*, uso de sitios de recogida o uso de *botnets*.
- Acceso. El último aspecto en que debemos fijarnos es la capacidad del grupo de riesgo para acceder a la organización o a un activo en concreto.

Para finalizar este apartado, es importante conocer los motivos que los grupos de riesgo pueden tener para querer dañar la organización. La mayoría de ellos se relacionan con:

- Lucro (directo o indirecto)
- Activismo
- Resentimiento
- Diversión o mejora de reputación
- Acceso a socios o sistemas conectados

Una vez finalizada esta fase debemos tener claro cuáles son los activos principales y secundarios, los principales grupos de riesgo y sus habilidades, así como estrategias de mitigación para los riesgos más probables.

2.4 Análisis de vulnerabilidades

Atendiendo a la definición proporcionada por [10] “una vulnerabilidad de seguridad es una debilidad en un producto que podría permitir a un usuario malintencionado comprometer la integridad, disponibilidad o confidencialidad de dicho producto”. Por lo tanto, un análisis de vulnerabilidades consistirá en la búsqueda de estas debilidades. Esta fase está muy relacionada con la de recogida de información e incluso comparten algunas herramientas, lo que las diferencia es el enfoque.

Basándonos en [10], que a su vez se basa en [7], podemos dividir esta fase en tres apartados: pruebas, validación e investigación.

A su vez, el apartado de pruebas puede ser dividido en dos: activas y pasivas. Para la realización de las pruebas activas es necesaria la interacción directa con el componente a auditar. Esta interacción puede ser automática o manual.

La interacción automatizada consiste en el uso de herramientas que escanean servicios realizando ciertas peticiones para examinar sus respuestas y así encontrar

indicios de vulnerabilidad. Las herramientas automatizadas pueden ser de cuatro tipos. [10] [7]

El primero son los escáneres genéricos, que permiten identificar versiones de servicios con vulnerabilidades conocidas. Pertenecen a esta categoría los escáneres de puertos, servicios y *banners*. Uno de los máximos exponentes de esta categoría es Nmap.

Además de las funciones mencionadas en la fase de recogida de información como descubrir si un puerto está abierto o cerrado, Nmap también nos permite averiguar qué servicio está escuchando en dicho puerto. Si, además, utilizamos NSE tendremos acceso a funciones avanzadas que la herramienta original no cubre como detección de versiones y servicios no basada en el *banner;* detección de vulnerabilidades e, incluso, explotación de vulnerabilidades. [12] [7] [10]

Asimismo, hay algunas herramientas centradas completamente en la búsqueda de vulnerabilidades como OpenVAS, Nessus y Nexpose.

Estas herramientas realizan un análisis de vulnerabilidades automatizado de forma transparente al usuario. Normalmente solo necesitaremos proporcionar una IP o un rango, los puertos que queramos escanear y la agresividad de las pruebas. El problema de estas herramientas es que poseen un grado de abstracción alto que impide al usuario saber que está haciendo exactamente la herramienta. [9] [7]

El segundo tipo son los escáneres de aplicaciones web, que pueden clasificarse según su funcionamiento. En primer lugar, los que se basan en rastrear todos los enlaces del servidor buscando malas configuraciones o posibilidad de inyección de código y, en segundo lugar, aquellos que buscan un listado de directorios y archivos asociados con vulnerabilidades conocidas. En esta última categoría quedaría englobada la herramienta Nikto nombrada en la fase de recogida de información y también Burp Suite muy conocida en el ámbito de la auditoría web.

La herramienta *Burp Suite* permite encontrar vulnerabilidades del tipo XSS o *SQL Injection* entre otras además de poder realizar un escaneo activo que es capaz de visitar todos los enlaces de un sitio web y realizar las pruebas necesarias para la detección de todo tipo de vulnerabilidades y errores de configuración. [10] [12]

El tercer tipo de escáner, normalmente llamado escáner de vulnerabilidades de red, es especial ya que analiza protocolos para los que no están preparados los escáneres genéricos como VPN o IPv6. Yersinia es una herramienta perteneciente a este apartado que permite analizar en busca de vulnerabilidades en protocolos tan diversos como STP, CDP, ISL o VTP entre otros. [10]

Por último, debemos mencionar los escáneres VoIP. Este caso es similar al anterior excepto porque al ser protocolos para el transporte de voz hacen falta herramientas específicas para, por ejemplo, pinchar conversaciones. En este sentido, podemos usar la herramienta Wireshark para capturar y reproducir una conversación VoIP siempre que sea sobre un protocolo soportado por la herramienta y no encriptado. [17]

También debemos tener en cuenta la evasión de IDS, IPS y WAF, para ello debemos evitar un comportamiento regular y predecible de las herramientas automatizadas para evitar la detección. Así pues, debemos hacer que las herramientas tengan un comportamiento aleatorio: variando la IP de salida; alternando el análisis de

puertos entre diferentes equipos en lugar de probar todos los de un solo equipo; variando la velocidad de escaneo, etcétera. [10] [7]

En el análisis pasivo de vulnerabilidades usamos las mismas herramientas que en la fase de recogida de información pasiva excepto porque no buscamos conseguir información, sino encontrar comportamientos extraños o datos en la información recopilada que puedan significar una vulnerabilidad. [10] [7]

Tras realizar pruebas con distintas herramientas surge la necesidad de validar sus resultados. Por lo tanto, deberemos comprobar si los resultados de los escáneres automatizados y las pruebas manuales coinciden, así como analizar por separado y manualmente algunos protocolos como VPN, Citrix, DNS, etcétera con herramientas especiales. [10] [7]

A continuación, debemos asegurarnos de no obviar ningún vector de ataque posible. Para ello es muy útil la elaboración de árboles de ataque que además nos ayudaran en la realización del informe, la fase final de la auditoría. El árbol de ataque debe ser actualizado cada vez que se descubran nuevos sistemas o servicios vulnerables ya que nos proporcionara una guía a seguir en la siguiente fase, la de explotación. [10] [7]

La última parte de un análisis de vulnerabilidades es la investigación. Tras identificar una vulnerabilidad, necesitaremos conocer su gravedad y sus posibilidades de explotación. En muchos casos, la vulnerabilidad será un error de código en una aplicación software, pero en otros casos puede ser una mala configuración o el uso de contraseñas por defecto. [10] [7]

Normalmente, el primer sitio al que debemos acudir en busca de información sobre una vulnerabilidad es a una base de datos de vulnerabilidades. La mayoría de herramientas usan un identificador CVE (*Common Vulnerabilities and Exposures*) para identificar una vulnerabilidad concreta.

Los CVE son referencias para identificar una vulnerabilidad concreta. El formato usado para identificar los elementos de la lista es, CVE-YYYY-NNNN donde las Y indican el año de descubrimiento de la vulnerabilidad y las N el número de vulnerabilidad. Debido al aumento de vulnerabilidades descubiertas se puede ampliar el número más allá de las cuatro cifras en caso de ser necesario. [18]

Estos identificadores son muy útiles para ampliar información en bases de datos de vulnerabilidades como *National Vulnerability Database* [19] y *Bugtraq* [20] o para encontrar un *exploit* en una base de datos de *exploits*. Esto es importante porque, por ejemplo, puede existir una vulnerabilidad que solo funciona en la versión Windows del servidor web Apache, pero no en su versión Linux, este hecho podría pasar desapercibido a un escáner automatizado. [10] [7]

A menudo, los administradores eligen contraseñas débiles, no cambian las contraseñas por defecto e, incluso, no usan contraseña alguna. En la página web del vendedor y en foros especializados es común encontrar listas de contraseñas por defecto para un producto o errores de configuración comunes. [10] [7]

Otro sitio donde podemos encontrar información importante como es la configuración por defecto y errores de configuración es en las guías de fortificación ya que señalan las partes más débiles de un sistema o programa. Las listas de correo y los

foros también proporcionan información valiosa sobre los problemas que suelen tener los administradores configurando y securizando un sistema. [10] [7]

Por último, no debemos obviar las vulnerabilidades relacionadas con la red Wi-Fi ya que es un punto de acceso a la red de la organización que puede ser fácilmente explotado si no está correctamente configurado.

Uno de los primeros aspectos en los que fijarnos es el tipo de encriptación usada, ya que si es cualquier tipo de WEP o WPA-PSK existen formas rápidas de conseguir la contraseña a causa de la mala implementación de estos estándares. WPA2 es el único estándar para el que todavía no se ha encontrado un método más rápido de recuperación de la contraseña que el uso de diccionarios o fuerza bruta.

Existen diferentes ataques contra WEP como los de tipo FMS, PTW o chop-chop pero todos ellos son fácilmente realizables utilizando las herramientas de la *suite* Aircrack-ng, por ejemplo, airmon-ng para activar el modo monitor en la tarjeta inalámbrica, airodump-ng para capturar los IV y aireplay-ng para reinyectarlos.

El caso de WPA-PSK es diferente puesto que para romper la contraseña no son necesarios IVs sino que puede hacerse de forma local una vez hayamos conseguido el *four-way-EAPOL-handshake*. Este *handshake* es generado normalmente cuando un usuario que conoce la contraseña se conecta al punto de acceso y puede ser capturado. En caso de que esto no ocurra podemos desautenticar a un usuario ya conectado usando aireplay-ng para forzar la reconexión.

Una vez conseguido el *handshake* podemos usar la herramienta CoWPAtty para automatizar el ataque por diccionario al que es vulnerable WPA-PSK. Debido a que WPA usa 4096 iteraciones del algoritmo SHA1 para la creación del *hash*, la creación de *hashes* para comparar con el capturado puede resultar lenta. Además, durante la creación del *hash* se utiliza el SSID del punto de acceso como *salt* por lo que no se pueden utilizar tablas de *hashes* ya creados a menos que tengan el mismo SSID.

El *salt* se utiliza como entrada en la función de *hash* junto con la contraseña para dificultar los ataques por diccionario. Existen tablas de *hashes* ya computados para los SSID más comunes, pero en caso de no estar el SSID en alguna de las tablas podemos crearlas usando la herramienta genpmk. Este proceso puede acelerarse utilizando tarjetas gráficas y *cloud computing* como comentaremos en la fase de post-explotación. Debemos resaltar que solo es posible romper la contraseña en sistemas WPA-PSK ya que no hay vulnerabilidades conocidas en WPA-RADIUS.

Por otra parte, existe un ataque que afecta a cualquier punto de acceso que utilice WPS incluso a los que utilicen WPA2. Esto es posible porque el funcionamiento de WPS consiste en realizar la primera conexión con el punto de acceso con un pin de 8 dígitos en lugar de con el protocolo y contraseña configurados. El hecho de reducir la complejidad a 8 dígitos unido a otros errores del protocolo hace que la duración de un ataque por fuerza bruta sea menor a un día en un equipo actual. Una herramienta creada especialmente para realizar este ataque es Reaver. [18]

Para finalizar con el apartado de Wi-Fi nos gustaría hablar sobre CROZONO, un *framework* que nos permite usar drones y robots en un test de intrusión. El *framework* tiene multitud de opciones y es capaz de realizar parte del pentest de forma autónoma si está correctamente configurado, pero en nuestro caso solo nos centraremos en sus posibles usos relacionados con el Wi-Fi.

Por ejemplo, podría usarse junto con un receptor GPS para geolocalizar los puntos desde los que es accesible la red inalámbrica en el exterior de la organización y saber, así, la cobertura de la misma. Otra posibilidad sería programar el dron para recoger toda la información posible sobre la red Wi-Fi e intentar llevar a cabo alguno de los ataques comentados anteriormente en caso de ser vulnerable a ellos. [21]

Otro vector de ataque mencionado en los medios de comunicación es utilizar un dron para acceder a impresoras Wi-Fi sin protección en las plantas superiores de un edificio que suelen quedar desprotegidas ya que se suele considerar que, al no ser accesibles en la planta baja, no es necesario cambiar la configuración por defecto. [22]

Tras realizar todos los pasos anteriores tendremos una lista de posibles vulnerabilidades ordenadas según su gravedad que podremos organizar en un árbol de ataque que usaremos en la fase de explotación.

2.5 EXPLOTACIÓN

En general, entendemos la explotación como el proceso de obtener control sobre un sistema, aunque es importante señalar que no todos los *exploits* tienen como resultado el acceso total al sistema.

Una definición más exacta señala que un *exploit* es una herramienta para aprovechar un fallo de seguridad. Este proceso puede tomar formas muy variadas, pero en este trabajo el objetivo final será siempre el mismo: lograr acceso administrativo a un equipo.

Así pues, un *exploit* es un programa que permite al atacante ejecutar un *payload* en el equipo objetivo. Los *payloads* pueden alterar el funcionamiento de un software y permiten realizar todo tipo de acciones, como instalar nuevo software, desactivar servicios en funcionamiento, añadir usuarios, abrir puertas traseras al equipo y mucho más. De ahora en adelante consideraremos *exploit* y *payload* como equivalentes. [12]

Los *exploits* aprovechan errores como el *off-by-one error* o expansión Unicode incorrecta, así como errores que causan corrupción de memoria como el *buffer overflow* o el *format string exploit* para realizar acciones no planeadas por el creador del programa. El objetivo de estos *exploits* es tomar el control del flujo de ejecución del programa y forzarlo a ejecutar código que hemos introducido previamente en memoria. Este proceso es conocido como ejecución de código arbitrario, ya que el atacante puede provocar que el programa haga prácticamente cualquier cosa. [23]

De todas las fases de una auditoría de seguridad, la explotación es probablemente la más amplia dado el gran abanico de opciones y herramientas disponibles para completar el proceso. Esto es así porque cada sistema es diferente y cada objetivo es único, por lo cual los vectores de ataque variaran de un equipo a otro dependiendo de su sistema operativo, los servicios en funcionamiento, software instalado, etc.

