UNIVERSITAT DE BARCELONA

Trabajo de fin de grado

GRADO EN INGENIERÍA INFORMÁTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

PentesTool

Autor: Rubén Pellicero Álvarez

Director: Dr. Raül Roca Cànovas
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 13 de enero de 2024

Resumen

El propósito fundamental de este proyecto de Trabajo de Fin de Grado (TFG) se centra en el desarrollo de una plataforma web destinada a la automatización integral de la generación de informes detallados concernientes a la seguridad asociada a direcciones IP o URL específicas. La esencia principal de esta plataforma radica en la compilación exhaustiva de información relevante que abarcará no solo los datos vinculados a estas direcciones, sino también identificará y analizará potenciales vulnerabilidades inherentes a las mismas.

Este sistema se concibe como una herramienta de gran utilidad, permitiendo a los usuarios obtener informes detallados que ofrecerán una visión comprehensiva de la seguridad asociada a direcciones IP o URL específicas. Esto incluirá un análisis minucioso de los datos disponibles y una identificación precisa de las posibles amenazas o debilidades de seguridad, proporcionando así una visión holística del estado de seguridad de la dirección consultada.

La funcionalidad principal de la plataforma se basará en la ejecución de scripts diseñados para recopilar información relevante de manera automatizada. Estos scripts serán configurables y adaptables para satisfacer las necesidades específicas de los usuarios. Posteriormente, la información recopilada será procesada y presentada en un informe detallado y comprensible que ofrecerá una visión integral de la seguridad relacionada con la dirección consultada.

En resumen, este proyecto aspira a ofrecer una solución completa y eficiente para la evaluación de la seguridad de direcciones IP o URL específicas a través de una plataforma web intuitiva y automatizada, brindando informes detallados y precisos que contribuyan significativamente al análisis y mejora de la seguridad informática.

Resum

El propòsit fonamental d'aquest projecte de Treball de Fi de Grau (TFG) es basa en el desenvolupament d'una plataforma web destinada a l'automatització integral de la generació d'informes detallats concernents a la seguretat associada a adreces IP o URL especifiques. La principal essència d'aquesta plataforma radica en la compilació exhaustiva d'informació rellevant que abasta no sols les dades vinculades a aquestes direccions, sinó també a identificar i analitzar potencials vulnerabilitats inherents en aquestes.

Aquest sistema es concep com una eina de gran utilitat, permetent als usuaris obtenir informes detallats que oferiran una visió comprehensiva de la seguretat associada a adreces IP o URL específiques. Això inclourà un minucios anàlisi de les dades disponibles i una identificació precisa de les possibles amenaces o febleses de seguretat, proporcionant així una visio holística de l'estat de seguretat de la direcció consultada.

La funcionalitat principal de la plataforma es basarà en l'execució de scripts dissenyats per a recopilar informació rellevant de manera automatitzada. Aquests scripts seran configurables i adaptables per a satisfer les necessitats específiques dels usuaris. Posteriorment, la informació recopilada serà processada i presentada en un informe detallat i comprensible que oferirà una visió integral de la seguretat relacionada amb la direcció consultada.

En resum, aquest projecte aspira a oferir una solució completa i eficient per a l'avaluació de la seguretat d'adreces IP o URL específiques a través d'una plataforma web intuïtiva i automatitzada, brindant informes detallats i precisos que contribueixin significativament a l'anàlisi i millora de la seguretat informàtica.

Abstract

The fundamental purpose of this Bachelor's thesis project focuses on the development of a web platform aimed at the comprehensive automation of the generation of detailed reports concerning the security associated with specific IP addresses or URLs. The main essence of this platform lies in the exhaustive compilation of relevant information that covers not only the data linked to these addresses, but also to identify and analyze potential inherent vulnerabilities. in the same.

This system is conceived as a very useful tool, allowing users to obtain detailed reports that will offer a comprehensive view of the security associated with specific IP addresses or URLs. This will include a thorough analysis of the available data and a precise identification of possible security threats or weaknesses, thus providing a holistic view of the security status of the consulted address.

The main functionality of the platform will be based on the execution of scripts designed to collect relevant information in an automated manner. These scripts will be configurable and adaptable to meet specific user needs. Subsequently, the information collected will be processed and presented in a detailed and understandable report that will offer a comprehensive view of the security related to the consulted address.

In summary, this project aims to offer a complete and efficient solution for the security evaluation of specific IP addresses or URLs through an intuitive and automated web platform, providing detailed and accurate reports that contribute significantly to the analysis and improvement of security. computing.

Agradecimientos

Quiero expresar mi sincero agradecimiento a todas las personas que han contribuido a la realización de mi Trabajo de Fin de Grado. Este proyecto no habría sido posible sin el apoyo incondicional de mi familia, amigos y profesores.

Agradezco a mi familia por su paciencia, comprensión y ánimos durante este período de intensa dedicación. Su apoyo ha sido mi fuente de inspiración y motivación.

Agradezco a mis amigos por estar a mi lado, brindándome su amistad y desconexión en momentos de estrés.

Quiero expresar mi gratitud al Dr.Raül Roca por su orientación y asesoramiento. Su conocimiento y experiencia han sido cruciales para el desarrollo y éxito de mi proyecto.

Además, quiero reconocer la importancia de la empresa en la que trabajo en el campo de la ciberseguridad. La experiencia adquirida ha enriquecido mi perspectiva y ha sido fundamental para la elaboración de mi TFG.

Finalmente, agradezco a todos aquellos que han contribuido directa o indirectamente a mi proyecto. Vuestra colaboración ha sido esencial en la consecución de este logro académico.

Índice

Índice de figuras

Índice de cuadros

1. Introducción

Este proyecto se centrará en el desarrollo de PentesTool, una página web muy útil para auditores de seguridad informática, la cual permite tener controladas diferentes direcciones, obteniendo informes detallados, donde encontraremos información como el sistema operativo, puertos abiertos con la versión de sus servicios, geolocalizaión entre otras muchas cosas.

1.1. Motivación

En la era digital actual, la integridad, confidencialidad y disponibilidad se han convertido en grandes preocupaciones para todos nosotros. El incremento constante de amenazas cibernéticas, ataques maliciosos y la exposición a vulnerabilidades en la red hacen que la protección de datos sea crucial para individuos y organizaciones.

La identificación y evaluación de las vulnerabilidades en direcciones IP y URL son fundamentales para mantener la seguridad en línea. La exposición a riesgos como intrusiones no autorizadas, fugas de datos sensibles o la posibilidad de ser objeto de ciberataques está en constante aumento, exigiendo soluciones efectivas que fortalezcan la seguridad digital.

En este contexto, el análisis exhaustivo de direcciones IP y URL se convierte en una piedra angular para mitigar riesgos. Sin embargo, las herramientas existentes a menudo dependen de métodos centralizados y pueden no ser lo suficientemente exhaustivas o ágiles para identificar y abordar las vulnerabilidades de manera completa y eficiente.

Además, las actualizaciones y nuevas versiones representan un aspecto crucial. Mantener todos los servicios actualizados a la versión estable más reciente posible se vuelve fundamental en el ámbito de la seguridad cibernética. Sin embargo, la recomendación no suele ser adoptar la última versión inmediatamente, ya que esta podría contener fallos no contemplados o aún no identificados.

Las actualizaciones periódicas no solo introducen nuevas funcionalidades, sino que también corrigen vulnerabilidades conocidas y fortalecen la seguridad de las plataformas digitales. Esta práctica es esencial para mitigar posibles riesgos derivados de vulnerabilidades previamente descubiertas y parcheadas por los desarrolladores.

La capacidad de realizar análisis de seguridad no solo se centra en detectar vulnerabilidades existentes, sino también en promover una cultura proactiva de mantenimiento y actualización. La implementación de esta funcionalidad permitirá a los usuarios mantener sus servicios digitales en un estado óptimo de seguridad, minimizando riesgos potenciales y asegurando un entorno en línea más protegido y resistente frente a amenazas emergentes.

Este enfoque hacia la gestión inteligente de actualizaciones no solo mejora la seguridad general, sino que también garantiza una experiencia más confiable y estable para los usuarios, respaldando así la integridad y la protección de sus activos digitales.

Mi proyecto de TFG tiene como objetivo desarrollar una plataforma innovadora que permita la generación de informes de seguridad detallados a partir de direcciones IP o URL. Al emplear scripts y recursos específicos ejecutados desde una máquina especializada, mi propuesta busca proporcionar un análisis minucioso y automatizado de la seguridad de estos puntos de acceso digitales.

En resumen, este proyecto aborda la necesidad de herramientas avanzadas y eficaces para evaluar riesgos de seguridad en línea. Al ofrecer una solución automatizada y detallada para la identificación de vulnerabilidades en direcciones IP y URL, se pretende facilitar a usuarios individuales y entidades comerciales una forma ágil y confiable de proteger sus activos digitales.

El desarrollo de esta plataforma enfrentará desafíos técnicos y de seguridad significativos, incluyendo la precisión del análisis, la gestión de grandes volúmenes de datos y la identificación efectiva de las amenazas potenciales. Además, será fundamental garantizar que la plataforma sea ágil y adaptable a los cambios constantes en el panorama de la ciberseguridad.

1.2. Objetivos

Para este proyecto, se han definido una serie de objetivos fundamentales que apuntan a abordar los desafíos específicos asociados con la evaluación de seguridad de direcciones IP y URLs, así como la generación de informes detallados sobre estas vulnerabilidades. A pesar de que en el transcurso del desarrollo del proyecto se han identificado objetivos adicionales, se ha priorizado el enfoque en los siguientes:

- Investigación y evaluación de herramientas de seguridad: El objetivo principal de este proyecto es realizar una investigación exhaustiva y evaluar diversas herramientas de seguridad disponibles para la identificación de vulnerabilidades en direcciones IP y URLs. Esta fase comprenderá la identificación de herramientas reconocidas y la evaluación de sus capacidades para detectar amenazas potenciales, como intrusiones no autorizadas y vulnerabilidades explotables en ciberataques. Se establecerán criterios específicos para la selección de las herramientas que se integrarán en el proceso de análisis de seguridad.

- Creación y Desarrollo de Scripts Especializados: El siguiente objetivo será el desarrollo de scripts especializados adaptados a las herramientas seleccionadas durante la fase de investigación. Estos scripts se diseñarán para realizar tareas específicas, como el escaneo y la evaluación de las direcciones IP y URLs. Además, se programarán para recopilar datos relevantes de seguridad de manera eficiente y precisa.

- Procesamiento y filtrado de datos obtenidos: Una vez recopilados los datos mediante los scripts, se procederá al procesamiento y filtrado de la información obtenida. Este paso implicará la identificación y recopilación de únicamente la información relevante y crítica para la generación del informe de seguridad, descartando datos superfluos y centrándose en los aspectos esenciales de las vulnerabilidades identificadas.

- Diseño e implementación del informe de seguridad: El desarrollo de un informe detallado y comprensible será un objetivo clave. Se diseñará e implementará un informe estructurado que proporcione una visión completa de las vulnerabilidades descubiertas en las direcciones IP y URLs evaluadas. Este informe contendrá detalles sobre las amenazas identificadas, niveles de riesgo asociados, así como recomendaciones para su mitigación y resolución.

- Desarrollo de una plataforma web intuitiva: Se creará una plataforma web de fácil acceso que permita a los usuarios ejecutar los análisis de seguridad y visualizar los informes generados. La interfaz de usuario será amigable, ofreciendo funcionalidades claras y accesibles para facilitar la interacción y comprensión de los informes de seguridad.

- Evaluación rigurosa de la efectividad y utilidad: Se llevará a cabo una evaluación detallada y rigurosa para determinar la efectividad y utilidad de la plataforma desarrollada. Esta evaluación comprenderá pruebas exhaustivas de la funcionalidad, precisión, facilidad de uso y capacidad para proporcionar

información crítica sobre las vulnerabilidades en las direcciones IP y URLs analizadas.

Estos objetivos establecen una base sólida para el desarrollo de la página, abordando la investigación, el diseño, la implementación y la evaluación de su efectividad para usuarios individuales y organizaciones.

1.3. Planificación

Este proyecto de TFG comprende una serie de objetivos fundamentales que requieren un margen de tiempo adecuado para afrontar posibles contratiempos imprevistos. He desarrollado un plan detallado de actividades, reflejado en la Figura 1, donde se detallan las diferentes tareas completadas a lo largo de mi investigación. Es crucial destacar que tanto el diseño como la implementación han demandado más tiempo del inicialmente previsto debido a cambios en la definición de objetivos y alcances. Específicamente, se produjeron ajustes en los objetivos, como la creación de la página web para visualizar los informes. Estos ajustes han contribuido a mejorar el estado final del proyecto.

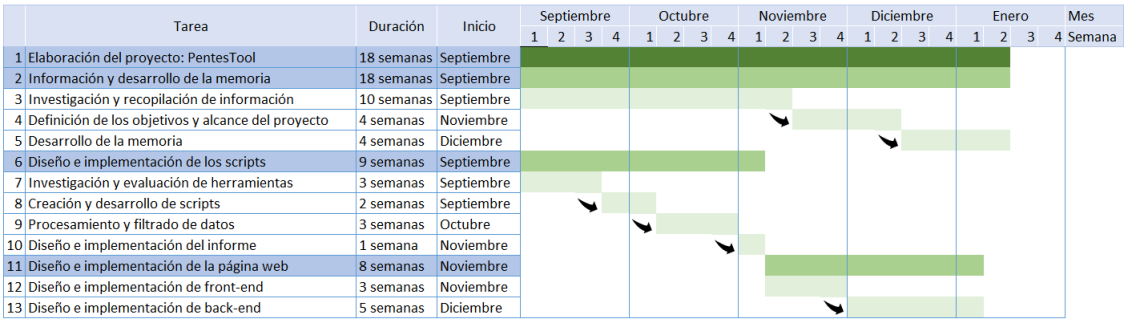

Figura 1: Diagrama de Gantt sobre la planificación de PentesTool

La planificación se ha estructurado en tres secciones principales, cada una abordando aspectos específicos del proyecto. La primera sección se enfoca principalmente en la recopilación de información y el desarrollo de la memoria, identificada como Tarea 2 en nuestro plan. La segunda sección (Tarea 6) se centra en el diseño e implementación de los scripts. Por último como tercera sección (Tarea 11) tenemos el diseño e implementación de la página web. Al dividir el proceso de esta manera, podre prestar la atención necesaria a cada etapa del proyecto, asegurando una gestión efectiva y una ejecución exitosa.

Tarea 1. Elaboración del proyecto PentesTool: Esta es la tarea principal la cual la desglosamos en tres secciones.

Tarea 2. Información y desarrollo de la memoria: La primera sección consta de tres tareas destinadas a configurar una memoria acorde con el proyecto.

Tarea 3. Investigación y recopilación de información: La tarea de investigación y recopilación de información para la memoria del TFG se centra en analizar exhaustivamente las vulnerabilidades de direcciones IP y URL. Esta labor implica revisar fuentes especializadas, estudios relevantes y metodologías existentes sobre ciberseguridad. El objetivo es reunir datos fundamentales que respalden el desarrollo de una plataforma innovadora para generar informes de seguridad detallados, empleando scripts y recursos específicos para un análisis automatizado. Se busca garantizar la precisión, actualización constante y adaptabilidad de la información recopilada para enfrentar desafíos técnicos y de seguridad.

Tarea 4. Definición de objetivos y alcance del proyecto: Es crucial definir objetivos realistas que podamos lograr dentro del alcance disponible. Analizaremos y evaluaremos meticulosamente las herramientas a nuestra disposición para seleccionar las más adecuadas a nuestras necesidades.

Tarea 5. Desarrollo de la memoria: La creación de una memoria exhaustiva que documente el trabajo realizado será esencial. Esta memoria detallará las decisiones, diseños, implementaciones y análisis efectuados, así como los resultados obtenidos.

