Escaneo de vulnerabilidades en servidores web: una solución móvil

Alex Soriano Faiges

ScanWebServer
Ciberseguridad

Tutor:
Prof. Joan Caparrós Ramírez
Consultor:
Dr. Andreu Pere Isern Deyà

13/06/2023

Universitat Oberta
de Catalunya

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Vulnerabilidades en redes IP: una solución móvil*
Nombre del autor:	*Alex Soriano Faiges*
Nombre del consultor/a:	*Joan Caparrós Ramírez*
Nombre del PRA:	*Andreu Pere Isern Deyà*
Fecha de entrega (mm/aaaa):	*06/2023*
Titulación o programa:	*Máster universitario Online de Ciberseguridad y Privacidad*
Área del Trabajo Final:	*Ciberseguridad*
Idioma del trabajo:	*Castellano*
Palabras clave	*Aplicación móvil, Vulnerabilidades, Ciberseguridad*

Resumen del Trabajo

En el marco de este proyecto, se ha desarrollado una aplicación móvil modular y facilidad de usar. Esta herramienta permite llevar a cabo análisis de vulnerabilidades en servidores web utilizando como punto de partida una dirección IP o una URL. La aplicación ofrece distintos tipos de escaneo, cada uno de los cuales brinda información detallada y permite al usuario configurar los parámetros según sus necesidades. Finalmente, los resultados obtenidos se presentan de manera clara y concisa al usuario, quien tiene la posibilidad de exportarlos si así lo requiere.

Abstract

As part of this project, a modular and user-friendly mobile application has been developed. This tool allows for vulnerability analysis on web servers, using either an IP address or a URL as a starting point. The application provides various scan types, each of which offers detailed information and allows users to configure parameters according to their needs. Lastly, the obtained results are presented clearly and concisely to the user, who has the option to export them if necessary.

Agradecimientos

Quiero aprovechar este espacio para expresar mi más sincero agradecimiento a todas las personas que han sido parte fundamental en el desarrollo de este proyecto.

En primer lugar, agradecer a mi familia y amigos, por estar ahí mostrando su apoyo durante todo el desarrollo del proyecto, aportando sus opiniones e interesándose por mi trabajo.

Mencionar también a dos grandes grupos de amigos "MDT" y "Alsa" que han estado durante estos últimos años y en este último proyecto de mi máster, haciendo más ameno mi camino aportando su compañía y amistad.

Por último, pero no menos importante, deseo manifestar mi más profundo y genuino agradecimiento al Profesor Joan Caparrós Ramírez, quien ha sido la persona clave en el desarrollo de este proyecto y ha sido fundamental en mi motivación para alcanzar los objetivos planteados.

Índice

Índice de figuras

1. Introducción

En un mundo cada vez más interconectado, la seguridad de los sistemas informáticos se ha convertido en un tema crítico para empresas y organizaciones de todo tipo. Los ataques informáticos están a la orden del día, lo que hace que cada vez sea más importante protegerse, debido al gran riesgo que implica la violación de la confidencialidad, integridad y/o disponibilidad de los datos. Es por eso que la identificación y corrección de vulnerabilidades en los sistemas es un punto clave para la protección de los datos y la privacidad de los usuarios y clientes. [Petrosyan, 2023]

Por esta rázon, el presente proyecto tiene como objetivo desarrollar una aplicación móvil que automatice el análisis de vulnerabilidades de un servidor a partir de su dirección IP. Utilizando herramientas open-source y otras adicionales que se desarrollarán por cuenta propia si se considera necesario para enriquecer el informe que se generará de los resultados.

La portabilidad y la facilidad de uso son dos grandes ventajas de la aplicación móvil. A diferencia de las herramientas actuales que requieren una configuración compleja y/o una instalación en un equipo específico, la aplicación móvil puede utilizarse en cualquier lugar y en cualquier momento a través de un dispositivo móvil Android. Esto permitirá a los profesionales de la seguridad informática llevar a cabo pruebas de penetración y detectar posibles vulnerabilidades de manera más rápida y conveniente.