Debido a la gran cantidad de herramientas disponibles solo explicaremos brevemente el conjunto de herramientas más conocido y utilizado en esta fase: Metasploit.

Metasploit es un *framework* dedicado a desarrollar y ejecutar *exploits*, además ayuda en el proceso de desarrollo al proporcionar unas directrices sobre cómo deben crearse las piezas que forma un *exploit* y cómo deben interactuar entre ellas. El *framework* también posee herramientas que permiten realizar docenas de funciones diferentes,

incluyendo algunas pertenecientes a otras fases (como por ejemplo realizar un escaneo de puertos). [12]

Según [24] los exploits pueden dividirse dependiendo de su complejidad en tres tipos:

- *Singles*, son auto contenidos y completamente autónomos. Pueden realizar acciones simples como añadir un usuario a un equipo o ejecutar un programa.
- *Stagers*, este tipo realiza una conexión entre el atacante y el objetivo y están diseñados para ser pequeños y confiables. Son usados por los de tipo *Staged* para introducirse en huecos de memoria donde realizar la explotación.
- *Staged*, estos *payloads* son descargados por los de tipo *Stagers* y permiten funciones avanzadas como Meterpreter o inyección VNC.

Otro aspecto que debemos tener en cuenta al elegir un *exploit* es si está diseñado para evadir DEP. El cometido de este sistema es dejar claro al procesador que son datos y que código en memoria, el atacante tal vez consiga inyectar código, pero no podrá ejecutarlo. Esto es importante ya que como apunta [25] "Cuando un atacante aprovecha un desbordamiento, lo que hace es inyectar en esa memoria del proceso un flujo hexadecimal de información. Si este flujo se interpreta como instrucciones, el procesador se pondrá a trabajar y el atacante conseguirá su objetivo".

Además de los exploits, hay otras formas de conseguir acceso no autorizado a un equipo, por ejemplo, podríamos aprovechar la existencia de servicios de acceso remoto como SSH o VNC en el equipo auditado. Para acceder a estos servicios existen herramientas como Medusa o Hydra, que son rompedores de contraseñas online, es decir, funcionan realizando un ataque por diccionario o fuerza bruta a los servicios de acceso remoto. Las probabilidades de éxito aumentan exponencialmente si combinamos estas herramientas con la información obtenida en la fase de recogida de información como nombres de usuario, correo electrónico, etc. Debemos señalar que algunos de estos servicios tienen un límite de *logins* incorrectos y que pueden bloquear la IP que realiza los intentos o el usuario usado. [12]

Asimismo, existe una técnica conocida como *evilgrade* que puede proporcionarnos acceso a un equipo totalmente actualizado y sin ningún *exploit* conocido. Este ataque consiste en hacer creer al equipo atacado que hay una actualización disponible de alguna de las aplicaciones que tiene instaladas y al intentar actualizarla descargará e instalará la herramienta que nosotros deseemos (normalmente un *backdoor*).

Esto es posible porque la mayoría de actualizaciones no están encriptadas y las aplicaciones no usan certificados para asegurar que el emisor es correcto. Para realizar el ataque necesitaríamos tener el control del tráfico de la víctima (por ejemplo, mediante un MITM), lo que nos permitiría suplantar al servidor al que el equipo está pidiendo la actualización y así conseguiríamos nuestro objetivo.

La herramienta más usada para realizar este ataque es *evilgrade.* Este *framework* permite automatizar parte del proceso para aprovechar que aplicaciones como Winzip, OpenOffice, iTunes o Safari, entre muchas otras, no comprueban que la actualización provenga realmente de un emisor de confianza. [26]

Como hemos comentado anteriormente, no todos los *exploits* conceden acceso total al sistema, puede ocurrir que se obtenga acceso al sistema con un usuario sin privilegios, dado que nuestro objetivo es conseguir acceso administrativo deberemos efectuar lo que se conoce como escalado de privilegios. Existen muchas formas de realizarlo, aprovechar una vulnerabilidad interna solo explotable desde dentro del equipo o romper los *hashes* de las contraseñas almacenadas en el equipo son algunas de las más comunes. [12]

Antes de poder romper los *hashes* de las contraseñas debemos localizar el archivo donde se almacenan, en sistemas Windows se guardan en un archivo especial llamado SAM que en sistemas Windows 2000 y superiores está situado en la carpeta C:\Windows\System32\Config\. Dado que este archivo contiene información muy importante posee varias medidas de seguridad. [25]

La primera es que este archivo queda bloqueado cuando el sistema operativo se inicia por lo que no se puede abrir ni copiar, la segunda es que el archivo está encriptado. Una forma de evitar estas restricciones, si se tiene acceso físico al equipo, seria utilizar un *Live CD* para iniciar el sistema, dado que Windows no llega a iniciarse tampoco bloquea el acceso a la SAM. Ahora tenemos el archivo, pero este sigue encriptado, para desencriptarlo necesitamos el archivo *system* situado en la misma carpeta que la SAM y una herramienta como Samdump2 que usando los dos archivos nos proporcionara los *hashes* que normalmente estarán en formato NTLM. [25]

Además, existe la posibilidad de conseguir los *hashes* de forma remota y con el equipo en funcionamiento. Para ello, usaremos herramientas como pwdump que funciona conectándose al proceso LSASS que es el encargado de autorizar y manejar las contraseñas introducidas en el sistema y extrayendo los *hashes*. [25]

En el caso de sistemas Linux y OS X las contraseñas están situadas en el llamado *shadow file* localizado en la carpeta /etc/shadow, este archivo, como su homólogo en Windows, está encriptado y además solo es accesible para usuarios con privilegios de administrador. Esto podría representar un problema si no fuera por la existencia de un archivo localizado en /etc/passwd que normalmente es legible para todos los usuarios y que podemos utilizar junto con una función especial de la herramienta John The Ripper para combinarlo con el *shadow file* y obtener los *hashes* originales que suelen estar en formato SHA. [12]

Una vez obtenidos los hashes e independientemente del sistema operativo, usaremos una herramienta como el anteriormente mencionado John The Ripper para romper la contraseña. Esto no es directamente posible ya que es imposible revertir una función de *hash* para obtener la palabra que la originó, pero si es posible usar la función *hash* sobre un diccionario o mediante fuerza bruta para comparar el resultado obtenido con el *hash* extraído del equipo auditado dado que no hay dos palabras que tengan como resultado el mismo *hash*. [12]

Existen varios métodos para acelerar la generación de *hashes* pues utilizando un solo procesador, aún con varios núcleos, suele ser un proceso lento y en ocasiones imposible. Uno de estos métodos es usar la potencia de cálculo en paralelo que proporcionan las tarjetas gráficas modernas mediante la herramienta oclHashcat que puede acelerar la generación de *hashes* en un factor 10 o 100 e incluso más dependiendo de la cantidad de tarjetas gráficas disponibles. [9]

Otro método novedoso es el uso del *cloud computing* como el proporcionado por Amazon Web Services que nos permite alquilar por meses, días o incluso horas la capacidad de proceso que necesitemos. La herramienta a usar para generar los *hashes* es la misma, aunque requiere de un mayor tiempo de configuración. Este método puede ser útil si no disponemos de las tarjetas gráficas necesarias ya que es una buena solución en relación con su coste/beneficio. En caso de utilizar constantemente una capacidad de proceso alta sería más económico optar por la anterior opción. [27]

Como hemos visto anteriormente, las contraseñas en Windows son almacenadas normalmente en forma de *hash* y deben ser obtenidas en texto plano recurriendo a diccionarios o fuerza bruta para poder ser utilizadas. Sin embargo, existe una tercera opción, el ataque *Pass The Hash.*

Este ataque permite autenticarse en un equipo o servicio usando un *hash* obtenido previamente en otro equipo (siempre y cuando el usuario y la contraseña coincidan). Esto es muy útil, dado que si conseguimos el *hash* de administrador de un equipo es posible que podamos usarlo en otros equipos de la organización pues probablemente el administrador no haya cambiado la contraseña para cada equipo. De este modo, si estuviéramos en un dominio y obtuviéramos un *hash* de administrador de dominio la auditoría finalizaría porque tendríamos acceso a todos los equipos y carpetas en red.

Existen varias herramientas para realizar este ataque como Windows Credential Editor o el módulo psexec de Metasploit, también podemos utilizar PStools para ejecutar procesos de forma remota con dichas credenciales. [18]

Del mismo modo, otra forma de escalar privilegios es escuchar la red, si escuchamos todo el tráfico entrante y saliente del equipo auditado es posible que encontremos información que nos permita obtener privilegios de administrador. La herramienta más utilizada para esta tarea es Wireshark, el famoso analizador de protocolos. Este programa nos permite capturar tráfico y analizarlo posteriormente, además, cuenta con un potente sistema de filtros para extraer solo la información que necesitemos. [6] [12]

Por otro lado, no debemos olvidar que en una auditoría además de PC y servidores también existen *routers*, *switches* y demás equipamiento de red que también puede ser explotado. Es común encontrar contraseñas por defecto en el equipamiento anteriormente nombrado, pero también en impresoras de red o sistemas de almacenamiento de copias de seguridad. Solemos creer que estos equipos son solo hardware, pero suelen contar con un motor web para configuración y otros servicios que podrían contener vulnerabilidades. [10]

Por último, debemos tener en cuenta que al utilizar *exploits* encontrados en internet existe la posibilidad de que el *exploit* haga más de lo que dice su autor o que no funcione correctamente, pudiendo provocar daños inesperados e irrecuperables en el equipo auditado, por lo que se aconseja probar cualquier *exploit* en un entorno controlado antes de usarlo en el equipo real. [6]

Una vez hayamos conseguido el nivel de acceso deseado (normalmente de tipo administrativo) podremos continuar a la siguiente fase, la de post-explotación.

2.6 POST-EXPLOTACIÓN

El objetivo de la fase de post-explotación es determinar el valor de la máquina auditada y mantener el control sobre ella. Su valor depende de la importancia de los datos almacenados y de lo útil que pueda ser en futuros ataques a equipos de la misma red. Para ello debemos conocer la configuración del equipo, los canales de comunicación disponibles y los equipos accesibles a través de ellos. [7]

Esta fase puede ser dividida a grandes rasgos en tres apartados: mantenimiento del acceso, obtención de información y cubrir huellas. [6]

La mayoría de los exploits no son persistentes, es decir, si una vez ejecutado el exploit y conseguido el acceso al equipo hubiera un corte de luz, una caída de red o incluso un simple reinicio tendríamos que volver a realizar todo el proceso de explotación, que, en ocasiones, puede llevar horas. Para evitarlo, el primer paso en la fase de post-explotación es mantener el acceso y para conseguirlo es común el uso de *backdoors*. [7]

Un *backdoor* es un tipo de herramienta que podemos instalar en el equipo atacado para acceder remotamente, algunos de los más usados son el polifacético Ncat y Cryptcat. Esta última es una variante de Ncat que usa un canal de datos cifrado para evadir los sistemas de seguridad de la organización auditada.

El funcionamiento de ambas para usarlas como *backdoor* es el mismo, *ncat* puede funcionar tanto en modo cliente como en modo servidor, por lo cual solo tendremos que instalarlo, elegir un numero de puerto y configurarlo para que proporcione una línea de comandos a quien se conecte a ese puerto.

Además, también existe la posibilidad de hacer que Ncat de manera automática haga la conexión con un equipo que configuremos con lo cual evitaremos muchos sistemas de seguridad que controlan las conexiones entrantes, pero no las salientes. Esto último es llamado una *Reverse Shell*. [6]

Por otro lado, debemos señalar que la utilización de *backdoors* es un tema que suele preocupar a las empresas auditadas ya que temen que sean descubiertos y utilizados por alguien no autorizado, por lo que es imprescindible acordar su utilización antes de usarlos. [12]

Una vez hemos asegurado el acceso, realizaremos el proceso conocido como *Internal Footprinting* nombrado en la fase de recogida de información. Para ello utilizaremos las herramientas usadas en la fase de recogida de información que puedan ser útiles en el análisis de la red auditada y además trataremos de extraer la máxima información de los equipos auditados.

Uno de los primeros sitios a los que acudir para obtener información de un equipo es la configuración de sus interfaces de red, en ella podremos encontrar datos como sus direcciones IP, máscaras de subred y puertas de enlace. Además, debemos comprobar la existencia de otras subredes accesibles por el sistema buscando en tablas de enrutamiento, tablas ARP, NetBIOS y otros protocolos de descubrimiento de equipos. Asimismo, debemos identificar los servidores DNS en uso ya que pueden ser útiles para descubrir nuevos sistemas.