Tarea 6. Diseño e implementación de los scripts: La segunda sección consta de cuatro tareas destinadas a la creción de los scripts.

Tarea 7. Investigación y evaluación de herramientas: Se enfoca en identificar y analizar diversas soluciones existentes para la detección de vulnerabilidades en direcciones IP y URL. Esto implica revisar herramientas de análisis de seguridad disponibles en el mercado, evaluar sus capacidades, ventajas y limitaciones, así como comparar diferentes metodologías y enfoques utilizados en estas soluciones. El objetivo es seleccionar las herramientas más adecuadas que puedan servir como referencia o complemento para el desarrollo de la plataforma, considerando su precisión, capacidad de análisis, actualizaciones y adaptabilidad a entornos cambiantes de ciberseguridad.

Tarea 8. Creación y desarrollo de scripts: Se concentra en la escritura y programación de códigos informáticos especializados que permitirán automatizar el análisis de seguridad de direcciones IP y URL. Estos scripts se diseñan para llevar a cabo tareas específicas, como escanear y recopilar datos para posteriormente hacer un análisis de vulnerabilidades. El enfoque radica en la implementación de algoritmos eficientes y precisos que mejoren la capacidad de detección de amenazas.

Tarea 9. Procesamiento y filtrado de datos: El procesamiento y filtrado de datos implica trabajar con la información recopilada durante el análisis de vulnerabilidades en direcciones IP y URL. Este proceso se centra en la limpieza, organización y estructuración de los datos para eliminar ruido o información redundante, así como para asegurar la precisión y relevancia de los datos recopilados. Además, implica la aplicación de filtros y técnicas de procesamiento para identificar patrones, clasificar la información y extraer los elementos clave necesarios para generar informes de seguridad detallados y útiles en la identificación de amenazas cibernéticas.

Tarea 10. Diseño e implementación del informe: Se enfoca en juntar todos los datos útiles en un documento. Además, se lleva a cabo la implementación de un script que genere automáticamente estos informes a partir de los datos recopilados y procesados, asegurando su coherencia, precisión y utilidad para los usuarios.

Tarea 11. Diseño e implementación de la página web: tercera y última sección consta de dos tareas destinadas a desarrollar la página web.

Tarea 12. Diseño e implementación de front-end: Creación de la interfaz de usuario de la plataforma de seguridad. Esto implica el desarrollo de la parte visual y funcional con la que los usuarios interactúan. Incluye la definición de la apariencia, disposición y navegación de la aplicación, asegurando una experiencia de usuario intuitiva y amigable. La implementación del front-end implica utilizar tecnologías como HTML, CSS, JavaScript y otros frameworks para construir la interfaz que permita a los usuarios acceder, ingresar datos y visualizar los informes de seguridad generados por la plataforma, facilitando así su comprensión y uso efectivo.

Tarea 13. Diseño e implementación de back-end: El diseño e implementación del back-end engloba la parte lógica y funcional que respalda la plataforma de seguridad. Incluye la creación de la estructura de la base de datos para almacenar y gestionar la información recopilada, así como el desarrollo del servidor que ejecuta los scripts de análisis de seguridad. Además, implica la configuración y despliegue de la aplicación en la nube, asegurando su disponibilidad y accesibilidad. Esta fase une todos los componentes del sistema, garantizando la integración fluida entre el servidor, la base de datos y la funcionalidad de la aplicación para ofrecer una solución completa y operativa para la detección de vulnerabilidades en direcciones IP y URL.

Este plan estructurado garantiza una gestión efectiva y una ejecución rigurosa de las tareas, lo que permitirá un desarrollo exitoso de nuestro proyecto de TFG.

1.4. Estado del arte

En la actualidad, la seguridad cibernética se ha convertido en un tema crucial debido al constante aumento de amenazas en línea. La detección de vulnerabilidades en direcciones IP y URL es esencial para proteger los activos digitales contra intrusiones no autorizadas y ataques malintencionados.

Herramientas y metodologías existentes, como los escáneres de vulnerabilidades, han sido fundamentales para identificar debilidades en sistemas informáticos. Estas soluciones utilizan técnicas de escaneo exhaustivo para encontrar posibles brechas de seguridad y evaluar la exposición a riesgos.

Además, se ha observado un crecimiento significativo en el desarrollo de plataformas basadas en la nube para la gestión y análisis de seguridad. Estas soluciones ofrecen escalabilidad, accesibilidad y flexibilidad, permitiendo a los usuarios realizar evaluaciones de seguridad de manera remota y eficaz.

No obstante, existen desafíos persistentes, como la necesidad de actualizaciones constantes para abordar nuevas vulnerabilidades y la adaptabilidad a entornos en constante evolución en el panorama de la ciberseguridad.

Herramientas como Shodan[1] y censys[2] entre otras, han sido fundamentales para la identificación de vulnerabilidades en sistemas informáticos. Sin embargo, su uso está acompañado por limitaciones notables. Shodan, es la mas conocida, y a veces pasa por alto ciertas páginas web, proporciona información incompleta y carece de claridad al indicar el estado de seguridad de una dirección IP o URL analizada.

En este contexto, el desarrollo de una plataforma innovadora para la generación de informes detallados a partir de direcciones IP y URL representa un avance significativo. Esta herramienta se propone como una solución automatizada y exhaustiva para identificar y evaluar vulnerabilidades, contribuyendo así a fortalecer la seguridad digital de individuos y organizaciones. Además de poder tener guardados estos informes en la propia plataforma.

En conclusión, la evolución constante del panorama de la ciberseguridad ha destacado la importancia de herramientas de detección de vulnerabilidades como Shodan y Censys, aunque con limitaciones inherentes. El desarrollo de una plataforma innovadora y automatizada para generar informes detallados a partir de direcciones IP y URL representa un paso adelante en la protección de activos digitales. Esta solución integral, al abordar la identificación y evaluación exhaustiva de riesgos, promete fortalecer la seguridad en línea, ofreciendo una visión más clara y acciones preventivas para individuos y organizaciones en un entorno digital cada vez más complejo y dinámico.

2. Tecnologías

En la sección de tecnologías, nuestro objetivo es brindar una introducción a cada una de las tecnologías que vamos a utilizar en el proyecto. Esta categoría servirá como punto de partida para comprender y contextualizar cada tecnología, lo que facilitará las explicaciones posteriores.

2.1. Herramientas de seguridad informática

Nos enfocaremos en herramientas que pueden ser ejecutadas mediante líneas de comandos en sistemas operativos, lo que permite interactuar con ellas a través de la terminal o línea de comandos. Estas herramientas, se ejecautan directamente mediante instrucciones específicas que se ingresan en la línea de comandos del sistema operativo.

2.1.1. Recopilación de información y enumeración

Estas herramientas son esenciales en las fases iniciales de un análisis de seguridad, ya que su objetivo principal es recopilar una amplia gama de datos sobre la organización, red, o sistema objetivo. Su propósito fundamental es identificar posibles debilidades, vulnerabilidades, y áreas susceptibles de ser aprovechadas por potenciales atacantes, proporcionando así información valiosa para fortalecer y mejorar la seguridad.

- Amass:
  - Función: Amass es una herramienta de código abierto que se utiliza para la recopilación de información sobre la infraestructura de red. Permite la búsqueda de activos en línea, descubrimiento de subdominios y otras entidades asociadas con un dominio específico.
  - Características clave: Utiliza diversas fuentes de datos, como motores de búsqueda, certificados SSL/TLS, bases de datos públicas y colaborativas para recopilar información sobre la infraestructura de red de una organización.
- Dig:
  - Función: Dig (Domain Information Groper) es una utilidad de línea de comandos que realiza consultas al sistema de nombres de dominio (DNS). Es utilizada para buscar y obtener información detallada sobre registros de DNS, resolución de nombres y diagnósticos relacionados con los dominios.
  - Utilidad: Es una herramienta fundamental para verificar la configuración del sistema de nombres de dominio y diagnosticar problemas de resolución de nombres en servidores DNS.

- Nslookup:
  - Función: Similar a Dig, Nslookup es otra herramienta de línea de comandos que permite consultar información del sistema de nombres de dominio (DNS). Proporciona datos sobre la resolución de nombres, diagnósticos y otros detalles útiles sobre los registros DNS de un dominio.
  - Utilidad: Permite a los administradores de sistemas y profesionales de redes verificar la configuración de DNS y realizar diagnósticos relacionados con la resolución de nombres.
- Whois:
  - Función: Whois es una utilidad que proporciona información detallada sobre un dominio en particular, como el propietario registrado, datos de contacto, registrador, fecha de creación y vencimiento, servidores de nombres, entre otros.
  - Utilidad: Es útil para obtener información básica sobre la propiedad de un dominio y para realizar investigaciones en la identificación de propietarios de dominios.
- Nmap:
  - Función: Nmap es una herramienta de escaneo de red que permite descubrir dispositivos, puertos abiertos y servicios en una red. Es ampliamente utilizada para la evaluación de seguridad, descubrimiento de vulnerabilidades y mapeo de la red.
  - Utilidad: Ayuda a los profesionales de seguridad cibernética a identificar activos de red, servicios expuestos y posibles puntos débiles en la seguridad de la red.
- Wappalyzer (Wappy) [3]:
  - Función: Wappalyzer es una extensión de navegador que identifica tecnologías utilizadas en sitios web, como el software del servidor, frameworks, bibliotecas JavaScript, entre otros.
  - Utilidad: Es útil para los desarrolladores web, investigadores de seguridad y profesionales de marketing para identificar las tecnologías subyacentes utilizadas en un sitio web, lo que puede ser útil para analizar la estructura tecnológica, identificar posibles vulnerabilidades y comprender mejor la arquitectura de un sitio.
- ipinfo:
  - Función: ipinfo.io es una plataforma que proporciona información detallada sobre direcciones IP. Ofrece datos como ubicación geográfica, proveedor de servicios de Internet (ISP), tipo de conexión, código postal, coordenadas geográficas y detalles del dominio.

    - Utilidad: Es útil para desarrolladores, administradores de red, investigadores de seguridad y profesionales de marketing que necesitan información específica sobre direcciones IP. Permite comprender la geolocalización, la infraestructura de red utilizada y ayuda en la identificación de posibles amenazas o uso indebido de direcciones IP. Además, puede ser útil en el análisis de tráfico, en la configuración de sistemas de seguridad y en el seguimiento de visitantes en sitios web.

- hunter [4]:

    - Función: Hunter.io es una plataforma que permite buscar direcciones de correo electrónico asociadas a un dominio específico. Proporciona información detallada sobre correos electrónicos encontrados, como el formato de la dirección (ejemplo@dominio.com), la probabilidad de que sea válida, y en algunos casos, el nombre del propietario asociado a esa dirección.
    - Utilidad: Es útil para profesionales de marketing, reclutadores, equipos de ventas y profesionales de relaciones públicas que buscan establecer contacto con personas específicas en función de sus roles o empresas. También es útil para verificar la autenticidad de direcciones de correo electrónico, en campañas de correo electrónico, y para entender la estructura de correos electrónicos dentro de una organización o dominio específico. Además, puede ayudar en estrategias de prospección y en la construcción de redes profesionales.

2.1.2. Análisis de vulnerabilidades

Las herramientas de análisis de vulnerabilidades tienen como objetivo identificar y evaluar posibles debilidades, fallos de seguridad o riesgos en los sistemas, aplicaciones, redes o infraestructuras de una organización. Su propósito principal es llevar a cabo un escaneo exhaustivo en busca de vulnerabilidades conocidas, tales como configuraciones inseguras, fallos de software, puertos expuestos, y otros posibles vectores de ataque. Estas herramientas utilizan bases de datos de exploits conocidos y técnicas de análisis avanzadas para descubrir y reportar vulnerabilidades potenciales. Su función es crucial para priorizar y remediar las vulnerabilidades encontradas, fortaleciendo así la seguridad y mitigando posibles riesgos de ataques informáticos.

- Searchsploit:

    - Función: Searchsploit es una herramienta incorporada en Kali Linux que simplifica la búsqueda y recuperación de exploits desde la base de datos de exploits de Offensive Security [5], conocida como exploit-db. Su función principal radica en facilitar la identificación y acceso a exploits previamente documentados y disponibles públicamente.
    - Búsqueda de exploits: Permite realizar búsquedas en una amplia base de datos de exploits para identificar y recuperar información detallada sobre vulnerabilidades conocidas.

- Análisis detallado: Ofrece descripciones, códigos de explotación y referencias a las fuentes originales de los exploits, proporcionando a los profesionales de seguridad cibernética información esencial para evaluar y comprender las vulnerabilidades documentadas.
- Apoyo a la investigación: Ayuda a los analistas de seguridad a investigar y comprender exploits conocidos, evaluar su impacto potencial y tomar medidas para mitigar los riesgos asociados con dichas vulnerabilidades en sistemas y aplicaciones.

- Qualys SSL Labs [6] [14]:
  - Función: Qualys SSL Labs es una plataforma en línea que evalúa la configuración de seguridad de los servidores web que utilizan protocolos SSL/TLS. Realiza pruebas exhaustivas para analizar la configuración de cifrado, la validez del certificado SSL, la configuración de protocolos y la seguridad general del servidor.
  - Utilidad: Es una herramienta esencial para administradores de sistemas, ingenieros de seguridad y desarrolladores web, ya que les permite evaluar y mejorar la seguridad de sus servidores web. Proporciona una evaluación detallada de la implementación SSL/TLS, identifica posibles vulnerabilidades, y ofrece recomendaciones para mejorar la seguridad y la configuración de los servidores. Además, ayuda a garantizar el cumplimiento de estándares de seguridad y a mantener la integridad de las comunicaciones en línea.

- SecurityHeaders [7]:
  - Función: SecurityHeaders.com es una herramienta en línea que escanea un sitio web en busca de encabezados HTTP y HTTPS relacionados con la seguridad. Evalúa la presencia y configuración de encabezados de seguridad como Content Security Policy (CSP), Strict-Transport-Security (HSTS), X-Frame-Options, X-XSS-Protection, entre otros.
  - Utilidad: Es una herramienta valiosa para desarrolladores web, administradores de sistemas y profesionales de seguridad informática. Permite verificar y mejorar la configuración de seguridad en un sitio web, ayudando a prevenir ataques como XSS (Cross-Site Scripting), ataques de clicjacking, y otros riesgos de seguridad. Además, ayuda a garantizar el cumplimiento de las mejores prácticas de seguridad recomendadas por OWASP (Open Web Application Security Project) [8] y otros estándares de seguridad web.

2.2. Scripts

Primero de todo empezaremos por saber qué es un script, este es un conjunto de instrucciones o comandos que se utilizan para automatizar tareas o realizar operaciones específicas en un sistema informático. Un script puede ser escrito en diferentes lenguajes de programación, como Python, Perl, Ruby, Bash, entre otros. Los scripts se ejecutan en un entorno de línea de comandos, lo que significa que no necesitan una interfaz gráfica de usuario para ser ejecutados. Los scripts pueden ser utilizados para una variedad de tareas, como automatizar la instalación de software, configurar sistemas, realizar tareas de mantenimiento, realizar copias de seguridad, procesar grandes cantidades de datos, entre otros. Una de las ventajas de utilizar scripts es que permiten a los usuarios automatizar tareas repetitivas, lo que puede ahorrar tiempo y aumentar la eficiencia en el trabajo. Además, los scripts pueden ser compartidos y reutilizados por otros usuarios, lo que los convierte en herramientas valiosas en el desarrollo de software y la administración de sistemas informáticos.

Para este proyecto he decidido utilizar bash, este es un lenguaje de programación de scripting de línea de comandos que se utiliza comúnmente en sistemas operativos Unix y Linux. Bash significa "Bourne-again shell", que es una referencia a la shell original de Unix, llamada "sh". Bash se ha convertido en la shell predeterminada en muchos sistemas operativos Unix y Linux.

