Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Grado en Ingeniería Informática

Metodología de Pentesting para Windows y Linux automatizada con Script

Trabajo fin de estudio presentado por:	Elisa Alises Núñez
Línea de investigación:	Ciberseguridad
Director/a:	Manuel Sánchez Rubio
Fecha:	21/03/2022

Resumen

El Pentesting es una disciplina que pertenece al campo de la ciberseguridad, cuyo objetivo es analizar la seguridad de un sistema o entorno. Para ello, se deben realizar múltiples ataques informáticos al objetivo, para examinar si presenta vulnerabilidades, fallos de configuración… Y sugerir posibles soluciones que se deben implantar para incrementar la protección de éste.

El factor tiempo es de elevada importancia en esta labor, ya que durante ese tiempo el sistema, y, por tanto, la información que contiene puede estar en peligro.

El objetivo de este trabajo es ofrecer una metodología clara que llevar a cabo, utilizando softwares ya existentes, así como conseguir realizar dichas pruebas de intrusión ayudándose de una herramienta que se ha diseñado para tal fin, de forma que un pentester puede realizar una auditoría a cualquier entorno (Linux o Windows) de forma muy sencilla y eficaz, gracias a que dicho software interactúa constantemente con el usuario, realizando todo de forma automatizada, tanto los ataques y la recopilación de información, como la generación del informe final.

Palabras clave: Auditoría, Hacking, Pentesting, Seguridad, Software.

Abstract

Pentesting is a discipline that belongs to the field of cybersecurity, with the objective to analyze the security of a system or environment. To do this, multiple computer attacks must be carried out on the target, to examine if it presents vulnerabilities, wrong configurations, etc., and to suggest possible solutions that should be implemented to increase its protection.

The time factor is highly important in this task, because during this time, the system, and therefore the information it contains, may be in danger.

The objective of this work is to offer a clear methodology to carry out, using available software, as well as being able to carry out said intrusion test with the help of a tool that has been designed for this purpose, so that a pentester can carry out an audit of any environment (Linux or Windows) in a very simple and effective way, thanks to the fact that said software constantly interacts with the user, performing everything automatically, both the attacks and the collection of information, as well as the generation of the final report.

Keywords: Audit, Hacking, Pentesting, Security, Software.

Índice de contenidos

Índice de figuras

1. Introducción

Hace no mucho tiempo pocos nos podíamos permitir el acceso a dispositivos con conexión a Internet, sin embargo, hoy en día la mayoría de nosotros podemos hacer uso de ello con asiduidad.

Esto influye directamente en la cantidad de datos e información que manejamos y generamos diariamente cada uno de nosotros. Esto ha facilitado cambios en nuestros hábitos actuales realizando todo tipo de gestiones, compras, relaciones laborales y sociales de forma online, a través de páginas web, aplicaciones móviles, etc.

Además, la ciberseguridad cobra especial importancia en tiempos de Covid-19 (Navarro, 2020), debido a que numerosas empresas se han visto obligadas a permitir el teletrabajo, teniendo por tanto que mejorar la protección de sus sistemas y comunicaciones.

Por lo que, no solo se genera una gran cantidad de información sensible continuamente, sino que, si no se protege correctamente, nuestra privacidad y seguridad está en constante peligro.

En un estudio reciente se obtiene que el 86% de las empresas españolas carecen de cultura de ciberseguridad (Sánchez, 2021), lo cual puede atraer graves problemas y consecuencias.

“Se puede tener el mejor sistema de seguridad, pero un error puede ponerlo todo en riesgo” (Añover, 2021). Podemos visualizar la magnitud del problema si nos centramos y analizamos cifras reales obtenidas en estadísticas realizadas por ESET en el año 2021 (Computing, 2022).

Algunos de los datos más importantes extraídos de dicha web son:

- Se produjo el mayor coste de una filtración de datos de las últimas décadas según un informe de IBM, siendo la causa más común el robo de credenciales de usuario.
- Se registra el mayor rescate de sistemas infectados por malware solicitado hasta ahora, llevado a cabo por los creadores del ransomware Sodinokibi al proveedor de software de gestión TI Kaseya, llegando a los 70 millones de dólares.

- El 36% de las infracciones han estado relacionadas con ataques de Phising, un aumento del 11% debido a la pandemia de Covid-19.
- El 69% de todas las filtraciones se dan a ataques de ingeniería social, lo cual constituye la amenaza más grave para la Administración Pública.
- Estafas de criptomonedas.
- El malware bancario en Android también ha aumentado notablemente, a principios de año incrementó un 158,7%.
- El troyano WannaCryptor (también denominado WannaCry) sigue considerándose una amenaza a nivel mundial, ya que infecta a máquinas vulnerables al exploit EternalBlue.
- Un reciente estudio realizado por Cybersecurity Workforce Study declara que "Para poder compensar el déficit de profesionales de la ciberseguridad necesarios para defender eficazmente los activos críticos de las organizaciones, la mano de obra mundial de la ciberseguridad tendría que crecer un 65%".
- Un 82% de las organizaciones ha admitido haber aumentado sus presupuestos en ciberseguridad en este último año.

El último Índice de Ciberseguridad Global (ICG) de la Unión Internacional de Telecomunicaciones (UIT), EEUU, Canadá y Brasil encabezan el ranking de seguridad cibernética en América (Valadés, 2021).

Un artículo interesante publicado el 18 de junio de 2021 hace eco de las palabras de Esther Naylor, investigación de seguridad internacional en Chatham House, "*Estonia se digitalizó mucho antes que otros países, se centró en cosas como la educación en línea y los servicios gubernamentales en línea y adoptó un enfoque más proactivo de la tecnología*". "*Con solo 1,3 millones de habitantes, el país báltico está muy por encima de su peso en lo que respecta a la seguridad en línea. La capital es Tallin, que alberga el Centro de Ciberdefensa de la OTAN, el centro de excelencia cooperativo de ciberdefensa*" (Añover, 2021).

Aunque los gobernantes de todo el mundo son conscientes de la importancia de la ciberseguridad y la ciberdefensa, no todos se implican de igual forma, se van dando pasos importantes, pero no es suficiente, como es el caso de España. Según recogen los medios de comunicación de nuestro país, el 29 de marzo de 2022, el Consejo de ministros, aprobó invertir más de 1.200 millones de euros en el **Plan Nacional de Ciberseguridad**. Tiene previsto, entre

otras muchas actuaciones, el desarrollo de un sistema integrado de indicadores de ciberseguridad a nivel nacional y la creación de nuevas infraestructuras de ciberseguridad en las comunidades autónomas y entidades locales, la puesta en marcha del centro de operaciones de ciberseguridad de la Administración General del Estado (AGE), adjudicado a Indra y Telefónica.

Figura 1: La UIT destaca que los países están trabajando para mejorar sus capacidades de seguridad cibernética.
Fuente: https://www.segurilatam.com/actualidad/ciberseguridad-eeuu-canada-y-brasil-primeros-paises-americanos-en-el-indice-de-ciberseguridad-global_20210702.html

Analizando estos datos, y otros similares, tan aterradores, podemos hacernos una idea de lo importante que es la Ciberseguridad como área de la informática que permite la protección de todas estas infraestructuras y, por tanto, de todos los datos que se van generando a través de Internet.

El objetivo principal de la Ciberseguridad es proteger los sistemas ante cualquier ataque informático, accesos no autorizados, etc. La tecnología evoluciona a velocidad de vértigo,

siendo a veces imposible impedir que se produzca un robo de información confidencial, suplantaciones de identidad, denegación de servicio…

El éxito de un ataque informático provoca consecuencias que pueden ser realmente críticas y devastadoras, tanto a nivel empresarial como personal. A veces no somos del todo conscientes del gran riesgo al que nos exponemos cuando la seguridad implantada no es la adecuada o cuando no se corrigen vulnerabilidades encontradas. Aquí es donde entra en juego la importancia de la seguridad de la información.

1.1. Justificación del tema elegido

Cada vez son más los delitos informáticos que se producen, y esta cifra se está incrementando de forma exponencial.

Todas las nuevas herramientas que faciliten un avance en fiabilidad, esfuerzo y tiempo, suponen una aportación importante tanto para los profesionales, como para quienes requieren de estos servicios.

Por lo tanto, se requiere de la investigación de las posibles amenazas y vulnerabilidades, así como de servicios encargados de proteger los sistemas debidamente, con el objetivo de mejorar la privacidad y seguridad frente a estos ataques, además de asegurarse de corregir posteriormente las vulnerabilidades encontradas en los sistemas.

El presente trabajo está enfocado con este planteamiento, cuyo objetivo es **determinar una metodología de Pentesting muy eficaz y sencilla**, con ayuda de una **herramienta de creación propia diseñada para realizar una auditoría de seguridad de forma automatizada y guiada**, tanto a la propia máquina en la que se está ejecutando la herramienta (si es Linux), como a un dominio web, como a una máquina a la que se tenga acceso en la red (independientemente del sistema operativo que utilice) en la que está la máquina atacante.

Gracias al seguimiento de esta metodología y con ayuda de esta herramienta, **se podrá analizar la seguridad de cualquier sistema o dominio web**, desde la fase inicial hasta la fase de explotación (así como información para la fase de post-explotación), descubriendo así vulnerabilidades existentes, errores en las asignaciones de permisos, llevando a cabo ataques que prueban la seguridad, etc., con el fin de notificar los resultados y que quede constancia

de ello, para que posteriormente se puedan tomar medidas al analizar el reporte obtenido en la auditoría realizada.

Esta herramienta se ha creado pensando en que cualquier persona (con unos conocimientos mínimos) pueda interactuar con ella. Esto es interesante debido a que cualquiera puede realizar esta primera parte de la auditoría y obtener el informe de resultados (aún sin tener conocimientos en la materia), para que después un experto en el área pueda analizarlo, encontrando vulnerabilidades y evaluar cómo explotarlas.

Por tanto, este programa **es totalmente guiado y automatizado**. Dispone de menús de navegación y colores aclarativos para el usuario. **De forma automática y a tiempo real, se encarga de ir almacenando todos los resultados de lo que se va ejecutando y obteniendo en un fichero, que será el reporte de auditoría que analizará el experto**.

Esto repercute de forma notable en la disminución del tiempo total de auditoría y fiabilidad en la toma de notas y generación de informes de los resultados por parte del experto, sobre todo en la fase inicial de recopilación de información. De esta forma, el experto tendrá más tiempo para las fases más complicadas de análisis de los resultados obtenidos o de explotación de vulnerabilidades o incluso para la fase de post-explotación de un sistema. Siendo por tanto una **herramienta muy útil en su trabajo**.

1.2. Problema y finalidad del trabajo

Realizar una auditoría informática exhaustiva puede ser realmente complejo, existen multitud de vulnerabilidades diferentes, hay numerosos vectores de ataque posibles... Además, normalmente son costosas en tiempo, y esto, por tanto, repercute en los costes económicos para la empresa o entidad que la solicita.

Sobre todo, la fase inicial, así como registrar todos los datos obtenidos, consume mucho tiempo al experto. Aunque realmente todas las fases son costosas, ya que analizar o encontrar vulnerabilidades, evaluar cómo explotarlas, realizar ataques de prueba y analizar la post-explotación de un sistema, también son etapas muy complicadas.

Por tanto, se crea esta herramienta que se encarga de registrar todo de forma totalmente automatizada y transparente de cara al usuario, además de ser muy fácil de utilizar por ser

completamente guiada a través de menús, preguntas, etc. Además de todo esto, se muestran continuamente mensajes explicativos de lo que está pasando en tiempo real, si hay errores, si todo va bien, si se ha almacenado en el fichero, etc. Por lo que otra persona (de menor cualificación y menor salario) puede realizar esta fase de la auditoría de forma muy simple, puede ir adelantando ese trabajo inicial para que finalmente el experto analice el reporte, o bien, si no dispone de personal de apoyo, el experto tendrá que realizar el proceso completo de la auditoria pero esta herramienta le facilitará el trabajo y consumirá menos tiempo en estas tareas.

Este ahorro de tiempo al auditar hace posible detectar los fallos de seguridad existentes mucho antes, para analizarlos e intentar corregirlos antes de ser descubiertos por un atacante.

La finalidad de este trabajo es la de establecer un marco metodológico en el que basarse al realizar una auditoría informática, donde se establecen pasos a seguir, herramientas útiles, así como la explicación de la gran ayuda y avance que aporta la herramienta creada, además de buenas prácticas.

1.3. Objetivos del TFE

Realizar una auditoría de seguridad es complejo, tanto en los aspectos técnicos, como en la parte legal. En este apartado se plantean los objetivos de carácter general y específicos que se pretenden cumplir en la realización de este TFG.

1.3.1. Objetivo General

El objetivo general es **diseñar una propuesta de intervención a través de una metodología de Pentesting muy completa**, compuesta por los posibles procedimientos que se pueden llevar a cabo en cada una de las fases, **haciendo uso de la herramienta diseñada y desarrollada para tal fin**. Este programa se realiza con la finalidad de analizar la seguridad de un sistema o un dominio web, y obtener el reporte de auditoría de forma totalmente automatizada y a tiempo real, de forma muy sencilla y eficaz, para que después un experto en la materia pueda analizarlo y actuar frente a los resultados obtenidos, lo antes posible para sugerir medidas de protección y corrección antes de que se produzca un ataque.

1.3.2. Objetivos específicos

Para conseguir el objetivo general, serán necesarios fijar una serie de objetivos específicos:

- Conocer en todo momento los requisitos legales y acuerdos firmados (tipos de ataques permitidos, consecuencias asumidas, etc.), antes de proceder a analizar un sistema o dominio web.
- Exponer en qué consiste cada fase, la problemática y detalles a tener en cuenta.
- Comprender los procedimientos posibles que se pueden llevar a cabo, así como herramientas disponibles en la red, en cada una de ellas.
- Entender en qué forma ayuda la herramienta creada en cada una de estas etapas.
- Observar el modelo de reporte de auditoría emitido de forma totalmente automatizada por la herramienta.
- Concienciar a los usuarios que utilizan el sistema para conseguir que se respeten en todo momento las normas de actuación marcadas por la empresa.

2. Marco teórico

Revisada la bibliografía relacionada con la Ciberseguridad y las auditorías informáticas, se constata que existen diferentes publicaciones, artículos y revistas científicas que tratan diversos aspectos relacionados con el tema de estudio de este trabajo, pero en este caso, se va a ajustar toda la explicación con un enfoque que permita dar un guion que sirva de orientación para poder comenzar a investigar posibles vulnerabilidades existentes, aunque después el experto deba analizar de forma más exhaustiva cada una de ellas.

Además, hay que tener presente que las pruebas de penetración (Pentesting) se ven condicionadas por recomendaciones, así como los acuerdos firmados con la parte que solicita el servicio que hay que respetarlos en todo momento, así como notificar e informar de las posibles consecuencias antes de comenzar el análisis, para dejar reflejado de manera formal que están de acuerdo ambas partes, ya que si no podría desembocar en problemas legales.

En este apartado se abordan los conceptos y términos teóricos más relevantes que serán objeto de referencia en el desarrollo del estándar metodológico a la hora de abordar una auditoría de seguridad, y su posterior análisis y redacción del reporte de resultados.

2.1. Ciberseguridad

La Ciberseguridad (también llamada seguridad informática) es el área relacionada con la informática cuyo objetivo se centra en proteger la infraestructura computacional y todo lo relacionado con ésta, especialmente la información contenida en ella. En resumen, es la habilidad de identificar y eliminar vulnerabilidades, así como salvaguardar la información y equipos físicos. Debe asegurarse de cumplir y respetar la confidencialidad, integridad y disponibilidad de la información y los sistemas.

2.2. Vulnerabilidad

Una vulnerabilidad es una debilidad existente en un sistema que puede ser utilizada maliciosamente por una o varias personas para comprometer su seguridad.

2.2.1. Clasificación de las vulnerabilidades según su nivel de criticidad

- **Peligrosidad nivel bajo**

  Son las menos peligrosas, pero pueden ayudar al atacante a elegir qué ataque realizar. Suelen ser aspectos relacionados con las configuraciones de un servicio, rutas encontradas…

- **Peligrosidad nivel medio**

  Estas vulnerabilidades no comprometen al sistema, pero pueden ayudar a dicha función. Por ejemplo, un fallo que permita realizar una escalada de privilegios.

- **Peligrosidad nivel alto**

  Son muy peligrosas porque permiten acceder a recursos sensibles que la máquina debería tener muy protegidos, como la utilización de credenciales por defecto.

- **Peligrosidad nivel crítico**

  Son similares a las altas, pero estas pueden causar repercusiones serias para el objetivo del ataque. Si se encuentra una vulnerabilidad de este tipo se debe notificar de inmediato al cliente, ya que, si alguien la ha detectado y realiza un ataque contra ella en ese momento, cae la responsabilidad en el auditor puesto que está realizando la prueba en ese periodo de tiempo.

2.2.2. Exploit

Un exploit es un programa informático, una parte de un software o una secuencia de comandos que se aprovecha de un error o vulnerabilidad para provocar un comportamiento no intencionado o imprevisto en un software, hardware o en cualquier dispositivo electrónico.

Dicho de otra forma, tratan de explotar una debilidad encontrada, con el fin de tomar el control del sistema, conseguir privilegios de administrador o incluso el lanzamiento de un ataque de denegación de servicio (DoS o DDoS).

2.3. Auditoría informática o pentesting (pruebas de penetración)

El término de auditoría informática es equivalente al de Pentesting (test de penetración). Se trata de una técnica para localizar vulnerabilidades en los sistemas. Consiste en realizar pruebas con el objetivo de evaluar el estado de seguridad de los sistemas o entornos que se están analizando, para detectar fallos, y posteriormente corregirlos con el fin de prevenir posibles ataques.

2.3.1. Auditor informático o Pentester

También se le denomina hacker ético. Se trata de una persona encargada de llevar a cabo auditorías informáticas con consentimiento explícito por parte de la entidad que lo solicita, cuya finalidad se centra en verificar y garantizar que todos los procesos y sistemas estén funcionando correctamente y no dispongan de vulnerabilidades.

Por tanto, su principal función es analizar y encontrar vulnerabilidades, así como aportar posibles soluciones o estrategias para la mejora de los sistemas o corrección de errores. Además, debe presentar informes con los resultados de cada auditoría que realiza.

2.3.2. Fases del Pentesting

Las fases son cuatro y se recomienda realizarlas en orden secuencial, ya que las fases tempranas ayudan a las posteriores.

- **Fase inicial**

  En esta fase es donde se lleva a cabo la recopilación de información sobre el objetivo de la auditoría. No son peticiones directas al objetivo, sino que se utilizan fuentes de

Internet (denominadas fuentes OSINT) en todo momento, por lo que es legal, aún sin poseer un consentimiento firmado.

- **Fase de búsqueda y análisis de vulnerabilidades**

  A partir de esta fase se necesita de forma obligatoria un consentimiento explícito por la entidad que solicita el servicio, al igual que un acuerdo firmado por ambas partes, ya que si no sería ilegal continuar con la auditoría.

  Aquí es donde se llevan a cabo los escaneos de puertos y vulnerabilidades. Se analizan los resultados en busca de debilidades, errores en asignaciones de permisos, malas configuraciones, etc.

- **Fase de explotación de vulnerabilidades**

  En esta etapa es donde se llevan a cabo las pruebas de explotación contra las vulnerabilidades que se han encontrado, para comprobar que no sean falsos positivos. En este punto hay que tener excesivo cuidado de no llevar a cabo ataques cuyas consecuencias no están permitidas por la entidad que lo solicita. Por ejemplo, hay que tener especial cuidado con ejecutar ataques de denegación de servicio, si en el consentimiento firmado la entidad no asume una pérdida de disponibilidad en los servicios que ofrece.