El resultado del análisis de vulnerabilidades se presentará con informe detallado que incluirá todas las vulnerabilidades encontradas en el servidor. Esta aplicación móvil será una herramienta útil para los profesionales de la seguridad informática, permitiéndoles llevar a cabo pruebas de penetración y detectar posibles vulnerabilidades de manera eficiente y efectiva. Además, la aplicación móvil puede ser actualizada y mejorada de manera constante para incorporar nuevas técnicas y herramientas para el análisis de vulnerabilidades.

En resumen, el presente proyecto no solo pretende ser una herramienta valiosa para los profesionales de la seguridad informática, sino que también pueda ofrecer ventajas significativas

en términos de portabilidad, facilidad de uso, generación de informes detallados y actualización constante. Este proyecto tiene como objetivo contribuir al desarrollo de herramientas que permitan a los profesionales de la seguridad proteger mejor los sistemas informáticos de posibles ataques.

2. Objetivos

Como propósito principal de este proyecto se quiere desarrollar una aplicación móvil modular, fácil de usar y segura, que permita realizar análisis de vulnerabilidades en servidores a partir de su dirección IP o una URL concreta, con el fin de contribuir a la seguridad de los sistemas informáticos. Para lograr esta meta, se abordarán varios objetivos específicos que se describen a continuación.

Se quiere identificar las herramientas open-source existentes que se puedan utilizar en el proceso de análisis de vulnerabilidades. Una vez listadas, se busca realizar una investigación sobre cada una de ellas para dictaminar cuáles pueden aportar información relevante al proyecto. Esto permitirá aprovechar al máximo las herramientas ya disponibles y no reinventar la rueda.

En lo que a modular se refiere, se busca crear y/o utilizar herramientas propias de la aplicación que puedan utilizarse de la misma forma que las herramientas open-source, creando diferentes módulos. De este modo, la aplicación no dependerá únicamente de servicios externos y ofrecerá diferentes formas de obtener información, para obtener resultados más detallados y completos.

Una vez marcados los objetivos más funcionales, se tiene la intención de diseñar e implementar una interfaz de usuario amigable e intuitiva para la aplicación móvil, que permita configurar y ejecutar las herramientas de análisis de vulnerabilidades, lo que facilitará su uso por parte de profesionales y no profesionales.

Además, como subpropósito del objetivo anterior, se requiere procesar los resultados de las herramientas de análisis de vulnerabilidades y presentarlos en un informe detallado. Esto permitirá una fácil comprensión de los resultados obtenidos y facilitará la toma de decisiones.

Como ultimo punto adicional, cuando todo el proyecto esté realizado, se pretende establecer protocolos de seguridad y privacidad para garantizar que la información confidencial del usuario y del servidor se proteja adecuadamente durante el proceso de análisis, lo que contribuirá a una gestión responsable de los datos.

3. Metodología

En este punto de la metodología, se definen las herramientas y técnicas que se emplearán en la gestión de las diversas tareas que se presentarán en el proyecto de desarrollo enfocado en la seguridad informática.

En el desarrollo del aplicativo móvil, se cuenta con un nivel de conocimiento previo en la creación de aplicaciones móviles, adquirido a través del trabajo de final de grado. Este conocimiento previo permitirá enfocar los esfuerzos en la funcionalidad del aplicativo, en lugar de en la curva de aprendizaje del desarrollo de aplicaciones móviles.

Para gestionar eficazmente el proyecto de desarrollo, se ha seleccionado la metodología Scrum, siendo un marco de trabajo ágil que se enfoca en la colaboración y la entrega incremental, lo que permite una mayor adaptabilidad a los cambios en los requisitos del proyecto. La metodología Scrum divide el proyecto en Sprints, que son períodos de tiempo predefinidos, que en este de 2 semanas, durante los cuales se ha de completar un conjunto específico de tareas. Al final de cada Sprint, se realiza una revisión para evaluar el progreso y realizar los ajustes necesarios para el siguiente Sprint, como la corrección de errores o bugs y la selección de las nuevas tareas a completar.

Para la gestión de tareas y seguimiento del progreso del proyecto en cada Sprint, se empleará Trello, que es una herramienta de gestión de proyectos en línea, que permite organizar y visualizar las tareas y su estado de avance, así como asignar responsabilidades y plazos a cada tarea. Trello será una herramienta clave para mantener el enfoque en la entrega incremental y el seguimiento del progreso del proyecto.