Bash es un lenguaje de programación interpretado, lo que significa que los comandos se ejecutan en tiempo real, línea por línea, a medida que se escriben en la línea de comandos. Bash se utiliza comúnmente para escribir scripts de automatización de tareas, como la gestión de archivos y directorios, la automatización de procesos de compilación de software y la gestión de servidores y redes. Bash es un lenguaje de programación de alto nivel que admite variables, operadores, bucles, condicionales, funciones y otros elementos comunes de la programación estructurada. También admite la redirección de entrada y salida, lo que permite que los datos se redirijan a través de tuberías y se procesen de manera efectiva.

2.3. Desarrollo web

2.3.1. Lenguajes y frameworks de programación

Los lenguajes de programación y frameworks son los cimientos de la ingeniería de software moderna, brindando a los desarrolladores las herramientas necesarias para dar vida a aplicaciones innovadoras y funcionales. Los lenguajes de programación, como Python, JavaScript, Java, C++, entre otros, actúan como idiomas con reglas y sintaxis específicas que permiten a los programadores comunicarse con las computadoras y crear software.

Por otro lado, los frameworks son estructuras y conjuntos de herramientas predefinidos que agilizan el proceso de desarrollo al ofrecer soluciones comunes para problemas recurrentes. Frameworks como React, Angular, Django, Ruby on Rails, entre muchos otros, proporcionan una base sólida y componentes reutilizables que permiten a los desarrolladores construir aplicaciones de manera eficiente, escalable y mantenible.

Esta combinación de lenguajes de programación y frameworks no solo impulsa la creación de aplicaciones web, móviles y de escritorio, sino que también alimenta la innovación tecnológica, permitiendo a los desarrolladores explorar nuevos horizontes y llevar a cabo proyectos ambiciosos en el mundo digital.

- Python:
  - Descripción: Python es un lenguaje de programación versátil, de alto nivel y fácil de aprender. Es conocido por su sintaxis clara y legible, lo que lo hace ideal para una amplia gama de aplicaciones, desde desarrollo web hasta inteligencia artificial y ciencia de datos.
  - Versatilidad: Se utiliza en múltiples dominios, incluyendo desarrollo web, scripting, desarrollo de juegos, automatización, entre otros.
- Vue.js:
  - Descripción: Vue.js es un framework progresivo de JavaScript utilizado para la construcción de interfaces de usuario en el lado del cliente (frontend). Es conocido por su enfoque modular y su facilidad de integración en proyectos existentes.
  - Reactividad: Proporciona reactividad y una estructura simple que facilita la creación de interfaces de usuario dinámicas.
  - Escalabilidad: Permite desarrollar desde pequeñas aplicaciones hasta proyectos a gran escala debido a su enfoque progresivo.
- CSS:
  - Descripción: CSS (Cascading Style Sheets) es un lenguaje de estilo utilizado para definir la presentación visual de las páginas web. Es fundamental para el diseño y la apariencia de las aplicaciones web.

    - Estilo y diseño: Permite definir la apariencia, diseño y formato de los elementos HTML en una página web.

- React:
    - Descripción: React es un framework de JavaScript mantenido por Facebook. Se utiliza para la creación de interfaces de usuario y se centra en la eficiencia y la construcción de componentes reutilizables.
    - Componentización: Permite crear componentes reutilizables que facilitan el desarrollo y mantenimiento de aplicaciones.

- Node.js:
    - Descripción: Node.js es un entorno de ejecución de JavaScript en el servidor. Es útil para construir aplicaciones web escalables y de alto rendimiento.
    - JavaScript en el servidor: Permite a los desarrolladores utilizar JavaScript tanto en el frontend como en el backend de una aplicación.

- FastAPI:
    - Descripción: FastAPI es un framework de desarrollo rápido para crear APIs web con Python. Es conocido por su alta velocidad, sencillez y capacidad de auto documentación.
    - Rendimiento: Ofrece un rendimiento rápido y eficiente para el desarrollo de APIs, aprovechando las características de Python.

2.3.2. Herramientas de desarrollo

Estas herramientas son esenciales en el desarrollo de software moderno, facilitando la gestión de aplicaciones, el trabajo colaborativo, el control de versiones y la comunicación con servidores, lo que mejora la eficiencia y la productividad en el desarrollo de aplicaciones y proyectos.

- Docker [9]:
    - Descripción: Docker es una plataforma de software que permite contener aplicaciones y sus dependencias en entornos aislados llamados contenedores.
    - Virtualización ligera: Permite la creación, implementación y ejecución de aplicaciones en contenedores, lo que facilita la portabilidad y distribución de software.

- GitHub:
    - Descripción: GitHub es una plataforma de desarrollo colaborativo que utiliza el sistema de control de versiones Git. Proporciona herramientas para alojar, revisar y colaborar en proyectos de software.

- Control de versiones: Permite a los equipos de desarrollo gestionar el control de versiones de su código, facilitando la colaboración y el seguimiento de cambios.

- Axios:
  - Descripción: Axios es una biblioteca de JavaScript utilizada para realizar solicitudes HTTP desde el navegador o Node.js.
  - Facilidad de uso: Proporciona una interfaz simple y fácil de usar para realizar solicitudes HTTP, lo que la hace popular para la comunicación entre aplicaciones web y servidores.

- pytest:
  - Descripción: pytest es un marco de prueba en Python que se utiliza para escribir y ejecutar pruebas de software de manera eficiente. Ofrece una amplia gama de características y soporte para diversos tipos de pruebas, incluyendo pruebas unitarias, de integración y funcionales.
  - Proporciona una interfaz simple y amigable para escribir y ejecutar pruebas en Python. Su sintaxis clara y su capacidad para integrarse con otros marcos de prueba lo hacen popular entre los desarrolladores. Además, pytest ofrece características avanzadas como parametrización de pruebas, fixtures flexibles y la posibilidad de escribir extensiones personalizadas, lo que lo convierte en una herramienta versátil para la automatización de pruebas en proyectos de cualquier tamaño.

2.3.3. Servicios en la nube y alojamiento

Los servicios en la nube han transformado la gestión de datos y recursos informáticos al proporcionar una red global de servidores remotos accesibles a través de Internet. Esta infraestructura ofrece una amplia gama de servicios, desde almacenamiento hasta cómputo y análisis, permitiendo a empresas y usuarios acceder a recursos según sus necesidades sin invertir en infraestructura física costosa. Por otro lado, el alojamiento web se centra en proporcionar espacios en servidores para sitios web y aplicaciones en línea, garantizando su accesibilidad constante. Ambos, los servicios en la nube y el alojamiento web, son fundamentales en la era digital actual, ofreciendo soluciones flexibles y eficientes para el almacenamiento, gestión y acceso a datos y aplicaciones en línea.

- Azure:
  - Descripción: Azure es la plataforma de servicios en la nube de Microsoft que ofrece una amplia gama de servicios, incluyendo cómputo, almacenamiento, bases de datos, redes y más, permitiendo a los usuarios construir, implementar y administrar aplicaciones en la nube.

- Versatilidad: Proporciona una amplia gama de servicios que se adaptan a diversas necesidades de desarrollo y despliegue de aplicaciones en la nube.
- Escalabilidad: Permite escalar recursos de manera flexible para satisfacer las demandas de las aplicaciones.

- Render:
  - Descripción: Render es una plataforma de alojamiento en la nube que se centra en la simplificación del despliegue y la administración de aplicaciones web.
  - Facilidad de uso: Ofrece una experiencia de despliegue simple y directa para aplicaciones web, facilitando la configuración y el mantenimiento.
  - Automatización: Proporciona herramientas para automatizar tareas de despliegue, actualización y gestión de aplicaciones web.

2.3.4. Base de datos

Las bases de datos constituyen el pilar fundamental en la gestión y almacenamiento de información en la era digital. Estas estructuras organizadas permiten almacenar, organizar, y recuperar datos de manera eficiente, facilitando la manipulación de información para aplicaciones, empresas y sistemas informáticos. Con una amplia diversidad de tipos, como bases de datos relacionales, NoSQL y bases de datos en la nube, estas plataformas son esenciales para mantener la integridad, disponibilidad y seguridad de los datos en un mundo cada vez más interconectado y orientado a la información.

- Supabase [10]:
  - Descripción: Supabase se basa en PostgreSQL, un sistema de gestión de bases de datos relacional potente y de código abierto. Además, proporciona una interfaz de API RESTful que permite a los desarrolladores interactuar con los datos almacenados en la base de datos utilizando endpoints de API predefinidos.
  - Base de datos escalable: Utiliza PostgreSQL como su motor, que es conocido por su confiabilidad, rendimiento y escalabilidad en entornos de bases de datos.
  - Interfaz de API RESTful: Ofrece una interfaz de programación de aplicaciones basada en REST que facilita la comunicación y manipulación de datos almacenados en la base de datos.

3. Implementación, código y ejecución

En el análisis detallado que sigue, examinaremos exhaustivamente las partes clave del proyecto, proporcionando una visión profunda de su implementación y funcionamiento. Estas secciones son fundamentales para comprender en profundidad como se han abordado los desafíos específicos y como se ha logrado la funcionalidad requerida. Sin embargo, es importante mencionar que, debido a la extensión del proyecto y el enfoque en las partes más relevantes, algunas secciones pueden ser tratadas de manera más superficial o incluso no ser abordadas en este contexto. Esto no implica que su importancia sea menospreciada, sino que nos enfocaremos en los aspectos que requieren mayor atención y explicación detallada

3.1. Estructura del proyecto

La estructura del proyecto la explicaré en tres fases:

- Estructura de la página web
- Estructura de los scripts
- Despliegue del proyecto

3.2. Página web

La estructura de la página web de PentesTool es la siguiente:

- .github:
  - workflows:
    - azure-static-web-apps-delightful-flower-067234403.yml: configura un flujo de CI/CD (Integración Continua/Entrega Continua) para Azure Static Web Apps en GitHub, donde se definen trabajos para compilar y desplegar la aplicación en la rama principal (main), así como para cerrar automáticamente solicitudes de extracción (pull requests) en el evento de cierre en la misma rama principal. Utiliza tokens de API y realiza acciones de construcción, despliegue y cierre utilizando la acción Azure/static-web-apps-deploy@v1.
- backend:
  - src:
    - __init__.py: Marcador para que Pyhton reconozca el directorio como un paquete.
    - database.py: Configura la conexión con la base de datos utilizando SQLAlchemy.
    - main.py: Archivo principal que configura las rutas y maneja las solicitudes HTTP (GET, POST, PUT, DELETE) utilizando el framework FastAPI en Python.
    - models.py: Define los modelos de datos utilizando SQLAlchemy, para representar entidades en la base de datos. Crea las relaciones, tablas, columnas, restricciones...
    - repository.py: Funciones que interactúan con la base de datos utilizando SQLAlchemy, para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar)
    - schemas.py: Definiciones de clases BaseModel utilizando la librería Pydantic. Estas clases definen los esquemas de datos que se utilizan para la validación y serialización de datos en la aplicación.
    - test_user.py: Contiene diferentes pruebas unitarias.
  - Dockerfile: Archivo de configuración utilizado para construir una imagen de Docker de la parte del backend.
  - requirements.txt: Lista con las diferentes dependencias de Python.
- frontend:
  - src:
    - components:
      - ◇ FooterView.vue: Contiene nuestro pie de página.

        - ◇ HeaderMenu.vue: Contiene nuestro encabezado de página.
        - ◇ Inicio.vue: Página principal donde podremos visualizar nuestros informes actuales y su estado. Además de poder volver a escanear una IP o URL de la lista.
        - ◇ Login.vue: Se utiliza para registro e inicio de sesión.
        - ◇ NewScan.vue: Sirve para empezar un nuevo escaneo no registrado.
        - ◇ Settings.vue: Se utiliza para visualizar y añadir información de usuario.
        - ◇ VerInforme.vue: Nos permite visualizar el informe.
      - router:
        - ◇ index.js: Configuración de enrutamiento para nuestra página web.
      - main.js: Archivo principal donde se configuran y se inicializan todas las dependencias necesarias.
      - userState.js: Nos ayuda a almacenar información del usuario.
    - Dockerfile: Archivo de configuración utilizado para construir una imagen de Docker de la parte del frontend.

- Docker-compose.yml: Archivo principal de configuración utilizado para construir una imagen de Docker compuesta por tres servicios (backend, frontend y base de datos).

No he añadido archivos como Bootstrap, build, config, dist, node_modules, public, static, package.json, entre otros. Estos archivos son comunes y considero que no es necesario mencionarlos. Para obtener más información, puede revisar el archivo adjunto que contiene el proyecto.

En cuanto a la base de datos tenemos la tabla users que almacena información de los usuarios, mientras que la tabla reports almacena los informes creados por esos usuarios. Estas dos tablas están relacionadas a través del campo user_id en la tabla reports, que se asocia con el username en la tabla users. Esta estructura de base de datos permite relacionar informes con usuarios específicos en la aplicación.

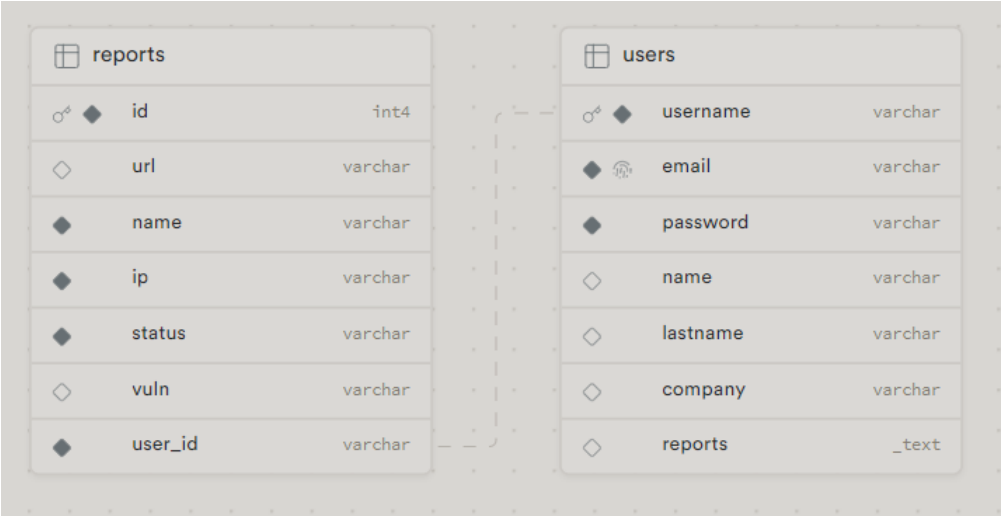

Figura 2: Esquema de la base de datos

Los informes se vinculan en la base de datos a través de una URL que representa la ubicación del recurso asociado. En la tabla 'reports', el campo 'url' guarda la dirección web que permite acceder al recurso relacionado con el informe almacenado externamente a la base de datos.

Al ingresar a la página web, lo primero que encontramos es la opción de "signin/login", donde podemos llevar a cabo ambas acciones desde la misma interfaz.

En la sección de "signin", se solicitará un nombre de usuario (único y con un mínimo de 15 caracteres), una contraseña (de al menos 8 caracteres, que incluya al menos una mayúscula, un número y un carácter especial), así como una dirección de correo electrónico (verificada mediante expresiones regulares y única).