- **Fase de post-explotación**

  Esta es la última fase antes de realizar el reporte de auditoría. Solo se lleva a cabo si se ha conseguido un ataque exitoso y con ello el acceso al sistema objetivo. Aquí es donde se intentan elevar privilegios, realizar técnicas de pivoting, etc.

Por último, el auditor debe realizar el informe de resultados obtenidos en la auditoría. Esto podría considerarse una fase más, pero la autora de este documento no lo va a considerar como tal, ya que la mayoría de los pentesters redactan dicho reporte al mismo tiempo que llevan a cabo las fases nombradas anteriormente.

2.3.3. Tipos de auditorías informáticas

Al realizar una auditoría se puede conocer información previa sobre el objetivo como estructura, tecnologías utilizadas, etc., o puede no conocer ningún tipo de dato. Se utiliza el término caja para declarar el tipo de conocimiento que se tiene inicialmente de las infraestructuras.

Tipos de auditorías según el conocimiento previo que posee el auditor:

- **Caja blanca**

  Entre todos los tipos de auditoría, las de caja blanca son las más sencillas y fáciles de realizar para un auditor. El cliente ofrece información previa sobre las IP's que hay que investigar, información de software, de hardware, servicios de los puertos abiertos, etc. Esto acelera el proceso de auditoría, y además es más precisa, ya que permite descartar fácilmente los falsos de los verdaderos positivos. Este tipo de auditoría la suele pedir el cliente cuando tiene necesidad de conocer los resultados en el menor tiempo posible, por lo que ayuda al auditor a realizar su tarea.

- **Caja negra**

  Se trata de una auditoría mucho más laboriosa que la de caja blanca. No se tiene ningún tipo de información previa por parte del cliente. Se extiende mucho más en tiempo, aunque será mucho más feraz, ya que el pentester se pone en el papel de un ciberatacante.

- **Caja gris**

  Es una mezcla de las dos anteriores. Este tipo de auditoría está orientada a los usuarios o a un departamento en concreto. Por ejemplo, se otorga acceso al auditor a un área determinada y éste debe comprobar si desde ese rol de usuario se pueden llevar a cabo ataques, o si se puede escapar de los privilegios que tiene asignados en ese momento. Es muy útil para evaluar el nivel de seguridad que tiene un departamento.

Otra manera de categorizar los tipos de auditorías sería dependiendo de si la auditoría se realiza desde dentro de la red o de forma externa:

- **Auditoría interna**

  Es más sencilla de realizar que la externa. Se llevan a cabo cuando el cliente da acceso al auditor a sus infraestructuras de forma física o remota con VPN. Estando por tanto dentro de su red de área privada (LAN, MAN o WAN).

- **Auditoría externa**

  Es lo contrario a la auditoría interna. El auditor no está dentro de la red del cliente. Pueden tener medidas perimetrales que impidan fases de la auditoría (cortafuegos cortando puertos, etc.). Una de las ventajas es que las IPs que se localizan son menos porque no se encuentran tantos servidores, solo infraestructuras externas.

Otros tipos de auditorías menos comunes:

- **Auditorías Wireless**

  Se encargan de comprobar la correcta segmentación de la red, robar el handshake para intentar obtener la contraseña, probar los pivotajes a otras máquinas o redes…

- **Auditorías móviles**

  Se encargan de análisis de puertos, servicios, vulnerabilidades… Necesitan que cumplan tareas (rooteo, jailbreak, etc.).

- **Auditorías a aplicación web**

  La batería de pruebas para este tipo de testing es muy extensa, por lo que conlleva mucho tiempo. Se debe analizar el código fuente, su funcionamiento, etc.

- **Auditoría a aplicaciones**

  Se suele requerir el desarrollo de exploits. Probar con un debugguer y fuzzing que manda cadenas para ver si encuentra buffer underflow, etc. Hay muchas pruebas

posibles para ejecutar como mandar comandos de forma remota, etc. Ya no se suele pedir porque existen programas que las analizan.

2.3.4. Metodología manual vs automatizada

La metodología manual consiste en aplicar solo técnicas manuales de análisis y explotación de vulnerabilidades. Es decir, con comandos, pero sin usar herramientas automatizadas como por ejemplo Metasploit, Nessus, etc.

Sin embargo, la metodología automatizada consiste en aplicar métodos automatizados, es decir, herramientas que realicen las labores de análisis y ataque de forma automática. Como por ejemplo Searchexploit, Metasploit, etc.

2.3.5. Importancia de la realización de auditorías regulares en una empresa

Es importante realizar auditorías informáticas de forma periódica a los sistemas para evitar, medir y controlar los posibles riesgos como pueden ser los ciberataques, así como asegurar el correcto funcionamiento de los servicios.

Los objetivos son evaluar las operaciones de los sistemas y gestión administrativa del área informática, verificar que se estén cumpliendo las políticas, normas y estándares que controlan el correcto funcionamiento de las responsabilidades, actividades y usos de los sistemas y equipos de cómputo que se utilizan dentro de la empresa, determinar si son eficientes o implementar cambios necesarios, inspeccionar si los empleados están utilizando de manera correcta los equipos informáticos y sus complementos para verificar que sean los adecuados para la realización de su trabajo, así como realizar control y evaluación de todos los softwares que se utilizan en la organización.

2.3.6. Delito informático

Cuando se realizan labores de pruebas de penetración sin consentimiento explícito por la entidad contratante, se trata de un delito.

Un delito informático se define como el acto delictivo en el que se hace uso de la informática para su comisión, ya sea como medio o como fin de este.

2.3.7. Hacker

Persona con elevado conocimiento en informática que se dedica a detectar fallos de seguridad en los sistemas y explotarlos llevando a cabo ataques cibernéticos. Si posee un consentimiento explícito se denomina hacker ético, porque realiza la evaluación de seguridad de forma completamente legal y sin fines maliciosos. Sin embargo, sin este consentimiento, el Pentesting es considerado ilegal.

Hoy en día se entiende la palabra Hacker como persona que realiza estas acciones de forma ilegal y con fines maliciosos, y Hacker ético o Pentester cuando lo lleva a cabo de forma legal. Aunque en realidad, el término Hacker no tiene por qué conllevar una connotación negativa.

Categorización del Hacker según su moralidad:

- **Sombrero blanco (White hat)**
  Hacker cuyo único fin es ayudar al cliente y nunca actúa para la obtención de beneficio propio. Esta es la moralidad de los Pentesters o Hackers éticos. Siempre cuentan con permiso explícito y contrato consensuado con la parte que lo contrata (realizan todo de forma legal).

- **Sombrero gris (Grey hat)**
  Hacker que no tiene ningún fin malicioso, pero no cuenta con el permiso expreso del dueño de esas infraestructuras informáticas. Es un delito, aunque sea menor que el sombrero negro.

- **Sombrero negro (Black hat)**
  Hacker que actúa en su propio beneficio. Puede ser de forma directa (por ejemplo, ransomware) o indirecta (por ejemplo, perjudicar las infraestructuras o servicios que aportan para beneficiarse). Son considerados como criminales, ya que actúan de forma

completamente ilegal y delictiva. Actualmente al término Hacker se le asocia esta idea, pero no siempre es así.

Algunos tipos de Hacker de sombrero negro:

- **Cracker**

  Hacker especializado en ingeniería inversa. Normalmente es de sombrero gris, pero puede ser blanco si su finalidad consiste en avisar a una empresa para que solucione los fallos de seguridad detectados, o de sombrero negro si obtiene beneficios propios. “To crack” en inglés significa romper. Por lo que son personas que rompen o vulneran algún sistema de seguridad y pueden estar motivados por multitud de razones, como por ejemplo fines de lucro, protesta, desafío…

- **Lamer**

  Son personas que carecen de cualquier conocimiento en Hacking. Este es quizás el grupo que más peligro acontece en la red, ya que ponen en práctica todo el Software de Hackeo que encuentran en la red, sin conocer sus posibles consecuencias.

- **Script Kiddies**

  Individuo no cualificado que utiliza scripts o programas desarrollados por otros para atacar sistemas informáticos, redes y defectos de sitios web.

- **Phreaking**

  Hacker especializado en las redes de telefonía. Se trata de personas que estudian, experimentan o exploran sistemas de telecomunicaciones, tales como equipos y sistemas conectados a redes telefónicas públicas. También puede referirse al uso de varias frecuencias de audio para manipular un sistema telefónico.

2.3.8. Defensa vs ataque

El departamento de seguridad se divide en dos secciones diferenciadas:

- **Equipo de ataque (Red Team)**

  Se centran en la parte ofensiva (ataque). Son los encargados de analizar sistemas, localizar vulnerabilidades y explotarlas. La herramienta de Pentesting que se ha diseñado, así como el presente documento, se centran más en esta parte de ataque, aunque después los resultados que se han obtenido se pasarían al equipo de defensa para que se aseguren de corregir los problemas que se han encontrado.

- **Equipo de defensa (Blue Team)**

  Se centran en la parte defensiva (defender o prevenir un ataque). Son los encargados de solucionar las vulnerabilidades encontradas por el Red Team, así como proteger lo máximo posible los sistemas o entornos de posibles ataques o fallos futuros.

2.4. Herramientas de pentesting

Una herramienta de Pentesting es un programa encargado de facilitar las técnicas de obtención de información y búsqueda de vulnerabilidades en un sistema o entorno, así como recabar posible información de cómo explotarlas. Estos softwares pueden ser gratuitos o comerciales, ambas opciones son totalmente lícitas y válidas.

Una auditoría se puede realizar con ayuda de estas herramientas que automatizan algunos procesos o de forma manual, siendo lo ideal que ambas se complementen para obtener la máxima información posible.

2.4.1. Distribuciones más utilizadas en el sector de la seguridad informática

Para llevar a cabo labores de seguridad informática normalmente se utilizan entornos Linux, aunque la distribución concreta dependerá del caso de estudio en cuestión. Si el objetivo es analizar un sistema o un dominio web se suelen utilizar Kali Linux, Parrot... Sin embargo, si se pretende realizar un análisis Wireless la distribución más utilizada es Wifislax.

2.4.2. Frameworks y software para Pentesting

Los frameworks son suites de herramientas software estandarizadas para un área determinada. Para el análisis de seguridad el más conocido es Metasploit Framework. Es gratuito y contiene multitud de exploits para analizar de forma automatizada cualquier sistema.

Existen numerosos programas para la realización de pruebas de penetración y su uso dependerá del análisis que se quiera realizar. Algunas de ellas son automatizadas y otras más manuales, pero ambos procedimientos son igual de lícitos.

Para el análisis del tráfico de red se pueden utilizar herramientas como WireShark, tcpdump, entre otras. Si el objetivo es analizar sistemas, pueden ser útiles programas como Nessus, Metasploit, nmap… Así como whatweb, Burp Suite, etc. para examinar dominios web.

2.4.3. Fuentes OSINT

En la fase inicial de una prueba de penetración la información que se obtiene del objetivo debe ser obtenida a través de fuentes OSINT (Open Source INTelligence), traducido como Inteligencia de Fuentes Abiertas. Este término hace referencia al conjunto de técnicas y herramientas para recopilar información pública, analizar los datos y correlacionarlos convirtiéndolos en conocimiento útil. La adquisición de esta información es completamente legal, aún sin tener consentimiento.

2.4.4. Evasión de detecciones

La evasión de detecciones son métodos para evitar la detección de un escaneo de puertos, de vulnerabilidades, o la realización de un ataque. Si el pentester dispone de permiso, no es necesario utilizar dichas técnicas, ya que ralentizan aún más el tiempo de auditoría. Llevar a cabo un escaneo de puertos de forma silenciosa, utilizar redes TOR, uso de VPN… Son ejemplos de métodos de evasión.

Estos métodos se llevan a cabo por parte del pentester por si la empresa cuenta por ejemplo con IDS (sistemas de detección de intrusiones) o IPS (sistemas de prevención de intrusiones).

Métodos de evasión que pueden utilizar los ficheros maliciosos:

Ciertos ficheros maliciosos se ocultan en ficheros legítimos, pero adicionalmente existen métodos de evasión en la detección de motores antivirus.

- Troyanización a un fichero legítimo.
- Cifrado mediante Crypter.
- Modificación de bucles y esperas en el proceso malicioso.

Un Crypter es un programa cuya finalidad consiste en cifrar y/o ofuscar código malicioso mediante diversas técnicas, con el fin de evitar la detección de software malicioso por parte de los motores antivirus.

Existen dos tipos:

- **Runtime:** el programa no es detectado por el motor antivirus al ser ejecutado.
- **Scantime:** el programa solo es indetectable ante el escaneo, pero al ejecutarlo, el antivirus lo detecta.

2.5. Script

Un script (guion) es una palabra en inglés que puede traducirse como el guion que dirige una escena o secuencia. Para explicarlo de forma más simple, se puede decir que un script está compuesto por una serie de instrucciones escritas en código (en algún lenguaje de programación), que sirven para ejecutar diversas funciones en un dispositivo electrónico.

La herramienta de creación propia diseñada por la autora del presente documento está compuesta por multitud de scripts que interaccionan entre sí.

2.6. Puerto

En informática, un puerto es una interfaz o conector que se utiliza para comunicar diferentes tipos de elementos hardware o software, a través del cual se pueden enviar y recibir los diferentes tipos de datos de un equipo a otro. Puede ser interno o externo, y si no se protegen de forma correcta, pueden ser una vía de acceso al sistema por parte de un atacante.

2.6.1. Tipos de puertos

- **Puertos hardware**

  Tienen como finalidad establecer comunicaciones entre dispositivos hardware (físicos). Estos permiten comunicar periféricos como teclados, módems, routers, ratón y demás elementos a un ordenador u otro dispositivo informático.

  Entre los puertos de hardware más utilizados que se pueden encontrar en diferentes dispositivos informáticos, se tienen los puertos basados en tecnología USB (mini USB, USB, micro USB), puertos de memoria, puertos de red, puertos VGA, puertos HDMI, puertos PCI, entre otros conectores que se encuentran en la parte posterior del ordenador, o en los laterales de dispositivos como Smarphones y tablets.

- **Puertos software**

  También conocidos como puertos lógicos, estos se encuentran ubicados dentro del equipo informático y permiten establecer comunicaciones con diferentes programas, así como, realizar la distribución de servicios y flujo de datos entre dispositivos informáticos o incluso dentro del mismo ordenador. Un ejemplo de esto es el puerto 21 que está asociado por defecto al servicio FTP, el 80 a HTTP, el 443 a HTTPS…

2.7. Dominio y servidor web

Un dominio web es un nombre único que identifica a una subárea de Internet. El propósito principal de los nombres de dominio y del sistema de nombres de dominio, es traducir las direcciones IP de cada activo en la red a términos memorizables y fáciles de encontrar. Es decir, es cualquier página web disponible en Internet.

Un servidor web, también llamado servidor HTTP, es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales o unidireccionales y síncronas o asíncronas con el cliente, y generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. Es decir, es el software que se encarga de despachar el contenido de un sitio web al usuario.

2.8. Ataques informáticos

Existen numerosos ataques cibernéticos diferentes, entre los que se encuentran:

2.8.1. Ataques de denegación de servicio

Existen ataques de denegación de servicio (DoS) o de denegación de servicio distribuido (DDoS), la diferencia entre ambos depende de si el ataque proviene de más de una máquina a la vez o no. Su finalidad es atentar contra la disponibilidad de un servicio en la máquina objetivo.

Los DDoS son mucho más difíciles, o incluso imposibles, de prevenir, detectar y frenar. Un ejemplo de DDoS puede ser una Botnet y un ejemplo de DoS puede ser IP Flood que consiste en agotar el ancho de banda disponible e impedir el acceso a usuarios legítimos.

- **Botnet**

  Es una red de equipos informáticos que han sido infectados con software malicioso que permite su control remoto, obligándoles a enviar spam, propagar virus o realizar ataques de denegación de servicio distribuido (DDoS) sin conocimiento ni consentimiento de los propietarios reales de los equipos.

2.8.2. Ataques de intermediario

Son llamados ataques “man in the middle”o MITM. Son realmente peligrosos puesto que el atacante busca poder observar e interceptar mensajes entre las dos víctimas y procurar que ninguna de las víctimas conozca que el enlace entre ellos ha sido violado. De esta forma, adquiere la capacidad de leer, insertar y modificar a voluntad. Por ejemplo, con Wireshark se puede interceptar el tráfico de la red.

2.8.3. Ataques de suplantación

- **Suplantación de un servidor**

Suplantación de un servidor légitimo con uno falso para poder robar credenciales válidas. Es indispensable haber hecho un reconocimiento previo del servidor legítimo para conocer qué servicios están corriendo, para crear el servidor falso idéntico.

- **IP y ARP spoofing**

  Consiste en enviar mensajes falsos, suplantando la IP o la MAC del objetivo, para interceptar la información.

2.8.4. Ingeniería social

La ingeniería social es una técnica que ayuda a revelar debilidades en las defensas de una organización. Consiste en utilizar la manipulación y el engaño para lograr que una persona haga algo indebido, como revelar su nombre de usuario y contraseña, su correo electrónico... o cualquier dato o información que pueda ser de interés para llevar a cabo ataques como por ejemplo Phising (técnicas que persiguen confundir a una víctima ganándose su confianza haciéndose pasar por una persona, empresa o servicio de confianza, para manipularla y hacer que realice acciones que no debería realizar).

Algunas de las técnicas basadas en técnicas informáticas son: phising, spam mail, pop-ups… Y algunas basadas en telefonía móvil como APPs maliciosas, infectar APPs legítimas, aplicaciones de seguridad falsas, SMS Phising…

2.8.5. Ataque por fuerza bruta

Un ataque por fuerza bruta consiste en intentar recuperar u obtener credenciales válidas (contraseñas, nombres de usuarios existentes, etc.), probando todas las posibles combinaciones hasta encontrar aquella que permite el acceso, o hasta que se decide finalizar el ataque sin éxito.

Este ataque se puede realizar de forma manual o automatizada con ayuda de un diccionario y una herramienta software, por ejemplo, Hydra.

Tipos de ataques de diccionario

- **Ataques de Diccionario Online**

  Este tipo de ataque realiza peticiones directas contra el servicio que está corriendo en la máquina remota, y en el momento en que el servicio acepte las credenciales en vez de rechazarlas, la aplicación las considerará correctas.

  La gran desventaja es que puede desembocar en ataque DoS, debido a que el servidor se está saturando con las peticiones y no puede contestar las de sus clientes legítimos. Para que esto no ocurra, se deben ajustar los Tasks (conexiones en paralelo contra un objetivo) y el tiempo de espera por cada hilo. Herramientas que realizan este tipo de pruebas son: Hydra, Medusa, módulos de Metasploit, Wpscan…

- **Ataques por diccionario Offline**

  En este caso el ataque se realiza mediante la rotura de un cifrado. Herramientas existentes para llevar a cabo esta labor: John the Ripper, Hashcat, Hash-identifier, Crakstation…

2.8.6. Footprinting vs fingerprinting

El footprinting, también denominado reconocimiento, es el proceso de recogida de información a través de Internet. Es totalmente legal de realizar sin consentimiento, ya que la información que se obtiene es pública (acudiendo a documentos que contienen metadatos, a redes sociales, medios de comunicación, etc.). Esta técnica se conoce como OSINT y trata de obtener toda la información posible del sistema, de la red o usuario objetivo.

Existen dos tipos de reconocimiento:

- **Activo:** son peticiones directas al dominio, como por ejemplo whois, traceroute, etc.
- **Pasivo:** son peticiones indirectas a través de motores de búsqueda, servicios online…

El fingerprinting es una técnica que se utiliza para recopilar información más específica del objetivo, y no es pública, por lo que puede considerarse como delito si se lleva a cabo sin consentimiento. Dicha información puede ser el estado de los puertos del sistema objetivo, vulnerabilidades existentes, versiones de software, sistema operativo, etc. Una herramienta utilizada para este procedimiento puede ser por ejemplo nmap.