Para la elaboración del informe se utilizará LaTeX, una herramienta adecuada para la creación de documentos científicos y técnicos. LaTeX proporciona una gran cantidad de funciones para la creación de fórmulas matemáticas, tablas, gráficos y referencias bibliográficas, lo que resulta muy útil en la redacción de informes técnicos. Mientras que para la elaboración del aplicativo se utilizara Android Studio, plataforma estándar para el desarrollo de aplicaciones Android.

El foco principal de la aplicación se centra en una parte experimentada de la sociedad en la seguridad informática. Por lo tanto, se espera que los usuarios tengan conocimientos básicos en informática y seguridad informática para poder aprovechar al máximo la funcionalidad de la aplicación. Es importante destacar que la aplicación no será de utilidad para personas sin conocimientos previos en informática y seguridad informática.

Como definición del proyecto, la aplicación debe disponer al usuario de toda la información que se pueda obtener desde la propia red exterior, recalcando los puntos que podrían ser explotados como vulnerabilidades. Aunque, en ningún caso la aplicación realizara una prueba de intrusión en estos puntos para clarificar si son o no una vulnerabilidad real.

4. Listado de las tareas

En este punto se enumera el orden lógico de las tareas globales necesarias para llevar a cabo el proyecto de desarrollo, incluyendo una breve descripción de cada tarea y la cantidad de horas estimadas que se requerirán para su realización.

A continuación se listan las 11 tareas que llevan a cabo en el proyecto de desarrollo, las cuales pueden dividirse en subtareas adicionales:

1. **Redacción del Plan de trabajo (30 horas)**: Se trata de la elaboración de un documento detallado que establece los objetivos, la descripción de la metodología y estrategias a seguir durante todo el proyecto de desarrollo de la aplicación móvil enfocada en la seguridad informática.

2. **Búsqueda de vulnerabilidades en Servidores (10 horas)**: Esta tarea se enfoca en la identificación de los diferentes tipos de vulnerabilidades en los servidores para establecer los posibles escáneres de la aplicación.

3. **Búsqueda de aplicaciones open-source y herramientas propias (20 horas)**: Se busca en la red aplicaciones y herramientas que sean útiles para el desarrollo de la aplicación y se identifican las que se puedan adaptar a las necesidades del proyecto.

4. **Diseño de la interfaz de usuario y los flujos de navegación (15 horas)**: Esta tarea implica la creación de un diseño visual atractivo e intuitivo para los usuarios y el establecimiento de los flujos de navegación para facilitar la interacción.

5. **Desarrollo de la aplicación en sí misma (30 horas)**: Se trata de la codificación de la aplicación móvil utilizando las herramientas y recursos seleccionados previamente. El objetivo de esta tarea es disponer de todas las pantallas y flujos de navegación correctamente diseñados para poder dar paso a añadir las funcionalidades.

6. **Desarrollo de herramientas adicionales (25 horas)**: Esta tarea implica la creación de herramientas que puedan realizar la detección de vulnerabilidades por parte de la aplicación, como por ejemplo un escaneo de puertos o detección del sistema operativo de la víctima.

7. **Inserción de las funcionalidades open-source a la aplicación (40 horas)**: Se integran las herramientas y aplicaciones open-source seleccionadas previamente a la aplicación móvil desarrollada.

    a) Pruebas de las diferentes herramientas open-source encontradas (20 horas).

    b) Comparación de los resultados obtenidos para poder seleccionar las más efectivas. (10 horas).

    c) Implementación de las funcionalidades en la aplicación (10 horas).

8. **Pruebas de calidad y fiabilidad (60 horas)**: Se divide en las siguientes subtareas:

    a) Búsqueda de servidores web vulnerables (10 horas).

    b) Prueba de herramientas propias (10 horas).

    c) Prueba de las herramientas open-source (10 horas).

    d) Corrección de errores y problemas encontrados. (30 horas).

9. **Diseño de los resultados producidos por la aplicación (20 horas)**: Se establece la forma en que se presentarán los resultados de la aplicación para los usuarios finales.