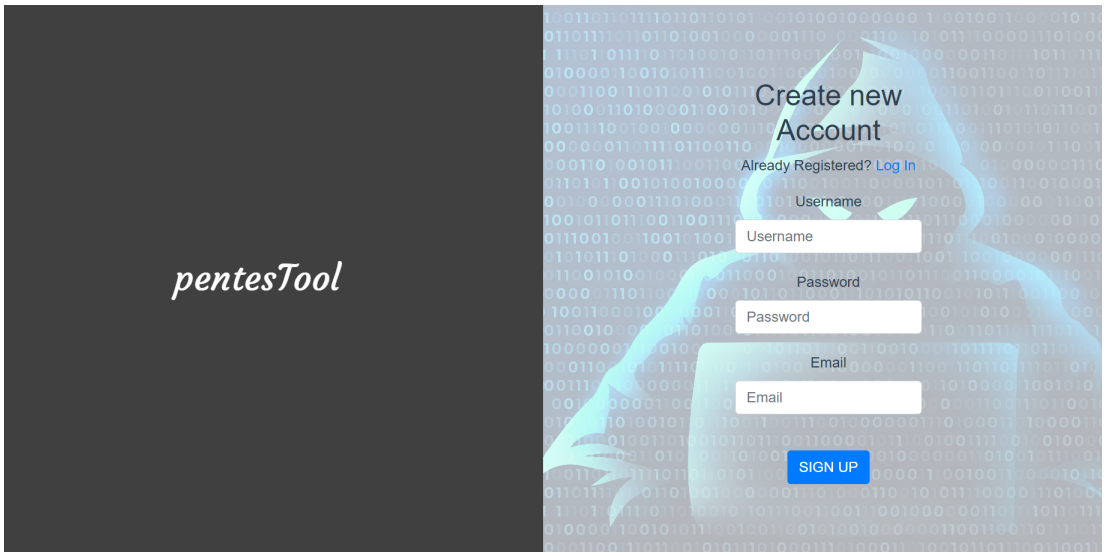

Figura 3: Registro de usuario

Al hacer uso de la funcionalidad de LocalStorage, se guardarán estos datos para

facilitar futuros accesos. Esta característica permite actualizar la página sin dificultades una vez iniciada la sesión y, al cerrar la página web, al volver a ingresar nos llevará directamente a la página de inicio, evitando pasar por el signin/login nuevamente.

Una vez que LocalStorage almacena nuestros datos, nos redirigirá a la página de inicio, que estará vacía la primera vez. La página mostrará una cabecera con el nombre del sitio, enlazando de vuelta a la página de inicio; seguido por una pestaña para realizar nuevos escaneos y un menú desplegable con opciones de ajustes y cerrar sesión. La última opción nos llevará al login y eliminará los datos guardados en localStorage. Además, encontraremos un pie de página con datos de contacto e información adicional.

Al acceder a la pestaña de "NewScan", completaremos el formulario con el nombre del informe y una dirección IP o URL a escanear de manera obligatoria, visualmente lo veremos con una estrella roja, opcionalmente podemos añadir una frecuencia, para que cada ciertas horas se genere un nuevo reporte automáticamente, esta frecuencia va de 1 a 999 horas y saltará error en caso contrario, además añadirá automáticamente un estado de "In Progress".

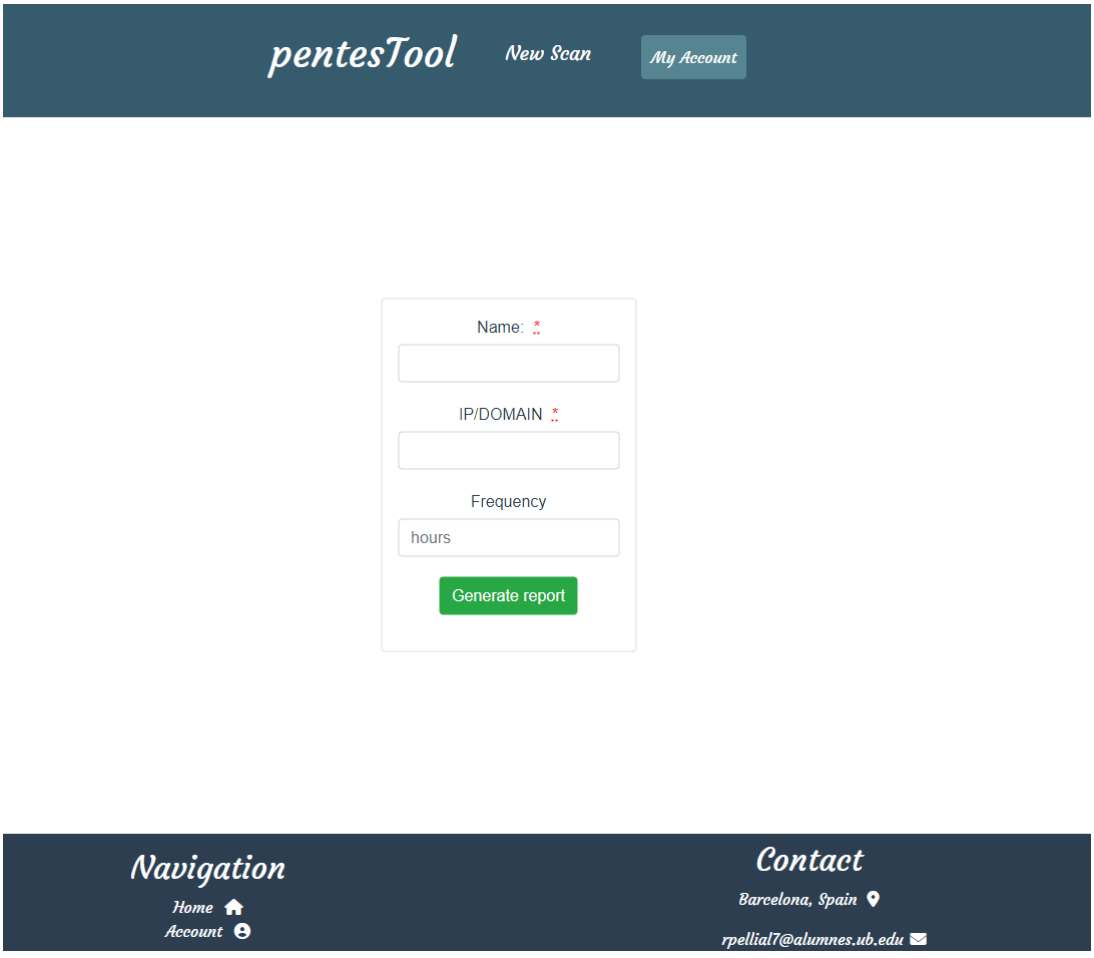

Figura 4: Nuevo escaneo

En la sección de “Inicio“, podremos visualizar los informes creados y su estado. También tendremos la opción de eliminarlos utilizando el botón correspondiente. Una vez que un informe pasa de “In Progress“ a “Completed“, se habilitará un botón para visualizar dicho informe. Asimismo, tendremos la opción de escanear nuevamente una o varias IPs o dominios. Para ello, seleccionaremos los elementos deseados y pulsaremos el botón “Start Report“, cambiando así el estado de todos los seleccionados de nuevo a “In Progress“. Si la IP o dominio no existe o no es alcanzable, se mostrará el estado de “No Connection“.

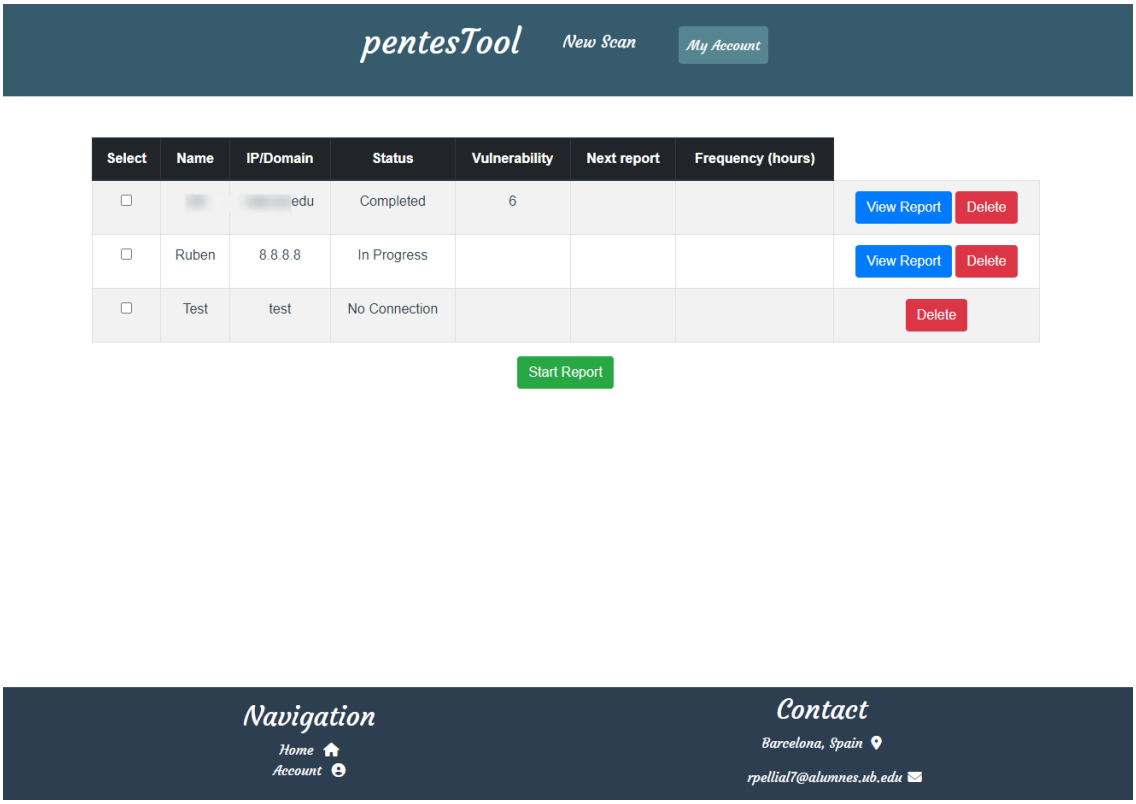

Figura 5: Inicio

Al hacer clic en “View Report“, seremos redirigidos a otra página donde podremos visualizar diferentes secciones del informe, cada una en cartas separadas con un título y su información adicional. Estos datos pueden ser puramente informativos o detallar vulnerabilidades o riesgos relacionados con alguna función específica.

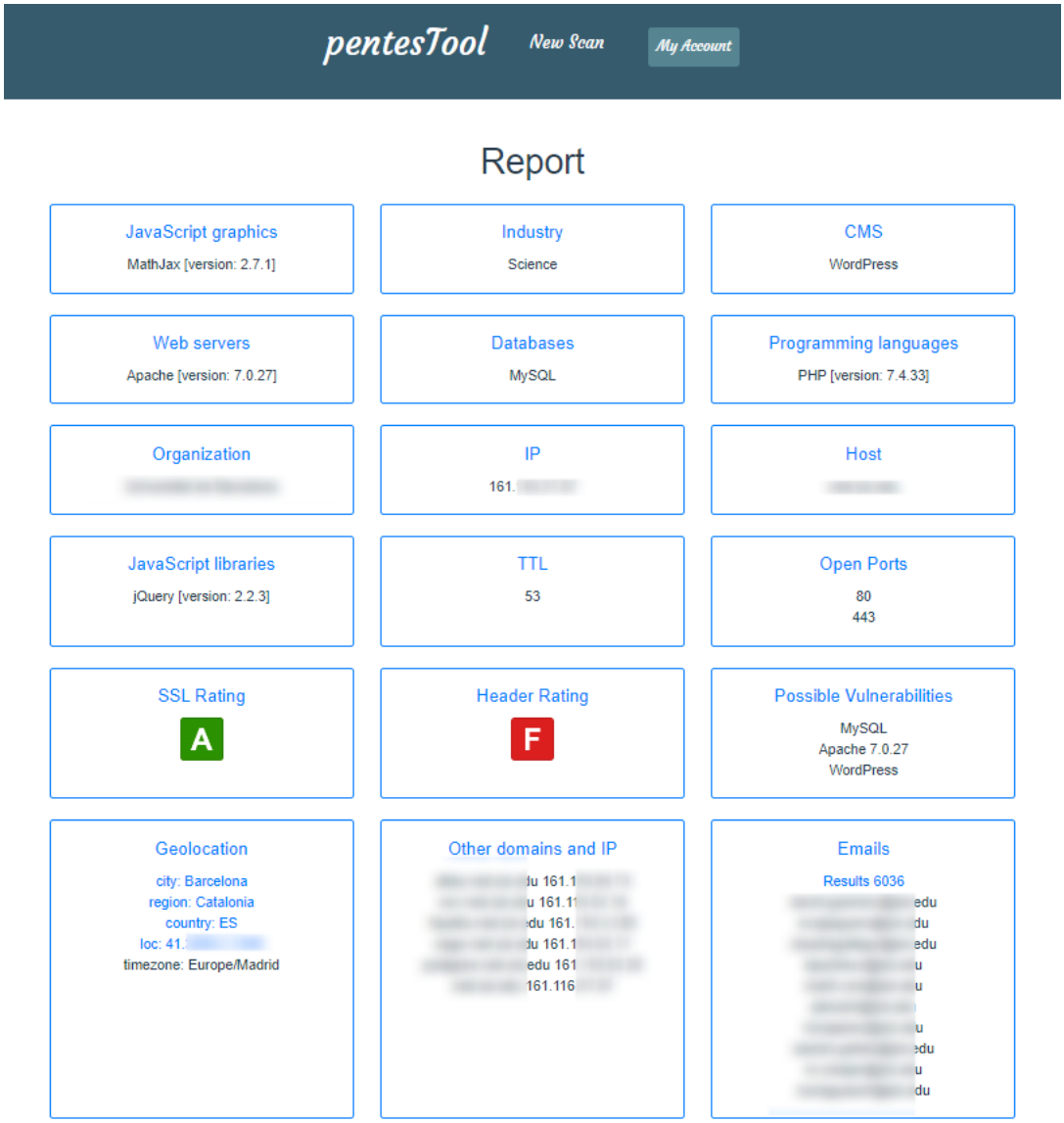

Figura 6: Ver Informe

Finalmente, en la sección de “Settings“, podremos visualizar los datos que se tienen sobre nosotros para simplemente consultar, añadir, modificar o eliminar. Los datos adicionales incluyen nombre, apellido y empresa, y se ofrece la opción de eliminar la cuenta.

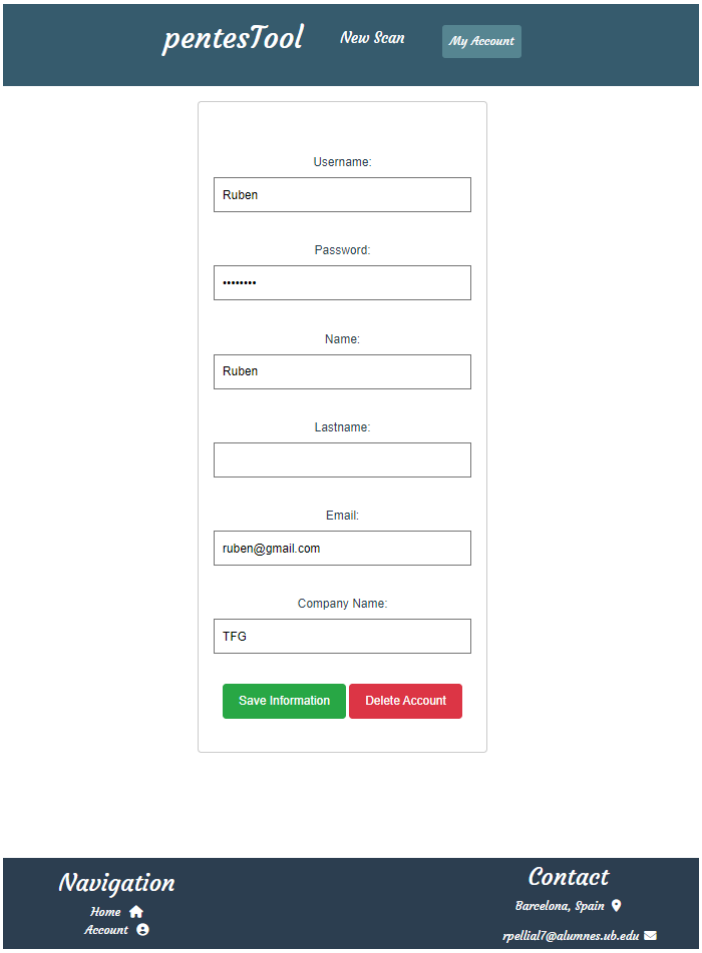

Figura 7: Ajustes de perfil

3.3. Scripts

Los scripts de mi proyecto son ejecutados desde una máquina Parrot [11], la cual es una distribución Linux altamente especializada y configurable. Parrot OS ofrece una variedad de herramientas preinstaladas diseñadas específicamente para actividades relacionadas con la seguridad informática, análisis forense, anonimato en línea, desarrollo y programación.