También puede haber servicios que no hayan sido identificados en las anteriores fases que puedan usarse en el descubrimiento de otras redes o equipos, por lo que es imprescindible realizar un perfil de cada sistema al que accedamos y tomar nota de la

configuración de estos servicios, su propósito y si pueden ser útiles para conseguir los objetivos de la auditoría o como pivote para un mayor acceso a la red.

Especial relevancia tienen los servicios de seguridad como *firewalls*, IDS/IPS o antivirus ya que conocer en profundidad su configuración nos dará una idea de que esperar en otras máquinas de la red y de que alertas puedan haberse producido durante la auditoría lo cual puede resultar muy útil para ayudar al cliente a mejorar sus políticas de seguridad. Del mismo modo, debemos fijarnos en la configuración de los proxis, sea a nivel de red o de aplicación, ya que normalmente son iguales en toda la empresa y podrían ser útiles para saber cómo extraer información de la organización.

Asimismo, las conexiones VPN existentes en un equipo pueden ser útiles para la extracción de información ya que no suelen ser bloqueadas por los *firewalls* o IPS y además pueden permitir el acceso a redes de socios comerciales, etc. [7]

En lo que respecta a la información contenida en el equipo, debemos priorizar la búsqueda de la que haya sido definida como objetivo en la fase de comunicación previa o en la de modelado de amenazas. Normalmente consiste en archivos que contengan información personal, planos, código fuente, información sobre tarjetas de crédito o datos bancarios, contraseñas, etcétera. Para ello es imprescindible estar familiarizado con las aplicaciones más comúnmente usadas en el entorno empresarial ya que muchas de ellas guardan los archivos en sitios y formatos diferentes. [7]

Después de haber asegurado el acceso futuro al equipo y haber obtenido toda la información posible de él, es recomendable escuchar la red en la que está situado el sistema y usar las técnicas de ataques a redes para obtener acceso a la información que circule por ella.

Los ataques a redes se dividen según [28] en tres tipos diferentes:

En primer lugar, el *Sniffing* es una técnica que permite procesar la información que circula por la red siempre que esta llegue al adaptador de red. Esta distinción es importante ya que en una red que use un *hub* toda la información se reenvía a todos los equipos conectados, pero no es el caso en una red que use un *switch*. Una herramienta que permite ver esta información y filtrarla es Wireshark.

Por su parte, el *Spoofing* no es una técnica concreta sino un conjunto de ellas. Su objetivo común es suplantar al objetivo. Existen muchos tipos diferentes según la tecnología que se utilice, MAC *spoofing*, ARP *spoofing*, DNS *spoofing*…

La más conocida y utilizada es la conocida como ARP *spoofing* que aprovecha el funcionamiento del protocolo ARP, un ejemplo básico de cómo funciona sería:

El equipo B (atacante) quiere conocer el tráfico que circula entre los equipos A y C, por lo que manda un paquete ARP *Reply* al equipo A y otro a C. Este paquete hace que A y C asocien la dirección MAC del atacante con la dirección IP del otro equipo, por lo que cualquier paquete entre A y C pasará por B. Lo único que le resta hacer a B es enrutar los paquetes que lleguen de C en dirección A y viceversa. Con esto B ha conseguido interponerse entre A y C y poder leer la información que intercambian sin que estos se den cuenta. Esto es posible ya que no es necesario que un equipo haya realizado un ARP *Request* para que acepte un ARP *Reply*.

Este tipo de ataques son conocidos popularmente como MITM. Una de las herramientas más usadas para realizar este ataque es Ettercap, por su potencia y facilidad de uso.

Otro ataque de tipo MITM ampliamente usado es el conocido como *Rogue* DHCP, que basa su funcionamiento en el uso de un servidor DHCP falso. Esto es posible debido al funcionamiento de este protocolo, que consiste en que, si hay dos o más servidores DHCP en una LAN determinada, el equipo que ha hecho la petición usando una trama *DISCOVERY* escoge la respuesta *OFFER* proporcionada por el DHCP que más rápido se la haga llegar. De esta forma el atacante puede interponerse entre ese equipo y cualquier otro consiguiendo así su objetivo.

Finalmente, el *Hijacking* también se refiere a un conjunto de técnicas, pero, en este caso, de secuestro de información. Continuando con un ejemplo anterior, ¿que impide a B alterar un paquete de A en dirección C? Sin embargo, el *Hijacking* es mucho más que la alteración de datos, ya que puede alterar sesiones o comportamientos de diferentes protocolos como HTTP, Telnet, etc.

Uno de los ataques más conocidos es el conocido como *Browser Hijacking*, este consiste en controlar el navegador web pudiendo lanzar *popups*, modificar la página de inicio, etcétera. [28]

Mención aparte merecen los ataques al protocolo IPv6, ya que está ampliamente implantado en las empresas debido a que viene activado por defecto en las últimas versiones de Windows y Linux, pero parece olvidado por los administradores que siguen centrándose en proteger IPv4.

Muchos sistemas de detección de intrusos detectan ataques de tipo ARP *spoofing*, pero no su equivalente en IPv6 el *Neighbor Spoofing*. Este ataque aprovecha el funcionamiento del *Neighbor Discovery Protocol* ya que en IPv6 no existen ARP ni RARP.

Este sistema utiliza cinco tipos distintos de mensajes ICMPv6 distintos, de los cuales dos son equivalentes al protocolo ARP, los mensajes de tipo *Neighbor Solicitation* piden la resolución de la dirección MAC de una dirección IPv6 conocida y los de tipo *Neighbor Advertisement* son la respuesta. El fallo del que se aprovecha el atacante para realizar el ataque MITM es el mismo error que existía ya en IPv4, un equipo no necesita haber realizado un *Neighbor Solicitation* para aceptar un *Neighbor Advertisement* por lo que bastará con enviar un mensaje NA a los dos equipos atacados poniendo la dirección IPv6 del otro y la dirección MAC del atacante para lograr interponerse entre ellos.

Además del método anterior también podemos realizar el ataque Rogue DHCPv6 que funciona como su análogo en IPv4. También existen otros métodos para realizar MITM en IPv6 como, por ejemplo, aprovechando la función SLAAC, pero son más complejos y necesitan ciertas condiciones previas.

Casi todas las herramientas necesarias para realizar ataques a este protocolo están presentes en The IPv6 Attack Toolkit que contiene utilidades para la realización de ataques MITM o descubrimiento de equipos entre otras. [28]

Un caso especial en los ataques a redes es el de las VLAN. En lo que respecta a una VLAN los puertos de un *switch* pueden ser de dos tipos, de acceso o de *trunk*. Un puerto de acceso solamente es miembro de una VLAN mientras que un puerto de tipo *trunk* es por defecto miembro de todas. Cuando se usan VLAN el *switch* es el encargado

de identificar las etiquetas de los paquetes y enrutarlos solo a la VLAN de la que es miembro el equipo transmisor. Conociendo estos datos podemos hablar de los dos principales ataques a las VLAN que pretenden saltarse la seguridad inherente a estas.

El primero es el *Switch Spoofing*, este ataque aprovecha una mala configuración del switch, concretamente de la función *auto-trunking* (llamada *Dinamics Trunking Protocol* en dispositivos Cisco). Esta función está pensada para poder ampliar la infraestructura de forma fácil añadiendo nuevos *switches* y consiste en que un puerto pueda estar configurado en modo pasivo esperando que otro equipo en la capa de enlace le pida pasar a modo *trunk.* Por lo tanto, si un equipo está conectado a un puerto con la función *auto-trunking* activada podría hacerse pasar por un *switch* y tendría acceso a cualquier VLAN existente.

El segundo ataque llamado *Double Tagging* basa su funcionamiento en etiquetar dos veces el tráfico generado por un sistema. Para ello se encapsula el trafico dirigido a un equipo en otra VLAN en una trama de la VLAN en la que se encuentra el atacante por lo que el switch realiza el desencapsulado a un solo nivel y el paquete es remitido a la otra VLAN como pretendía el atacante. Aunque este ataque suele ser efectivo solo es válido en una sola dirección ya que el equipo receptor no manda su tráfico doblemente encapsulado, aun así, puede ser muy útil para un ataque de denegación de servicio.

Ambos ataques pueden ser realizados usando la herramienta *Yersinia* nombrada anteriormente en la fase de análisis de vulnerabilidades. [28]

El último paso en la fase de Post-Explotación es cubrir huellas, es decir, eliminar todo resto de nuestra presencia en el sistema. Para ello tendremos que eliminar o modificar registros de actividad (*logs*) y esconder los archivos y carpetas donde hayamos alojado las herramientas usadas.

Antes de empezar a manipular registros debemos pensar bien que pretendemos conseguir puesto que, si los eliminamos, nos aseguramos de que nuestra actividad no sea rastreable ya que al administrador del sistema le resultara muy difícil recrear nuestro ataque, esto es útil si necesitamos eliminar cualquier rastro de quien somos o de dónde venimos, pero tiene un inconveniente, ser detectado.

Esto es debido a que cuando un registro es borrado hay grandes posibilidades de que el administrador se dé cuenta, los registros no son solo usados para detectar actividades no autorizadas, sino también para conocer el estado del equipo, por lo que un administrador acudirá en primer lugar a los registros si nota algún comportamiento extraño en el equipo, por lo que si de repente hay registros ausentes o de tamaño incorrecto podría sospechar de un usuario no autorizado. [6]

Una forma rápida y automatizada de eliminar los *logs* es usar Meterpreter. Meterpreter es un *payload* perteneciente al *framework* Metasploit que además de permitir el borrado de *logs* con un solo comando también incorpora otras funciones útiles en la fase de post-explotación como, registro de pulsaciones, volcado de *hashes* de contraseñas, subida y descarga de archivos entre muchas otras. [12]

Por otra parte, si en vez de eliminar todos los *logs* nos tomamos el tiempo de modificarlos podemos borrar solo los que impliquen nuestra presencia en el equipo y mantener los que no tengan relación con nosotros, de esta forma es probable que el administrador no lo advierta con lo cual evitaríamos la detección. Sin embargo, existe el posible inconveniente de que no eliminemos todos los necesarios o que eliminemos tantos que el hueco en los *logs* sea muy visible. [6]

Otro tipo de herramienta empleada en este apartado son los *rootkits*, estos tienen múltiples utilidades que permiten desde esconder archivos, procesos o programas como si nunca hubieran sido instalados, hasta registrar teclas, además la mayoría son muy difíciles de detectar ya que funcionan a nivel de *kernel* pudiendo así evitar la detección por herramientas que funcionan a un nivel más alto del sistema.

Un conocido *rootkit* es Hacker Defender, funciona solo en sistemas Windows y además también puede usarse como *backdoor*, entre sus funciones están las anteriormente mencionadas, pero también esconder directorios enteros, partes del registro, puertos o hacer que se inicien programas en el arranque del equipo. [12]

Por último, después de conseguir toda la información posible sobre la red objetivo y sus equipos pasaremos a la última fase del test de penetración, la elaboración del informe.

2.7 ELABORACIÓN DEL INFORME

Una vez terminadas las anteriores fases de la auditoría, queda la elaboración del informe que es probablemente la que más dice de un auditor y su empresa. Es importante recordar que cuanto mejor hagamos nuestro trabajo como auditores menos percibirá el cliente nuestra presencia, por lo tanto, es común que el informe sea la única prueba tangible que el cliente recibirá del proceso de auditoría.

Una vez terminada esta, es imprescindible que presentemos nuestros descubrimientos de manera organizada y fácil de entender ya que en muchas ocasiones el cliente no sabe con seguridad que hemos estado haciendo y cuántas horas hemos invertido. Por consiguiente, el informe se convierte en el elemento para evaluar nuestra competencia. Así pues, no debemos subestimar esta fase pues, a menudo, el esfuerzo dedicado o el éxito conseguido será juzgado más por el informe que por el propio éxito o fracaso al acceder a una red.

Es una buena práctica enfatizar bajo el título de cada informe que el pentest es solo una instantánea del sistema en el momento en que se hizo la auditoría y solo es válida y correcta hasta el día en que se realizó el informe. Esto es debido a que la seguridad de redes, equipos y software es un ámbito en continuo cambio y lo que es seguro hoy puede no serlo mañana si se descubre una nueva vulnerabilidad [12].

Así pues, ¿qué debe contener un buen informe? Las diferentes metodologías dan consejos sobre cómo realizarlo y que debe incluir, sin embargo, no hay ningún estándar en la industria. La respuesta ideal sería “cualquier cosa que el cliente necesite”. Desafortunadamente, a la mayoría de los clientes les resulta tan ajena la auditoría que no saben qué esperar. Esto significa que debemos dedicar tiempo a hablar con el cliente, averiguar sus objetivos de negocio y ver cómo encajamos en su plan de seguridad global. [6]

Para empezar, cualquier informe debe contener en su portada un control de versiones y los nombres de los autores, revisores y responsables además de la fecha para poder verificar las diferentes versiones en caso de duda. El control de versiones puede ser en progresión de números naturales o bien con la nomenclatura X.X más usada en software.

Según [7] un buen informe debe contener como mínimo un informe ejecutivo y otro técnico. A continuación, explicaremos qué se espera de cada informe y algunas directrices para llevarlos a cabo.