2.8.7. Ataques XSS (Cross Site Scripting)

Gracias a un formulario vulnerable se puede incrustar código (HTML, JS…) en la misma aplicación que se está utilizando. Si en la aplicación se puede incrustar este código, significa que es vulnerable a XSS.

Tipos de ataques XSS

- **Reflejo**
  Solo permanece en ejecución mientras se haya mandado la cadena por el canal GET o POST.

- **Permanente**
  El código se queda de forma permanente hasta que el web master lo borre. Por ejemplo, en un post de un foro.

Estos ataques pueden ser inofensivos, como por ejemplo un mensaje de alerta que se ejecute en el navegador, pero también se puede insertar código malicioso (por ejemplo, que se obtenga la cookie que ha generado la sesión de ese usuario y la mande a un servidor malicioso para que se almacene y así poder impersonificar a dicho usuario).

Existen diccionarios para poder agilizar la batería de pruebas con XSS. También existen los Local (o Remote) File Incusion, que consiste en poder acceder a información sensible del sistema por medio de un fallo en el código de la aplicación web.

2.8.8. Malware

Un malware es un programa software malicioso, que realiza tareas dañinas en un sistema informático de forma intencionada y sin el conocimiento del usuario.

Algunos tipos de malware

- **Virus**
  Normalmente ralentizan el funcionamiento de la máquina.

- **Gusano**
  Su característica principal es la propagación utilizando uno o varios exploits contra un determinado servicio remoto de las máquinas que funcionan en la red en la que trabaja la víctima.

- **Ransomware**
  Se encarga de cifrar el contenido accesible al usuario que ejecuta el fichero malicioso, algunos incluso imposibilitan el uso de la máquina remota hasta el pago de un rescate.

- **Spyware**
  Espía el contenido de la máquina, sea en sus ficheros o interfaces de entrada y/o salida. Un keylogger es un tipo de spyware.

- **Troyano**
  Se presenta al usuario como un programa aparentemente legítimo e inofensivo, pero que, al ejecutarlo, le brinda a un atacante acceso remoto al equipo infectado.

- **Rootkits**
  Se encarga de ofuscar la detección de códigos maliciosos como los citados previamente e intentan adquirir privilegios de administrador y con ello ofuscar los procesos e incluso realizar tareas maliciosas.

2.8.9. Pivoting y escalada de privilegios

El pivoting es una prueba dentro de un test de intrusión, se lleva a cabo una vez que se ha conseguido acceso en el sistema objetivo (fase de post-explotación).

Se trata de intentar saltar a través del sistema comprometido a otras máquinas de la red o de otra diferente, utilizando técnicas de redirección y enrutamiento. Es decir, acceder a equipos en otras redes que no deberían ser accesibles. De esta forma, se podrían propagar las consultas maliciosas contra más máquinas.

Escalar privilegios consiste en encontrar y explotar una vulnerabilidad o fallo de configuración, una vez que se dispone de acceso a la máquina objetivo, con permisos limitados, de forma que se consigue elevar el acceso a un usuario con más permisos. También es una técnica de post-explotación.

3. Contextualización

Actualmente, tanto la Ciberseguridad, como el papel de auditor informático, han tomado una especial relevancia en los sectores TI.

Un auditor informático (o Pentester) es el encargado de analizar la seguridad de un sistema o entorno informático. Se centra en encontrar vulnerabilidades y sugerir soluciones para prevenir posibles ataques cibernéticos.

Realizar un test de intrusión conlleva mucho tiempo, lo que puede ser un gran problema debido a que durante ese periodo de tiempo en el que estén expuestas vulnerabilidades, y no se implanten medidas de prevención, se pueden recibir ataques informáticos por parte de ciberdelincuentes, pudiendo provocar consecuencias devastadoras.

Cuanto menos tiempo se tarde en identificar debilidades, antes se podrán implantar medidas de prevención. Por lo tanto, el factor tiempo es de elevada importancia en este sector.

Es precisamente **por este motivo por el que se decide diseñar una herramienta automatizada** de Pentesting, la cual contemple todas las fases de auditoría, **desde la etapa inicial, hasta la generación del reporte con los resultados**.

Gracias a esta herramienta se consigue decrementar notablemente el tiempo total de auditoría, sobre todo en las fases iniciales de recopilación de información, así como en la toma de notas de los resultados que se van obteniendo, ya que la herramienta almacena toda esta información de forma autónoma y a tiempo real, evitando por tanto errores humanos al registrar dicha información.

La herramienta **está programada pensando en la idea de que cualquier persona puede utilizarla**, aunque no disponga de conocimientos en esta área, siendo capaz de interactuar con ella y obtener el reporte, de forma muy sencilla.

Si esta persona no dispone de conocimientos en la materia, no sabrá interpretar la información obtenida, y tampoco sabrá investigar a través de los resultados, así como comprobar si las vulnerabilidades encontradas se tratan de falsos positivos o no, etc., pero

supone una gran ventaja para el auditor, ya que no consume tiempo en ello y puede dedicarlo al análisis y exploración más exhaustiva a raíz de estos resultados.

Sin embargo, si la persona que interactúa con la herramienta es el auditor (experto en la materia), será menor el tiempo dedicado que si realizara todo de forma manual.

Todo esto es de gran importancia, ya que cada día se producen más ataques cibernéticos y además cada vez más peligrosos.

Actualmente son cada vez más los servicios que disponen de una gran vulnerabilidad, a través de la cual existe la posibilidad de acceso al sistema de posibles atacantes. De ahí la prioridad en detectarlas y corregirlas antes, para impedir que los programas o ficheros maliciosos que se reciben a través de cualquier medio informático pueden provocar daños irreparables con consecuencias muy peligrosas…

Si un atacante consigue el acceso al sistema, las consecuencias pueden ser totalmente devastadoras. Dispondría de acceso a información sensible y confidencial, pudiendo visualizarla, descargarla, modificarla, borrarla… Además, podría editar las configuraciones de forma incorrecta en servicios para poder seguir realizando ataques en un futuro, etc.

Hace relativamente poco tiempo se utiliza Internet e inicialmente se utilizaban muchas menos utilidades que actualmente. Hoy en día utilizamos Internet para casi todo (compras, comunicaciones, transacciones, etc.). La realidad es que nuestra vida es más sencilla, pero también está más expuesta a ataques.

Por tanto, se deben realizar auditorías informáticas de forma periódica para comprobar que el nivel de seguridad de las infraestructuras y, por tanto, de la información que contienen y protegen, es la correcta. También se debe comprobar que existen las medidas correctas de prevención y detección de intrusos, con la configuración debida.

Cada día se localizan nuevas vulnerabilidades en servicios que utilizamos constantemente, sin darnos cuenta de que un ataque de una o varias personas con fines maliciosos puede ser totalmente dañina y prácticamente imposible de parar si no se tienen implantadas las medidas correctas.

Algunas de estas vulnerabilidades dependen de nosotros mismos como, la utilización de contraseñas débiles, la protección de estas, no cambiar las contraseñas por defecto… Y otras

son ajenas a nosotros, pero debemos ser conscientes de que existen y de cómo podemos protegernos frente a ellas.

La explotación de algunas vulnerabilidades por parte de un atacante también depende del impacto que éstas tengan. Algunas se catalogan como de baja importancia, debido a que, aunque se exploten, las consecuencias no son graves. Pero otras pueden ser totalmente críticas como, por ejemplo, que un ciberatacante consiga acceso a nuestros sistemas, pudiendo realizar cualquier actividad delictiva.

Por ello, se deben realizar auditorías con asiduidad y además las debe llevar a cabo un profesional que esté al tanto de las novedades del momento, para comprobar si las nuevas vulnerabilidades se presentan o no en el sistema analizado, o si las medidas de prevención actuarían frente a dicho ataque si se llevase a cabo.

Existen multitud de herramientas útiles en Internet para cualquiera de las fases de la auditoría, así como exploits o softwares más concretos para comprobar vulnerabilidades determinadas, etc.

Nessus es posiblemente el programa más utilizado en cualquier auditoría de seguridad en la actualidad. La autora de este documento se ha basado en la idea general de dicho programa, para diseñar una herramienta que además de realizar la auditoría y generar el informe de forma automática, que sea totalmente interactiva con el usuario.

La herramienta está compuesta por un conjunto de Scripts interconectados en Bash, por lo que ocupa poco espacio en disco y es muy fácil de ejecutar. Además, **no requiere de instalación**.

Actualmente no existía ninguna herramienta de este tipo, que dispusiese de módulos para llevar a cabo todas las fases de una auditoría, que fuese **automatizada desde la fase inicial hasta la generación del informe, totalmente interactiva con el usuario, muy fácil de utilizar e intuitiva, capaz de comprobar las dependencias de forma autónoma** y puede ser utilizada por cualquier persona, sin necesidad de conocimientos en la materia, etc.

Funciona en cualquier distribución de Linux. Además, es capaz de realizar una auditoría a un dominio web, a la propia máquina que ejecuta la herramienta (si es Linux) y a cualquier máquina dentro de la red (sea Linux o Windows).

Una vez obtenido el reporte, el auditor debe analizarlo y tratar de explotar las vulnerabilidades que se han encontrado, así como aportar soluciones para poder corregirlas y prevenir futuros ataques.

A modo de resumen o conclusión, cada día se encuentran nuevas vulnerabilidades y estamos en constante peligro, por lo que se debe llevar un control lo más continúo posible para comprobar que nuestros sistemas, y, por tanto, nuestra información, esté a salvo. Cuanto menos tiempo esté una vulnerabilidad expuesta, más seguro será el sistema, por lo que el tiempo que se tarde en realizar una auditoría de seguridad es de suma importancia. Es por este motivo que se diseña una herramienta automatizada que acelera notablemente el proceso y abarata costes.

4. Diseño de la propuesta

La propuesta del presente documento es la redacción de una **metodología para realizar una auditoría de seguridad a un sistema o entorno de forma muy eficiente y sencilla, utilizando una herramienta diseñada y desarrollada para tal fin.**

4.1. Objetivos

El objetivo general de la metodología automatizada de Pentesting propuesta, es **facilitar el trabajo a la hora de realizar una auditoría, simplificando las tareas, acortando los tiempos y aumentando la fiabilidad, utilizando la herramienta diseñada y desarrollada para ello**.

Para ello, se deben cumplir los siguientes objetivos específicos:

- Dar a conocer este proyecto y esta herramienta al sector, mediante inserciones en publicaciones de ciberseguridad, videos promocionales y demos para que se puedan comprobar los resultados.
- Actualizarla y ampliarla continuamente.
- Redactar una guía sencilla y atractiva.
- Evaluar la metodología y la herramienta por quienes la utilicen para sugerencias de mejora.

4.2. Alcance de la herramienta creada

El alcance de esta herramienta es muy amplio, ya que contempla todas las etapas de una prueba de intrusión, desde la fase inicial, hasta la fase de post-explotación y la de generación del reporte de resultados.

Está compuesta por módulos claramente diferenciados, que se presentan a través de las opciones de los menús que se muestran de forma gráfica al usuario. Además, contempla la evaluación de seguridad en diferentes ámbitos como el análisis a un dominio web cualquiera,

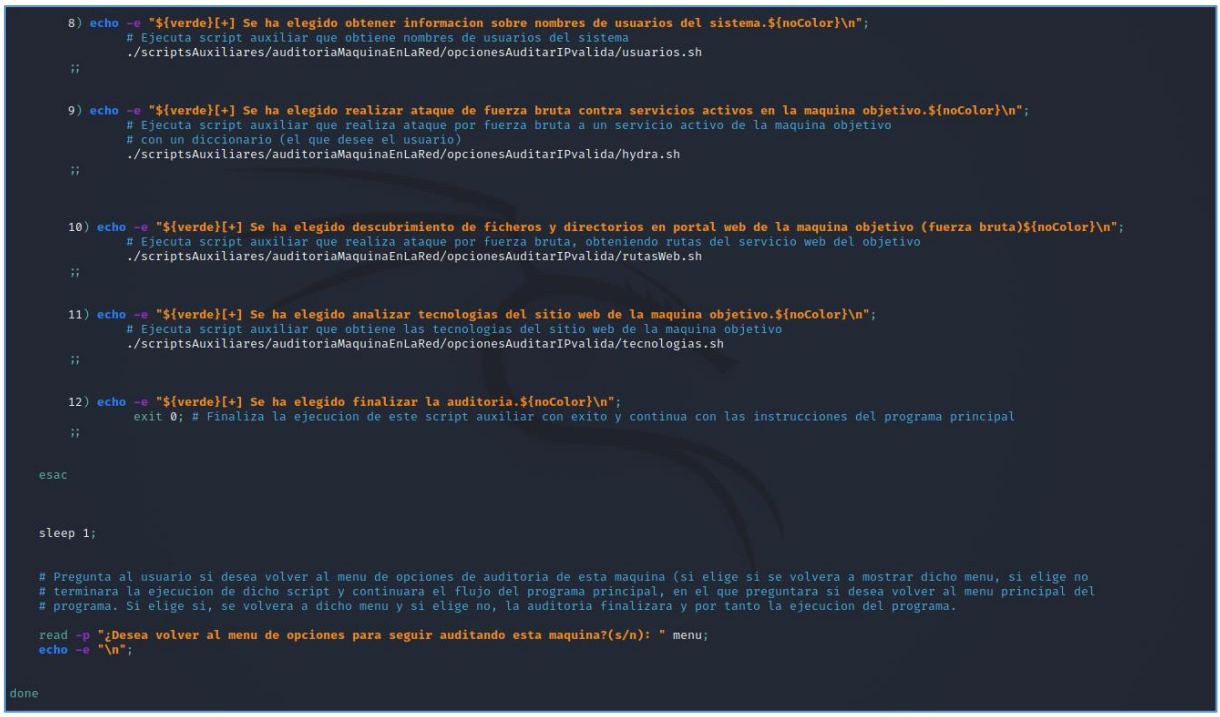
auditar la propia máquina si se trata de un entorno Linux, e incluso analizar cualquier máquina dentro de la red, pudiendo ser éste un sistema Linux o Windows.

Este software está implementado bajo el lenguaje de programación Bash y está compuesto por múltiples scripts interconectados (para aumentar la modularidad y claridad del programa), por tanto, debe ser ejecutado en un entorno Linux.

Ejemplo de script auxiliar que muestra y ejecuta las opciones del menú para auditar una máquina de la red (llamando a otros scripts auxiliares):

```bash
#!/bin/bash

# Script auxiliar que muestra el menu de opciones para auditar una maquina de la red (se ejecuta cuando el programa principal recibe una IP valida)

sleep 3; # Tiempo de espera para que el usuario pueda leer los mensajes informativos del programa

menu='s'; # Variable inicializada en 's' (si) para que se ejecute al menos una vez el bucle que muestra el menu

while [ "$menu" = 's' ] || [ "$menu" = 'S' ]
do
    # Muestra por consola el menu de opciones para auditar una maquina de la red
    echo -e "${azul}~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
    echo -e "-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-";
    echo -e "|            MENU DE OPCIONES DISPONIBLES           |";
    echo -e "-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-•-\n";

    echo -e "1. Obtener informacion sobre la IP consultada.\n";
    echo -e "2. Consultar puertos abiertos y servicios activos en la maquina objetivo.\n";
    echo -e "3. Consultar la version de los servicios corriendo en los puertos abiertos en la maquina objetivo.\n";
    echo -e "4. Consultar Sistema Operativo de la maquina objetivo.\n";
    echo -e "5. Analisis vulnerabilidades de los puertos abiertos en la maquina objetivo.\n";
    echo -e "6. Analisis vulnerabilidades web en la maquina objetivo.\n";
    echo -e "7. Crear un diccionario (para realizar ataque fuerza bruta).\n";
    echo -e "8. Obtener informacion sobre nombres de usuarios del sistema (servicio smb).\n";
    echo -e "9. Realizar ataque fuerza bruta contra servicios activos en la maquina objetivo.\n";
    echo -e "10. Descubrimiento de ficheros y directorios en portal web de la maquina objetivo (fuerza bruta).\n";
    echo -e "11. Descubrir tecnologias sitio web en maquina objetivo.\n";
    echo -e "12. SALIR (Finalizar auditoria).\n";
    echo -e "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~${noColor}\n\n";


    opcion=0; #Variable para que se ejecute al menos una vez el bucle que solicita una opcion a ejecutar al usuario
    while [ "$opcion" -lt 1 ] || [ "$opcion" -gt 12 ] # Pide continuamente una opcion hasta recibir una valida (existente en el menu)
    do
        read -p "¿Que opcion desea ejecutar?:   " opcion; # Pide la opcion, lee lo que inserta el usuario y lo almaceno en la variable 'opcion'

        if [ "$opcion" -lt 1 ] || [ "$opcion" -gt 12 ]; then # Si inserta un valor no contemplado, muestra mensaje informativo de error y se repite el bucle
                echo -e "\n${rojo}[!] Inserte un valor valido.${noColor}\n\n";
        fi

    done
```

```bash
    case $opcion in # Swich case para cada una de las opciones que elija el usuario en el menu (cada una ejecuta otro script auxiliar)

        1) echo -e "${verde}[+] Se ha elegido la opcion de obtener mas informacion sobre la IP consultada.${noColor}\n";
                # Ejecuta script auxiliar que obtiene informacion de la IP (ubicacion, propietario, etc.)
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/info.sh
        ;;

        2) echo -e "${verde}[+] Se ha elegido la opcion de comprobar puertos abiertos de la maquina objetivo.${noColor} \n";
                # Ejecuta script auxiliar que realiza escaneo simple de puertos y servicios activos en la maquina objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/puertosYserviciosSimple.sh
        ;;


        3) echo -e "${verde}[+] Se ha elegido la opcion de obtener la version de los servicios activos en los puertos abiertos de la maquina objetivo.${noColor}\n";
                # Ejecuta script auxiliar que realiza un escaneo mas avanzado, obteniendo tambien la version de los servicios activos
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/versionServicios.sh
        ;;

        4) echo -e "${verde}[+] Se ha elegido la opcion de conocer el Sistema Operativo de la maquina objetivo.${noColor}\n";
                # Ejecuta script que obtiene el sistema operativo de la maquina objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/SO.sh
        ;;

        5) echo -e "${verde}[+] Se ha elegido la opcion de analisis de vulnerabilidades en la maquina objetivo.${noColor}\n\n";
                # Ejecuta script auxiliar que obtiene las vulnerabilidades de los servicios abiertos en la maquina objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/analisisVuln.sh
        ;;

        6) echo -e "${verde}[+] Se ha elegido la opcion de analisis web en la maquina objetivo.${noColor}\n\n";
                # Ejecuta script auxiliar que realiza analisis de vulnerabiliadades en servicios web de la maquina objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/analisisWeb.sh
        ;;

        7) echo -e "${verde}[+] Se ha elegido crear un diccionario para realizar ataque de fuerza bruta.${noColor}\n";
                # Ejecuta script auxiliar que crea un diccionario a gusto del usuario, para utilizar en ataques de fuerza bruta
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/crearDicc.sh
        ;;
```



```
        8) echo -e "${verde}[+] Se ha elegido obtener informacion sobre nombres de usuarios del sistema.${noColor}\n";
                # Ejecuta script auxiliar que obtiene nombres de usuarios del sistema
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/usuarios.sh
        ;;


        9) echo -e "${verde}[+] Se ha elegido realizar ataque de fuerza bruta contra servicios activos en la maquina objetivo.${noColor}\n";
                # Ejecuta script auxiliar que realiza ataque por fuerza bruta a un servicio activo de la maquina objetivo
                # con un diccionario (el que desee el usuario)
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/hydra.sh
        ;;



        10) echo -e "${verde}[+] Se ha elegido descubrimiento de ficheros y directorios en portal web de la maquina objetivo (fuerza bruta)${noColor}\n";
                # Ejecuta script auxiliar que realiza ataque por fuerza bruta, obteniendo rutas del servicio web del objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/rutasWeb.sh
        ;;


        11) echo -e "${verde}[+] Se ha elegido analizar tecnologias del sitio web de la maquina objetivo.${noColor}\n";
                # Ejecuta script auxiliar que obtiene las tecnologias del sitio web de la maquina objetivo
                ./scriptsAuxiliares/auditoriaMaquinaEnLaRed/opcionesAuditarIPvalida/tecnologias.sh
        ;;


        12) echo -e "${verde}[+] Se ha elegido finalizar la auditoria.${noColor}\n";
                exit 0; # Finaliza la ejecucion de este script auxiliar con exito y continua con las instrucciones del programa principal
        ;;

    esac



    sleep 1;


    # Pregunta al usuario si desea volver al menu de opciones de auditoria de esta maquina (si elige si se volvera a mostrar dicho menu, si elige no
    # terminara la ejecucion de dicho script y continuara el flujo del programa principal, en el que preguntara si desea volver al menu principal del
    # programa. Si elige si, se volvera a dicho menu y si elige no, la auditoria finalizara y por tanto la ejecucion del programa.

    read -p "¿Desea volver al menu de opciones para seguir auditando esta maquina?(s/n): " menu;
    echo -e "\n";


done
```

Figura 2: *Código Bash del Script auxiliar de la herramienta que muestra el menú de opciones para auditar una máquina en la red.*

Fuente: *Elaboración propia.*

Al iniciar la herramienta, se obtienen de forma automática los permisos del usuario que la está ejecutando y si es “root” (usuario con permisos máximos en el sistema) se comprueban las dependencias. Si alguna no se encuentra en el sistema, se instalará.