Figura 8: Logo de Parrot OS

La elección de utilizar Parrot como plataforma para ejecutar los scripts se basa en su conjunto completo de herramientas incorporadas, su robustez en cuanto a seguridad y privacidad, además de su flexibilidad para adaptarse a distintos requerimientos de proyectos. Desde Parrot, se pueden aprovechar y utilizar las utilidades y comandos integrados para automatizar procesos, realizar análisis de datos, y llevar a cabo pruebas o tareas específicas que requieran de un entorno confiable y potente.

Al utilizar Parrot como entorno de ejecución para los scripts, se garantiza una sólida infraestructura que respalda la ejecución de las tareas de manera eficiente y segura, aprovechando las capacidades integradas en esta distribución para alcanzar los objetivos del proyecto con confianza y precisión.

La estructura de los scripts de PentesTool es la siguiente:

- auto:
  - archivos:
    - txt: carpeta donde guardaremos los archivos .txt con la información procesada.
    - xml: carpeta donde guardaremos los archivos .xml con la información procesada.
    - informe.md: este es el archivo con formato markdown con todos los datos de los archivos .txt y .xml procesados.
  - scripts:
    - actualizar_md.sh: Revisa los archivos .txt y .xml y los junta en un mismo archivo creando el informe.md.
    - all.sh: Este script es quien se encarga de llamar a los diferentes scripts según la información recibida. También llama al script de crear el informe y lo sube a la base de datos, actualizando el estado.
    - amass.sh: Recoge los datos de otros dominios e IPs relacionadas a la escaneada y procesa estos datos guardandolo en un archivo amass.txt.
    - check_sslscan.py: Recoge una valoración de la A+ hasta la F dependiendo de los resultados del certificado SSL obtenidos.

    - dig.sh: Recoge la IPs de un dominio y la guarda en dig.txt.
    - geo.sh: Recoge los datos de la geolocalización como ciudad, región , pueblo y coordenadas, junto al hostname y lo guarda en geolocation.txt.
    - header.sh: Recoge una valoración de la A+ hasta la F dependiendo de los resultados del header de la página web obtenidos.
    - hunter.sh: Guarda correos electrónicos, nombre de la organización e industria de la empresa en hunter_emails.txt.
    - json.sh: Se encarga de transformar el archivo informe.md, en el archivo json que subiremos a la base de datos.
    - nmap.sh: Hace un reconocimiento de puertos abiertos y guarda la información más relevante en nmap.xml.
    - nslookup.sh: Recoge el dominio de una IP y la guarda en nslookup.txt.
    - searchsploit.sh: Revisa si alguna información guardada tiene alguna posible vulnerabilidad, en caso de que así sea se añadirá al archivo vuln_number.txt.
    - ssl.sh: Ejecuta el archivo check_sslscan.py y guarda la información en ssl.txt.
    - storage.js: Sube el informe .json a la base de datos.
    - wappy.sh: Recoge la información de las tecnologías utilizadas en la página web y las guarda en wappy.txt.
    - xml.sh: Transforma el archivo .xml en un archivo .txt para poder juntarlo todo más fácil.

- bdd.sh: Cuando en la página web algún informe tiene estado "In Progress", este script se encarga de recoger el name, IP o URL y user_id y enviarlo a al script all.sh.

Dentro de estos scripts aparte de usar las herramientas mencionadas anteriormente, se hace uso de otros comandos básicos como ping, mkdir, rm, sudo... y para manipular y procesar datos utilizamos jq, curl, tee, awk, grep, tail, sed entre otros.

3.4. Despliegue del proyecto

El despliegue de este proyecto se organiza meticulosamente para asegurar su funcionamiento y sincronización efectiva entre sus diferentes componentes.

Utilizamos Docker para encapsular la aplicación en un contenedor. Esto ofrece una portabilidad excepcional y permite la ejecución consistente de la aplicación en una variedad de entornos. La modularidad y la capacidad de escalabilidad inherente a Docker facilitan enormemente el desarrollo y el despliegue.

La parte del backend está alojada en Render, un servicio de alojamiento en la nube altamente confiable y escalable. La implementación de actualizaciones se realiza con fluidez a través de un simple push, garantizando una rápida puesta en marcha de las últimas versiones. Para la gestión de la base de datos, confiamos en Supabase, que proporciona una capa sólida y segura para el almacenamiento de datos con funcionalidades avanzadas.

El frontend, que constituye la interfaz de usuario, se encuentra alojado en Azure como una aplicación web estática. Se configura para tomar automáticamente la rama principal (main) del repositorio de GitHub, lo que asegura que los usuarios accedan siempre a la última versión disponible sin demoras ni inconvenientes.

El componente de monitoreo está a cargo de una máquina virtual equipada con Parrot OS. Esta máquina realiza verificaciones constantes del estado de la base de datos. Si detecta que algún reporte está en estado 'In Progress', recoge los datos que necesita de la base de datos y ejecuta scripts predefinidos para procesar la información y actualizar el estado a 'Completed' y genera informes detallados para su revisión y resolución almacenándolos en la base de datos. En caso de no encontrar conexión en la IP o URL, simplemente cambiará el estado a 'No Connection'.

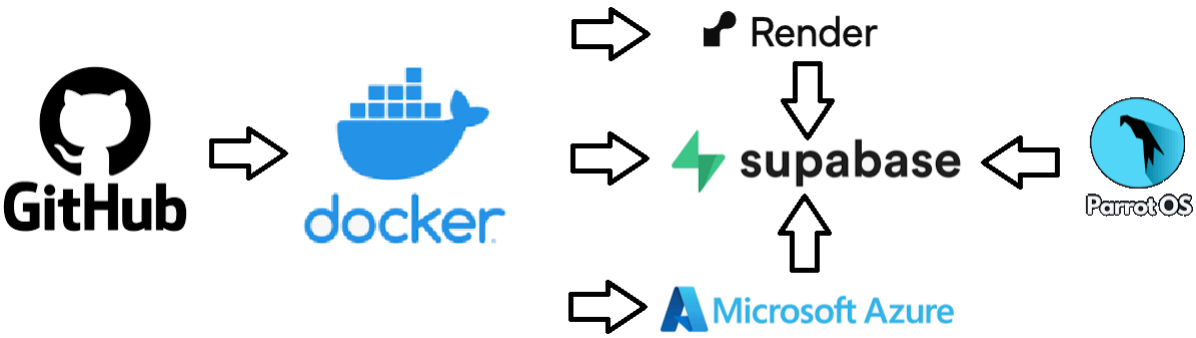

Figura 9: Flujo de pentesTool

En resumen, como podemos ver en la figura 9, github crea los contenedores en docker, donde este tendrá como contenedores backend (Render), base de datos (Supabase) y Frontend (Azure), donde tanto Render como Azure harán solicitudes a la base de datos. Por otro lado, tenemos la máquina virtual (Parrot OS) que también hará solicitudes a la base de datos.

Este enfoque integral de despliegue permite mantener una actualización cons-

tante, garantizar la disponibilidad y llevar a cabo una supervisión proactiva del proyecto en todas las etapas de su funcionamiento. La combinación estratégica de estas herramientas y servicios crea un entorno robusto y confiable para la implementación exitosa de la aplicación.

4. Análisis de los resultados

En el transcurso del proyecto, se realizó una meticulosa serie de pruebas destinadas a asegurar la calidad y confiabilidad tanto de los scripts individuales como de la plataforma web en su conjunto. Este enfoque de pruebas se llevó a cabo en diferentes etapas del desarrollo para garantizar un producto final robusto y funcional.

- Pruebas de scripts individuales:

  Cada script utilizado en el proyecto fue sometido a rigurosas pruebas en su propio entorno. Se ejecutaron diversos escenarios y se simularon diferentes condiciones para verificar su rendimiento, precisión y comportamiento bajo diversas circunstancias. Esta fase de pruebas individuales fue esencial para identificar y corregir posibles fallos, asegurando que cada script cumpliera su función específica de manera confiable y eficiente.

  Las pruebas exhaustivas no solo validaron la funcionalidad, sino que también permitieron detectar posibles cuellos de botella o áreas susceptibles de mejora, lo que llevó a la optimización de cada script para su mejor desempeño.

- Pruebas incrementales de la página web:

  En el desarrollo de la página web, se adoptó un enfoque incremental para las pruebas. Cada nueva característica o cambio introducido en la interfaz web fue sometido a pruebas exhaustivas de funcionalidad y compatibilidad. Se realizaron tests de usuario, pruebas de carga y evaluaciones de usabilidad para garantizar que cada componente nuevo o modificado funcionara sin problemas y no afectara la estabilidad de las características existentes.

  La realización de pruebas paso a paso permitió identificar y solucionar posibles conflictos entre diferentes secciones de la página, asegurando una experiencia de usuario fluida y consistente a medida que se agregaban nuevas funcionalidades.

- Iteraciones basadas en resultados:

  Los resultados obtenidos de estas pruebas jugaron un papel fundamental en la toma de decisiones para mejorar el sistema. Se analizaron minuciosamente para identificar áreas de mejora y se implementaron cambios significativos en los scripts y en la estructura de la página web en base a estas conclusiones.

  Las mejoras fueron diversas, desde optimizaciones de rendimiento hasta ajustes de usabilidad y correcciones de errores. Cada cambio se aplicó cuidadosamente, manteniendo un enfoque centrado en la mejora continua basada en los datos recopilados durante las pruebas.

- Pruebas de integración completa:

  Una vez que se logró un nivel de rendimiento satisfactorio en cada componente individual, se llevaron a cabo pruebas integrales. Estas pruebas simularon situaciones reales de uso, integrando los scripts, la base de datos, el backend y el frontend en un escenario de prueba unificado.

Estas pruebas de integración integral permitieron validar la interacción entre todos los elementos del sistema, desde las solicitudes de los scripts hasta la respuesta de la página web, garantizando su funcionamiento coordinado y eficiente en un entorno realista.

El enfoque iterativo, combinado con pruebas exhaustivas en diferentes niveles, permitió no solo identificar y corregir problemas, sino también garantizar un sistema robusto y confiable que responde a las necesidades del usuario y se adapta a diferentes situaciones.

4.1. Comparativa con otras páginas similares

En este apartado, se llevará a cabo una comparativa detallada con algunas plataformas relevantes en este campo, como son, Shodan, Censys, ZoomEye [12] y FOFA [13], con el objetivo de evaluar la efectividad y las capacidades de nuestra página en relación con la información expuesta.

En nuestra comparativa, nos enfocaremos en aspectos clave como la profundidad de análisis, la precisión de los datos recopilados, la interfaz de usuario, la capacidad de identificar y mitigar vulnerabilidades. Evaluaremos cómo nuestra página se compara con estas plataformas líderes, destacando sus fortalezas y mejoras potenciales en términos de seguridad, visibilidad en línea y protección de la información.

Esta comparativa nos permitirá identificar áreas de mejora y destacar las fortalezas de nuestra página en comparación con estas plataformas líderes en el ámbito de la seguridad cibernética.

Empezaremos la comparativa con una página muy famosa cuyo uso principal es ser un traductor, corrector ortográfico y gramatical de la lengua catalana para fomentar su uso.

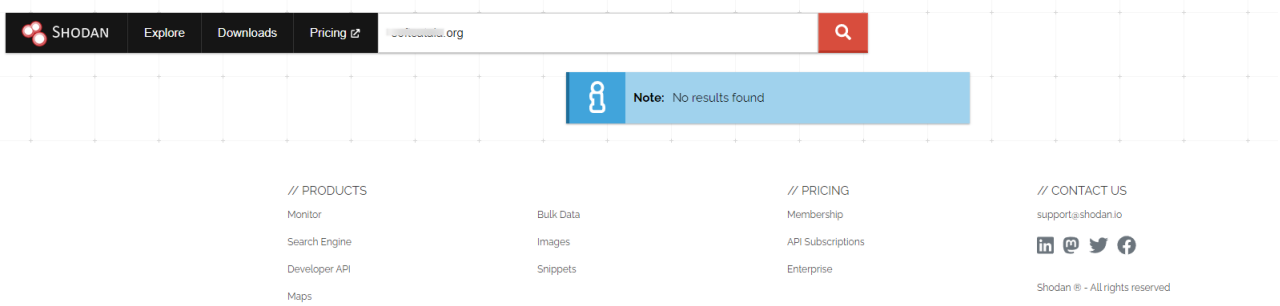

Figura 10: Ejemplo de error en Shodan.io

Como podemos ver en la Figura 10, Shodan no encuentra resultados para esta página. Algo que me sorprende de la página similar más utilizada. Así que buscaremos otra para ver los resultados.

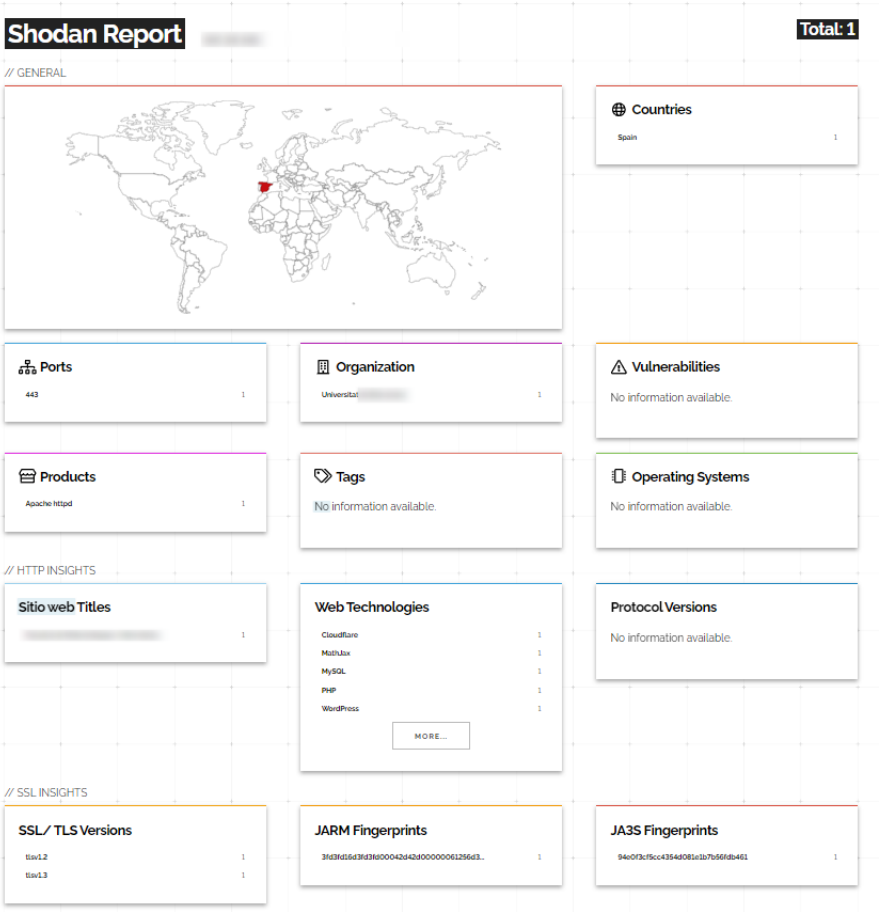

Figura 11: Ejemplo en Shodan.io

Censys, en cambio, sí que encuentra información como puertos abiertos, localización geográfica, servicios, otros DNS entre otros. La información generada eta bien, pero hecho en falta algún punto donde hable de vulnerabilidades o posibles vulnerabilidades. Así que podríamos decir que es una página que su función es únicamente recopilar información sin hacer después un análisis de vulnerabilidades. Como otro punto negativo pondría que no es muy visular de primeras y requiere de hacer mucho scroll.