En lo que respecta al informe ejecutivo, pretendemos comunicar a un lector no técnico los aspectos más importantes de la auditoría sin usar tecnicismos ni proporcionar detalles técnicos. Debe tener una longitud por lo general no mayor a las 2 páginas y centrarse en nuestros descubrimientos y cómo pueden afectar al negocio. Además, debe proporcionar referencias al informe técnico para que las partes interesadas puedan revisar los detalles. También es una buena idea recordar el alcance de la auditoría e incluir una estimación general del riesgo al que está sometida la organización. [12]

Por otra parte, el informe técnico debe centrarse en proporcionar todos los detalles necesarios para que los encargados de subsanar los fallos puedan hacerlo. Por lo tanto, debemos empezar mostrando los fallos que puedan suponer un riesgo más inmediato para el cliente. [12]

Clasificar las vulnerabilidades encontradas puede ser una tarea difícil, afortunadamente la mayoría de herramientas como, por ejemplo, Nessus vienen con un sistema de clasificación predeterminado. Eso hará el informe más fácil de leer y permitirá al cliente tomar acciones correctoras en primer lugar en los aspectos más graves. [12]

Por otro lado, es vital que no dejemos información relevante fuera del informe aunque no hayamos conseguido darle uso. Por ejemplo, si encontramos una vulnerabilidad crítica en uno de los *routers* perimetrales debemos informar de ella aun cuando nos haya sido imposible explotarla, ya que, aunque no hayamos podido, el sistema no deja de ser vulnerable. Asimismo, nunca debemos exagerar ni atenuar nuestros descubrimientos, nuestro objetivo es informar al cliente de una manera lo más neutra posible y dejar que él decida qué es más importante. [12]

En esta línea, también debemos tener en cuenta los problemas que estén fuera del alcance de la auditoría, los cuales podemos dividir a grandes rasgos en dos tipos. Por una parte, están los servicios o programas que encontramos en un equipo y no están dentro del alcance, así como los equipos que estén fuera del alcance, pero conectados a equipos que si están dentro. Por otra parte, están los problemas de tipo sistémico como por ejemplo contraseñas débiles en los equipos. En el primer caso, deberemos aconsejar al cliente auditar esos equipos en el futuro y en el segundo, informar para que se tomen las medidas oportunas. [6]

Otro apunte relevante es que nunca debemos utilizar capturas de pantalla de pruebas de concepto genéricas pues es peligroso y poco ético. En su lugar, debemos incluir capturas de pantalla cuando hayamos conseguido explotar con éxito una vulnerabilidad para que sirva como prueba innegable de que el error existe.

Es bueno recordar que no todas las auditorías resultarán en acceso administrativo a un sistema, ya que, en muchas ocasiones, la auditoría estará limitada por reglas externas que reducen su realidad como, el alcance, el tiempo o el presupuesto, sin olvidar otras como las restricciones legales y éticas. Habrá ocasiones en las que realizando un test de penetración no consigamos nada (ninguna vulnerabilidad descubierta, ninguna debilidad, ningún dato útil…). En estas situaciones es igualmente importante que realicemos el informe para que quede constancia de todos los vectores de ataque utilizados.

Siempre que sea posible debemos incluir en el informe sugerencias y estrategias de mitigación para corregir los problemas encontrados. Algunas herramientas nos proporcionan esta información al descubrir una vulnerabilidad, pero si no es el caso es importante que encontremos soluciones por nuestra cuenta.

Un buen sitio donde buscarlas es en las propias páginas donde encontramos los *exploits* y vulnerabilidades puesto que suelen incluir sugerencias sobre cómo arreglar el fallo. Normalmente, las soluciones son descargar un parche o actualizar una aplicación, aunque también pueden ser necesarios cambios en la configuración o mejoras en el hardware. Dar solución a cada uno de los problemas encontrados es una parte vital del informe técnico y puede ayudar a fidelizar al cliente.

Otra buena práctica es proporcionar los resultados de las herramientas utilizadas como anexo al informe y referenciarlas desde este. Procediendo así evitaremos, en gran medida, llamadas del cliente preguntando como descubrimos cierta vulnerabilidad.

Por otra parte, no debemos olvidar como entregaremos el documento al cliente. Este es un punto que debe ser tratado en la comunicación previa, pero en el que no está de más insistir llegado el momento.

En el caso de que se haya decidido entregar una copia en papel deberemos asegurarnos de imprimir y encuadernar profesionalmente y de enviar el documento mediante correo certificado para verificar que ha sido recibido. En cambio, si se ha optado por realizar él envió electrónicamente debemos asegurarnos de que el documento esté como mínimo encriptado. [12]

Finalmente, debemos tomarnos un tiempo para leer, releer y editar el informe. Es tan importante entregar un informe que sea técnicamente preciso como que esté libre de errores de gramática y ortografía. Un informe que contuviera estos errores indicaría al cliente que realizamos un trabajo poco riguroso e influiría negativamente en nuestra reputación. No debemos olvidar que se juzgará nuestro trabajo no solo por su nivel técnico sino también por su presentación y legibilidad.

Asimismo, es imprescindible tener un informe de muestra que poder enseñar al cliente porque puede influir en su decisión final de contratar nuestros servicios. Es fundamental que este informe sea realmente una muestra y no incluya datos reales de ningún tipo.

Para finalizar con esta fase, cabe mencionar que, dada la complejidad del informe, muchos clientes esperarán que estemos disponibles tras su entrega para poder realizar preguntas. Esta es una buena oportunidad para impresionar al cliente y, de este modo, conseguir futuros negocios juntos. Sin embargo, no debemos olvidar que nuestra voluntad de ayudar al cliente también debe ser rentable económicamente y que no estamos obligados a prestar nuestros servicios de forma gratuita, por lo que deberemos encontrar un equilibrio entre una atención al cliente excepcional y el beneficio económico. [12]

3.PENTEST EN OTROS SISTEMAS OPERATIVOS

Aunque el entorno Linux es el más utilizado y en el que más herramientas hay disponibles para el pentester, no podemos olvidar sistemas operativos como Windows y MacOS ya que con cierta adaptación pueden convertirse en plataformas de pentest completas. Además, al utilizar aplicaciones nativas en estos SO obtendremos un mejor rendimiento en comparación con la utilización de una máquina virtual Linux, uno de los métodos más comunes de funcionamiento.

En línea con lo anterior, hay algunas herramientas que solo tienen versión para un sistema operativo por lo que es recomendable conocer y saber utilizar varios SO. Una de estas aplicaciones es la FOCA [29] desarrollada por la empresa española Eleven Paths.

Esta herramienta es un potente analizador de metadatos, su funcionamiento consiste en realizar búsquedas en Google, Bing y Exalead para encontrar todos los archivos ofimáticos públicos de un dominio y descargarlos para obtener sus metadatos. Actualmente soporta archivos Office, Open Office y PDF entre otros. Asimismo, también permite añadir archivos locales para extraer la información.

Una vez ha obtenido toda la información posible la herramienta tratara de unirla para proporcionar datos como, servidores y equipos encontrados, software usado, rutas de archivos, nombres de usuarios, servidores DNS, etcétera. Además, puede ser ampliada mediante plugins para aumentar sus capacidades.

Por otra parte, tenemos l0phtcrack [30], esta herramienta se especializa en la recuperación de contraseñas de Windows, aunque también puede recuperar las de Linux. Además, permite el uso de hashes pre computados para acelerar el proceso e incluye una función para conocer la fuerza de una contraseña en función de la dificultad para romperla.

Otra de estas herramientas exclusivas del SO Windows es Cain & Abel [31], una herramienta de recuperación de contraseñas fuera de línea tan conocida y potente como el anteriormente nombrado John The Ripper, sin embargo, no se limita a las contraseñas, sino que también permite realizar captura de tráfico, ataques ARP Spoofing e incluso captura y reproducción de conversaciones VoIP entre otras funciones.

Además, tenemos disponible la herramienta Evil FOCA que permite realizar gran parte de las funciones de Cain & Abel, pero en el protocolo IPv6 además de IPv4. Entre otros ataques posibles tenemos el *Neighbor Advertisement Spoofing*, el ataque SLAAC o el *Rogue DHCPv6*, además de ataques DoS sobre IPv4 y IPv6. Para poder llevar a cabo estos ataques cuenta con un escáner para identificar todos los dispositivos y sus interfaces de red, así como sus direcciones físicas.

Uno de los mayores problemas para el auditor en Windows y MacOS es la necesidad de buscar y descargar una por una cada aplicación que desee utilizar. La mejor solución es descargar un pack de aplicaciones como Pentest Box Tools [32] para Windows, esta recopilación de herramientas contiene algunas de las nombradas anteriormente como los escáneres de vulnerabilidades web Burp Suite y Nikto, Nmap para la recogida de información o Metasploit para la fase de explotación.

Su interfaz es similar a la que tendríamos en Linux ya que consiste en una línea de comandos que además incluye la mayoría de utilidades incluidas por defecto como *bash*, *cat*, *ssh* o *vim*. Además, gracias a que funciona en Windows de forma nativa consigue un rendimiento óptimo, un consumo de memoria RAM mucho menor que el de

una máquina virtual con funciones comparables y es portátil por lo que podemos usarlo en cualquier equipo Windows sin necesidad de instalarlo. Asimismo, es modular por lo que podremos añadir herramientas usando la utilidad toolsmanager presente en el propio pack.

En lo que respecta a MacOS tenemos disponible K0sasp [33] con la desventaja de que contiene algunas herramientas menos que su homólogo en Windows. Este pack no incluye ninguna de las opciones con las que sí cuenta Pentest Box Tools aparte de las herramientas, aunque esto queda suplido en parte debido a que la terminal de MacOS es similar en su funcionamiento y opciones a la de Linux.

En caso de querer instalar alguna herramienta no contenida en esta recopilación y si no hemos podido encontrar un paquete preparado para instalar directamente, tenemos la opción de instalarla usando MacPorts [34] o Homebrew [35].

Estas herramientas son similares en su propósito a los gestores de paquetes existentes en Linux como APT o pacman y aunque ambas son efectivas difieren en su implementación. Comprender que las diferencia puede ayudarnos a decidir cuál usar.

Homebrew intenta utilizar siempre que sea posible las librerías y programas ya instalados por Apple. Macports por otro lado instala sus propias librerías y programas independientemente de lo instalado. Es decir, si un programa necesita Perl 5.23 para funcionar Homebrew intentara usar la versión ya instalada en el equipo, aunque no sea exactamente la necesaria, mientras que Macports realizara una instalación por separado de la versión exacta que necesite el programa en cuestión.

Por consiguiente, Homebrew instala menos software y es más tolerante a versiones diferentes mientras que Macports nos proporciona exactamente la versión necesaria a costa de una mayor complejidad. Esta última herramienta, actualmente cuenta con más de 20000 paquetes entre los que se cuentan muchas de las herramientas usadas en un pentest. Por su parte Homebrew cuenta con más 3500 paquetes y tiene una desventaja adicional ya que al depender del software instalado por Apple puede dejar de funcionar si actualizamos el SO a una nueva versión.

En la siguiente sección, hablaremos sobre como practicar el pentest de forma gratuita ya sea en línea o fuera de línea.

4.LUGARES DONDE PRACTICAR EL PENTEST

En esta sección de nuestro trabajo pretendemos proporcionar información sobre recursos disponibles gratuitamente en la red para practicar y consolidar los conocimientos adquiridos.

Uno de los sitios más conocidos para practicar online es Root-me [36] que se define como la forma fácil, rápida y barata de poner en practica nuestros conocimientos en seguridad informática. Cuenta con 50 entornos vulnerables diferentes que pueden ser desde equipos individuales a redes completas de los que además se proporciona las diferentes soluciones.

También tiene disponibles más de 200 retos de todo tipo que van desde el desarrollo de scripts para automatizar tareas hasta la obtención de acceso administrativo en un servidor Active Directory. Asimismo, cuenta con una modalidad llamada capturar la bandera en la que se propone un objetivo (por ejemplo, el *hash* de la contraseña de administrador) y varios usuarios intentan obtenerlo lo más rápido posible.

El sitio también posee un sistema de recompensas económicas para los usuarios que aporten nuevas soluciones, nuevos retos e incluso nuevos entornos. En el momento de redacción de este trabajo las recompensas variaban entre los 25 euros para desarrollar un reto relacionado con un *Reflected XSS* hasta los 500 euros por la creación de una máquina virtual ARM y diez retos asociados a ella. Actualmente cuenta con más de 46000 miembros.

Otro sitio que nos permite practicar nuestras habilidades online es PentestIt [37]. Esta plataforma en su versión gratuita nos da acceso a una red que cuenta con entre 8 y 12 equipos y servidores en los que debemos encontrar una serie de *tokens* que demuestran los diferentes grados de acceso que hemos conseguido en la red.

Esta parte gratuita varia periódicamente ya que suelen crear redes nuevas y las antiguas pasan a ser de pago, actualmente en la versión de pago se puede acceder a 6 redes distintas con un total de 55 sistemas entre equipos y servidores. Actualmente cuenta con más de 9500 usuarios registrados.

A continuación, hablaremos de la web Hack This Site [38], este sitio cuenta con un IRC y unos foros muy activo en los que hablar sobre cualquier aspecto de la seguridad informática. Además, tiene disponibles “misiones” en las que practicar nuestras habilidades con páginas web completas.