10. **Creación del informe detallado del proyecto (40 horas)**: Se redacta un informe detallado que documenta todo el proceso de desarrollo de la aplicación y sus resultados.

    a) Redacción de la implementación (10 horas).

    b) Redacción de las Conclusiones y líneas a futuro (10 horas).

    c) Realización de ajustes y correcciones en la memoria (20 horas).

11. **Establecimiento de protocolos de seguridad y privacidad (10 horas)**: Esta tarea se realizará en caso de que la planificación se haya ejecutado sin errores y no se hayan utilizado más horas de las previstas en otras tareas. Su objetivo es establecer los protocolos y medidas de seguridad necesarios para proteger la privacidad y seguridad de las comunicaciones. Además, se incluirán los detalles de estos protocolos en el informe final del proyecto.

5. Planificación temporal

La planificación se encuentra representada en un diagrama de Gantt que detalla las tareas mencionadas previamente y su respectivo período de tiempo. Cada tarea se encuentra asignada a un periodo de tiempo, representado por un número que indica las horas estimadas para su realización. Aunque se espera que las tareas sean realizadas en orden numérico, algunas de ellas podrían ser llevadas a cabo de manera simultánea para optimizar el tiempo y recursos.

Como se puede ver en la siguiente imagen la planificación consta de 300 horas como máximo, con solo un periodo de descanso definido entre finales de marzo y principios de abril.

Nombre del proyecto **App móvil - Analizador devulnerabilidades de servidor web**

Fecha de Inicio **01/03/2023**

Escala de tiempo de Gantt

	TAREA	INICIO	FIN	01/03/2023	05/03/2023	09/03/2023	13/03/2023	17/03/2023	21/03/2023	25/03/2023	29/03/2023	02/04/2023	06/04/2023	10/04/2023	14/04/2023	18/04/2023	22/04/2023	26/04/2023	30/04/2023	04/05/2023	08/05/2023	12/05/2023	16/05/2023	20/05/2023	24/05/2023	28/05/2023	01/06/2023	05/06/2023	09/06/2023	13/06/2023
1	Redacción del Plan de trabajo	01/03/2023	13/03/2023																											
2	Búsqueda de vulnerabilidades en servidores	13/03/2023	16/03/2023																											
3	Búsqueda de aplicaciones open-source	17/03/2023	26/03/2023																											
4	Diseño de la interfaz y los flujos	09/04/2023	12/04/2023																											
5	Desarrollo de la aplicación	13/04/2023	20/04/2023																											
6	Desarrollo de herramientas adicionales	19/04/2023	26/04/2023																											
7	Inserción herramientas open-source	25/04/2023	06/05/2023																											
8	Pruebas de calidad y fiabilidad	06/05/2023	21/05/2023																											
8.1	Búsqueda de servidores web vulnerables	06/05/2023	08/05/2023																											
8.2	Prueba de herramientas propias	08/05/2023	11/05/2023																											
8.3	Prueba de las herramientas open-source	11/05/2023	14/05/2023																											
8.4	Corrección de errores y problemas encontrados	14/05/2023	21/05/2023																											
9	Diseño de los resultados de la aplicación	22/05/2023	27/05/2023																											
10	Redacción del informe final	27/05/2023	07/06/2023																											
10.1	Redacción de la implementación	27/05/2023	29/05/2023																											
10.2	Redacción de las Conclusiones y líneas a futuro	29/05/2023	02/06/2023																											
10.3	Realización de correciones en la memoria	02/06/2023	07/06/2023																											
11	Establecimiento de protocolos de seguridad	08/06/2023	13/06/2023																											

Figura 1: Diagrama de Gantt

6. Estado del arte

En el ámbito de la seguridad informática, el estado del arte está en constante evolución. La tecnología y las técnicas utilizadas por los hackers también evolucionan constantemente, lo que requiere que las empresas y organizaciones se mantengan actualizadas y adapten sus sistemas y procesos de seguridad.

Actualmente, existe mucho software conocido para realizar escaneos de vulnerabilidades externos, que aplican el escaneo de puertos, la identificación de servicios y de versiones, entre otras cosas. Algunos ejemplos populares podrían ser Nmap, Nessus, OpenVas y Qualys.