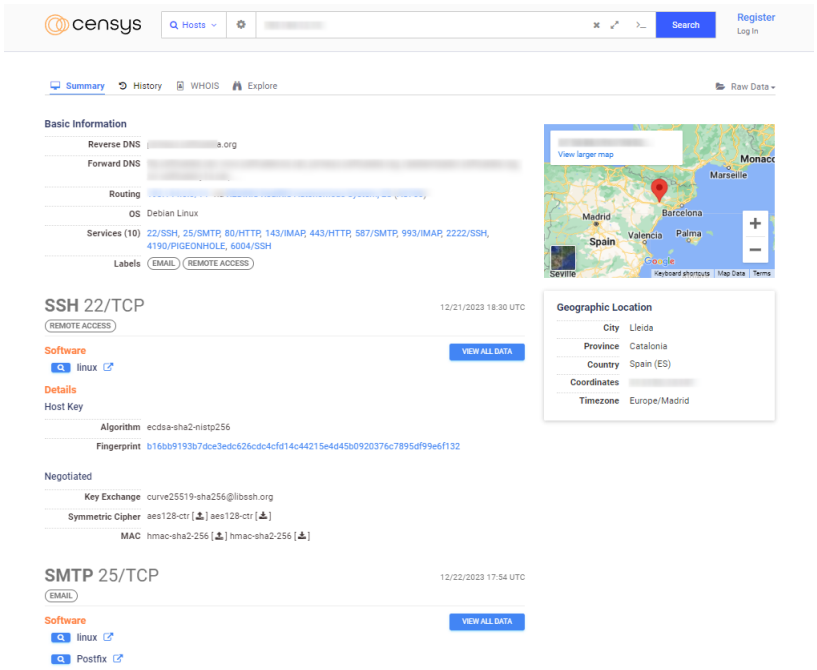

Figura 12: Ejemplo en Censys

FOFA es bastante parecido, incluso diría que muestra menos información y de manera menos clara.

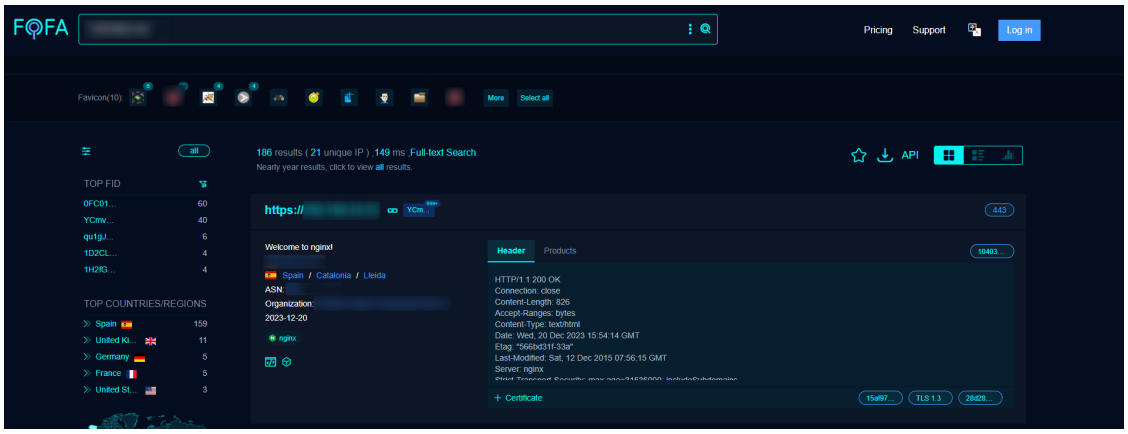

Figura 13: Ejemplo en FOFA

Por último una página menos conocida que Shodan y Censys es ZoomEye, una página que te dice menos información que estas dos, pero, en cambio, te dice si tiene vulnerabilidades y cuáles son.

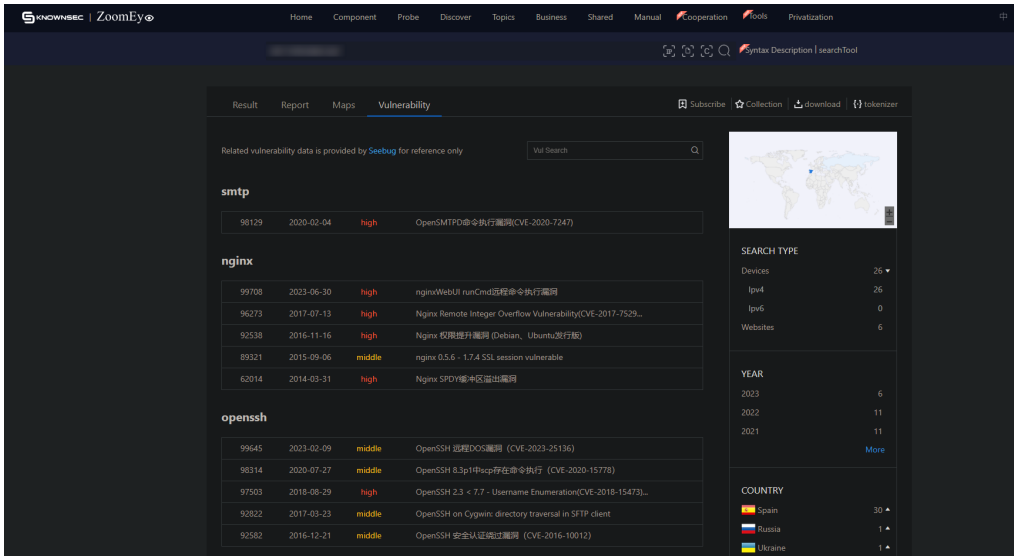

Figura 14: Ejemplo en ZoomEye

A continuación, se presenta una tabla comparativa con las herramientas de recopilación de información y análisis de con URL/IP mencionadas anteriormente, destacando sus principales características y funcionalidades. Gracias a esta tabla comparativa podremos ver las principales diferencias, puntos fuertes y flojos de cada herramienta.

Comparativa de páginas parecidas					
	Shodan	Censys	FOFA	ZoomEye	pentestTool
Utilidad	Recopilación de información y análisis	Recopilación de información	Recopilación de información	Recopilación de información y análisis	Recopilación de información y análisis
Información	Información mejorable pero filtrada	Mucha información poco filtrada	Poca información	Mucha información poco filtrada	Buena información filtrada
Vulnerabilidades	De pago	No tiene	No tiene	Muy buen análisis	Buen análisis
Interfaz	Interfaz web y API	Interfaz web y API (de pago)	Interfaz web y API	Interfaz web y API	Interfaz web
Privacidad	Datos públicos recopilados	Datos públicos recopilados	Datos públicos recopilados	Datos públicos recopilados	Datos públicos recopilados
Velocidad	<5min	<5min	<5min	<5min	>5min
Informes	Individual y sin revisión posterior	Individual y sin revisión posterior	Individual y sin revisión posterior	Individual y sin revisión posterior	Individual, múltiple y con revisión posterior
Costo	Versión gratuita y de pago (más búsquedas y vulnerabilidades)	Versión gratuita y de pago (más búsquedas e información)	Versión gratuita y de pago (más consultas API, búsquedas e información)	Versión gratuita y de pago (más consultas API, búsquedas e información)	Versión gratuita

Cuadro 1: Comparativa de páginas parecidas

5. Conclusiones

Durante el desarrollo de este proyecto, se han logrado alcanzar los objetivos principales establecidos, así como se han identificado metas adicionales que podrían ser exploradas en futuras etapas.

El proyecto pentesTool proporciona a los auditores y profesionales de este ámbito una herramienta versátil y poderosa. Desde su concepción hasta su desarrollo final, se ha trabajado para ofrecer una plataforma integral que permita un control detallado de múltiples direcciones, brindando informes exhaustivos que incluyen datos relevantes como sistemas operativos, puertos abiertos con las versiones de sus servicios, geolocalización, y una amplia gama de información crucial para evaluar la seguridad de los sistemas.

Este proyecto ha sido un ejercicio continuo de innovación tecnológica, impulsando la creación de una herramienta que facilita el análisis y control de direcciones de manera eficiente. La usabilidad y accesibilidad de pentesTool han sido aspectos fundamentales en su desarrollo. Se ha prestado especial atención a una interfaz amigable e intuitiva, permitiendo a usuarios de distintos niveles de experiencia acceder y utilizar la herramienta sin complicaciones.

Durante la fase inicial del proyecto, se llevó a cabo un exhaustivo estudio de las diferentes herramientas disponibles en el mercado. Tras un análisis meticuloso, se seleccionaron las más adecuadas, lo que permitió la creación de un conjunto de scripts personalizados. Estos scripts, ejecutados en una máquina virtual especialmente configurada, han sido fundamentales para recopilar datos precisos y generar informes detallados. Esta estrategia ha consolidado un informe integral y meticuloso que supera las expectativas establecidas.

La interfaz de pentesTool se ha diseñado meticulosamente con el objetivo de garantizar una experiencia intuitiva y cumplir con una amplia gama de requisitos. Más allá de las expectativas iniciales, la página ofrece una experiencia de usuario fluida y completa. Su diseño intuitivo facilita el acceso y la comprensión de las funcionalidades incluso para aquellos usuarios con niveles de experiencia variados en el ámbito de la auditoría y la seguridad informática.

La integración de las diversas etapas del proyecto, desde la creación de la página web hasta la implementación de la base de datos, la configuración de la máquina virtual con los scripts y la posterior migración a la nube, presentó desafíos significativos. A pesar de las complejidades y obstáculos encontrados en este proceso, se superaron con éxito. La dedicación y el esfuerzo dedicado a solventar estos desafíos llevaron finalmente a la culminación exitosa de esta integración compleja, logrando un producto final funcional y coherente.

Sin embargo, a pesar de los logros obtenidos, el desarrollo de pentesTool no concluye aquí. Reconocemos que existen oportunidades para mejorar y expandir sus capacidades. Se vislumbra la posibilidad de integrar nuevas funcionalidades que enriquezcan aún más la experiencia del usuario, así como mejorar la precisión y velocidad de la herramienta para adaptarse a entornos más complejos y exigentes. Además, se busca fomentar una colaboración más estrecha con la comunidad

de usuarios y expertos en seguridad, para recibir retroalimentación constante que permita mejorar la herramienta en futuras versiones.

5.1. Ampliaciones futuras

El desarrollo de esta plataforma ha sido un paso significativo hacia la gestión eficiente de informes de seguridad web. No obstante, se identifican áreas clave para mejoras y ampliaciones futuras que podrían potenciar su funcionalidad, accesibilidad y seguridad, así como enriquecer la experiencia del usuario y la utilidad de la herramienta en general.

Tarea 1. Mejora de la seguridad de la página web: La seguridad es un aspecto fundamental en cualquier plataforma, especialmente cuando se trata de datos sensibles. Para mejorar la seguridad de la página web, se pueden implementar medidas como la encriptación de datos sensibles, la autenticación con dos o más factores (MFA), el escaneo continuo de vulnerabilidades y la aplicación de parches de seguridad de manera proactiva. Además, realizar auditorías regulares de seguridad y cumplir con las mejores prácticas de desarrollo seguro sería crucial para proteger la integridad de la información almacenada y la privacidad de los usuarios.

Tarea 2. Botones de descarga en diferentes formatos: Se sugiere la incorporación de botones que permitan a los usuarios descargar los informes en formatos específicos, como archivos JSON para la manipulación de datos o en formato PDF para una visualización más accesible. Esta mejora facilitaría la utilización de los informes según las necesidades específicas de los usuarios, brindando flexibilidad en la manipulación o simplemente en la visualización de la información.

Tarea 3. Funcionalidades de selección masiva: La adición de un botón para seleccionar todos los informes desde la página de inicio, para mejorar la usabilidad y rapidez en la plataforma.

Tarea 4. Implementación del buscador: Inclusión de un buscador avanzado, permitirá los usuarios encontrar rápidamente los informes deseados. El buscador podría ser capaz de buscar informes por nombre, estado, vulnerabilidades u otros criterios relevantes, mejorando significativamente la usabilidad y la eficiencia de la plataforma.

Tarea 5. Filtro de datos en la página de inicio: Agregar un filtro en la página de inicio para la tabla de informes facilitaría la organización y clasificación de los informes, permitiendo a los usuarios visualizarlos de acuerdo con criterios específicos.

Tarea 6. Edición en la página de inicio: Añadir la capacidad de editar datos directamente desde la página de inicio será una adición valiosa, brindando mayor flexibilidad en la gestión y actualización de la información.

Tarea 7. Mejoras en los informes: Es esencial ampliar y mejorar los informes existentes para proporcionar información más detallada y valiosa a los usuarios. La

inclusión de detalles como los CVE (Common Vulnerabilities and Exposures) de las vulnerabilidades identificadas, sugerencias sobre áreas a mejorar, una valoración global del dominio/IP y otros datos relevantes, enriquecerían enormemente la utilidad de los informes.

Tarea 8. Integración de API_KEY: La integración de una API_KEY permitirá un acceso seguro a la API de la plataforma, brindando una capa adicional de seguridad y control en el intercambio de información entre sistemas. Esto facilitará a los usuarios autorizados aprovechar la funcionalidad de la plataforma de manera segura y controlada.

Tarea 9. Compartir informes: Habilitar la capacidad de compartir informes entre distintos miembros de una misma empresa. Esta funcionalidad permitirá a los usuarios compartir de manera eficiente y segura los informes generados, promoviendo la colaboración y el intercambio de información relevante dentro de la organización.

En conclusión, la implementación de estas mejoras y ampliaciones futuras fortalecerá la plataforma, proporcionando una experiencia más segura, eficiente y enriquecida para los usuarios. Estos avances no solo mejorarán la funcionalidad de la herramienta, sino que también contribuirán significativamente a la seguridad y utilidad en la gestión de informes de seguridad web.

6. Bibliografía

Referencias

[1] John Matherly, Shodan: Motor de búsqueda,
https://www.shodan.io/, 2009.

[2] Zakir Durumeric, Censys: Motor de búsqueda,
https://search.censys.io/, 2015.

[3] BlackArrowSec, Wappy,
https://github.com/blackarrowsec/wappy, 18 de noviembre 2021.

[4] Antoine Finkelstein and FranÃ§ois Grante, Hunter: Buscador de emails,
https://hunter.io/, 2015.

[5] Mati Aharoni and Devon Kearns, Offensive Security : Compañia de ciberseguridad,
https://www.exploit-db.com/, s.f.

[6] Philippe Courtot, Qualys: Gestión de vulnerabilidades,
https://www.qualys.com/certview/, 1999.

[7] Scott Helme, Security Headers: Analizador de cabeceras web,
https://securityheaders.com/, s.f.

[8] Mark Curphey and Dennis Groves, OWASP: Comunidad de ciberseguridad,
https://owasp.org/www-project-top-ten/, 2001.

[9] Solomon Hykes, Docker: Despliegue de aplicaciones,
https://www.docker.com/, 2013.

[10] Supabase: Base de datos,
https://supabase.com/, s.f.

[11] Frozenbox Team, Parrot OS: Distribución GNU/Linux,
https://www.parrotsec.org/, 10 de abril 2013.

[12] ZoomEye: Motor de búsqueda,
https://www.zoomeye.org/, s.f.

[13] FOFA: Motor de búsqueda,
https://en.fofa.info/, s.f.

[14] Philippe Kueck, check_sslscan,
https://github.com/philfry/check_sslscan, 2022.