Figura 3: *Proceso inicial de la herramienta, siendo root el usuario que la ejecuta.*

Fuente: *Elaboración propia.*

Si, por el contrario, el usuario que va a llevar a cabo la auditoría no dispone de estos permisos de superusuario, las dependencias no se podrán instalar, por lo que se pregunta al usuario si decide continuar o salir del programa, debido a que éste puede no funcionar de la forma esperada en algunos puntos donde se requiera del uso de una herramienta que no esté instalada en el sistema.

```
┌──(kali㉿kali)-[~/Desktop/HackSystemKiller]
└─$ bash HackSystemKiller.sh

Iniciando script ...
[+] Se inicia el script correctamente.


[+] El usuario que va a realizar la auditoria es kali.
[!] Se deben poseer permisos root para comprobar e instalar las dependencias.

El programa puede no funcionar correctamente si falta alguna dependencia. Desea continuar (s/n)?:
```

Figura 4: Proceso inicial de la herramienta, sin poseer permisos de superusuario.

Fuente: Elaboración propia.

Instaladas o no las dependencias necesarias, si no se decide finalizar la ejecución de la herramienta, se inicia el programa y se visualiza el logo de ésta junto con la leyenda informativa de colores, para facilitar la comprensión al usuario.

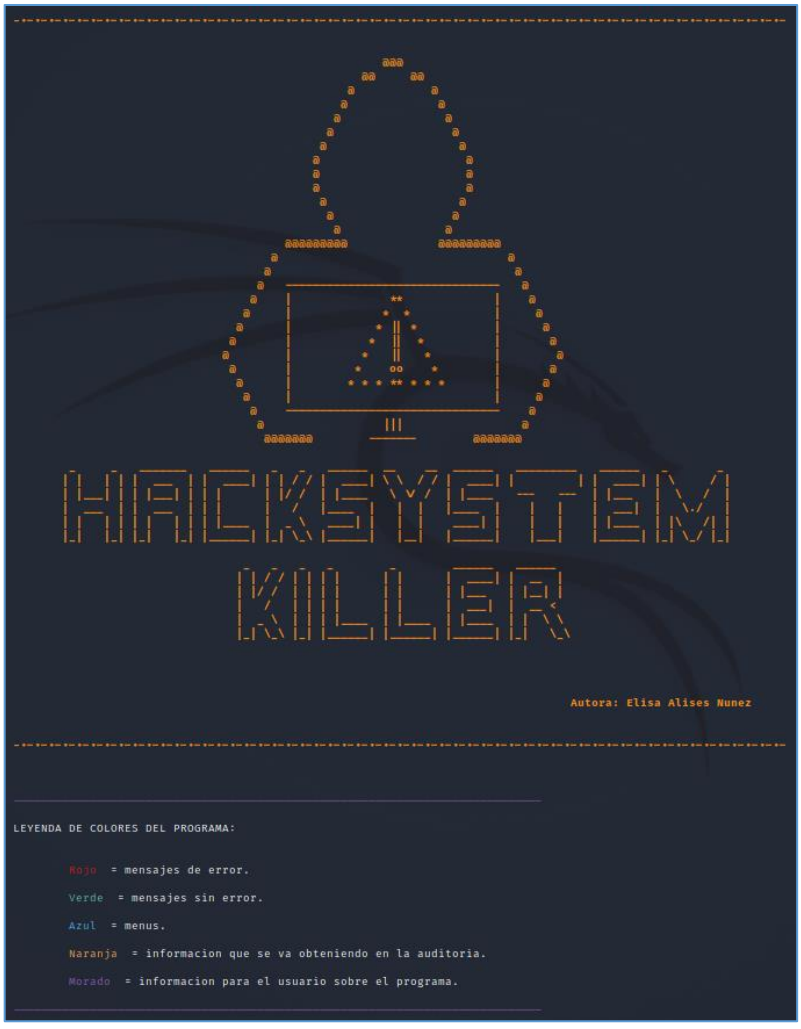


Figura 5: Visualización inicial de la herramienta.

Fuente: Elaboración propia.

A continuación, se va a comprobar si se ha realizado alguna auditoría con la fecha del día que se está ejecutando la herramienta. Si no se ha realizado ninguna, se creará de forma automática la estructura de ficheros donde se almacenará el reporte que se va a ir completando en tiempo real. Y después se pedirá al usuario el nombre que desea que tenga dicho documento, se creará y se mostrará el menú principal de opciones del programa.

Figura 6: Creación estructura de ficheros y menú principal de la herramienta.

Fuente: Elaboración propia.

Si, por el contrario, el programa detecta que ya existe un reporte con esta fecha, el usuario tendrá dos opciones, o mostrar los nombres de estos ficheros y poder crear uno nuevo, o finalizar la auditoría.

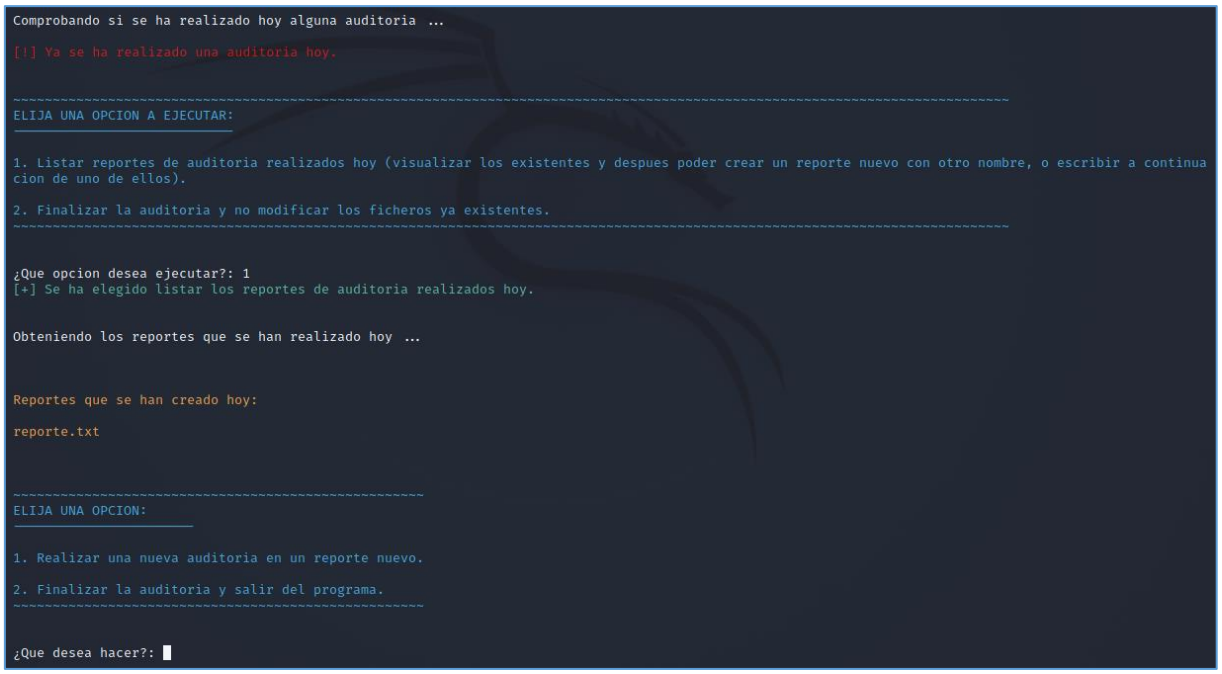


Figura 7: *Menú de opciones de la herramienta cuando ya se ha realizado alguna auditoría en esa fecha.*

Fuente: *Elaboración propia.*

Una vez se ha creado el fichero en el que se irán almacenando todos los resultados, el programa muestra el menú principal de opciones, las cuales se irán explicando en el siguiente apartado de este trabajo, cada una en su fase correspondiente para su comprensión y correcta utilización, a la hora de seguir una metodología eficaz.

Se presenta la estructura interna de dicho software, con todas las posibilidades que ésta presenta, para visualizar de forma más simple su alcance:

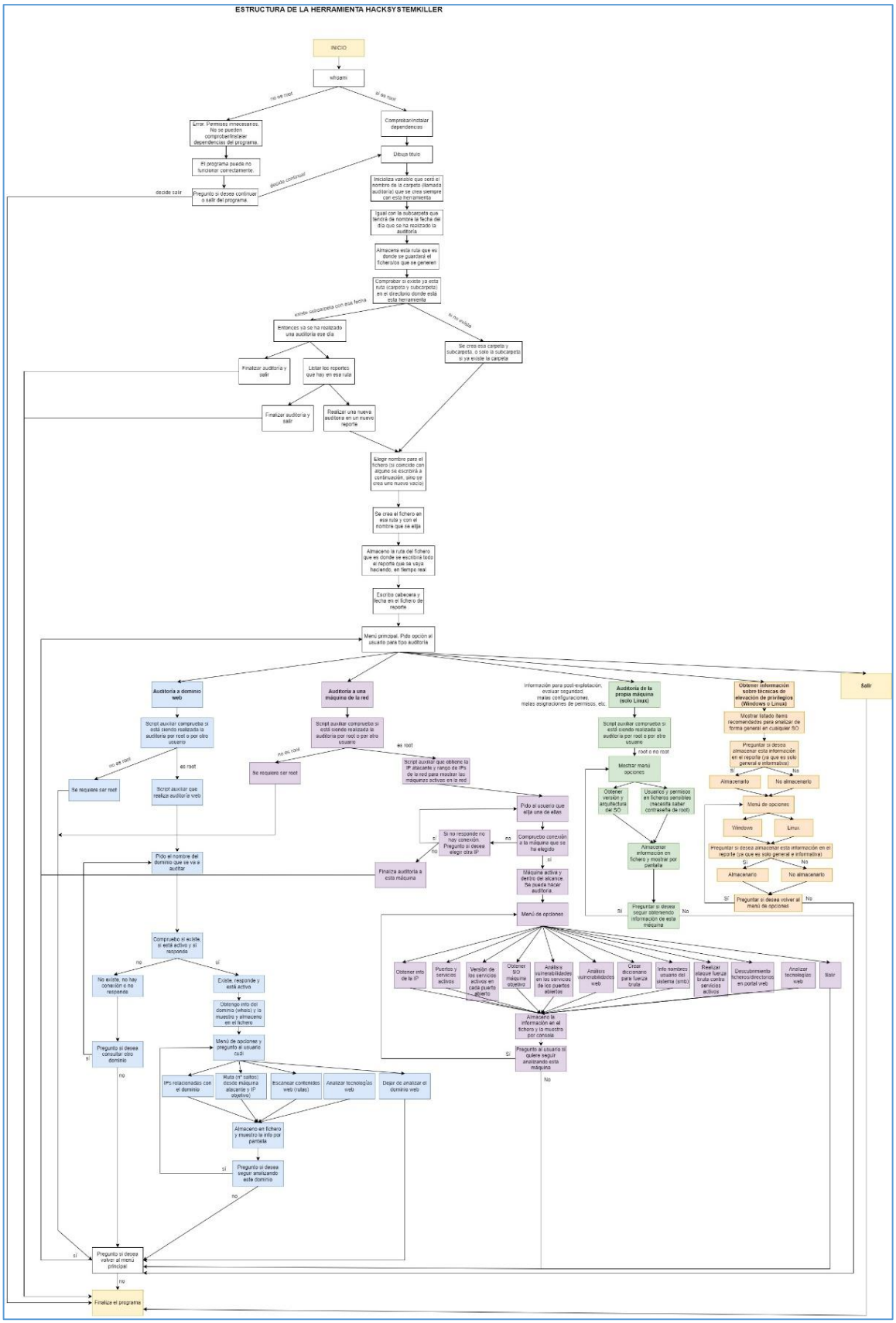


Figura 8: *Estructura interna de la herramienta.*

Fuente: *Elaboración propia.*

4.3. Metodología de pentesting

Una auditoría de seguridad consta de cuatro fases principales bien diferenciadas, las cuales se recomienda llevar a cabo en orden secuencial, debido a que las fases tempranas sirven de ayuda para las posteriores.

4.3.1. Fase inicial

Es la primera fase que se debe realizar. En esta etapa es donde se debe recopilar la máxima información posible acerca del objetivo al que se va a evaluar.

Es la única fase que se puede realizar sin consentimiento previo de forma legal, ya que la información se obtiene a través de fuentes abiertas (Internet), es decir, es pública.

Hay formas de recopilar esta información de forma manual, por ejemplo, con ayuda de herramientas básicas para comprobar el estado y comportamiento de la red como tracert, whois, ping, nslookup… También se puede analizar si por ejemplo existe el fichero "robots.txt" (archivo que se encuentra en la raíz de un sitio web e indica a qué partes no se permite que accedan los rastreadores de los motores de búsqueda) y, si es así, comprobar las rutas encontradas, otras posibles opciones pueden ser la búsqueda a través de motores de búsqueda como Shodan, Censys…

También es útil, por ejemplo, analizar posibles metadatos, o incluso utilizar herramientas como Google Dorks para realizar búsquedas avanzadas y poder encontrar rutas interesantes, Maltego, The Harverster y DMTRY para automatizar la recopilación de información de un dominio o IP, etc.

Sin embargo, la herramienta que se ha diseñado realiza todo esto de forma completamente automatizada en cualquiera de los tipos de auditoría, almacenando en tiempo real los resultados en el fichero de resultados, así como ir mostrándolos por consola al usuario e ir guiándole con mensajes aclarativos o preguntas.

- **Fase inicial de auditoría de un dominio web con la herramienta diseñada**

En el caso de auditar la seguridad de un dominio web (**ver figura 6**), el primer paso que ejecuta la herramienta es comprobar que se trata de un dominio válido, que está activo y que responde a las peticiones de conexión que se le envían.

Una vez se comprueba esto, se realiza la obtención de información inicial y después se muestra un menú de opciones disponibles al usuario. Todo esto se realiza de forma completamente automática, visualizándose en la consola y almacenándose en el fichero al mismo tiempo.

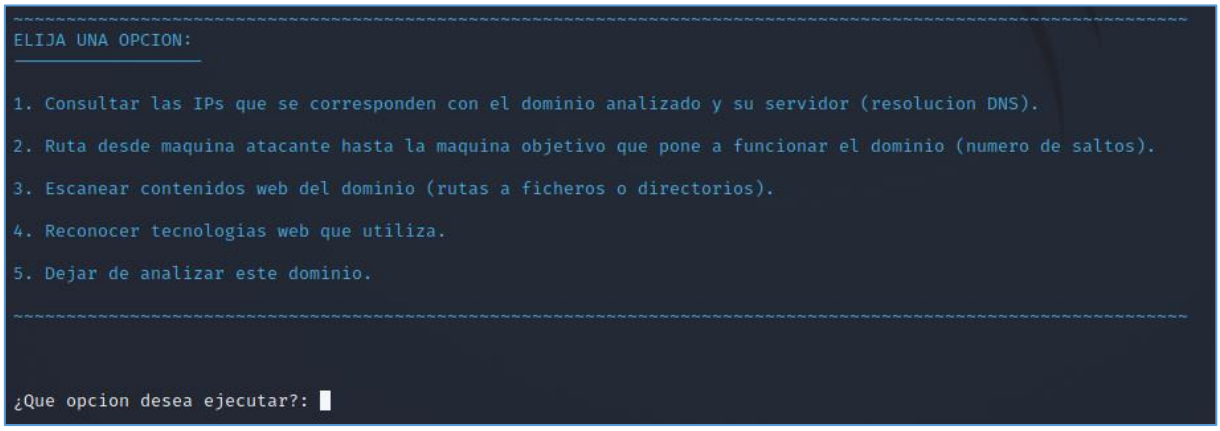


Figura 9: Menú opciones para auditar un dominio web válido con la herramienta creada.

Fuente: Elaboración propia.

Todas las opciones de este menú pueden ejecutarse en esta fase inicial, menos la tercera opción puesto que realiza un ataque por fuerza bruta para obtener las rutas existentes, por lo que se debería llevar a cabo en la siguiente fase, aunque este ataque sea transparente de cara al usuario, se necesita consentimiento.

Es decir, la herramienta en esta fase es capaz de automatizar la obtención inicial de información acerca del dominio, consultar las IP's correspondientes a este (pudiendo así localizar subdominios), así como reconocer las tecnologías que utiliza.

- **Fase inicial de auditoría de la propia máquina Linux con la herramienta diseñada**

En el caso de elegir la opción de auditar la seguridad de la propia máquina que está ejecutando la herramienta (**ver figura 6**), se muestra su correspondiente menú de opciones, siendo la segunda de ellas la que pertenece a esta fase inicial de obtención de información. Dicha máquina debe utilizar el sistema operativo Linux.

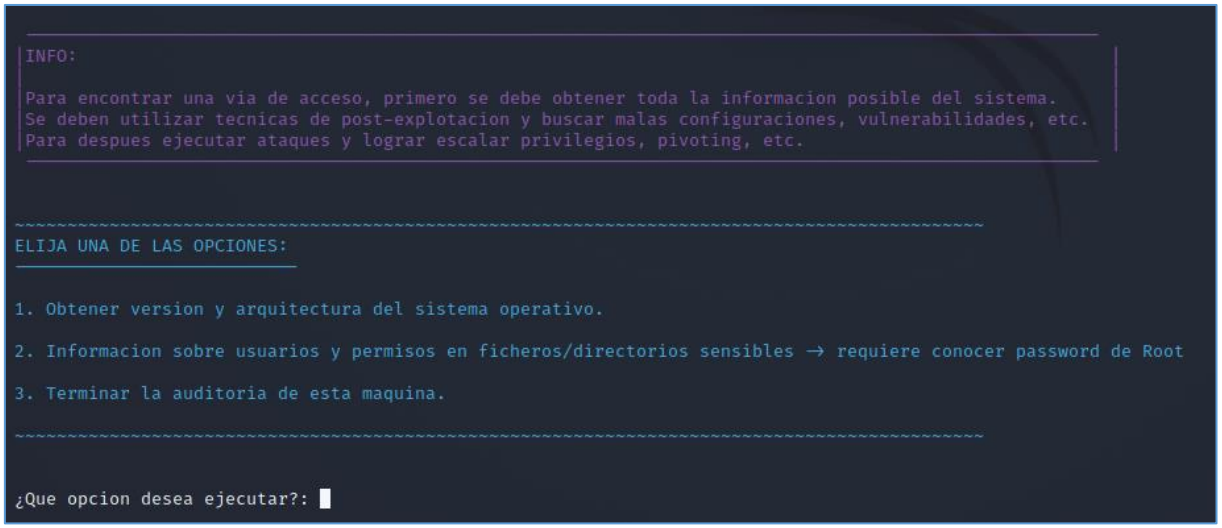


Figura 10: Menú de opciones para auditar la propia máquina Linux con la herramienta creada.