Además, existen organizaciones específicamente dedicadas a la seguridad informática. Una de las líderes en seguridad informática es OWASP, que se dedica a identificar y prevenir los riesgos de seguridad en aplicaciones web. Esta organización tiene una lista de los 10 principales riesgos que hay que prevenir para evitar presentar vulnerabilidades. A continuación se presenta un breve resumen:

- Realizar pruebas de seguridad en su aplicación.
- Mantener todos los componentes de la aplicación actualizados.
- Configurar la seguridad de la aplicación correctamente.
- No confiar únicamente en la autenticación y la autorización.
- Proteger los datos confidenciales en todas las etapas.
- Establecer controles de acceso adecuados y revise regularmente los permisos.
- Asegurarse de que todos los errores de seguridad sean registrados y monitoreados.
- Diseñar la aplicación pensando en la seguridad desde el principio.
- Usar criptografía adecuada y actualizada para proteger los datos.
- Ser consciente de las vulnerabilidades y riesgos comunes en las aplicaciones web.

Debido a que hoy en día no es popular el uso de aplicaciones móviles en el campo del escaneo de vulnerabilidades, este proyecto puede cobrar sentido, mejorando el uso de estas herramientas de forma portable y más fácil de usar.

De cara al futuro, con el uso de tecnologías emergentes, como la inteligencia artificial y el aprendizaje automático. Estas tecnologías tienen el potencial de mejorar y adaptarse más rápidamente que nunca.

7. Costes Asociados

En la siguiente tabla se presentan los recursos económicos necesarios para la realización del proyecto. Teniendo en cuenta que el proyecto tendrá una duración aproximada de 3 meses, se ha calculado el gasto correspondiente a los recursos que se pagan mensualmente.

Recursos	Uso	Coste
Ordenador personal	Ordenador donde se redacta la memoria y se realiza la programación del proyecto.	800 €
Conexión a Internet	Salida a internet.	25€/mes x (3 meses)
Teléfono móvil personal	Dispositivo donde se ejecutara el proyecto, para comprobar su funcionamiento.	200 €
Publicación de app Android	Coste para utilizar Google Play Developer.	25 €
Servidor VPS	Servidor que ejecutará las diferentes herramientas open-source. El precio puede variar según el tráfico.	5-30€/mes x (3 meses)
	PRECIO TOTAL:	1.175 €

Figura 2: Tabla de Costes Asociados

El precio total estimado es de uno 1.175 euros, como se puede observar. Aunque el precio, como se ha mencionado en la tabla, puede variar dependiendo del tráfico que genere la aplicación, a la hora de contratar el servidor VPS.

8. Riesgos y Soluciones

Existen múltiples riesgos que pueden surgir durante la realización del proyecto. En esta sección se expondrán los más relevantes, con sus posibles soluciones al respecto.

Uno de los principales riesgos es hipotética falta de herramientas open-source adecuadas para el análisis de vulnerabilidades en servidores. Es posible que las herramientas existentes no sean lo suficientemente precisas, estén desactualizadas o no sean compatibles. En caso de que esto suceda, se optaría por diseñar manualmente todas las funcionalidades necesarias para la detección de vulnerabilidades, complicando así mucho más el desarrollo de la parte técnica.

Otro riesgo se basa en el rendimiento, este puede ser poco eficiente a la hora de realizar el procesamiento de los datos. Provocando que pierda parte de su utilidad. Para minimizar este riesgo, se trataría de reducir las operaciones realizadas para realizar un escaneo de vulnerabilidades, dándole al usuario la posibilidad de elegir la profundidad de este.

El ultimo riesgo determinante seria el tiempo limitado, que puede afectar el desarrollo del proyecto, ya que algunas tareas pueden llevar más tiempo del esperado y afectar el cronograma general. En caso de que esto suceda, se priorizará la funcionalidad básica de todo el proyecto en lugar de enfocarse en un solo punto para maximizar el uso del tiempo disponible y garantizar la finalización del proyecto dentro del plazo establecido.

9. Impacto en sostenibilidad, ético-social y de diversidad del Trabajo

9.0.1. Sostenibilidad

El proyecto no tiene un impacto negativo en aspectos de sostenibilidad medioambiental y/o huella ecológica, incluso teniendo en cuenta todo el proceso de desarrollo, vida útil o eliminación. Este hecho se debe a que la aplicación principalmente solo necesita de software, y por tanto, la legislación al respecto de la sostenibilidad no le afectaría.