También incluye otras misiones para aprender a realizar ingeniera inversa o aprender a ver errores de programación. Actualmente están reconstruyendo RootThisBox un apartado del sitio en el que individuos o grupos configuran un equipo para ser usados como objetivo para que el resto de usuarios o equipos intenten penetrar en ellos.

Por último, hablaremos de PentesterLab [39], este sitio está centrado en el hacking web y proporciona cursos sobre diferentes temas desde cómo usar un exploit en concreto hasta como, a través de inyección SQL, obtener acceso a la consola de administración y al sistema.

Además, para la parte práctica de estos cursos se proporcionan maquinas vulnerables en las que practicar los conocimientos adquiridos. Este sitio cuenta también con una versión de pago que proporciona acceso a mas cursos y también a videos que complementan las explicaciones por escrito.

Asimismo, cuenta con una sección llamada Bootcamp en la que detallan los conocimientos necesarios para iniciarse en la seguridad informática y especialmente en la auditoría web. Esta sección está enfocada desde una perspectiva practica en la que para cada conocimiento a adquirir se sugieren diferentes tareas a llevar a cabo.

Por otra parte, existen páginas web como VulnHub [40] donde podemos encontrar una recopilación de decenas de máquinas virtuales vulnerables para descargar y auditar sin necesidad de estar conectados. En esta página podemos encontrar máquinas para practicar casi cualquier aspecto de la auditoría desde la recogida de información hasta la explotación de servidores web.

Asimismo, muchas de ellas tienen una historia de fondo que nos va guiando de una maquina a otra conforme las vamos completando, un ejemplo sería la serie de máquinas De-ICE que contiene 6 máquinas en orden de dificultad. Además, la mayoría de máquinas del sitio pueden explotarse de varias formas.

Entre las máquinas virtuales disponibles en esta web nos gustaría destacar Net in VM por su carácter innovador ya que permite emular una red entera en una sola máquina virtual utilizando *User Mode Linux*. Esta red está dividida en 3 subredes con un total de 11 equipos.

Este sitio también incluye una guía sobre como montar un laboratorio casero para practicar la auditoría y una sección con recursos útiles como libros y cursos de pentest y programación. Su única desventaja es que todas las maquinas usan sistemas operativos libres como Linux por lo que si estamos interesados en practicar nuestras habilidades en sistemas Windows deberemos buscar en otro lugar.

Así pues, el mejor sitio para obtener máquinas virtuales Windows es la web para desarrolladores de Microsoft [41]. Estas máquinas originalmente pensadas para probar las diferentes versiones de Internet Explorer y Microsoft Edge en los diferentes sistemas operativos son muy útiles ya que están preparadas para su uso como máquina virtual y pueden convertirse con poco esfuerzo en buenos objetivos para la práctica del pentest.

Por otro lado, desaconsejamos la descarga y utilización de versiones de Windows no originales ya que es difícil saber si han sido modificadas con algún propósito oculto y pueden provocar graves daños a nuestro sistema.

Actualmente Apple no da soporte para virtualizar MacOS y remarca que no debe instalarse en otro hardware que no sea el suyo, por lo que la única forma legal de hacer pentest a este SO sería comprar un equipo de esta compañía. Existen otras opciones como instalar el SO en un equipo no Apple [42] o en una máquina virtual modificada, pero estaríamos incumpliendo los términos de la licencia al hacerlo.

A continuación, realizaremos una pequeña auditoría en la que aplicaremos los conocimientos adquiridos durante el transcurso de este trabajo.

5. PRUEBAS

En este apartado usaremos los conocimientos adquiridos durante la realización del trabajo para llevar a cabo una pequeña auditoría. Para ello, utilizaremos dos máquinas virtuales elegidas aleatoriamente en el sitio web Vulnhub y seguiremos las fases definidas por el PTES para conseguir acceso administrativo en ellas.

5.1 METODOLOGÍA UTILIZADA

Para la realización del pentest seguiremos la metodología y herramientas definidas en este trabajo y explicaremos los pasos seguidos durante su realización. Nuestro objetivo es conseguir permisos de administrador ya que así tendremos control total sobre el equipo.

5.2 DESCRIPCIÓN DEL BANCO DE PRUEBAS

El banco de pruebas consta de:

- Una máquina virtual con la distribución Kali Linux
- Dos máquinas virtuales elegidas aleatoriamente en Vulnhub

La primera máquina será desde la que realizaremos la auditoría y las otras dos máquinas serán nuestro objetivo. Dado que no es una auditoría real no podremos llevar a cabo algunas de sus partes como el contacto inicial con el cliente o el modelado de amenazas ya que requieren la intervención de la empresa auditada. Por el mismo motivo la auditoría empezará en el interior de la red de la empresa ficticia.

5.3 PRUEBAS REALIZADAS

Para empezar con la fase de recogida de información utilizamos la herramienta Nmap indicándole que escanee la red 192.168.1.0/24 en busca de equipos, como resultado obtenemos las ilustraciones inferiores.

En la figura 2 podemos ver cuáles de entre los 1000 puertos más comunes están abiertos en el equipo 192.168.1.100. Tenemos varios servidores de correo, un servidor FTP, un servidor SSH y un servidor HTTP.

Figura 2:Escaneo básico Nmap 192.168.1.100

En la figura 3 vemos que en el equipo 192.168.1.108 existe un servidor SSH, un servidor HTTP y un servidor IRC.

```
Nmap scan report for 192.168.1.8
Host is up (0.00011s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
6667/tcp open  irc
MAC Address: 08:00:27:57:4F:AA (Oracle VirtualBox virtual NIC)
```

Figura 3:Escaneo básico Nmap 192.168.1.108

Para continuar con la recogida de información una vez identificados los equipos y habiendo realizado un escaneo básico de sus puertos pasaremos a centrarnos en cada equipo por separado, empezando por el 192.168.1.100.

El primer sitio donde debemos buscar información es en el servidor web que tras acceder nos muestra la página web que podemos ver en la figura 4.

Figura 4:Pagina web en puerto 80 192.168.1.100

De esta página nos interesan principalmente las direcciones de correo electrónico ya que el resto de la página es una broma. Especialmente interesantes son las direcciones del final ya que corresponden a los administradores del servidor por lo que probablemente sus correos coincidan con sus nombres de usuario y nos ayuden a acceder al equipo.

Como comentamos en la fase de recogida de información, es importante que hagamos listas con los correos electrónicos relacionados con la empresa y con variaciones comunes de ellos. Por ejemplo, poniendo la última letra delante o viceversa.

Otro sitio donde podemos buscar información es en el servicio SMTP que hemos nombrado anteriormente. SMTP dispone de tres opciones que pueden sernos útiles para descubrir que usuarios existen en el equipo. La primera de ella es EXPN que nos mostraría una lista de todos los miembros de esa lista de correo, lamentablemente esta desactivada. La segunda es VRFY que al introducir una dirección de correo nos dice si existe o no en ese servidor, por desgracia también esta desactivada.

La última opción es RCPT que sí está activa. Para no introducir una a una las direcciones de correo y sus variaciones podemos usar la herramienta smtp-user-enum que nos permite usar un archivo que contenga las direcciones de correo que queramos probar, el comando que hemos utilizado es: *smtp-user-enum -M RCPT -f user@slax.example.net -D slax.example.net -U usuariosycon.txt -t 192.168.1.100.* Y su resultado ha sido el que podemos ver en la figura 5:

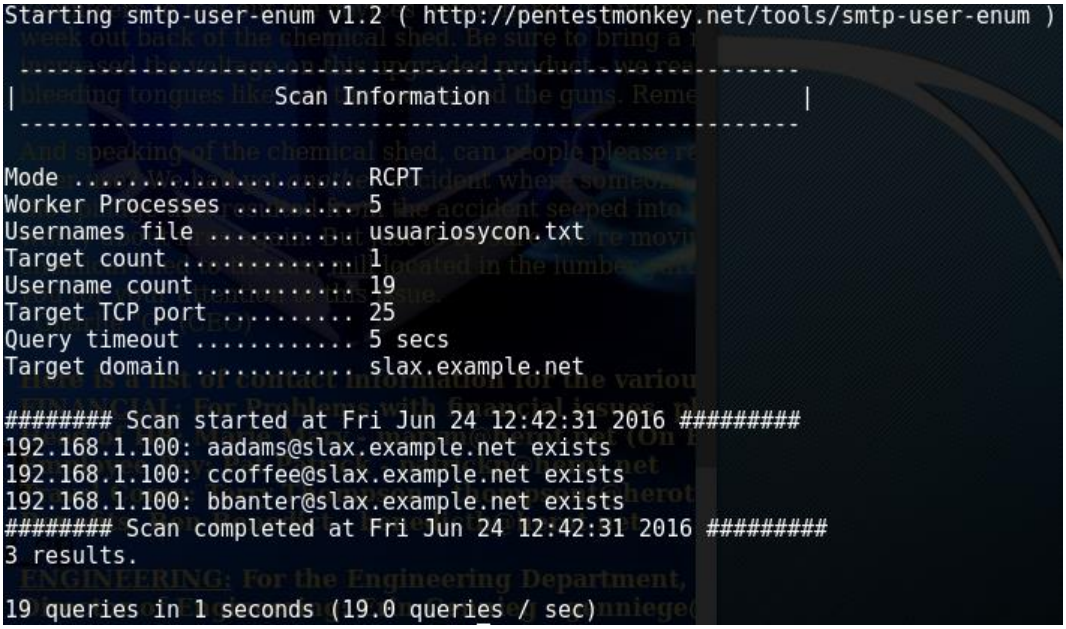

Figura 5:Resultado herramienta smtp-user-enum 192.168.1.100

Como podemos ver, existen en el servidor tres direcciones de las que tenemos en nuestra lista, que contiene tanto las direcciones conseguidas en la web como variaciones de ellas.

El siguiente paso es la búsqueda de vulnerabilidades por lo que realizaremos un escaneo en profundidad del equipo elegido incluyendo todos sus puertos, las versiones de software que están escuchando en ellos y el SO que utiliza.

Para ello volveremos a utilizar Nmap, pero esta vez utilizando los comandos necesarios para obtener la información que queremos. El comando usado es: *nmap -T 5 -A -p1-65535 192.168.1.100.*

Para ampliar información sobre que hace cada *flag* podemos recurrir al manual de Nmap disponible dentro de la propia herramienta, esto último es aplicable a todas las herramientas nombradas. El resultado obtenido es el que podemos ver en la figura 6.

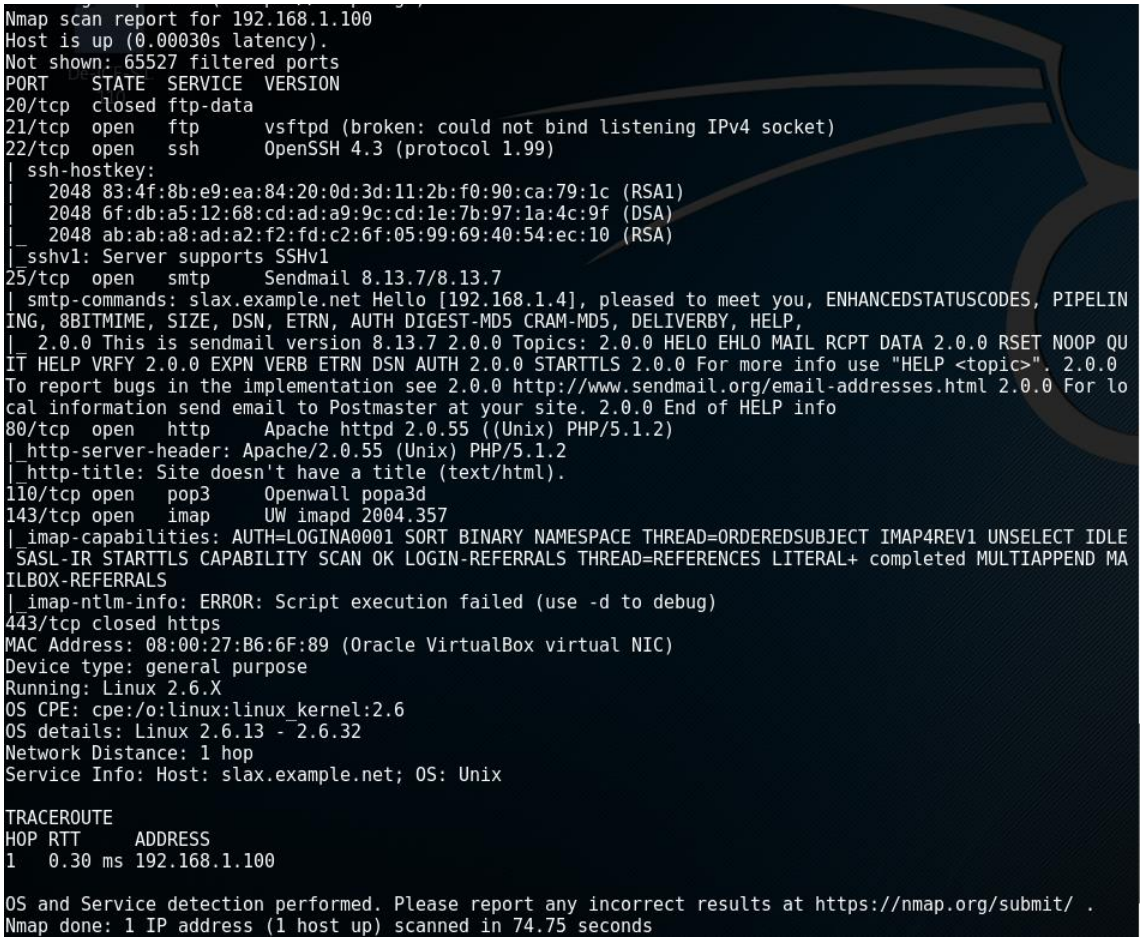

Figura 6: Escaneo en profundidad Nmap 192.168.1.100

Una vez investigadas las versiones de software funcionando en el equipo en busca de vulnerabilidades en NVD y Bugtraq podemos extraer algunas conclusiones:

Existe algún problema con el servidor FTP del puerto 21 ya que Nmap marca que está roto, tras usar Ncat confirmamos que es imposible conectarse a él.