Fuente: Elaboración propia.

En este caso, la herramienta es capaz de obtener información sobre usuarios y permisos en ficheros y directorios sensibles (identificando el usuario actual y el grupo al que pertenece, nombre del usuario que está ejecutando la auditoría, usuarios conectados al sistema en este momento, hora de arranque, últimos usuarios conectados, usuarios con permisos root en el sistema, contraseñas cifradas de cada usuario junto con su fecha de caducidad, permisos que posee el usuario actual, configuraciones SSH, etc.).

- **Fase inicial de auditoría de una máquina de la red con la herramienta diseñada**

Si se decide auditar una máquina activa en la red (**ver figura 6**), el programa detecta de forma automática la IP de la máquina atacante (la que está ejecutando la herramienta) y su rango de red, para obtener y mostrar las IP's de las máquinas activas en la red (las que se pueden auditar).

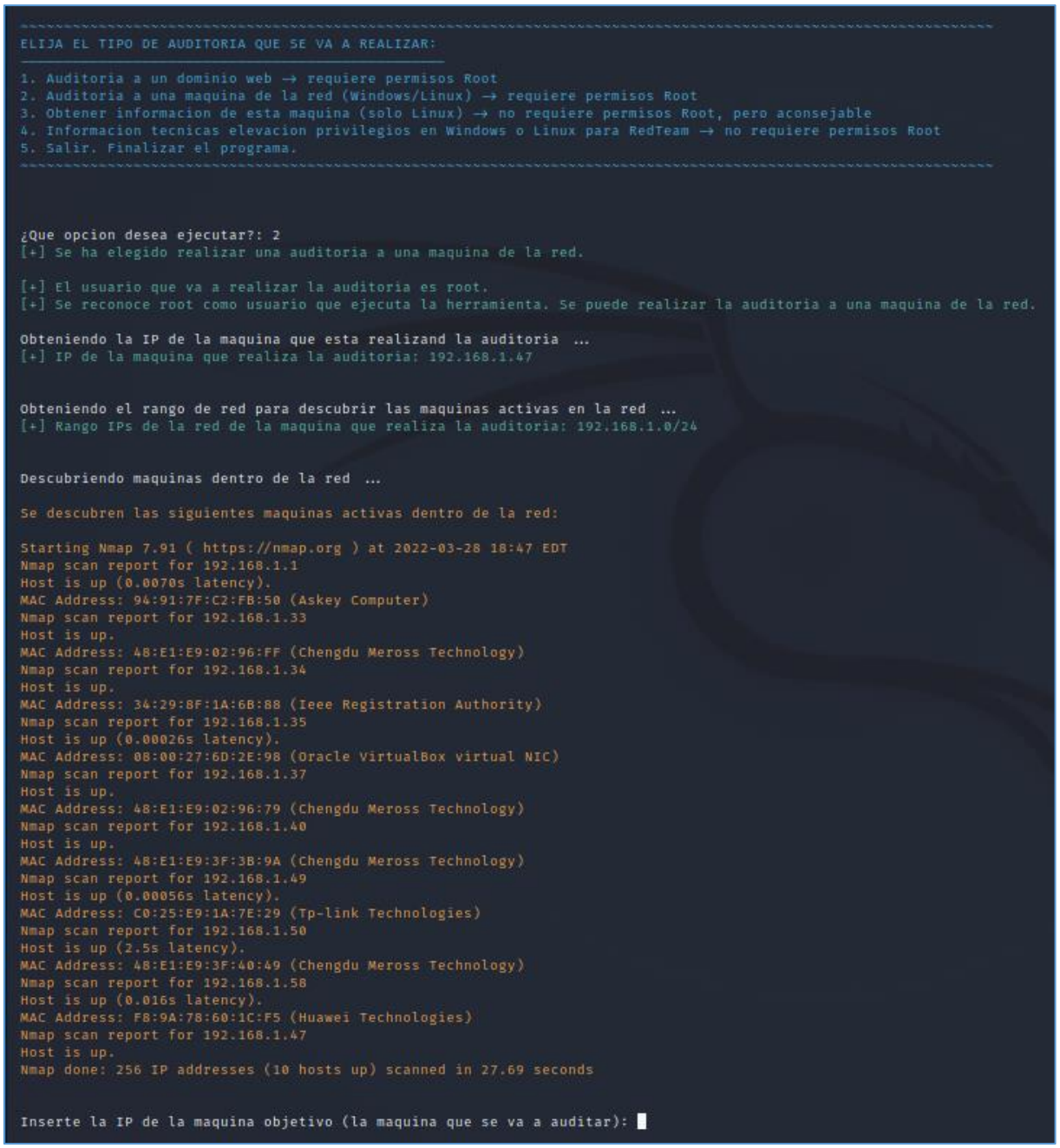


Figura 11: *Detección automática IP y rango de red de la máquina atacante, y las máquinas activas en dicha red.*

Fuente: *Elaboración propia.*

Una vez el usuario inserta una IP válida (dentro del alcance, activa y que responda a las peticiones) se muestra el menú de opciones para poder auditar dicha máquina.

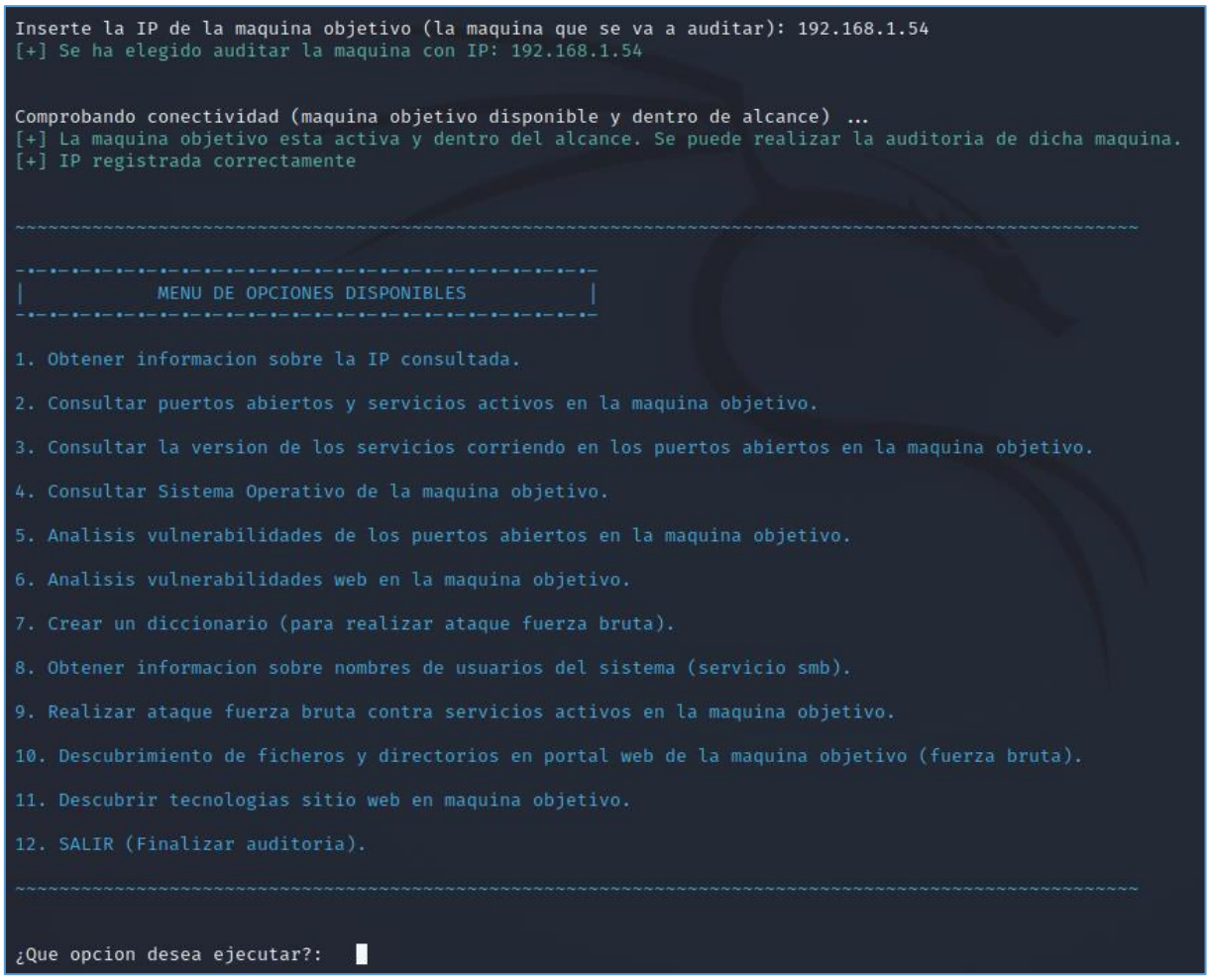


Figura 12: *Menú de opciones de la herramienta para auditar una máquina activa en la red.*

Fuente: *Elaboración propia.*

En este caso, las opciones que se pueden ejecutar en esta fase son la primera y la penúltima, las cuales son capaces de obtener información sobre la IP que se está analizando y descubrir las tecnologías del sitio web de la máquina objetivo.

4.3.2. Fase de análisis

Esta etapa debe comenzar cuando se haya obtenido la máxima información posible acerca del objetivo en la fase anterior. A partir de este punto, se requiere de un consentimiento explícito por parte de la entidad que solicita el servicio, sino es ilegal.

La función del auditor en este momento es examinar los resultados obtenidos hasta ahora, analizando las posibles vulnerabilidades que presenta el sistema o entorno, ejecutando un análisis de puertos, servicios y vulnerabilidades, para después investigar cómo explotarlas.

La información que se obtiene es vital, puesto que permite descubrir qué servicios están corriendo en la máquina objetivo y la versión concreta que están utilizando, así como el sistema operativo del sistema.

Además, un escaneo de puertos se puede realizar contra cualquier tipo de dispositivo que esté disponible en la red de la máquina atacante (teléfonos, puertas de enlace, routers, redes TCP/UDP…).

Gracias a esta información se pueden examinar numerosas debilidades como fallos en configuraciones del propio sistema o de servicios activos, analizar si alguno de ellos es vulnerable a algún exploit público existente, si se utilizan tecnologías obsoletas o versiones antiguas, etc.

También se deben realizar procedimientos para intentar obtener conexiones remotas y ejecutar comandos, recabar información sensible, evaluar si el servicio es seguro o no frente a diferentes técnicas como sniffers, envenenamiento de redes, etc.

En esta fase se puede averiguar qué tipo de medidas de protección tienen implementadas, configuraciones, arquitectura…

Para llevar a cabo estos procedimientos, existen herramientas manuales como nmap, que además presenta numerosos scripts para analizar diferentes vulnerabilidades, netdiscover para descubrir máquinas activas en la red, dirbuster para buscar ficheros o directorios, netcat para verificar qué servicio está corriendo en un puerto, Nikto para analizar vulnerabilidades web…

O también programas más automatizados como por ejemplo Zenmap que es similar a nmap, Nessus que se trata de una aplicación para analizar vulnerabilidades, Owasp-zap y Burpsuite para análisis de aplicaciones web… O incluso multitud de exploits disponibles en Metasploit que analizan si un servicio en concreto es vulnerable a una debilidad determinada.

De nuevo, la herramienta que se ha diseñado realiza todo esto de forma automatizada y guiada, por lo que es muy sencillo de realizar para el auditor.

- **Fase de análisis de un dominio web con la herramienta diseñada**

El análisis de un dominio web en dicha herramienta realiza un ataque por fuerza bruta, por lo que dicha operación pertenece a la siguiente etapa (fase de explotación).

- **Fase de análisis de la propia máquina Linux con la herramienta diseñada**

La herramienta es capaz de obtener la versión y arquitectura del sistema operativo (conociendo la distribución concreta que está utilizando, la versión del Kernel, el listado de mensajes importantes al iniciar el sistema…). Esta corresponde con la primera opción del menú de opciones de este tipo de auditoría, **ver figura 10**.

- **Fase de análisis de una máquina de la red con la herramienta diseñada**

Llegados a este punto, es cuando se deben llevar a cabo la detección del sistema operativo que utiliza la máquina objetivo (opción 4 del menú de opciones de este tipo de auditoría), los escaneos de puertos y servicios (opciones 2 y 3) y los análisis de vulnerabilidades, tanto en los puertos abiertos de la máquina (opción 5), como sus servicios web (opción 6). **Ver figura 12**.

Ejemplo de uso de la herramienta, realizando un escaneo de los servicios activos en los puertos abiertos de la máquina objetivo:

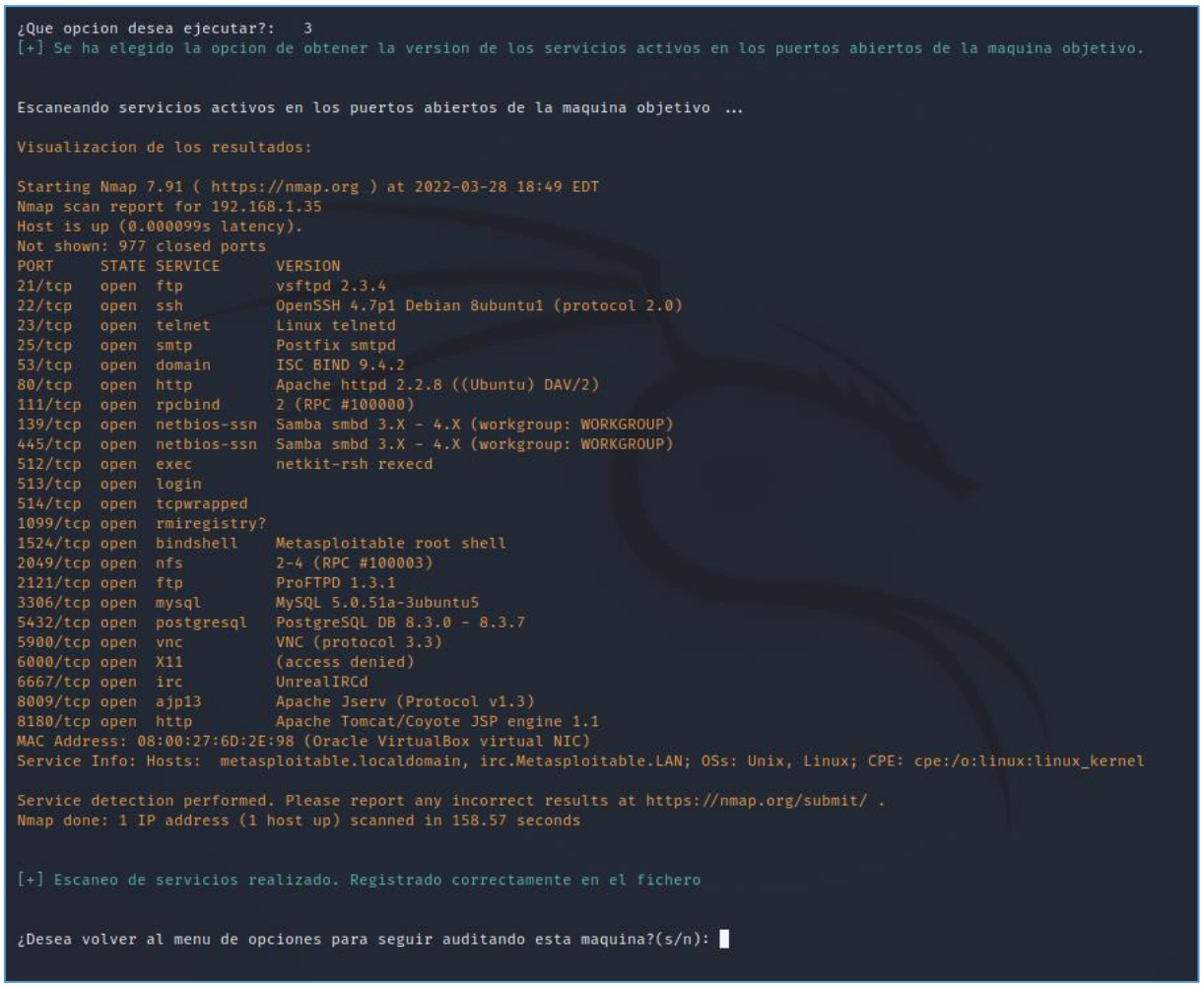


Figura 13: *Ejemplo ejecución escaneo de servicios activos en máquina de la red con la herramienta creada.*

Fuente: *Elaboración propia.*

Ejemplo de análisis de vulnerabilidades en los servicios activos en los puertos abiertos de la máquina objetivo:

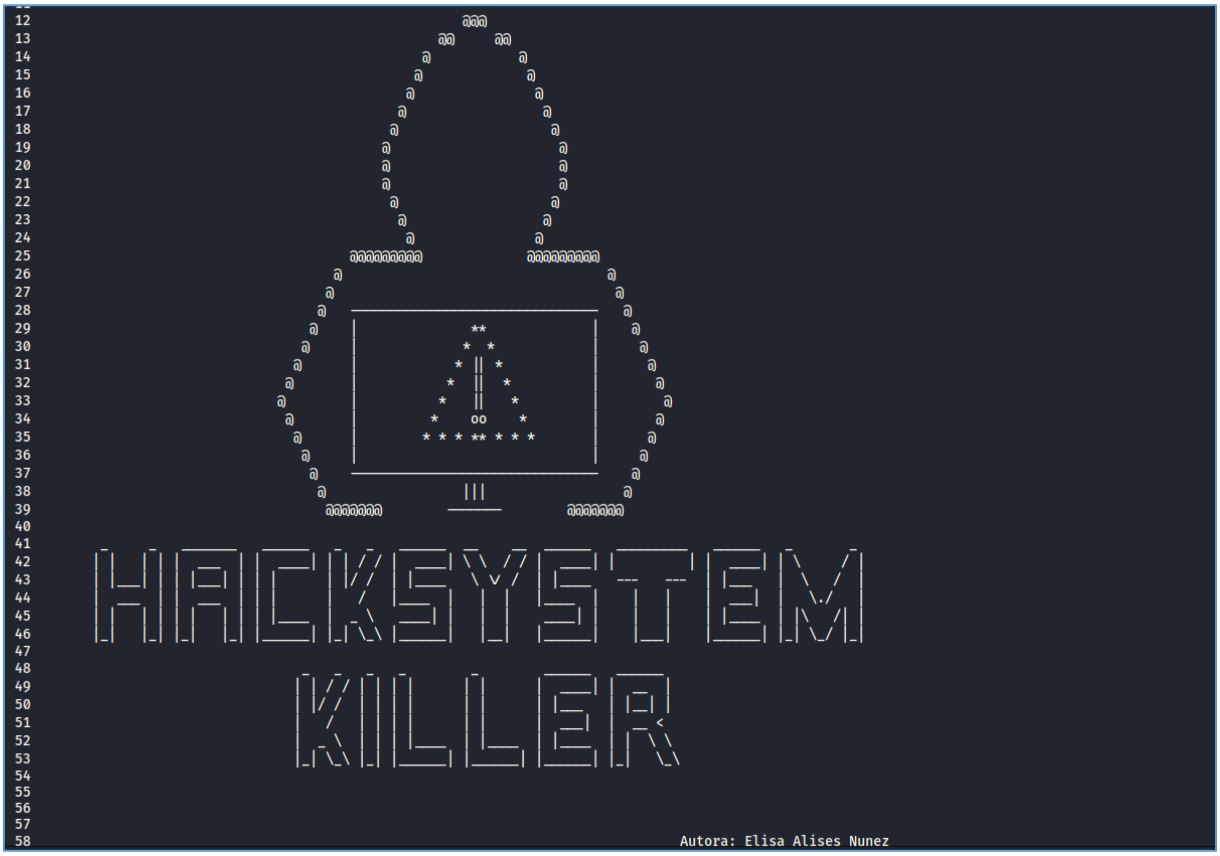
Figura 14: *Ejemplo de ejecución de análisis de vulnerabilidades a máquina de la red con la herramienta creada.*

Fuente: *Elaboración propia.*

Estos análisis tan exhaustivos a veces son muy lentos, por lo que se ha diseñado la herramienta de forma que pregunte al usuario el tiempo máximo de espera deseado antes de realizar el escaneo elegido, para aportar mayor flexibilidad. De esta forma, cuando finaliza este tiempo, se muestran los resultados que se han obtenido hasta ese momento en la consola y a la vez se almacena, como siempre, en el fichero de reporte.