En caso de una posible aplicación a gran escala, el uso de un grupo de servidores podría requerir un consumo eléctrico significativo. Sin embargo, la elección de servidores ubicados en territorios que produzcan energía renovable tendría un impacto positivo en la sostenibilidad.

En cuanto a los ODS, se podría argumentar que la aplicación contribuye indirectamente a ODS 9 - Industry, innovation and infrastructure, ya que fomenta el desarrollo de tecnologías de ciberseguridad.

9.0.2. Ético-social

En cuanto a la ética de este proyecto, puede tener impactos ético-sociales y legales significativos, debido a que puede ser usado de manera fraudulenta o en beneficio económico de manera indebida. Puede llegar a poner en riesgo la privacidad de los datos, seguridad y/o propiedad intelectual de sus objetivos, siempre que se use de forma maliciosa.

Por otro lado, puede suponer una gran herramienta para defenderse de usuarios con intenciones fraudulentas, mejorando la seguridad y la protección de los datos y/o propiedad intelectual. Esta dualidad de formas de uso, provoca una preocupación en la realización del proyecto.

En conclusión, es importante tener en cuenta la responsabilidad ética y social de garantizar que el resultado no tenga impactos negativos y que, en cambio, contribuya al bienestar social. Como se define en el ODS 16 - Peace, justice and strong institutions, esta pensada para

ayudar a mejorar el bienestar social, reduciendo la criminalidad y preocupándose por un uso responsable.

9.0.3. Aspectos de género

La aplicación móvil, al enfocarse en la seguridad y privacidad de los datos, está relacionada con la protección de los derechos humanos. La confidencialidad de estos, es un aspecto importante para garantizar la privacidad y protección de la información personal, evitando posibles violaciones de los derechos de las personas. En este sentido, la aplicación puede ser utilizada para detectar vulnerabilidades en sistemas que contengan información confidencial, como hospitales, gobiernos y escuelas. Lo que se relaciona con el ODS 10 - Reduced inequalities al reducir posibles violaciones de los derechos humanos y promover la igualdad en el acceso a la protección de la información personal.

Recalcar que la aplicación puede ser utilizada por cualquier identidad de género, orientación sexual, etnia, religión e ideología, lo que aporta en diversidad y puede ayudar a desarrollar una cultura de inclusión y respeto. Lo que contribuye al ODS 5 - Gender equality.

10. Investigación

En esta sección se detallarán las herramientas y decisiones que se han tomado antes de desarrollar la aplicación móvil, así como las razones por las que se han seleccionado sobre otras opciones disponibles. También se explicarán las características que estas herramientas aportarán al proyecto.

10.1. Implementación nativa vs. híbrida

En primer lugar, se ha tenido que elegir entre desarrollar una aplicación nativa específica para un sistema operativo u optar por una implementación híbrida que permitiera la compatibilidad entre diferentes sistemas operativos.

Aunque la implementación híbrida tiene la ventaja de ser compatible con varios sistemas operativos, reduciendo el tiempo y los costos de producción, también posee algunas desventajas en comparación con la aplicación nativa. Esta última ofrece mayor velocidad y eficiencia, y se adapta al sistema operativo en términos de funcionamiento, además de mantener la relación de aspecto necesaria para una buena calidad de imagen y gráficos.

Por lo tanto, se ha decidido desarrollar una aplicación nativa basándose en su mayor eficiencia y compatibilidad con los recursos que se emplearán en ella. [Vázquez, 2022]

10.2. Sistema operativo

Seguidamente, ha sido necesario escoger el sistema operativo donde estará disponible la aplicación móvil antes que empezar a desarrollarla. Entre Android y iOS se ha acabado optando por Android, debido al gran número de dispositivos en el mercado. Estos representan el 71 % de todos los dispositivos móviles. Lo que significa que la aplicación podrá llegar a una audiencia mucho más amplia y diversa. [Roa, 2023]

Además, se posee de una mayor facilidad para efectuar el desarrollo, gracias a la experiencia obtenida anteriormente en el software de Android Studio en el diseño de aplicaciones móviles.

Como futuro del proyecto, se buscaría desarrollar la aplicación para iOS, debido a que ocupa el 28 % del mercado restante.