Además, vemos que en el puerto 22 hay un servidor SSH, este podría ser una vía de entrada al equipo ya que acepta conexiones. Además, Nmap señala que es compatible con SSHv1 que es una versión obsoleta para la que existía una vulnerabilidad con su exploit correspondiente pero que ya había sido subsanada para esta versión del programa.

Por otra parte, tenemos el servidor SMTP en el puerto 25 que ya hemos utilizado para realizar la enumeración de usuarios existentes en el equipo y que no contiene ninguna vulnerabilidad que nos proporcione acceso al equipo.

Por otro lado, Nmap señala el servidor web Apache en el puerto 80. En principio no hay ninguna vulnerabilidad que nos dé acceso al equipo para esa versión del servidor. Aunque ya nos ha sido útil para obtener direcciones de correo.

Asimismo, tras investigar sobre los programas escuchando en los puertos 110 y 143 correspondientes a POP e IMAP vemos que no existen vulnerabilidades conocidas que nos sean útiles.

Para finalizar, descubrimos que el equipo utiliza un SO Linux e investigando encontramos que no existe ninguna vulnerabilidad de interés para esa versión del *kernel*.

Antes de continuar con la siguiente fase utilizaremos el escáner de vulnerabilidades OpenVAS para comprobar si hemos pasado por alto alguna vulnerabilidad que nos pudiera dar acceso al equipo. Los resultados de la herramienta son los mostrados en la figura 7:

Vulnerability	Severity	QoD	Host	Location
Sendmail NULL Character CA SSL Certificate Validation Security Bypass Vulnerability	7.5 (High)	80%	192.168.1.100	25/tcp
phpinfo() output accessible	7.5 (High)	80%	192.168.1.100	80/tcp
http TRACE XSS attack	5.8 (Medium)	99%	192.168.1.100	80/tcp
SSH Weak Encryption Algorithms Supported	4.3 (Medium)	95%	192.168.1.100	22/tcp
Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability	4.3 (Medium)	99%	192.168.1.100	80/tcp
TCP timestamps	2.6 (Low)	80%	192.168.1.100	general/tcp
SSH Weak MAC Algorithms Supported	2.6 (Low)	95%	192.168.1.100	22/tcp

Figura 7: Resultado OpenVAS 192.168.1.100

Ninguna de las vulnerabilidades encontradas permite acceso al equipo, aunque si estuviéramos en una red real las dos más graves podrían proporcionarnos información importante para continuar el ataque. La primera permite un ataque MITM y la segunda acceder al archivo *info.php* que contiene información como el nombre del usuario que instaló PHP, si este tenía permisos de administrador, la versión del SO utilizado…

Con los datos conseguidos en las fases de recogida de información y análisis de vulnerabilidades pasaremos a la fase de explotación obviando la de modelado de amenazas ya que el único vector de ataque posible es intentar autenticarnos en el servicio SSH con los usuarios obtenidos anteriormente.

Para automatizar el proceso de autenticación utilizaremos la herramienta Hydra en la que introduciremos la lista de nombres de usuarios que según SMTP existen en el servidor. Para empezar, comprobaremos si alguno de los usuarios ha utilizado su nombre de usuario como contraseña. Para ello usaremos el comando: *hydra -t 4 -L usuariosycon.txt -P usuariosycon.txt ssh://192.168.1.100:22*. Con ello obtenemos el resultado mostrado en la figura 8.

```
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations, o
r for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-06-24 19:37:31
[DATA] max 4 tasks per 1 server, overall 64 tasks, 361 login tries (l:19/p:19), ~1 try per task
[DATA] attacking service ssh on port 22
[STATUS] 223.00 tries/min, 223 tries in 00:01h, 138 todo in 00:01h, 4 active
[22][ssh] host: 192.168.1.100   login: bbanter   password: bbanter
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-06-24 19:39:13
```

Figura 8:Resultado Hydra Usuario=Contraseña 192.168.1.100

Como podemos ver *bbanter* uno de los usuarios del sistema ha utilizado su nombre de usuario como contraseña con lo que hemos conseguido acceso al equipo. Ahora procederemos a autenticarnos en él y ver que permisos tenemos.

Tras entrar al equipo e intentar leer el *shadow file* utilizando el siguiente comando: *cat /etc/shadow* descubrimos que no podemos acceder por lo que no podemos obtener el *hash* de la contraseña de administrador. Al intentarlo usando el comando *sudo* el equipo nos informa que el usuario actual no tiene permiso para ejecutar acciones como administrador por lo que deberemos buscar otro método.

A continuación, intentamos acceder al archivo */etc/passwd* en el que se almacenan los nombres de usuario, aunque no las contraseñas, de esta forma confirmamos que además de *bbanter* existen también los usuarios *aadams*, *ccoffee* tal y como puede verse en la figura 9 y como ya señalaba el servicio SMTP.

Figura 9: Usuarios existentes 192.168.1.100

Dado que no sabemos si alguno de ellos tiene acceso administrativo utilizamos el comando *group* seguido del nombre de usuario. Así descubrimos que *ccoffee* forma parte del grupo *users* y *aadams* del grupo *wheel*. Normalmente los usuarios del grupo *wheel* tienen la capacidad de utilizar *sudo* por lo que intentaremos autenticarnos con el usuario *aadams* usando Hydra.

En esta ocasión utilizaremos el usuario *aadams* y uno de los diccionarios que incluye Kali Linux llamado *rockyou.txt* que contiene las contraseñas más usadas alrededor del mundo obtenidas de diferentes filtraciones. Para ello utilizaremos Hydra para autenticarnos en SSH con el comando: *hydra -f -t 64 -vV -l aadams -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.100:22*. Tras más de una hora obtenemos que la contraseña para el usuario *aadams* es "nostradamus".

Procedemos a autenticarnos en el equipo con el usuario y contraseña obtenidos y tratamos de leer otra vez el *shadow file* usando el comando *sudo*. En esta ocasión tenemos éxito y se nos muestra los hashes de las contraseñas de todos los usuarios entre ellos las del usuario *root* como puede verse en la figura 10.

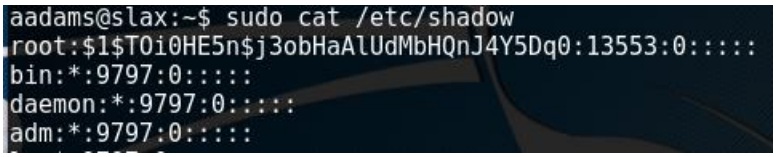

Figura 10: Hash administrador 192.168.1.100

Una vez obtenido acceso al *shadow file* solo nos resta obtener la contraseña de administrador para terminar con la fase de explotación. Para ello copiamos el *hash* del administrador a un archivo de texto y procedemos a recuperar la contraseña utilizando la herramienta John The Ripper en la cual usaremos el siguiente comando: *John –wordlist=/usr/share/wordlists/rockyou.txt passadmin.txt*. Tras media hora *John* nos devuelve la contraseña "*tarot*" por lo que podemos pasar a la fase de post-explotación al haber obtenido privilegios administrativos completos.

Tal y como comentamos anteriormente, el primer paso en la fase de post-explotación es el mantenimiento del acceso, pero dado que no hemos accedido al equipo mediante el uso de un *exploit* no necesitamos utilizar un *backdoor* ya que podemos usar el protocolo SSH.

El segundo paso es obtener la máxima cantidad de información del equipo por si pudiera ser útil posteriormente. Al ser una máquina virtual sabemos que sus interfaces de red no nos descubrirán ninguna subred nueva, pero si no fuera el caso deberíamos comprobarlo usando los comandos *ifconfig* o *ip a*. Asimismo debemos comprobar si existen puertos abiertos que no hayan sido descubiertos por Nmap. Para ello estando

dentro de la maquina atacada usamos el comando: *nmap -A localhost*. Este nos muestra los resultados de la figura 11.

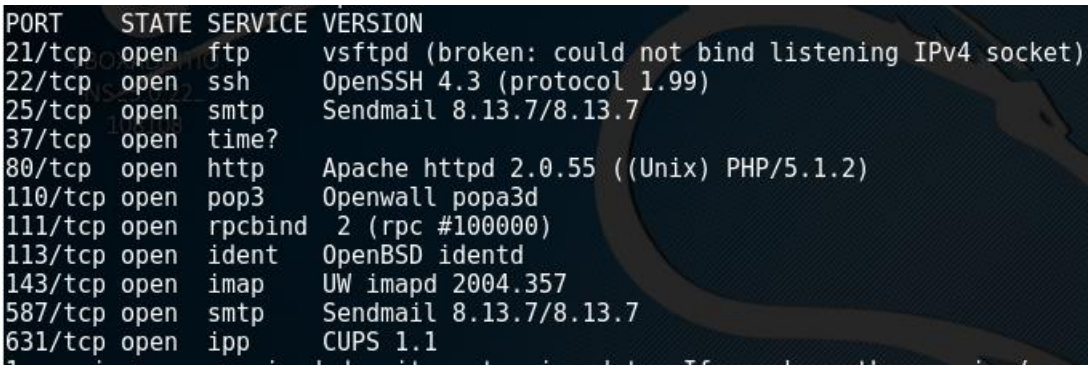

Figura 11: Escaneo Nmap desde dentro maquina objetivo

Como podemos ver hay cuatro puertos que Nmap no había detectado anteriormente por lo que procedemos a investigarlos.

Aparentemente, el servicio *time* en el puerto 37 es un protocolo precursor de NTP que actualmente está en desuso. Por su parte el servicio *rpcbind* también conocido como *port mapper* podría decirnos que otros servicios se encuentran en funcionamiento en la maquina objetivo, pero no es visible fuera del propio equipo. En lo que respecta al servicio *OpenBSD identd* del puerto 113 da acceso al protocolo del mismo nombre que identifica al usuario de una conexión TCP en particular. Ninguno de estos servicios tiene vulnerabilidades que pudieran permitir acceso al equipo.

Finalmente, el servicio *CUPS 1.1* corresponde a un servidor de impresión y tiene una vulnerabilidad conocida con número CVE-2004-1267. Esta vulnerabilidad, que además cuenta con un *exploit*, permitiría ejecutar código arbitrario por lo que podría obtenerse acceso administrativo si se utilizara. Lamentablemente este servicio no es visible desde fuera de la máquina y no ha podido ser explotado.

El último paso de la fase de post-explotación es el borrado de huellas. En nuestro caso dado que la discreción no es importante bastaría con borrar los *logs* del equipo localizados en la carpeta */var/logs*.

Ahora continuaremos con la recogida de información en la maquina con IP 192.168.1.108. Para ello visitaremos el servidor web que aparecía en nuestro primer escaneo con Nmap. Navegando por el sitio no encontramos ningún correo electrónico que pueda sernos útil para autenticarnos en el servicio SSH ni nada extraño hasta que llegamos a la pestaña *Documentatión* mostrada en la figura 12.

Figura 12: Pestaña Documentation

Una sección de la web vacía y con fondo negro nos hace sospechar que debe haber algo oculto por lo que echamos una ojeada al código HTML. Tras leer detenidamente el código en busca de algo fuera de lo común encontramos unas líneas escondidas en él tal y como muestra la figura 13.

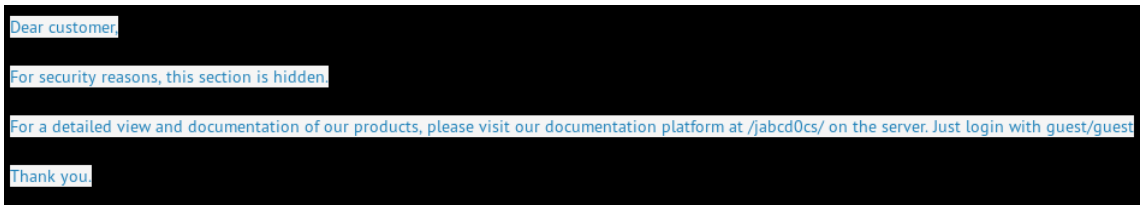

Figura 13: Pestaña Documentation oculta

Parece ser que hay una sección oculta de la web accesible en *http://192.168.1.8/jabcd0cs/* para la que además nos proporcionan un usuario y contraseña por lo que procedemos a visitarla. En la página de *login* de la sección oculta visible en la figura 14 observamos que utiliza *OpenDocMan* versión 1.2.7 una herramienta para gestión de documentos. Dado que existe una página de *login* es fácil suponer que existe una base de datos subyacente que podría ser vulnerable a ataques de inyección.