Ejemplo de reporte obtenido con la herramienta en esta fase a una máquina de la red:

```

.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.
|                                                                     |
.  INFORME DE RESULTADOS DE LA AUDITORIA INFORMATICA REALIZADA        .
|                                                                     |
.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.

[+] Auditoria realizada con HackSystemKiller.

```



```


-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

   FECHA DE LA AUDITORIA:
─────────────────────────────
2022-03-30 a las 17:12:28 h



                         ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        |                                                       |
                        | SE HA ELEGIDO OBTENER LAS MAQUINAS ACTIVAS EN LA RED |
                        |                                                       |
                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


──────────────────────────────────────────────────────────────────
| RANGO DE IPs DE LA RED DE LA MAQUINA QUE REALIZA LA AUDITORIA: |
──────────────────────────────────────────────────────────────────
192.168.1.0/24


─────────────────────────────────────
|MAQUINAS ACTIVAS DENTRO DE LA RED: |
─────────────────────────────────────
[+] Se detectan las siguientes maquinas activas dentro de la red:

Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-30 17:12 EDT
Nmap scan report for 192.168.1.1
Host is up (0.0029s latency).
MAC Address: 94:91:7F:C2:FB:50 (Askey Computer)
Nmap scan report for 192.168.1.35
Host is up (0.00020s latency).
MAC Address: 08:00:27:6D:2E:98 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.1.49
Host is up (0.00017s latency).
MAC Address: C0:25:E9:1A:7E:29 (Tp-link Technologies)
Nmap scan report for 192.168.1.36
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.01 seconds




                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```

                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        |                                                              |
                        | SE HA ELEGIDO REALIZAR AUDITORIA A UNA MAQUINA DE LA RED    |
                        |                                                              |
                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


─────────────────────────────────────────────
| IP DE LA MAQUINA OBJETIVO DE LA AUDITORIA: |
─────────────────────────────────────────────
192.168.1.35


────────────────
| CONECTIVIDAD: |
────────────────
[+] La maquina objetivo esta activa y dentro del alcance. Se puede realizar la auditoria.


```

```
125 ──────────────────────────────────────────────────────────────
126 | PUERTOS ABIERTOS Y SERVICIOS ACTIVOS EN LA MAQUINA OBJETIVO: |
127 ──────────────────────────────────────────────────────────────
128 [+] Escaneo de puertos y servicios realizado correctamente.
129
130 Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-30 17:13 EDT
131 Nmap scan report for 192.168.1.35
132 Host is up (0.00020s latency).
133 Not shown: 65505 closed ports
134 PORT      STATE SERVICE
135 21/tcp    open  ftp
136 22/tcp    open  ssh
137 23/tcp    open  telnet
138 25/tcp    open  smtp
139 53/tcp    open  domain
140 80/tcp    open  http
141 111/tcp   open  rpcbind
142 139/tcp   open  netbios-ssn
143 445/tcp   open  microsoft-ds
144 512/tcp   open  exec
145 513/tcp   open  login
146 514/tcp   open  shell
147 1099/tcp  open  rmiregistry
148 1524/tcp  open  ingreslock
149 2049/tcp  open  nfs
150 2121/tcp  open  ccproxy-ftp
151 3306/tcp  open  mysql
152 3632/tcp  open  distccd
153 5432/tcp  open  postgresql
154 5900/tcp  open  vnc
155 6000/tcp  open  X11
156 6667/tcp  open  irc
157 6697/tcp  open  ircs-u
158 8009/tcp  open  ajp13
159 8180/tcp  open  unknown
160 8787/tcp  open  msgsrvr
161 37198/tcp open  unknown
162 39827/tcp open  unknown
163 52357/tcp open  unknown
164 58702/tcp open  unknown
165 MAC Address: 08:00:27:6D:2E:98 (Oracle VirtualBox virtual NIC)
166
167 Nmap done: 1 IP address (1 host up) scanned in 2.16 seconds
168
```

```
170 ──────────────────────────────────────────────────────────────────────────
171 | VERSION DE LOS SERVICIOS ACTIVOS EN LOS PUERTOS ABIERTOS DE LA MAQUINA OBJETIVO: |
172 ──────────────────────────────────────────────────────────────────────────
173 [+] Escaneo de puertos y servicios realizado correctamente.
174
175 Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-30 17:13 EDT
176 Nmap scan report for 192.168.1.35
177 Host is up (0.00013s latency).
178 Not shown: 977 closed ports
179 PORT     STATE SERVICE      VERSION
180 21/tcp   open  ftp          vsftpd 2.3.4
181 22/tcp   open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
182 23/tcp   open  telnet       Linux telnetd
183 25/tcp   open  smtp         Postfix smtpd
184 53/tcp   open  domain       ISC BIND 9.4.2
185 80/tcp   open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
186 111/tcp  open  rpcbind      2 (RPC #100000)
187 139/tcp  open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
188 445/tcp  open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
189 512/tcp  open  exec         netkit-rsh rexecd
190 513/tcp  open  login?
191 514/tcp  open  tcpwrapped
192 1099/tcp open  rmiregistry?
193 1524/tcp open  bindshell    Metasploitable root shell
194 2049/tcp open  nfs          2-4 (RPC #100003)
195 2121/tcp open  ftp          ProFTPD 1.3.1
196 3306/tcp open  mysql        MySQL 5.0.51a-3ubuntu5
197 5432/tcp open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
198 5900/tcp open  vnc          VNC (protocol 3.3)
199 6000/tcp open  X11          (access denied)
200 6667/tcp open  irc          UnrealIRCd
201 8009/tcp open  ajp13        Apache Jserv (Protocol v1.3)
202 8180/tcp open  http         Apache Tomcat/Coyote JSP engine 1.1
203 MAC Address: 08:00:27:6D:2E:98 (Oracle VirtualBox virtual NIC)
204 Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
205
206 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
207 Nmap done: 1 IP address (1 host up) scanned in 158.55 seconds
208
```

```
┌──────────────────────────────────────────────────────────────────────────────┐
| ESCANEO VULNERABILIDADES DE LOS SERVICIOS ABIERTOS DE LA MAQUINA OBJETIVO: |
└──────────────────────────────────────────────────────────────────────────────┘
[+] Escaneo de vulnerabilidades realizado correctamente.


Resultado obtenido con el script auth (autenticacion):

Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-30 17:16 EDT
Nmap scan report for 192.168.1.35
Host is up (0.000095s latency).
Not shown: 65506 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
| smtp-enum-users:
|_  Method RCPT returned a unhandled status code.
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
111/tcp   open  rpcbind      2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/udp   nfs
|   100005  1,2,3      35234/udp   mountd
|   100005  1,2,3      37198/tcp   mountd
|   100021  1,3,4      39827/tcp   nlockmgr
|   100021  1,3,4      44163/udp   nlockmgr
|   100024  1          45021/udp   status
|_  100024  1          58702/tcp   status
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rexecd
```

```
513/tcp   open  login
514/tcp   open  tcpwrapped
1099/tcp  open  rmiregistry?
1524/tcp  open  bindshell    Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
| mysql-empty-password:
|_  root account has empty password
| mysql-users:
|   debian-sys-maint
|   guest
|_  root
3632/tcp  open  distccd      distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
6697/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
| http-default-accounts:
|   [Apache Tomcat] at /manager/html/
|     tomcat:tomcat
|   [Apache Tomcat Host Manager] at /host-manager/html/
|_    tomcat:tomcat
|_http-server-header: Apache-Coyote/1.1
8787/tcp  open  drb          Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
37198/tcp open  mountd       1-3 (RPC #100005)
39827/tcp open  nlockmgr     1-4 (RPC #100021)
52357/tcp open  java-rmi     GNU Classpath grmiregistry
58702/tcp open  status       1 (RPC #100024)
MAC Address: 08:00:27:6D:2E:98 (Oracle VirtualBox virtual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
| smb-enum-users:
|_  Domain: METASPLOITABLE; Users: backup, bin, bind, daemon, dhcp, distccd, ftp, games, gnats, irc, klog, libuuid, list, lp, mail, man, msfadmin,
    mysql, news, nobody, postfix, postgres, proftpd, proxy, root, service, sshd, sync, sys, syslog, telnetd, tomcat55, user, uucp, www-data
```

```
Post-scan script results:
| creds-summary:
|   192.168.1.35:
|     8180/http:
|       tomcat:tomcat - Valid credentials
|_      tomcat:tomcat - Valid credentials
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 190.70 seconds


-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

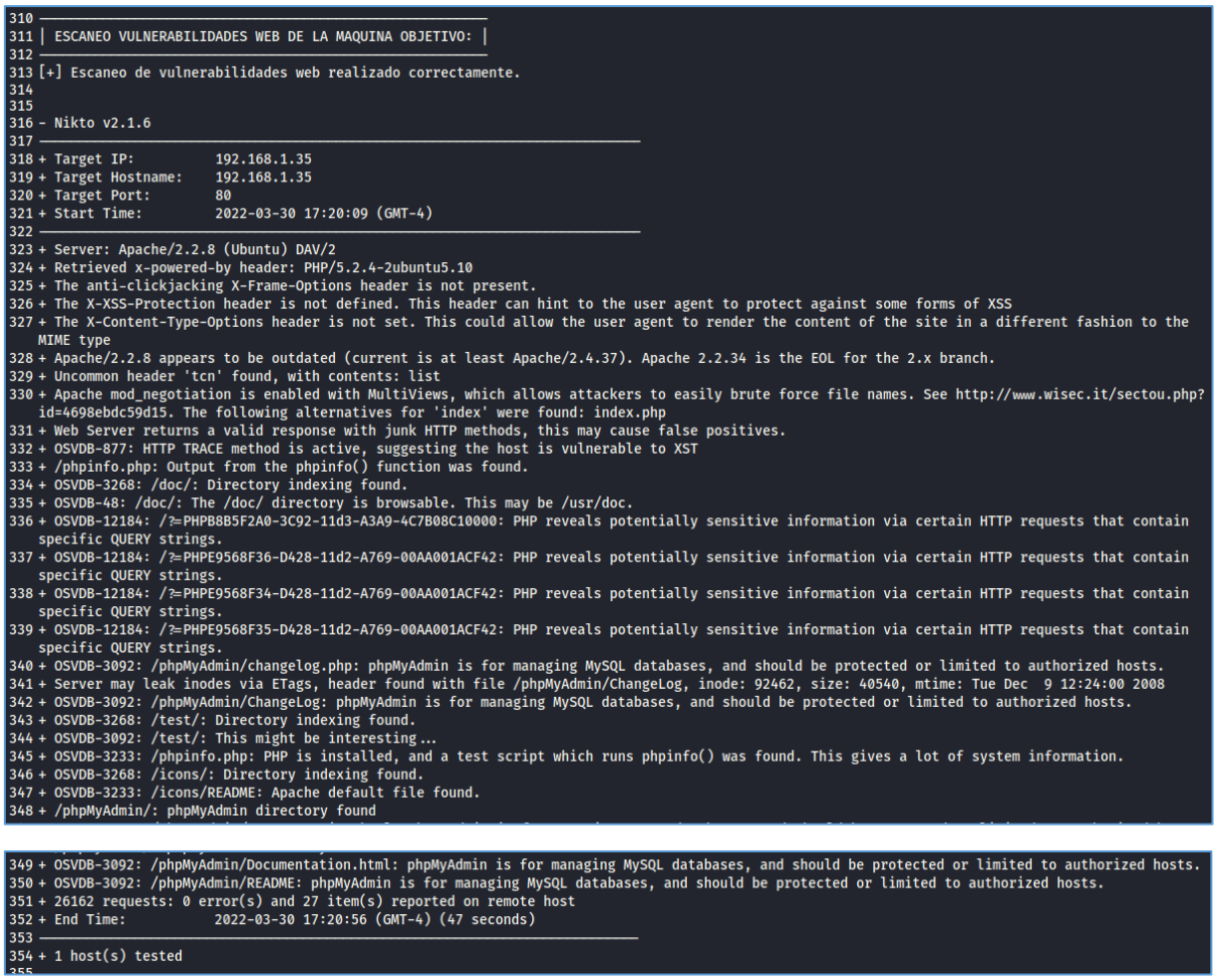


Figura 15: *Ejemplo de reporte obtenido de una auditoría a una máquina de la red con la herramienta creada.*

Fuente: *Elaboración propia.*

Por tanto, a modo de resumen, se recalca que en esta fase es cuando se deben ejecutar y analizar los escaneos de puertos, servicios y vulnerabilidades a la máquina objetivo, así como analizar todos los datos obtenidos en la fase anterior, para encontrar debilidades e intentar explotarlas en la siguiente etapa (fase de explotación).

Se puede apreciar que esta herramienta es capaz de realizar de forma automatizada un escaneo sencillo de puertos abiertos en la máquina objetivo y el servicio que está corriendo en cada uno, y un escaneo más concreto, de forma que también detecte la versión del servicio que está siendo utilizada. También permite realizar análisis de vulnerabilidades web o en dichos servicios activos.

4.3.3. Fase de explotación de vulnerabilidades

En esta fase es donde se llevan a cabo los ataques para probar las debilidades encontradas o buscar nuevas.

Se pueden crear diccionarios (o utilizar alguno ya existente) para realizar ataques por fuerza bruta para probar credenciales hasta dar con unas válidas. Existen herramientas manuales muy útiles para crear diccionarios de forma muy sencilla, como por ejemplo Crunch, Cewl, CuPP, entre otras.

Del mismo modo que hay programas que utilizan el diccionario elegido para automatizar dichas pruebas, como por ejemplo Hydra, Medusa, Wpscan o incluso módulos de Metasploit auxiliares.

Hay que tener presente y muy en cuenta, que estas herramientas realizan estos ataques de diccionario de forma online, lo que quiere decir que puede desembocar en DoS (denegación de servicio) en la máquina objetivo, ya que se satura el servidor con estas peticiones, no pudiendo responder a las consultas de los clientes legítimos. Para que esto no ocurra, se debe ajustar debidamente las conexiones en paralelo y el tiempo de espera en cada intento.

También existen herramientas que realizan dichos ataques de forma offline, es decir, mediante la rotura de un cifrado. Las más conocidas son John The Ripper (normalmente es capaz de detectar de forma automática el algoritmo que se está utilizando), Hashcat, Hash-identifier, Crakstation…

Puede ser de gran ayuda consultar bases de datos de vulnerabilidades publicadas, como por ejemplo Exploit-DB, Rapid7-DB, 0day.today, etc. Así como consultar las claves por defecto de los servicios que se encuentran activos en ese momento y probar si se permite la conexión con dichas credenciales.

En los procedimientos de esta fase, al igual que en la anterior, **se pueden utilizar métodos de evasión** (para evitar que el antivirus detecte las pruebas, ataques, análisis, etc. que se están llevando a cabo). Algún ejemplo puede ser utilizar redes TOR o VPN (que favorecen el anonimato de la IP atacante), o el uso de determinados parámetros en los comandos para que sea más difícil la detección.

En las auditorías **normalmente no se necesita ni se aconseja ocultar la identidad del auditor**, **solo se recomiendan si las medidas perimetrales del cliente bloquean algún análisis o se**

produce algún tipo de obstáculo que imposibilita el proceso de evaluación de seguridad del sistema o entorno.

Esta etapa es la más extensa y compleja, ya que se pueden llevar a cabo multitud de ataques diferentes y puede ser interminable. Y, además, **se debe respetar siempre en todo momento los requisitos legales y acuerdos firmados** (tipos de ataques permitidos, consecuencias asumidas, etc.), **antes de proceder a analizar un sistema o dominio web**.

Por ejemplo, siguiendo una metodología manual, se puede crear un fichero malicioso para después analizar la forma de subirlo al sistema objetivo y que se ejecute para ganar acceso al sistema.

También se pueden probar técnicas de envenenamiento de redes, llevando a cabo ataques MITM ("Man In The Middle"), esnifando el tráfico e interceptándolo para obtener credenciales, leer información confidencial, etc. Para ello hay herramientas como arpspoof, que se encarga de suplantar la MAC del objetivo, de forma que el tráfico pasará también por el atacante y se redirigirá a la víctima, de forma que ésta no lo perciba. Y existen programas para analizar el tráfico de red como Wireshark, Xplico, tcpdump…

Del mismo modo que se puede suplantar un servidor, la identidad de un dominio, o cualquier servicio a través de Metasploit, etc.

Para ello, se debe de poseer consentimiento, ya que si no es un delito.

Además, en esta etapa a veces es útil utilizar técnicas de ingeniería social. Esto consiste en conocer a los usuarios a los que se va a atacar. El objetivo es obtener información sin que lo noten, para poder engañarlos, por ejemplo, mandando un correo electrónico con un fichero malicioso y atraerlos de forma que sea creíble y lo ejecuten.

Para el análisis de una aplicación web se pueden utilizar herramientas como SQLmap para ejecutar inyecciones SQL (utiliza el lenguaje SQL para realizar consultas y obtener información sensible), Commix para ejecutar comandos de forma remota… También se deben probar ataques XSS si existe un formulario vulnerable, con ayuda de una herramienta como por ejemplo Owasp-zap, para interceptar las peticiones y respuestas que recibe la aplicación.

- **Fase de explotación de vulnerabilidades en dominio web con la herramienta creada**

Esta herramienta escanea los contenidos web del dominio que se está analizando, para obtener rutas a ficheros o directorios sensibles. Realiza de forma automática un ataque por fuerza bruta, pero de forma transparente al usuario. Se trata de la tercera opción del menú de opciones de la auditoría a un dominio web. **Ver figura 9.**

- **Fase de explotación de vulnerabilidades en la propia máquina Linux con la herramienta creada**

Esta fase no está contemplada en este tipo de auditoría, ya que, si se está realizando la evaluación a la propia máquina que ejecuta la herramienta, el acceso a ésta ya existe.

Si se ejecuta esta opción de auditoría, lo más probable es que se desee obtener información acerca del sistema, de la red, etc. con diferentes finalidades, por ejemplo, para analizar la forma de escalar privilegios o acceder a otras máquinas o redes.

Por tanto, en la herramienta se contemplan opciones para la fase inicial y la de análisis, y también se podrá utilizar posteriormente el módulo de post-explotación, para seguir analizando el sistema de forma más exhaustiva y concreta para tal finalidad.

- **Fase de explotación de vulnerabilidades en una máquina de la red con la herramienta creada**

Las opciones del menú de opciones para este tipo de auditoría que se pueden ejecutar en esta fase son la creación de un diccionario para realizar después un ataque por fuerza bruta (opción número 7 del menú), obtener información sobre nombres de usuarios del sistema con el servicio samba (opción 8), realizar ataque de fuerza bruta contra algún servicio activo en la máquina objetivo (opción 9) y descubrir ficheros y directorios en el portal web de la máquina objetivo por fuerza bruta (opción 10). Visualizar menú de la **figura 12**.

Samba (SMB) es un sistema de archivos compartidos en red utilizado en entornos Windows.

4.3.4. Fase de post-explotación

Esta fase **solo se lleva a cabo si se ha conseguido acceder al sistema objetivo** (**ver figura 6**). Normalmente se consigue la intrusión con un usuario con permisos limitados, por lo que cuando se llega a esta etapa es cuando se intenta escalar privilegios, así como realizar pivoting para llegar a más máquinas o a otras redes.

Existen multitud de herramientas interesantes para utilizar en esta etapa. En entornos Windows las más utilizadas son WindowsEnum, Windows Exploit Suggester, Empire Framework, entre otras. Y en sistemas Linux se recomiendan LinEnum, Linux Exploit Suggester, Linuxprivchecker, BeRoot, Linux-Smart-Enumeration, etc. Además, Metasploit cuenta con numerosos exploits que intentan el escalado de privilegios de forma automatizada en ambos sistemas.

También se pueden realizar técnicas de port-forwarding, que consiste en abrir un puerto al exterior con un servicio concreto, de forma que desde fuera se pueda acceder a él.

La herramienta diseñada cuenta con información detallada compuesta por numerosos comandos útiles para esta etapa, tanto si se trata de un entono Windows como si es Linux. Esta fase es la única que no se ejecuta de forma automática, debido a que requiere de acceso al sistema comprometido, pero sí se pregunta al usuario si desea almacenar dichas recomendaciones en el fichero para consultarlo más tarde.

Pero de igual forma es de gran ayuda, por ejemplo, para el caso concreto de evaluar la escalada de privilegios en la propia máquina, ya que se pueden probar todos los comandos de la lista, para analizar los resultados y encontrar la forma de conseguirlo. También es eficaz cuando el auditor consiga el acceso a un sistema objetivo, puesto que puede probarlos de forma manual, pero ganando tiempo sin necesidad de realizar búsquedas de posibles comandos.

Al elegir este tipo de auditoría, el programa muestra un listado de ítems que se deben verificar inicialmente independientemente del sistema operativo de la máquina objetivo.

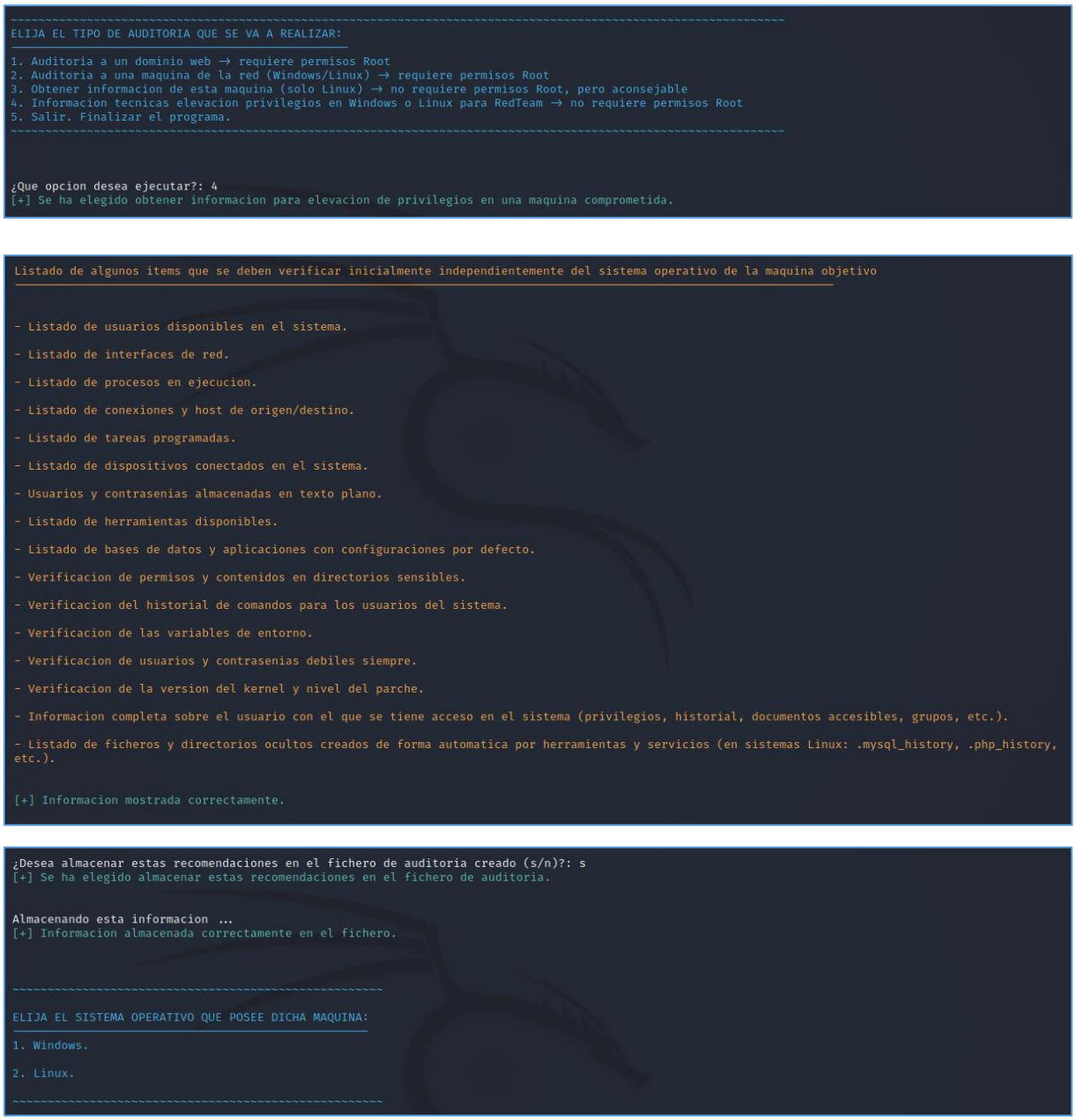
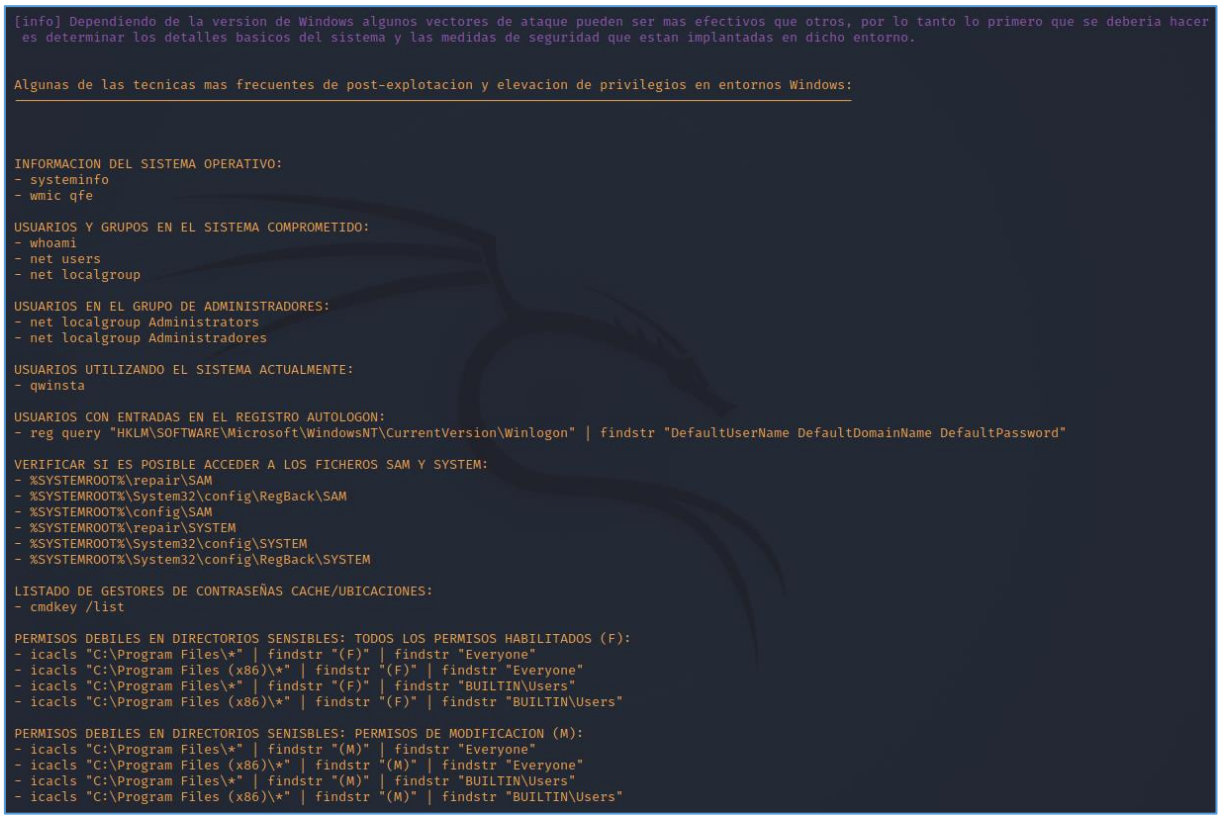


Figura 16: Información para post-explotación independientemente del sistema operativo de la máquina objetivo.

Fuente: Elaboración propia.

- **Fase de post-explotación de un sistema Windows con la herramienta creada**

Si se elige la primera opción del menú de post-explotación (**ver figura 16**), se muestra un listado de comandos útiles para ejecutar en un sistema Windows, una vez se cuenta con el acceso al sistema.



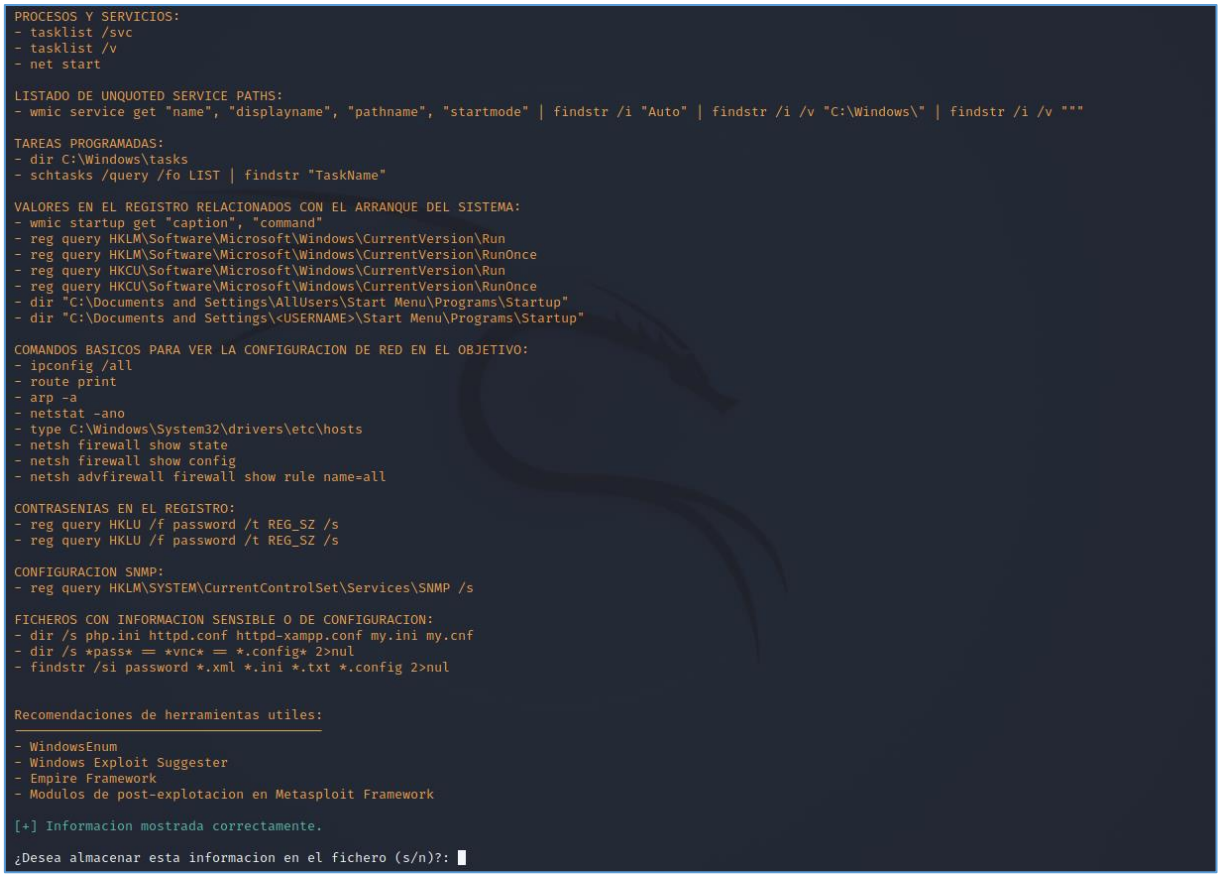
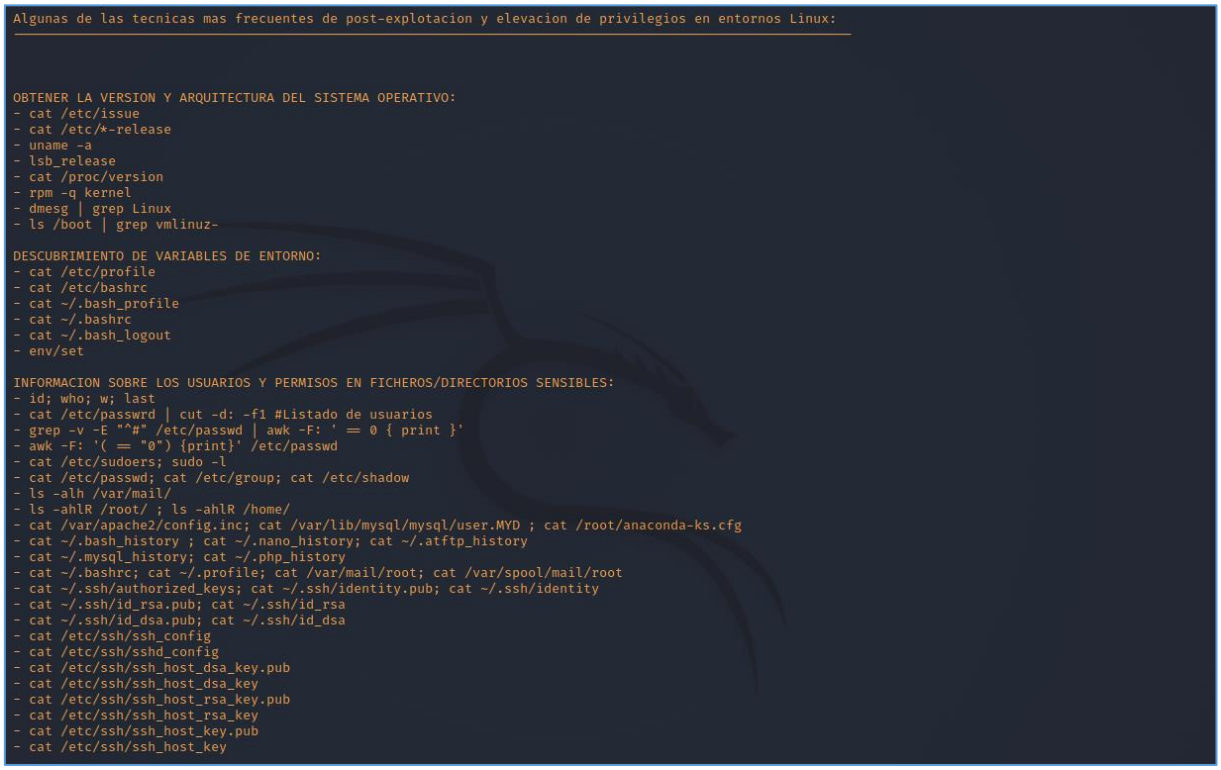


Figura 17: *Listado comandos útiles post-explotación en sistemas Windows.*

Fuente: *Elaboración propia.*

- **Fase de post-explotación de un sistema Linux con la herramienta creada**

Si se elige la segunda opción del menú de post-explotación (**ver figura 16**), se muestra un listado de comandos útiles para ejecutar en un sistema Linux, una vez se cuenta con el acceso al sistema.