A. Tecnologías

```yaml
version: '3.8'
services:
  backend:
    image: rubenpa97/pentestool:latest
    build: ./backend
    ports:
      - 8000:8000
    environment:
      - DATABASE_URL=postgresql://postgres:postgres@db:5432/appdb
      - PYTHONPATH=/app/src
    volumes:
      - ./backend:/app
    depends_on:
      - db
    networks:
        - app-network
  frontend:
    build: ./frontend
    volumes:
      - ./frontend:/app
      - /app/node_modules
    ports:
      - 8080:8080
    networks:
        - app-network
  db:
    image: postgres:15.1
    expose:
      - 5432
    environment:
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
      - POSTGRES_DB=appdb
    volumes:
      - postgres_data:/var/lib/postgresql/data/
    networks:
        - app-network
volumes:
  postgres_data:
networks:
```

Figura 15: Docker-compose.yml

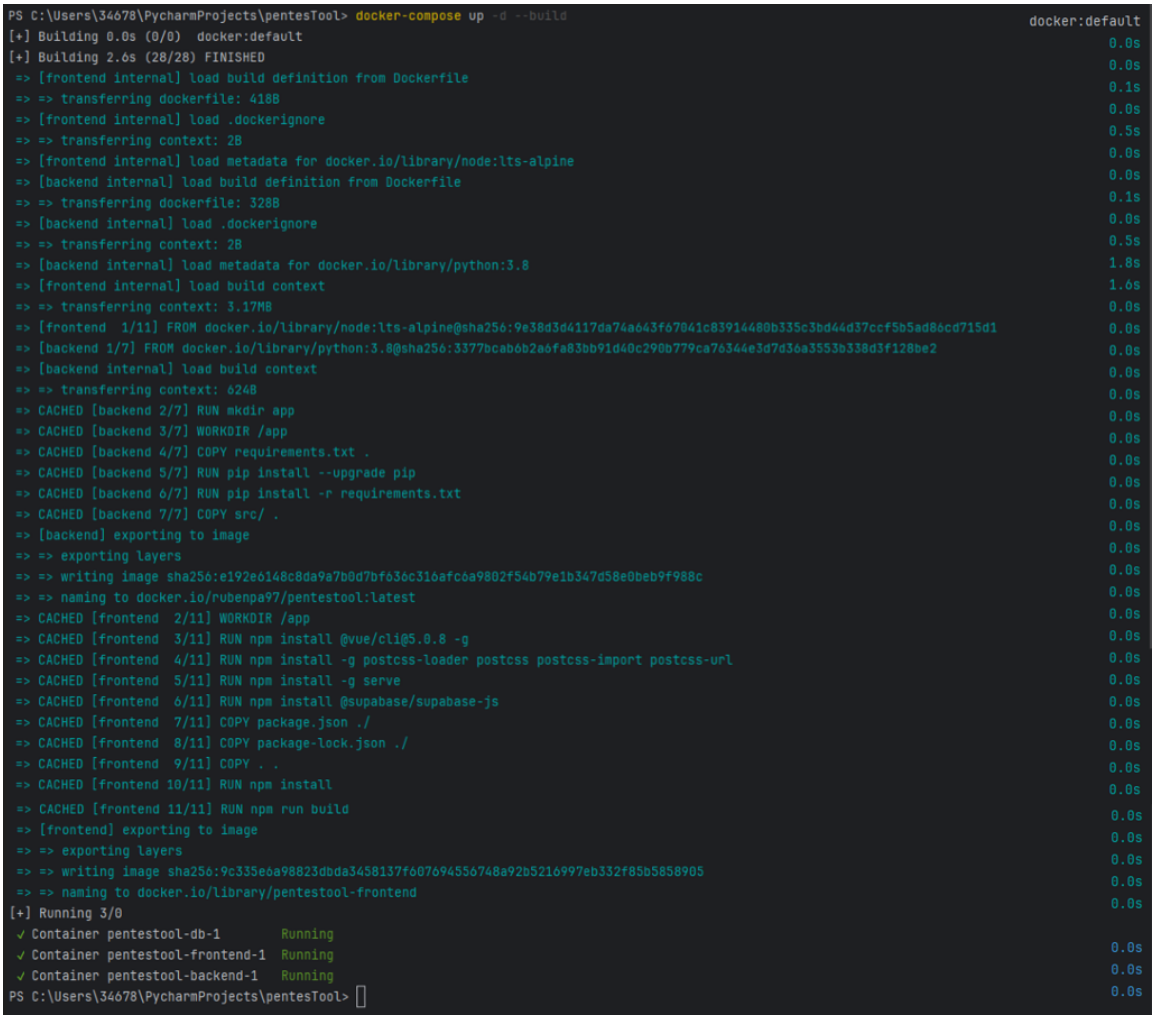

Figura 16: Crear contenedores en docker

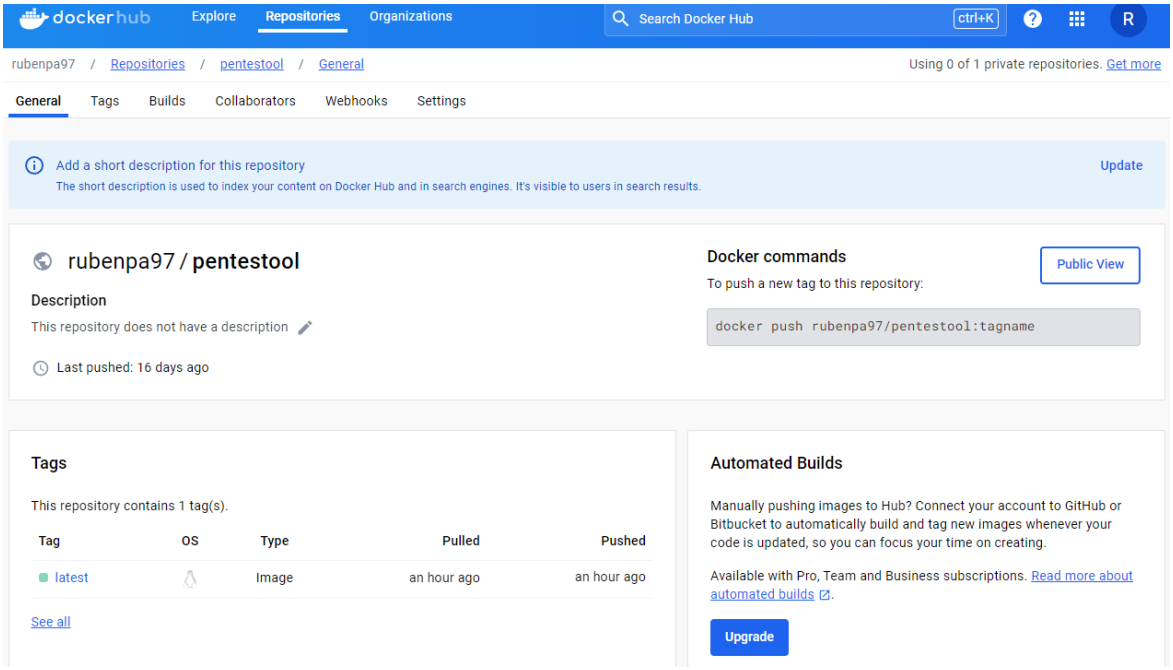

Figura 17: Dockerhub

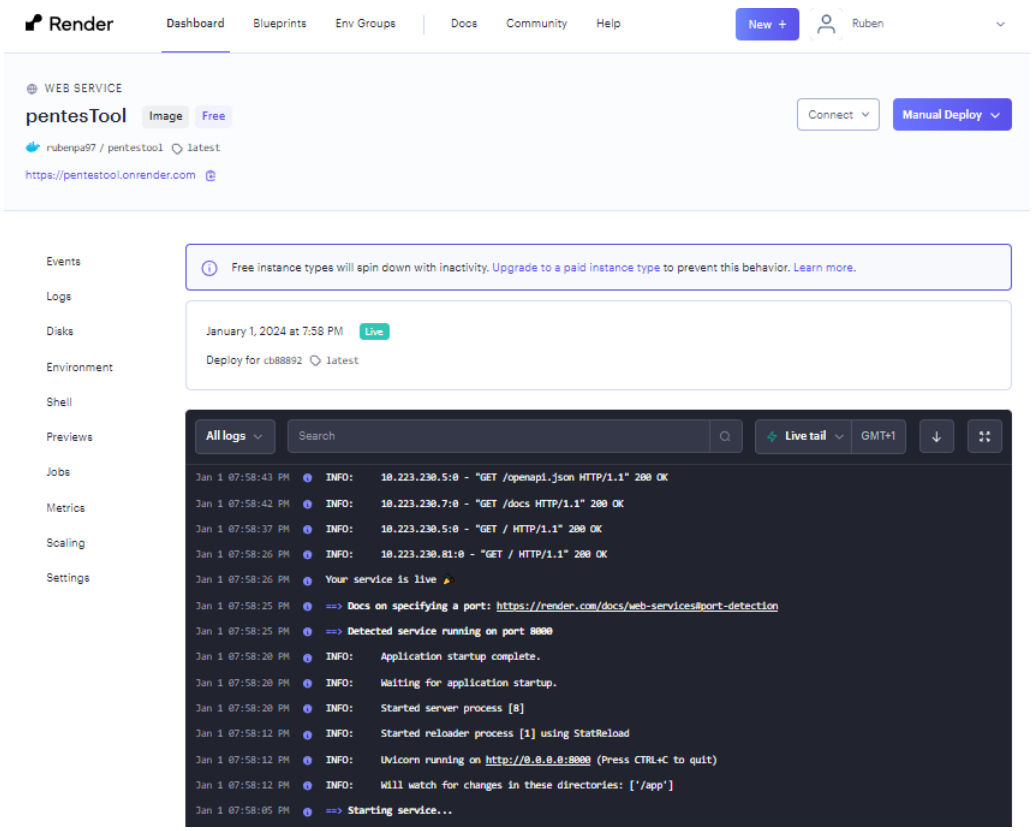

Figura 18: Render

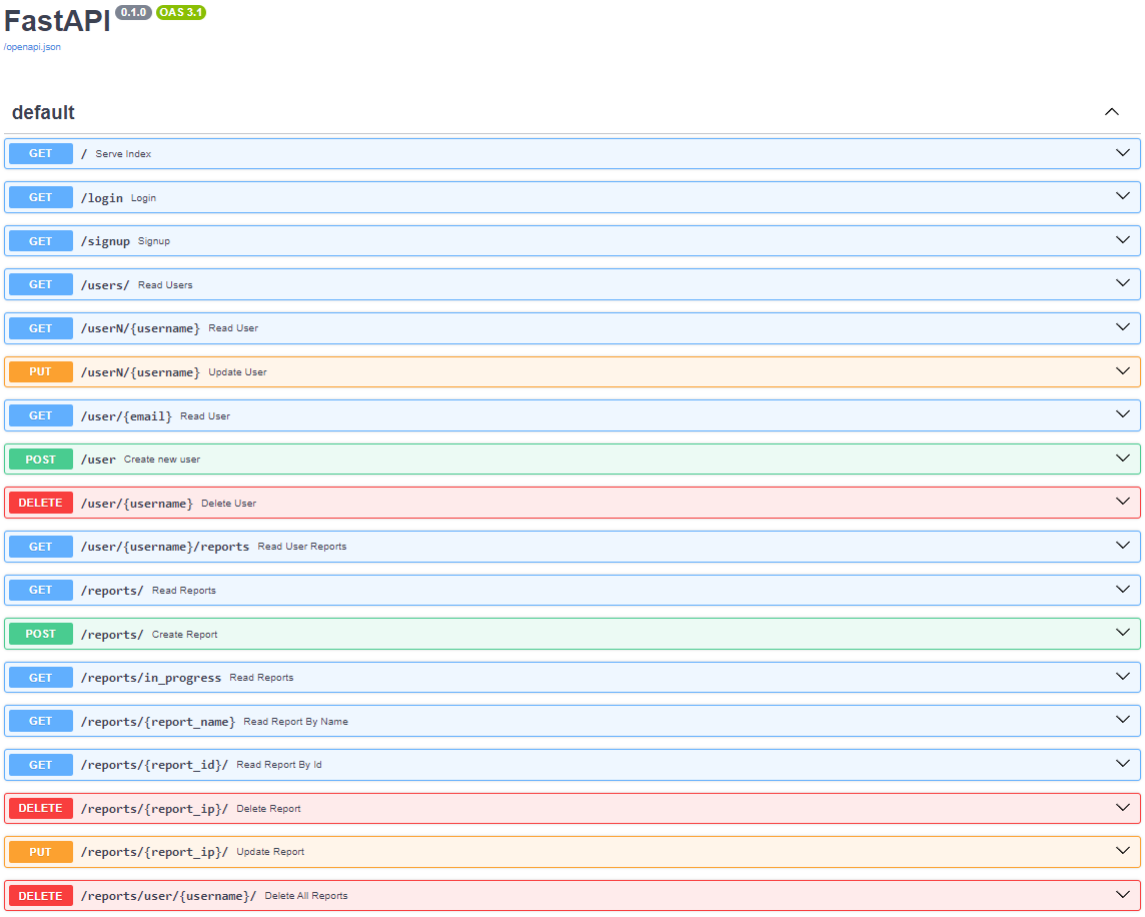

Figura 19: Backend - FastAPI

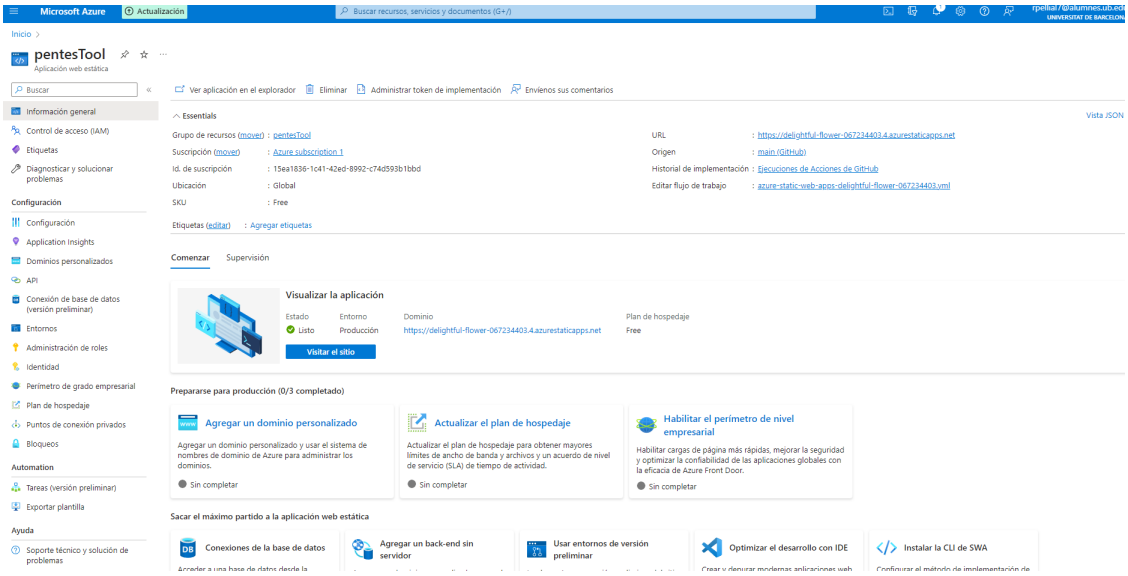

Figura 20: Microsoft Azure

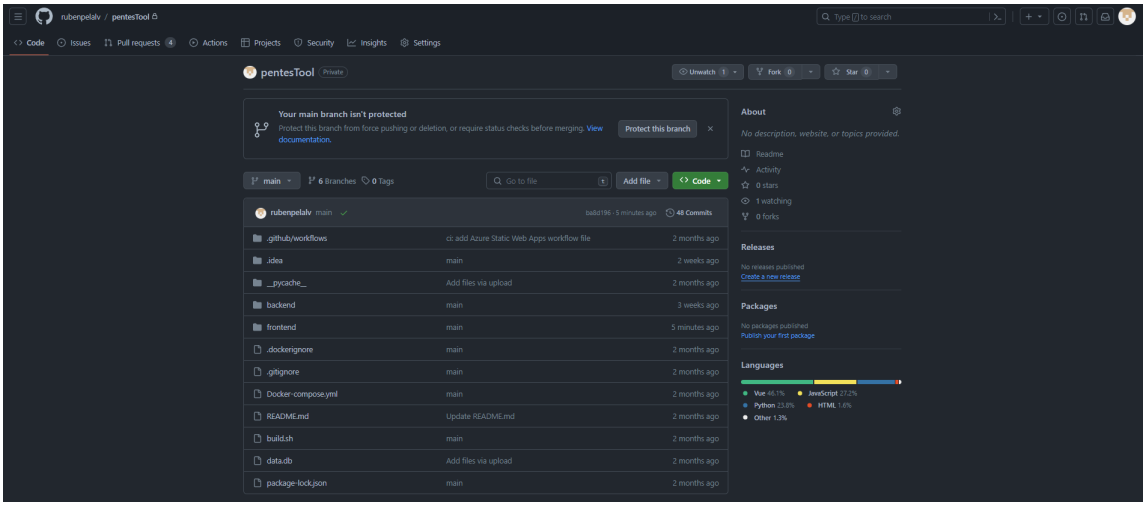

Figura 21: Github

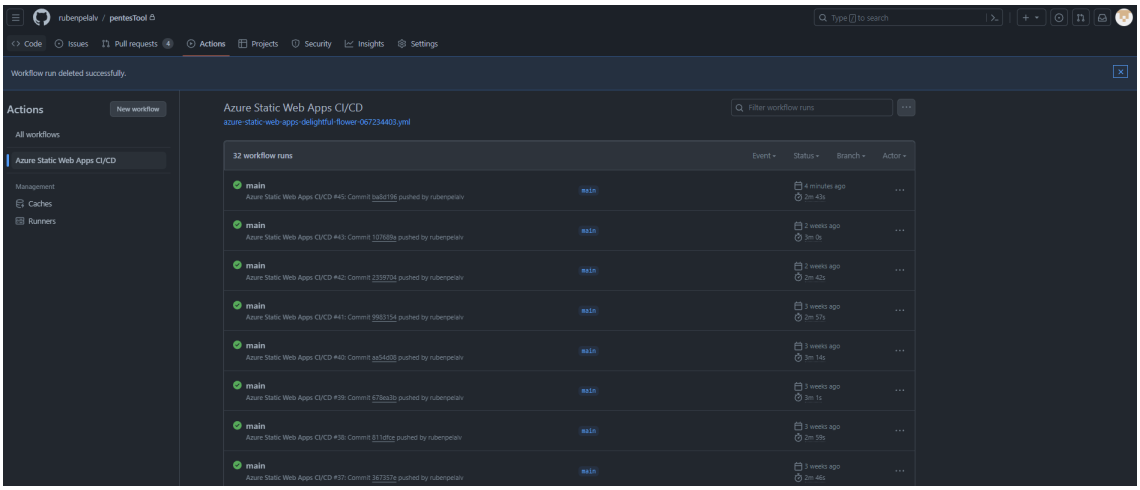

Figura 22: Github Actions Workflow Azure CI/CD

B. Scripts

```bash
#!/bin/bash

name=$1
IP_URL=$2
ip=$2
user_id=$3

timestamp=$(date +"%Y%m%d_%H%M%S")
name2="${name}_${timestamp}.json"

bdd="https://pentestool.onrender.com/reports/$ip/"

script_dir=$(dirname "$0")
output_dir="$script_dir/.."

if ! ping -c 1 "$ip" &> /dev/null; then
    echo "La dirección IP $ip no responde al ping. Finalizando el script."

    data=$(cat <<EOF
    {
      "user_id": "$user_id",
      "url": "",
      "name": "$name",
      "ip": "$ip",
      "status": "No Connection",
      "vuln": ""
    }
EOF
    )

    # Haciendo la solicitud PUT con curl
    curl -X PUT \
      "$bdd" \
      -H "accept: application/json" \
      -H "Content-Type: application/json" \
      -d "$data"

    exit 1
fi

if [ -f "$output_dir/archivos/informe.md" ]; then
    sudo rm "$output_dir/archivos/informe.md"
fi

sudo rm -f "$output_dir/archivos/txt"/*
sudo rm -f "$output_dir/archivos/xml"/*

is_ip() {
```

Figura 23: Parte del script all.sh

```python
#!/usr/bin/python3
# -*- coding: utf-8 -*-
#
# check_sslscan - check server against Qualys sslscan

from argparse import ArgumentParser

import ssl
from urllib.parse import urlencode, quote
from urllib.request import Request, urlopen, ProxyHandler, build_opener, install_opener
from urllib.error import URLError, HTTPError

import sys
import json
import signal
import time

__version__ = "1.3.1"

scores = {
    'A+': 7, 'A':  6, 'A-': 5, 'B':  4, 'C':  3,
    'D':  2, 'E':  1, 'F':  0, 'T':  0, 'M':  0
}

class SSLScan:

    def __init__(self, host, publish=False, maxAge=0, nocache=False, trust=True, proxy=None, debug=False):
        self.uri = "https://api.ssllabs.com/api/v2/analyze?host=%s&" % host
        self.args = ['publish=%s' % ("off","on")[publish]]
        self.trust = trust
        self.debug = debug
        if proxy is not None:
            proxy_opener = build_opener(ProxyHandler({"https":proxy}))
            install_opener(proxy_opener)

        if nocache: self.args += ['startNew=on']
        else:
            self.args += ['fromCache=on']
            if maxAge > 0: self.args += ['maxAge=%d' % maxAge]

        self.headers = {
            'Accept': "application/json",
            'User-Agent': f"nagios/check_sslscan {__version__}"
        }

    def get_results(self):
        return self.poll(True)
```

Figura 24: Parte del script check_sslscan.py

```bash
#!/bin/bash

IP=$1

script_dir=$(dirname "$0")
output_dir="$script_dir/.."

# Realizar la solicitud a ipinfo.io con curl y guardar la respuesta en un archivo temporal
temp_file=$(mktemp)
curl -s "ipinfo.io/$IP" > "$temp_file"

# Verificar si existen campos relevantes para la sección "### Geolocation"
if jq -e '.city, .region, .country, .loc, .timezone' "$temp_file" >/dev/null; then
    echo "### Geolocation" | sudo tee "$output_dir/archivos/txt/geolocation.txt" > /dev/null

# Extraer la información relevante y guardarla en un archivo de texto
jq -r '. | if has("loc") then .loc |= sub(","; "-") else . end | "city: \(.city // empty)\nregion: \(.region // empty)\ncountry: \(.country // empty)\nloc: \(.loc // empty)\ntimezone: \(.timezone // empty)"' "$temp_file" |
 sed '/^[[:space:]]*$/d' | sudo tee -a "$output_dir/archivos/txt/geolocation.txt" > /dev/null
fi

# Verificar si existe el campo "hostname" para la sección "### Host"
if jq -e '.hostname' "$temp_file" >/dev/null; then
    echo -e "\n### Host" | sudo tee -a "$output_dir/archivos/txt/geolocation.txt" > /dev/null
    jq -r '.hostname' "$temp_file" | sudo tee -a "$output_dir/archivos/txt/geolocation.txt" > /dev/null
fi

# Eliminar el archivo temporal
rm "$temp_file"
```

Figura 25: Script geo.sh

```bash
#!/bin/bash

IP=$1
KEY="6cacc9bd6b6e41bd4513c01582e6b1a28dcaa2cf"

# Extraer el dominio principal de la cadena IP
DOMAIN=$(echo "$IP" | awk -F'.' '{print $(NF-1)"."$NF}')

# Directorio de salida
script_dir=$(dirname "$0")
output_dir="$script_dir/.."

# Realizar la solicitud a la API de Hunter.io para obtener direcciones de correo electrónico y detalles adicionales
response=$(curl -s "https://api.hunter.io/v2/domain-search?domain=$DOMAIN&api_key=$KEY")

# Verificar si hay resultados antes de guardar en el archivo
results=$(echo "$response" | jq -r '.meta.results')
if [ "$results" -gt 0 ]; then
    # Guardar correos electrónicos en un archivo de texto
    echo "### Emails" >> "$output_dir/archivos/txt/hunter_emails.txt"
    echo "Results $results" >> "$output_dir/archivos/txt/hunter_emails.txt"
    echo "$(echo "$response" | jq -r '.data.emails[].value')" >> "$output_dir/archivos/txt/hunter_emails.txt"
fi

# Verificar si .data.organization existe antes de agregar al archivo
organization=$(echo "$response" | jq -r '.data.organization')
if [ ! -z "$organization" ]; then
    echo -e "\n### Organization" >> "$output_dir/archivos/txt/hunter_emails.txt"
    echo "$organization" >> "$output_dir/archivos/txt/hunter_emails.txt"
fi

# Verificar si .data.industry existe antes de agregar al archivo
industry=$(echo "$response" | jq -r '.data.industry')
if [ ! -z "$industry" ]; then
    echo -e "\n### Industry" >> "$output_dir/archivos/txt/hunter_emails.txt"