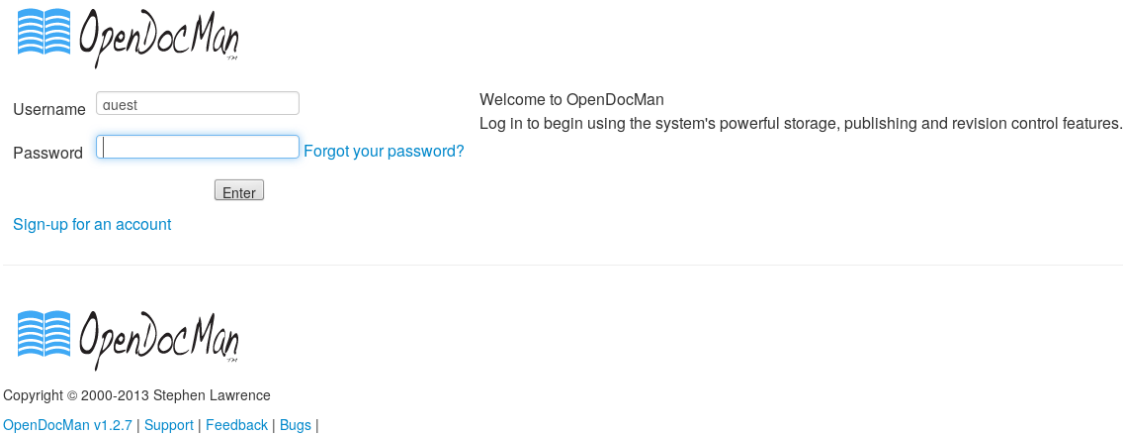

Figura 14: Pagina login sección oculta

Procedemos a autenticarnos con el usuario y contraseña proporcionados y observamos la página mostrada en la figura 15.

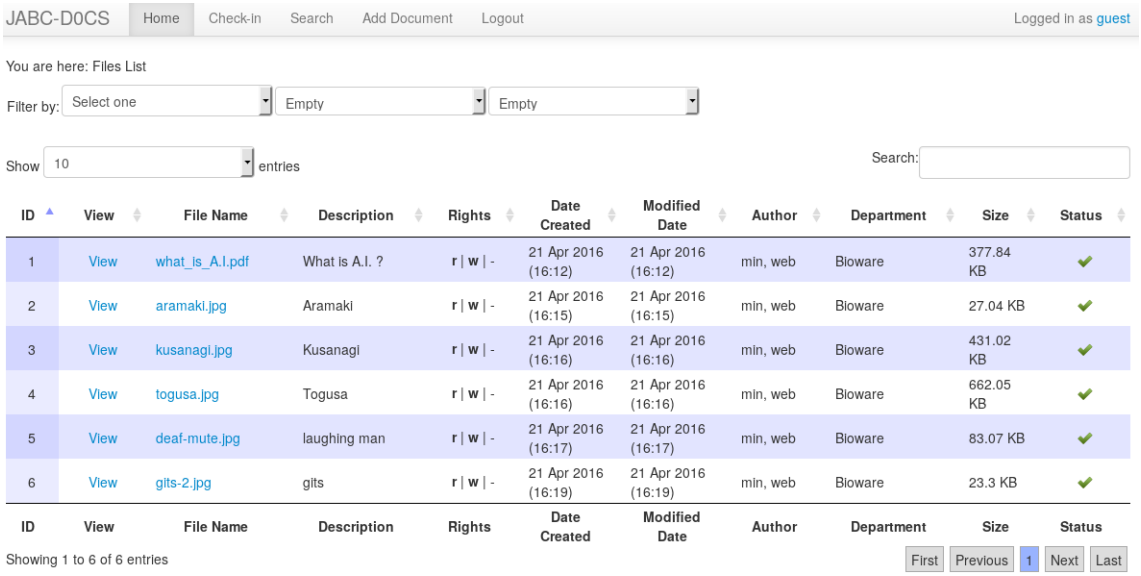

Figura 15: Página principal OpenDocMan

Como podemos ver hay varias imágenes y un archivo PDF ninguno de los cuales nos proporciona ninguna información. Sin embargo, observamos la pestaña *Add Document* que nos permite subir archivos al servidor.

Una vez hemos terminado de recoger información en la página web realizaremos un escaneo más detallado del equipo utilizando Nmap con el comando: *nmap -T 5 -A -p1-65535 192.168.1.8* lo que nos devuelve como resultado la figura 16.

```
Starting Nmap 7.12 ( https://nmap.org ) at 2016-06-28 12:49 CEST
Nmap scan report for 192.168.1.8
Host is up (0.00064s latency).
Not shown: 65532 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 f5:4d:c8:e7:8b:c1:b2:11:95:24:fd:0e:4c:3c:3b:3b (DSA)
|   2048 ff:19:33:7a:c1:ee:b5:d0:dc:66:51:da:f0:6e:fc:48 (RSA)
|_  256 ae:d7:6f:cc:ed:4a:82:8b:e8:66:a5:11:7a:11:5f:86 (ECDSA)
80/tcp   open  http    Apache httpd 2.4.7 ((Ubuntu))
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: VulnOSv2
6667/tcp open  irc     ngircd
MAC Address: 08:00:27:57:4F:AA (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.4
Network Distance: 1 hop
Service Info: Host: irc.example.net; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figura 16: Nmap detallado 192.168.1.8

Tras investigar las versiones de los servicios descubrimos que el servidor SSH no tiene ninguna vulnerabilidad que pueda darnos acceso al equipo y del IRC tenemos únicamente el nombre del programa y no el número de versión por lo que no podemos asegurar nada. Dado que en la página web hemos encontrado dos posibles vectores de

ataque (la página de *login* y la capacidad de subir archivos) realizaremos un escaneo más específico en busca de vulnerabilidades web.

El escaneo de vulnerabilidad web usando Nikto no nos devuelve ninguna vulnerabilidad relevante por lo que solo nos queda comprobar si la página de *login* en OpenDocMan descubierta anteriormente es vulnerable a *SQL Injection*. Para ello buscamos el programa usando la herramienta searchsploit que realiza una búsqueda en una copia local de la base de datos exploit-db y que nos devuelve el resultado mostrado en la figura 17.

```
root@kali:~# searchsploit opendocman 1.2.7
------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                          |  Path
                                                                        | (/usr/share/exploitdb/platforms)
------------------------------------------------------------------------ ---------------------------------
OpenDocMan 1.2.7 - Multiple Vulnerabilities                             | ./php/webapps/32075.txt
------------------------------------------------------------------------ ---------------------------------
```

Figura 17: Resultado searchsploit

Indica que existen múltiples vulnerabilidades en esta versión por lo que accedemos al archivo que nos indica. En la figura 18 vemos que existen dos vulnerabilidades y que una de ellas es de tipo *SQL Injection* situada en */ajax_udf$* y más concretamente en *add_value*.

```
1) SQL Injection in OpenDocMan: CVE-2014-1945

The vulnerability exists due to insufficient validation of "add_value" HTTP GET parameter in "/ajax_udf$

The exploitation example below displays version of the MySQL server:

http://[host]/ajax_udf.php?q=1&add_value=odm_user%20UNION%20SELECT%201,v
ersion%28%29,3,4,5,6,7,8,9
```

Figura 18:Descripción SQL Injection 192.168.1.8

Así pues, adaptamos la URL proporcionada a nuestro caso y entramos en: http://192.168.1.8/jabcd0cs/ajax_udf.php?q=1&add_value=odm_user y obtenemos la figura 19.

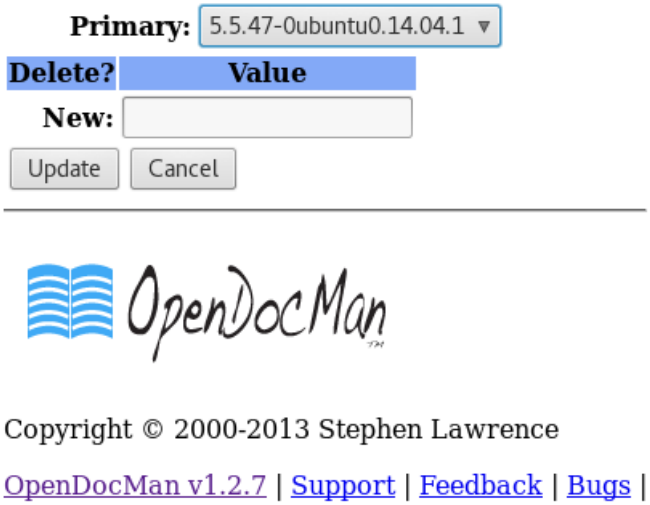

Figura 19: Base de datos vulnerable a SQL Injection

Una vez hemos descubierto la vulnerabilidad solo nos queda explotarla para ver qué información útil podemos extraer de la base de datos. Para ello utilizaremos la herramienta sqlmap que nos permite automatizar las consultas a la base de datos. Para

empezar, usamos sqlmap con las opciones por defecto para descubrir que base de datos está en funcionamiento en el equipo. Como resultado obtenemos la figura 20.

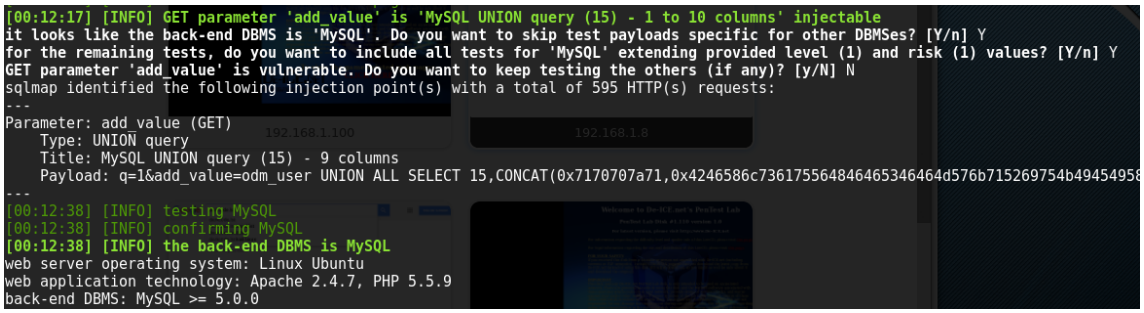

Figura 20:Resultado sqlmap 192.168.1.8

Como podemos ver la base de datos usada es MySQL en una versión igual o superior a la 5.0. A continuación consultamos cuantas bases de datos hay disponibles en el servidor con el comando:

sqlmap -u "http://192.168.1.8/jabcd0cs/ajax_udf.php?q=1&add_value=odm_user" -p add_value --dbs

Lo que nos devuelve seis bases de datos tal y como podemos ver en la figura 21.

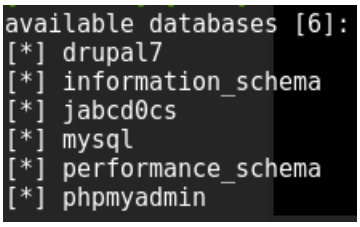

Figura 21: Lista bases de datos 192.168.1.8

A continuación, consultamos las tablas existentes en cada base de datos en busca de algún dato de interés. No encontramos nada excepto en la tabla odm_user de la base de datos jabcd0cs en la que obtenemos la figura 22.

Figura 22:Tabla user_odm de jabcd0cs 192.168.1.8

Como podemos ver, en esta tabla se almacenan todos los datos de los usuarios incluyendo los *hashes* de las contraseñas. Aparece el usuario *guest* del que ya tenemos la contraseña y el usuario *webmin* que suponemos es el administrador del sitio web.

Tras intentar recuperar la contraseña utilizando John The Ripper y el diccionario *rockyou.txt* disponible en Kali no conseguimos la contraseña. Procedemos a intentar recuperarla utilizando el diccionario *sqlmap.txt* que incluye la propia herramienta también sin éxito.

Dado que las contraseñas más conocidas no tienen éxito debemos haber pasado por alto algún dato en las bases de datos conseguidas o en la página web por lo que procedemos a buscar más en profundidad en ambos sitios. Tras la búsqueda, el único dato

que parece puesto intencionadamente es la frase que podemos ver en la figura 23 situada en la parte de debajo del sitio web.

Figura 23: Dato extraño 192.168.1.8

Puesto que no hemos encontrado ningún otro dato de interés procedemos a crear un diccionario con combinaciones de esas palabras como *since1980*, *vulnerable1980*, *vulnerablesince* y combinaciones de estas palabras con las contraseñas del diccionario sqlmap.txt que son las más utilizadas en bases de datos.

Tras volver a lanzar John The Ripper con el diccionario creado obtenemos la contraseña mostrada en la figura 24.

```
root@kali:~/Desktop# john --format=raw-md5 --wordlist=wordlistmodificada.txt hashwebmin.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 AVX 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
webmin1980       (?)
1g 0:00:00:00 DONE (2016-06-29 13:22) 6.250g/s 7248Kp/s 7248Kc/s 7248KC/s webmedia..webmultitech
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figura 24:Password webmin recuperado por John 192.168.1.8

La contraseña obtenida es webmin1980 que es una combinación del texto encontrado y del usuario *webmin*. Una vez tenemos este dato y al no haber ningún otro vector de ataque procedemos a autenticarnos en SSH con este usuario y contraseña. Conseguimos acceder al equipo, pero no podemos acceder al *shadow file* ya que no tenemos permisos de administrador y no podemos usar *sudo*.