```
INFORMACION QUE SE PUEDE EXTRAER DEL SISTEMA DE ARCHIVOS Y DIRECTORIOS COMUNES:
- ls -aRl /etc/ | awk ' ~/^.*w.*/' 2>/dev/null
- ls -aRl /etc/ | awk ' ~/^..w/' 2>/dev/null
- ls -aRl /etc/ | awk ' ~/^.....w/' 2>/dev/null
- ls -aRl /etc/ | awk ' ~/w.$/' 2>/dev/null
- find /etc/ -readable -type f 2>/dev/null
- find /etc/ -readable -type f -maxdepth 1 2>/dev/null
- ls -alh /var/log
- ls -alh /var/mail
- ls -alh /var/spool
- ls -alh /var/spool/lpd
- ls -alh /var/lib/pgsql
- ls -alh /var/lib/mysql
- mount
- df -h
- cat /etc/fstab

INFORMACION QUE SE PUEDE OBTENER DE LOS FICHEROS DE LOGS:
- cat /etc/httpd/logs/access_log
- cat /etc/httpd/logs/access.log
- cat /etc/httpd/logs/error_log
- cat /etc/httpd/logs/error.log
- cat /var/log/apache2/access_log
- cat /var/log/apache2/access.log
- cat /var/log/apache2/error_log
- cat /var/log/apache2/error.log
- cat /var/log/apache/access_log
- cat /var/log/apache/access.log
- cat /var/log/auth.log
- cat /var/log/chttp.log
- cat /var/log/cups/error_log
- cat /var/log/dpkg.log
- cat /var/log/faillog
- cat /var/log/httpd/access_log
- cat /var/log/httpd/access.log
- cat /var/log/httpd/error_log
- cat /var/log/httpd/error.log
- cat /var/log/lastlog
- cat /var/log/lighttpd/access.log
- cat /var/log/lighttpd/error.log
- cat /var/log/lighttpd/lighttpd.access.log
- cat /var/log/lighttpd/lighttpd.error.log
- cat /var/log/messages
- cat /var/log/secure
- cat /var/log/syslog
- cat /var/log/wtmp
- cat /var/log/xferlog
- cat /var/log/yum.log
- cat /var/run/utmp
- cat /var/webmin/miniserv.log
- cat /var/www/logs/access_log
- cat /var/www/logs/access.log
- ls -alh /var/lib/dhcp3/
```

```
- ls -alh /var/log/postgresql/
- ls -alh /var/log/proftpd/
- ls -alh /var/log/samba/

LISTADO DE SERVICIOS Y APLICACIONES EN EJECUCION:
- ps aux | ps -fea
- top
- cat /etc/services
- ps aux | grep root
- ps -ef | grep root
- ls -alh /usr/bin/
- ls -alh /sbin/
- dpkg -l
- rpm -qa
- ls -alh /var/cache/apt/archives
- ls -alh /var/cache/yum/

LISTADO DE FICHEROS DE CONFIGURACION EN UBICACIONES POR DEFECTO:
- cat /etc/syslog.conf
- cat /etc/chttp.conf
- cat /etc/lighttpd.conf
- cat /etc/cups/cupsd.conf
- cat /etc/inetd.conf
- cat /etc/apache2/apache2.conf
- cat /etc/my.conf
- cat /etc/httpd/conf/httpd.conf
- cat /opt/lampp/etc/httpd.conf
- ls -aRl /etc/ | awk ' ~/^.*r.*/

LISTADO DE TAREAS PROGRAMADAS:
- crontab -l
- ls -alh /var/spool/cron
- ls -al /etc/ | grep cron
- ls -al /etc/cron*
- cat /etc/cron*
- cat /etc/at.allow
- cat /etc/at.deny
- cat /etc/cron.allow
- cat /etc/cron.deny
- cat /etc/crontab
- cat /etc/anacrontab
- cat /var/spool/cron/crontabs/root