    echo "$industry" >> "$output_dir/archivos/txt/hunter_emails.txt"
fi
```

Figura 26: Script hunter.sh

```bash
#!/bin/bash

script_dir=$(dirname "$0")
output_dir="$script_dir/.."

# Nombre del archivo final
json_file="$1"
echo $json_file

create_json() {
    local md_file="$output_dir/archivos/informe.md"
    # Array asociativo en bash
    declare -A data

    # Leer el archivo MD línea por línea
    while IFS= read -r line || [ -n "$line" ]; do
        # Extracción de claves y valores
        if [[ $line =~ ^\#\#\# ]]; then
            key=$(echo "$line" | sed 's/^### \(.*\)/\1/')
        else
            # Mantener los espacios en los valores
            value=$(echo "$line")
            # Agregar valores al array asociativo
            if [ -n "$key" ] && [ -n "$value" ]; then
                if [ -z "${data[$key]}" ]; then
                    data[$key]="$value"
                else
                    data[$key]+=",$value"
                fi
            fi
        fi
    done < "$md_file"

    # Convertir datos a formato JSON
    echo "$output_dir/archivos/$json_file"
    printf '[\n' > "$output_dir/archivos/$json_file"
    for k in "${!data[@]}"; do
        if [[ ${data[$k]} == *","* ]]; then
            printf '{"%s":["%s"]},\n' "$k" "${data[$k]}"
        else
            printf '{"%s":"%s"},\n' "$k" "${data[$k]}"
        fi
    done | sed '$s/,$//' >> "$output_dir/archivos/$json_file"
    printf ']\n' >> "$output_dir/archivos/$json_file"

    echo "Archivo JSON creado: $output_dir/archivos/$json_file"

```

Figura 27: Parte del script json.sh

```bash
#!/bin/bash

while true; do
    # URL de la página web
    url="https://pentestool.onrender.com/reports/"

    # Realizar la solicitud GET a la página web y guardar la respuesta en un archivo temporal
    curl -s "$url" -o temporal.json

    # Verificar si la solicitud fue exitosa (código de estado 200)
    if [ $? -eq 0 ]; then
        # Verificar si el archivo temporal tiene contenido
        if [ -s temporal.json ]; then
            # Obtener el nombre de los archivos de salida
            output_file="datos_pagina.json"
            log_file="log.txt"

            # Iterar sobre cada objeto en el JSON
            while IFS= read -r line; do
                name=$(jq -r '.name' <<< "$line")
                ip=$(jq -r '.ip' <<< "$line")
                user_id=$(jq -r '.user_id' <<< "$line")

                if [ ! -z "$name" ] && [ ! -z "$ip" ]; then
                    # Simulación: Imprimir el comando que se ejecutaría
                    sudo /home/gargoly/TFG/pentesTool/auto/scripts/all.sh "$name" "$ip" "$user_id">> "$log_file" 2>&1
                fi
            done < <(jq -c '.[]' temporal.json)

            echo "Proceso completado. Se ha simulado la ejecución de los comandos."
        else
            echo "No se obtuvo contenido de la página web."
        fi

        # Eliminar el archivo temporal
        rm temporal.json
    else
        echo "No se pudo obtener la página. Error de solicitud."
    fi

    # Esperar 5 segundos antes de continuar con el siguiente ciclo
    sleep 5
done
```

Figura 28: Script bdd.sh

```javascript
const { createClient } = require('@supabase/supabase-js');
const fs = require('fs');

// Configura tu cliente de Supabase con las credenciales de tu proyecto
const supabaseUrl = 'https://kvzmpyiwmlbipvqorazv.supabase.co';
const supabaseKey = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6Imt2em1weWl3bWxiaXB2cW9yYXp2Iiwicm9sZSI6InNlcnZpY2Vfcm9sZSIsImlhdCI6MTcwMDU3ODUzNCwiZXhwIjoyMDE2MTU0NTM0fQ.D3LiPHr-UoPUSagAtxpkhfle35Cn39K2ZTHJtPm6wFU';
const supabase = createClient(supabaseUrl, supabaseKey);

const fileName = process.argv[2];

// Ruta donde quieres guardar el archivo en Supabase Storage
const storagePath = 'https://kvzmpyiwmlbipvqorazv.supabase.co/storage/v1/object/public/assets/'; // La ruta donde deseas almacenar el archivo

// Ruta local del archivo que deseas subir
const localFilePath = `/home/gargoly/TFG/pentesTool/auto/archivos/${fileName}`;

// Lee el archivo local y súbalo a Supabase Storage
const uploadFileToSupabase = async () => {
  try {
    const fileData = fs.readFileSync(localFilePath);

    // Sube el archivo a Supabase Storage
    const { data, error } = await supabase.storage
      .from('assets')
      .upload(`${fileName}`, fileData);

    if (error) {
      console.error('Error al subir el archivo:', error);
      return;
    }

    console.log('Archivo subido exitosamente:', data);
  } catch (err) {
    console.error('Error:', err.message);
  }
};

uploadFileToSupabase();
```

Figura 29: Script storage.sh

C. Logo

Figura 30: Logo de pentesTool

D. Glosario

- IP: Una dirección única que identifica un dispositivo en una red utilizando el protocolo de Internet (IP).
- URL (Uniform Resource Locator): También llamado link, la dirección que identifica la ubicación de un recurso en internet. Por ejemplo, "https://www.ejemplo.com".
- Línea de comandos, terminal o cmd: Interfaz de texto utilizada para dar instrucciones a una computadora mediante comandos escritos en texto.
- DNS (Domain Name System): Un sistema que traduce los nombres de dominio legibles por humanos en direcciones IP comprensibles por las máquinas.
- Certificado SSL/TLS: Un archivo que garantiza la seguridad de una conexión cifrada entre un servidor web y un navegador, proporcionando autenticación y privacidad.
- ISP (Internet Service Provider): Una empresa que ofrece servicios de acceso a internet a los usuarios.
- Análisis de tráfico: La evaluación y estudio de los datos de red para comprender patrones, identificar problemas de seguridad o mejorar el rendimiento.
- Exploit: Un programa, fragmento de código o técnica que aprovecha una vulnerabilidad en un sistema para lograr un comportamiento no deseado.
- Kali Linux: Una distribución de Linux especializada en seguridad informática y pruebas de penetración.
- XSS (Cross-Site Scripting): Una vulnerabilidad que permite a los atacantes inyectar scripts maliciosos en páginas web visitadas por otros usuarios.
- HTTP (Hypertext Transfer Protocol): Un protocolo de comunicación utilizado para la transferencia de datos en la web.
- HTTPS (Hypertext Transfer Protocol Secure): Una versión segura de HTTP que utiliza cifrado para proteger la transferencia de datos.
- Clickjacking: Una técnica de ataque donde los usuarios son engañados para hacer clic en algo diferente de lo que perciben, generalmente ocultando elementos maliciosos sobre elementos legítimos en una página web.