Tras investigar los archivos del equipo no encontramos nada de interés por lo que procedemos a investigar que versión del SO utiliza usando el comando: *uname -r* que nos devuelve que el núcleo Linux usado es la versión *3.13.0-24-generic* por lo que procedemos a buscar una vulnerabilidad.

Descubrimos que existe un exploit local disponible en exploit-db con número 37292 para obtener una *shell* con permisos *root* que aprovecha la vulnerabilidad CVE-2015-1328. Así pues, descargamos el *exploit* en la maquina atacada usando el comando *wget* y procedemos a compilarlo tal y como indica su creador. A continuación, lo ejecutamos y conseguimos la *shell* con permiso de administrador y procedemos a acceder al *shadow file* y al *hash* de administrador obteniendo la figura 25.

```
# cat /etc/shadow
root:$6$FUG2g0oZ$KSa0C5cB4IZKYUxSAlTHW3XpUXLNaZZOcMzw5Vj0t3JEVZ9rHMPmsWUkWSxV.c4FnOEt.BCW/e7GWQZyNLbXD.:
16923:0:99999:7:::
```

Figura 25: Hash administrador 192.168.1.8

Dado que ya tenemos permisos de administrador gracias al *exploit* no es necesario recuperar la contraseña y podemos pasar a la fase de post-explotación.

Como en la maquina anterior, al haber accedido mediante SSH no necesitamos instalar un *backdoor* para mantener el acceso. Pasamos pues a la recogida de información en el equipo que en este caso no devuelve ningún dato nuevo ni ningún puerto que *Nmap* no hubiera detectado por lo que solo restaría borrar los *logs* para dar por terminado el pentest.

6. CONCLUSIONES

6.1 CUMPLIMIENTO DEL OBJETIVO

El principal objetivo de este trabajo era esbozar las siete fases de un test de penetración siguiendo la metodología mostrada en el *Penetration Testing Execution Standard* y proporcionar información actual y contrastada de las técnicas y herramientas utilizadas durante un pentest profesional y a qué fase pertenecen cada una de ellas.

Creemos haber cumplido el objetivo, ya que nuestras fuentes son, en su mayoría, libros escritos por profesionales y docentes del sector de la auditoría de seguridad con una antigüedad menor por lo general a los cuatro años. Además, hemos descrito las técnicas y herramientas más utilizadas para cada una de las fases.

Otro de nuestros objetivos era encontrar y recopilar lugares donde practicar el pentest de forma gratuita -ya fuera en línea o fuera de línea-. También hemos cumplido este objetivo, puesto que hemos proporcionado abundantes lugares donde poder practicar el pentest, entre ellas el sitio web Vulnhub del que hemos extraído las máquinas virtuales para nuestra sección de pruebas.

Por último, nuestro objetivo de realizar una pequeña auditoría para comprobar que la metodología, técnicas y herramientas descritas eran correctas ha sido cumplido, dado que hemos conseguido acceso administrativo a los dos equipos preparados para ello siguiendo en todo momento las fases anteriormente descritas.

6.2 PROBLEMAS ENCONTRADOS

Nuestro primer problema ha consistido en definir con qué nivel de detalle abordar la auditoría, ya que dada la amplitud y profundidad del tema es imposible hablar de todo. Puesto que la extensión del trabajo es limitada, hemos preferido tratar muchos temas con poca profundidad y proporcionar una bibliografía lo más completa posible con la que ampliar detalles.

El otro problema ha sido de índole técnica, pues llevar la teoría a la práctica siempre entraña una serie de dificultades que se tienen que ir subsanando.

Detrás de la parte práctica del trabajo hay muchas más horas e investigación de las que pudiera parecer ya que las herramientas tienen multitud de opciones y no siempre es fácil saber cómo se utiliza una en particular. Para ello han resultado muy útiles los manuales incluidos con las herramientas, aunque frecuentemente ha sido necesaria una búsqueda más exhaustiva para descubrir cómo lograr el resultado esperado.

6.3 CONCLUSIONES SOBRE EL PROYECTO

Como hemos visto a lo largo de este trabajo, hay muchísimas formas de acceder a un equipo y de obtener información relevante sobre una empresa. En multitud de ocasiones es debido a la negligencia de los administradores de los sistemas (por ejemplo, al no actualizar las aplicaciones o al utilizar la misma contraseña de administración en todos los equipos). Sin embargo, aun haciendo correctamente lo anterior no estamos seguros.

Ningún sistema de seguridad es infalible y, en muchas ocasiones, ocurre lo que ya dijo el famoso hacker Kevin Mitnick “Las organizaciones gastan millones de dólares en firewalls y dispositivos de seguridad, pero tiran el dinero porque ninguna de estas medidas

cubre el eslabón más débil de la cadena de seguridad: la gente que usa y administra los ordenadores".

La única solución posible es la realización de auditorías periódicas y la creación de una cultura de la seguridad para concienciar a todos los empleados de los riesgos a los que se exponen.

BIBLIOGRAFÍA

[1] D. Ruiz Marull, «La Vanguardia - ¿Quién filtró los papeles de Panamá?,» 04 04 2016. [En línea]. Available: http://www.lavanguardia.com/economia/20160404/40865196126/papeles-panama-filtracion-mossack-fonseca.html. [Último acceso: 06 06 2016].

[2] D. Leigh, «El Pais - La era de las filtraciones,» 29 11 2015. [En línea]. Available: http://internacional.elpais.com/internacional/2015/11/26/actualidad/1448548222_604295.html. [Último acceso: 08 06 2016].

[3] EFE / 20MINUTOS, «20 minutos - Cronología del 'caso Snowden', el joven que reveló el espionaje masivo de Estados Unidos,» 07 07 2013. [En línea]. Available: http://www.20minutos.es/noticia/1850380/0/caso-snowden/cronologia/espionaje-ee-uu/. [Último acceso: 08 06 2016].

[4] R. Jiménez Cano, «El Pais - Un ataque informático paraliza Sony Pictures,» 25 11 2014. [En línea]. Available: http://tecnologia.elpais.com/tecnologia/2014/11/25/actualidad/1416904284_635758.html. [Último acceso: 01 06 2016].

[5] Instituto Nacional de Ciberseguridad de España, «Titulares de ciberseguridad del 2015,» 28 01 2016. [En línea]. Available: https://www.incibe.es/blogs/post/Seguridad/BlogSeguridad/Articulo_y_comentarios/Titulares_de_ciberseguridad_del_2015. [Último acceso: 07 06 2016].

[6] T. Wilhelm, Professional Penetration Testing, 2nd Edition, Waltham: Syngress, 2013.

[7] «The Penetration Testing Execution Standard,» 16 08 2014. [En línea]. Available: http://www.pentest-standard.org/. [Último acceso: 20 02 2016].

[8] G. Weidman, Penetration Testing: A Hands-On Introduction to Hacking, San Francisco: No Starch Press, Inc., 2014.

[9] J. Faircloth, Penetration Tester's Open Source Toolkit, 3rd Edition, Waltham: Syngress, 2011.

[10] P. González Pérez , G. Sánchez Garcés y J. M. Soriano de la Cámara, Pentesting con Kali 2.0, Mostoles: 0xWORD Computing S.L, 2015.

[11] Instituto Nacional de Ciberseguridad de España, «OSINT - La información es poder,» 28 05 2014. [En línea]. Available: https://www.incibe.es/blogs/post/Seguridad/BlogSeguridad/Articulo_y_comentarios/osint_la_informacion_es_poder. [Último acceso: 01 04 2016].

[12] P. Engebretson, The Basics of Hacking and Penetration Testing, 2nd Edition, Waltham: Syngress, 2013.

[13] G. Lyon, «Nmap - Intro,» 16 05 2016. [En línea]. Available: https://nmap.org/. [Último acceso: 26 05 2016].

[14] The OWASP Foundation, «OWASP Testing Project,» 17 9 2014. [En línea]. Available: https://www.owasp.org/images/5/52/OWASP_Testing_Guide_v4.pdf. [Último acceso: 20 5 2016].

[15] D. Song, «fragroute,» 18 05 2016. [En línea]. Available: https://www.monkey.org/~dugsong/fragroute/. [Último acceso: 24 05 2016].

[16] Microsoft Corporation, «Análisis de modelo de amenazas,» 02 09 2013. [En línea]. Available: https://msdn.microsoft.com/es-es/library/aa561499.aspx. [Último acceso: 03 05 2016].

[17] C. Maynard, «VoIP Calls,» 09 12 2011. [En línea]. Available: https://wiki.wireshark.org/VoIP_calls. [Último acceso: 26 05 2016].

[18] P. Gonzalez Perez, Ethical Hacking: teoría y practica para la realización de un pentesting, Móstoles: 0xWord Computing, 2014.

[19] NIST, «National Vulnerability Database,» 21 06 2016. [En línea]. Available: https://nvd.nist.gov/. [Último acceso: 24 06 2016].

[20] «Bugtraq Mailing List,» 06 04 2016. [En línea]. Available: http://seclists.org/bugtraq/. [Último acceso: 24 06 2016].

[21] C. Alonso, «CROZONO: Uso de drones y robots en tests de intrusión,» 31 03 2016. [En línea]. Available: http://www.elladodelmal.com/2016/03/crozono-uso-de-drones-y-robots-en-tests.html. [Último acceso: 16 06 2016].

[22] K. Zetter, «Hacking Wireless Printers With Phones on Drones,» 10 05 2015. [En línea]. Available: https://www.wired.com/2015/10/drones-robot-vacuums-can-spy-office-printer/. [Último acceso: 16 06 2016].

[23] J. Erickson, Hacking: The Art of Exploitation, 2nd Edition, San Francisco: No Starch Press, 2008.

[24] The Offensive Security Team, «Understanding Payloads in Metasploit,» 28 04 2016. [En línea]. Available: https://www.offensive-security.com/metasploit-unleashed/payloads/. [Último acceso: 29 05 2016].

[25] S. de los Santos, Máxima Seguridad en Windows: Secretos Técnicos. 3ª Edición, Mostoles: Informática64, 2015.

[26] F. Amato y F. Kirschbaum, «Defcon,» 18 09 2010. [En línea]. Available: https://www.defcon.org/images/defcon-18/dc-18-presentations/Amato-Kirschabum/DEFCON-18-Amato-Kirschabum-Evilgrade.pdf. [Último acceso: 15 06 2016].

[27] Amazon Web Services, Inc, «AWS | Cloud Computing,» 11 06 2016. [En línea]. Available: https://aws.amazon.com/es/?nc2=h_lg. [Último acceso: 15 06 2016].

[28] J. L. Garcia Rambla, Ataques en redes de datos IPv4 e IPv6 2ª edicion, Mostoles: Informatica64, 2014.

[29] Eleven Paths, «FOCA,» 26 05 2016. [En línea]. Available: https://www.elevenpaths.com/es/labstools/foca-2/index.html. [Último acceso: 09 06 2016].

[30] C. Rioux, C. Wysopal y P. Mudge Zatko, «l0phtcrack,» 26 04 2016. [En línea]. Available: http://www.l0phtcrack.com/. [Último acceso: 14 06 2016].

[31] M. Montoro , «Cain,» 16 03 2016. [En línea]. Available: http://www.oxid.it/cain.html. [Último acceso: 14 06 2016].

[32] ManifestSecurity, «Pentest Box,» 12 04 2016. [En línea]. Available: https://pentestbox.com/. [Último acceso: 14 06 2016].

[33] I. Gonzalez, «K0sasp - Haking con OS X,» 11 03 2016. [En línea]. Available: http://k0sasp.kontrol0.com/. [Último acceso: 14 06 2016].

[34] The MacPorts Project, «The MacPorts Project Official Homepage,» 09 06 2016. [En línea]. Available: https://www.macports.org/. [Último acceso: 14 06 2016].

[35] M. Howell y R. Prévost, «Homebrew,» 11 06 2016. [En línea]. Available: http://brew.sh/. [Último acceso: 14 06 2016].

[36] «Root-me,» 24 03 2016. [En línea]. Available: https://www.root-me.org/?lang=es. [Último acceso: 10 06 2016].

[37] «Pentestit - Penetration Testing Laboratories,» 11 06 2016. [En línea]. Available: https://lab.pentestit.ru/. [Último acceso: 14 06 2016].

[38] «Hack This Site,» 11 06 2016. [En línea]. Available: https://www.hackthissite.org/. [Último acceso: 17 06 2016].

[39] «PentesterLab,» 06 05 2016. [En línea]. Available: https://pentesterlab.com/. [Último acceso: 28 05 2016].

[40] g0tmi1k, «Vulnerable By Design,» 10 04 2016. [En línea]. Available: https://www.vulnhub.com/. [Último acceso: 18 04 2016].

[41] Microsoft Corporation, «Free Virtual Machines,» 25 04 2016. [En línea]. Available: https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/windows/.. [Último acceso: 15 05 2016].

[42] «InsanelyMac,» 16 06 2016. [En línea]. Available: http://www.insanelymac.com/. [Último acceso: 18 06 2016].