FICHEROS CON INFORMACION SENSIBLE, CON USUARIOS Y/O CONTRASENIA:
- grep -i user [filename]
- grep -i pass [filename]
- grep -C 5 "password" [filename]
- find . -name "*.php" -print0 | xargs -0 grep -i -n "var "
```

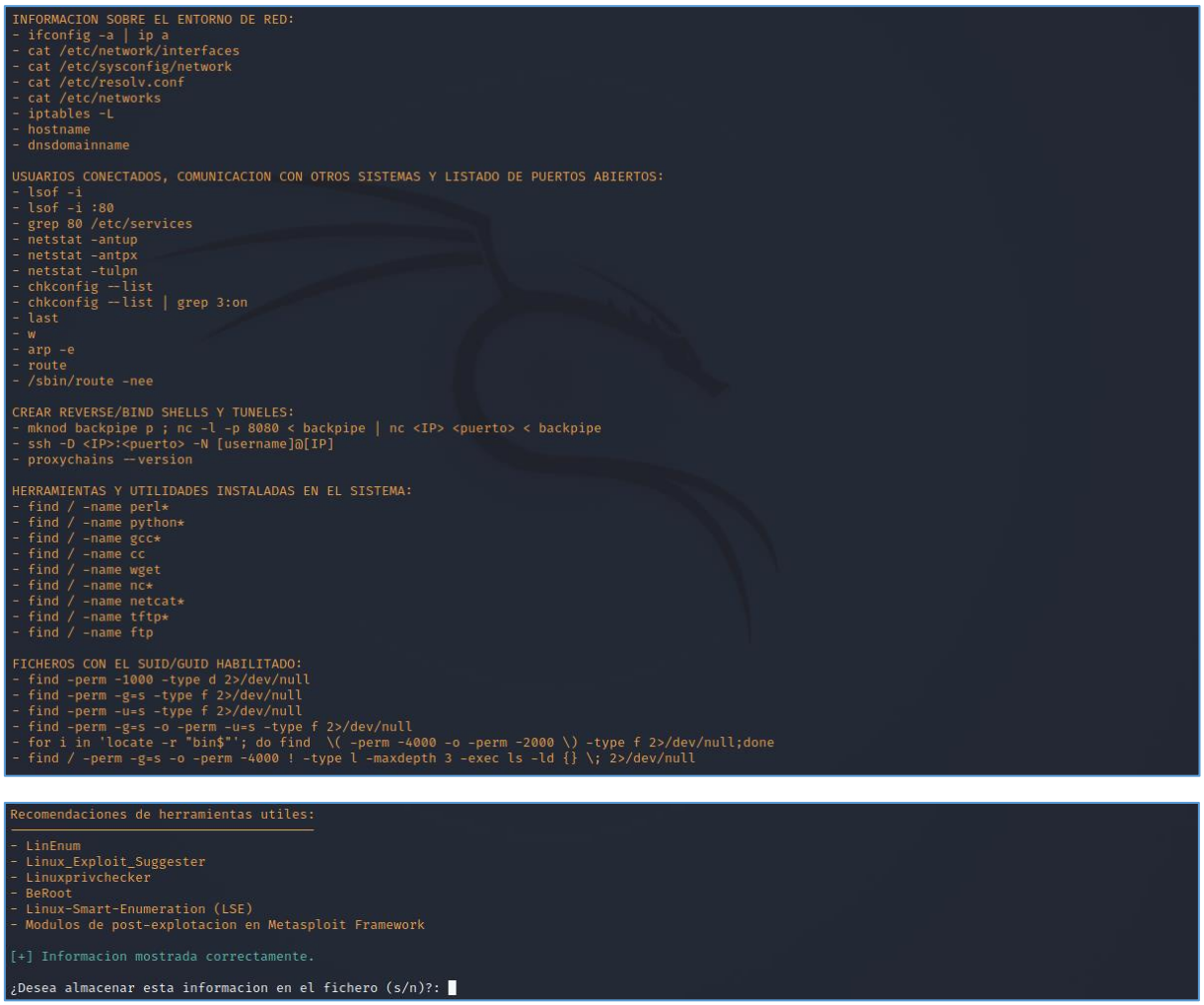


Figura 18: Listado comandos útiles post-explotación en sistemas Linux.

Fuente: Elaboración propia.

4.3.5. Generación del informe de auditoría

Por último, el auditor debe generar el informe completo de auditoría, que puede ir realizándolo de forma paralela mientras analiza y va obteniendo resultados.

Se recomienda ir completando el informe que va creando esta herramienta de forma automática, junto con las pruebas que se ejecuten de forma externa, o añadir la información que va obteniendo la herramienta a su informe, ganando mucho tiempo, que es la finalidad principal de dicho software y, por tanto, del presente trabajo.

4.4. Fragmento de código fuente de la herramienta

En este apartado se muestra un pequeño fragmento de código fuente del menú principal de la herramienta, implementado en lenguaje Bash.

Esta porción de software **se encarga de comprobar si se ha realizado alguna auditoría el día que se está ejecutando el programa**. Si es así, existen dos posibilidades que llevar a cabo, listar los nombres de dichos reportes y poder crear uno nuevo con otro nombre o escribir a continuación de uno existente, o también puede optar por finalizar la evaluación de seguridad que se iba a realizar.

Sin embargo, si no se ha realizado ninguna auditoría ese día, **se creará la estructura de ficheros correspondiente** y se pedirá al usuario el nombre que desea que tenga el reporte que se va a generar. **Todo esto de forma completamente automatizada**.

```
# Fragmento de código Bash del programa principal de la herramienta creada

#...


nombreCarpeta="auditoria";


fecha="`date +%Y%m%d`";
nombreSubcarpeta="$nombreCarpeta/$fecha";
export nombreSubcarpeta;
sleep 2;


echo -e "Comprobando si se ha realizado hoy alguna auditoria ...";
sleep 2;



# Si existe una carpeta y una subcarpeta con ese nombre, entonces ya se ha hecho alguna auditoria ese día
if [ -d "$nombreSubcarpeta" ]; then

		echo -e "\n${rojo}[!] Ya se ha realizado una auditoria hoy.${noColor}\n";
		sleep 2;


		echo -e
"\n${azul}~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";
		echo -e "ELIJA UNA OPCION A EJECUTAR:";
		echo -e "---------------------------\n";
```

```
		echo -e "1. Listar reportes de auditoria realizados hoy (visualizar los existentes y despues poder crear un reporte nuevo con otro nombre, o escribir a continuacion de uno de ellos).\n";
		echo -e "2. Finalizar la auditoria y no modificar los ficheros ya existentes.";
		echo -e "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~${noColor}\n\n";


		resp=0;
		while [ "$resp" -ne 1 ] && [ "$resp" -ne 2 ]
		do
			read -p "¿Que opcion desea ejecutar?: " resp;

			case $resp in

				1) echo -e "${verde}[+] Se ha elegido listar los reportes de auditoria realizados hoy.${noColor}\n";
				   sleep 2;

				   echo -e "\nObteniendo los reportes que se han realizado hoy ...\n";
				   sleep 2;

				   echo -e "\n\n${naranja}Reportes que se han creado hoy: \n";
				   ls $nombreSubcarpeta;
				   sleep 2;


				   echo -e "\n\n\n${azul}~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";
				   echo -e "ELIJA UNA OPCION:";
				   echo -e "-----------------------\n";
				   echo -e "1. Realizar una nueva auditoria en un reporte nuevo.\n";
				   echo -e "2. Finalizar la auditoria y salir del programa.";
				   echo -e "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~${noColor}\n\n";


				   op=0;
				   while [ "$op" -ne 1 ] && [ "$op" -ne 2 ]
				   do
				    read -p "¿Que desea hacer?: " op;

				    case $op in

					1) echo -e "${verde}[+] Se ha elegido realizar una nueva auditoria en un reporte nuevo.${noColor}\n";

					   sleep 2;
					;;

					2) echo -e "${verde}[+] Se ha elegido salir del programa.${noColor}\n";

					   exit 0;
					;;

				    esac
```

```
                                        if [ "$op" -ne 1 ] && [ "$op" -ne 2 ]; then
                                                echo -e "\n${rojo}[!] Inserte un valor valido.${noColor}\n\n";
                                        fi

                                      done
                                    ;;


                                    2) echo -e "${verde}[+] Se elige finalizar la auditoria y no modificar los
informes ya existentes.${noColor}\n";
                                      exit 0;
                                    ;;
                        esac


                        if [ "$resp" -ne 1 ] && [ "$resp" -ne 2 ]; then
                                echo -e "\n${rojo}[!] Inserte un valor valido.${noColor}\n\n";
                        fi
                done


# Si no existe esta estructura de directorios, o no existe una subcarpeta con el nombre de la fecha de este día,
entonces se crea
else

        echo -e "${verde}[+] No se ha realizado ninguna auditoria hoy.${noColor}\n";

        echo -e "\nCreando estructura de directorios en el directorio actual para almacenar el fichero de
reporte ...";
        sleep 2;

        # Se crea carpeta con nombre auditoria con todos los permisos en directorio donde está la
herramienta
        mkdir -p -m 777 $nombreCarpeta
        echo -e "${verde}[+] Se ha creado el directorio /$nombreCarpeta en la ruta donde se esta ejecutando
este script.${noColor}";
        sleep 2;

        echo -e "${verde}[+] La fecha en que se esta realizando la auditoria sera el nombre de la subcarpeta
que se va a crear. En este caso: $fecha${noColor}";
        sleep 2;

        # Se crea la subcarpeta (con nombre la fecha del día que se está realizando la auditoria) dentro de la
carpeta ya creada
        mkdir -p -m 777 $nombreSubcarpeta
        echo -e "${verde}[+] Se ha creado el directorio /$nombreSubcarpeta${noColor}";
        sleep 2;
fi



# Se pide al usuario el nombre que va a tener el reporte de auditoría que se va a crear
echo -e "\n\n";
read -p "Elija un nombre para el reporte de auditoria que se va a generar (sin extension): " nombreElegido;
extensionFichero=".txt"
nombreFichero=$nombreElegido$extensionFichero;
```

```
sleep 2;


touch $nombreSubcarpeta/$nombreFichero # Se crea fichero inicialmente vacío dentro de la subcarpeta creada
echo -e "${verde}[+] Se ha creado el fichero con nombre $nombreFichero en la ruta $nombreSubcarpeta
${noColor}";


# Se recoge en una variable la ruta del fichero que es donde se va a ir escribiendo todo de forma automatizada
rutaFichero="$nombreSubcarpeta/$nombreFichero"
export rutaFichero;
sleep 1;


echo -e "${verde}[+] Comienza la auditoria informatica... ${noColor}\n\n\n";
sleep 1;



# Se escribe la cabecera en el fichero de auditoria
echo -e "\n.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-." >> $rutaFichero;
echo "|                                                                                                                  |" >> $rutaFichero;
echo ".  INFORME DE RESULTADOS DE LA AUDITORIA INFORMATICA REALIZADA  ." >> $rutaFichero;
echo "|                                                  |" >> $rutaFichero;
echo -e ".-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.\n" >> $rutaFichero;

echo -e "[+] Auditoria realizada con HackSystemKiller.\n" >> $rutaFichero;

./scriptsAuxiliares/cabecera.sh # Se dibuja cabecera en el fichero



echo -e "   FECHA DE LA AUDITORIA:" >> $rutaFichero;
echo -e "-----------------------------" >> $rutaFichero;
echo -e "`date +%F` a las `date +%T` h \n" >> $rutaFichero; # Se añade la fecha


#...
```

5. Conclusiones y trabajo futuro

En este apartado se van a abordar las conclusiones del presente documento, así como el trabajo futuro que se puede llevar a cabo.

5.1. Conclusiones

Se puede apreciar la **gran dificultad de realizar correctamente pruebas de intrusión y de evaluación de seguridad de un sistema o entorno**, ya que hay numerosas maneras diferentes de testearlo e infinidad de vulnerabilidades posibles, así como fallos de configuración, etc.

Además, la seguridad de un sistema no solo depende de las medidas perimetrales, las actualizaciones de nuevas versiones en los servicios, las evaluaciones periódicas de seguridad, etc., sino que el error humano es el eslabón más débil de la cadena. Aunque las protecciones y monitoreos sean exhaustivos, si el usuario que utiliza el sistema realiza cualquier acción de forma incorrecta, como por ejemplo, ejecutar un programa malicioso recibido por correo electrónico (ejemplo de Phising), puede ocasionar consecuencias críticas si dicho software está ofuscado (herramienta más importante para evitar la ingeniería inversa, consiste en reordenar o alterar las instrucciones de un programa para que, aunque realice la misma función, sea más difícil su comprensión) de forma que ni el antivirus ni los sistemas de detección (IDS) o prevención (IPS) de intrusos lo detecten.

La única solución para esto es concienciar a los usuarios que utilizan el sistema y conseguir que se respeten en todo momento las normas de actuación marcadas por la empresa, así como auditar que los permisos que posee cada usuario con su rol en el sistema sean los correctos y solo los necesarios para poder realizar su tarea, nunca más de esto, de forma que se protejan lo máximo posible los ficheros sensibles, las instalaciones de herramientas, etc.

En general, aunque más en este aspecto en particular, **se puede incrementar la seguridad, pero asegurarla por completo, es prácticamente imposible**. Ya que por ejemplo un DDoS, es muy difícil de parar, ya que el ataque proviene de múltiples IP's diferentes y detectarlo es complejo, así como encontrar una manera de impedirlo o frenarlo cuando se perciba.

Por tanto, se puede concluir que **el tiempo total de auditoría y la detección de vulnerabilidades**, malas configuraciones, incorrectas asignaciones de permisos, etc. de un sistema o entorno, **es de suma importancia para detectarlo y corregirlo lo antes posible y así impedir ataques que conlleven, o puedan conllevar, consecuencias nefastas**.

De esta forma, no solo se comprueba el propio sistema que se está auditando, sino que también se validan si las medidas perimetrales están configuradas de forma debida.

El uso de la herramienta creada que automatiza una auditoría informática, así como seguir una correcta metodología, contribuye notablemente a dicho análisis con menor tiempo y trabajo, así como por fiabilidad, para después poder examinar y proponer las mejores medidas de protección o corrección que implantar, por lo que **es de gran utilidad y cumple con la finalidad que se buscaba en el presente trabajo**.

5.2. Trabajo futuro

La herramienta diseñada se puede ampliar implementando más opciones y, por tanto, **añadiendo más capacidades a cada tipo de auditoría, o incluso añadiendo funcionalidades nuevas, haciéndola aún más completa en cuanto a alcance**, y por supuesto debe estar en continua actualización para incorporar todo lo novedoso que se vaya descubriendo o conociendo en dicho campo.

Por ejemplo, se puede crear un módulo que se encargue de realizar pruebas de estrés al objetivo, para analizar si es capaz de provocar una denegación de servicio (ataques DoS o DDoS) y examinar hasta qué niveles aguanta el servicio sin afectar en su disponibilidad.

También se puede contemplar la posibilidad de incorporar un front-end (interfaz de usuario) más interactivo aún, por ejemplo, incorporando botones o campos de texto, con ayuda de un lenguaje de programación de más alto nivel, como puede ser por ejemplo Python, Java, C, etc. De forma que el back-end (lo que no ve el usuario) siga realizándose en Bash, para poder reutilizar el código fuente diseñado hasta ahora y mejorarlo.

Por otro lado, otra cuestión de trabajo futuro es cómo darla a conocer a los posibles usuarios, ya sea por dominio libre o a través de otras posibles fórmulas, comercializarla...

Todo esto no se ha llevado a cabo por falta de tiempo, además de que nunca se podrá disponer de una versión definitiva totalmente, ya que cada día se descubren nuevas vulnerabilidades y la herramienta debe estar en continua actualización y ampliación.

Referencias bibliográficas

Añover, A. (2021). Cómo Estonia se convirtió en el país experto en ciberseguridad. *La Razón*. https://www.larazon.es/internacional/20210618/2nhwdnpsbneb3dr7wxs3w5aexm.html

Astudillo, K. (2018). Hácking ético 101 (2ª ed.). Createspace Independent.

Computing, R. (2022). 22 estadísticas de ciberseguridad que hay que conocer para 2022. BPS. https://www.computing.es/seguridad/informes/1130465002501/22-estadisticas-de-ciberseguridad-hay-conocer-2022.1.html

López, JG. Simón, MAC. y Núñez, PG. (2014). Hackers. Aprende a atacar y a defenderte (2ª ed. acualizada). Ra-Ma.

Merino, B. y González, P. (2018). Hacking con Metasploit Advanced Pentesting (1ª ed.). 0xword.

Montoya, DE. (2020). 25 técnicas aplicadas a campañas de Red Team y Hacking (1ª ed.). Independently Published.

Navarro, M. (2020). La ciberseguridad cobra importancia en tiempos de Covid-19. Revistabyte. https://revistabyte.es/ciberseguridad/ciberseguridad-covid-19/

Orcero, DS. (2019). Pentesting con Kali (9ª ed.). Blurb.

Pacheco, J. (2022). Se puede tener el mejor sistema de seguridad, pero un error puede ponerlo todo en riesgo. Hispasec. https://unaaldia.hispasec.com/2022/03/jesus-pacheco-product-manager-se-puede-tener-el-mejor-sistema-de-seguridad-pero-un-error-puede-ponerlo-todo-en-riesgo.html

Pérez, PG. (2020). Metasploit para Pentesters (5ª ed.). 0xword.

Sánchez, LJ. (2021). El 86% de las empresas españolas carecen de cultura de ciberseguridad. Confilegal. https://confilegal.com/20210129-el-86-de-las-empresas-espanolas-carecen-de-cultura-de-ciberseguridad/

Valadés, B. (2021). EEUU, Canadá y Brasil, primeros países americanos en el Índice de Ciberseguridad Global. Segurilatam.

https://www.segurilatam.com/actualidad/ciberseguridad-eeuu-canada-y-brasil-primeros-paises-americanos-en-el-indice-de-ciberseguridad-global_20210702.html

Índice de acrónimos

A

ARP: Address Resolution Protocol (Protocolo de resolución de direcciones).

D

DDoS: Distributed Denial of Service (Denegación de Servicio Distribuida).

DoS: Denial of Service (Denegación del Sistema).

F

FTP: File Transfer Protocol (Protocolo de Transferencia de Ficheros).

H

HTML: HyperText Markup Language (Lenguaje de Marcado de HiperTexto).

HTTP: HyperText Transfer Protocol (Protocolo de Transferencia de Hipertexto).

HTTPS: Hyper Text Transfer Protocol Secure (Protocolo Seguro de Transferencia de Hipertexto).

I

IDS: Intrusion Detection System (Sistema de Detección de Intrusos).

IP: Internet Protocol (Protocolo de Internet).

IPS: Intrusion Prevention System (Sistema de Prevención de Intrusos).

J

JS: JavaScript.

L

LAN: Local Area Network (Red de Área Local).

M

MAC: Media Access Control (Control de Acceso a Medios).

MAN: Metropolitan Area Network (Red de Área Metropolitana).

MiTM: Man-In-The-Middle Attack (Ataque de hombre en el medio).

O

OSINT: Open Source INTelligence (Inteligencia de fuentes abiertas).

S

SSH: Secure Shell (Cápsula segura).

T

TCP: Transmission Control Protocol (Protocolo de Control de Transmisión).

TOR: The Onion Router (El router cebolla).

U

UDP: User Datagram Protocol (Protocolo de Datagramas de Usuario).

USB: Universal Serial Bus (Bus Universal en Serie).

V

VPN: Virtual Private Network (Red Privada Virtual).

W

WAN: Wide Area Network (Red de Área Amplia).