10.3. Lenguaje de programación

Al decantarse por Android, la utilización de la herramienta Android Studio es prácticamente obligatoria. Aunque aún queda elegir el lenguaje de programación que se utilizará en el desarrollo, pudiendo ser Java o Kotlin.

Java es uno de los lenguajes de programación más utilizados en el desarrollo de aplicaciones móviles y cuenta con una gran comunidad de desarrolladores, lo que se traduce en una amplia variedad de recursos y herramientas disponibles. Además, el entorno de desarrollo integrado (IDE) de Android Studio está pensado para ser utilizado con Java.

Kotlin, por su parte, ofrece ventajas en términos de sintaxis, ya que el número de líneas se reduce en un 40 %, aumentando también la seguridad. Sin embargo, se ha decidido emplear Java debido a la gran cantidad de recursos que se disponen en la red. Además, es un añadido disponer de experiencia en el uso de este lenguaje.

10.4. Funcionalidades de la aplicación

Se ha considerado la posibilidad de que la aplicación ofrezca diferentes funcionalidades según el tipo de suscripción que tenga el usuario. La versión gratuita permitirá el acceso a funcionalidades básicas implementadas manualmente para obtener información sobre un objetivo concreto, junto con el uso de diferentes APIs gratuitas que brindarán información adicional sobre el objetivo, incluyendo algunas vulnerabilidades. Además, existirá la opción de realizar el escaneo sobre una URL o sobre una IP.

En contraste, la versión de pago ofrecerá todas las características de la versión gratuita de forma ilimitada, mientras que la versión gratuita limitará el número de peticiones por nuevo usuario. Incluso pueden existir características exclusivas para la versión de pago.

10.4.1. Funcionalidades propias

Estas funcionalidades serán programadas manualmente en Java con el objetivo de obtener la mayor cantidad de información posible sobre el objetivo, ya sea una URL o una dirección IP.

En el caso de una URL, se buscará obtener la dirección IP del servidor, los puertos abiertos y el sistema operativo utilizado. Para una dirección IP, también se obtendrán los puertos abiertos y el sistema operativo, y se realizará una búsqueda de IP inversa para determinar si corresponde a algún nombre de dominio.

10.4.2. Herramientas para el escaneo de redes

Estas herramientas estarán disponibles para su uso en la versión gratuita, se pretende que ofrezcan más información sobre el objetivo, como los servicios específicos utilizados en el servidor, la ubicación y otras características.

Después de una larga búsqueda entre muchas de las APIs gratuitas, se han obtenido estas opciones:

- **VirusTotal**: La API de VirusTotal es una herramienta muy efectiva para analizar URLs en busca de amenazas de malware y virus, utilizando una amplia variedad de motores antivirus. Se ha seleccionado esta API sobre otras opciones debido a su alta tasa de detección de amenazas, su gran popularidad, su documentación clara y su uso gratuito.
- **Censys Search API**: Esta API permitirá analizar las diferentes direcciones IP en busca de los diferentes servicios activos y los puertos correspondientes a estos. Existen otras alternativas que son de pago, pero ofrecen las mismas características que Censys.
- **Shodan**: Esta famosa API gratuita nos permitirá obtener muchísimos datos interesantes de una IP, incluyendo la geolocalización y los puertos designados. Ha sido designada frente a otras APIs de geolocalización debido a que ofrece otros datos interesantes sobre la propia IP.

10.4.3. Herramientas de escaneo de vulnerabilidades

Para el escaneo de vulnerabilidades tenemos muchas herramientas, aunque no todas incluyen APIs para poder comunicarse con la aplicación. También existen APIs como servicio de escaneo de vulnerabilidades, aunque estas son de pago. [OWASP, 2023]

Se han buscado una serie de herramientas que puedan instalarse en un servidor para poder realizar peticiones a este, pudiendo realizar los escáneres y obtener los resultados. Aquí se listan algunas de las herramientas que se podrían utilizar en el desarrollo de la aplicación:

- **OWASP ZAP API**: es una herramienta de seguridad de aplicaciones web de código abierto desarrollada por OWASP que incluye una API RESTful. Se utiliza para detectar y explotar vulnerabilidades de seguridad en aplicaciones web, incluyendo inyecciones SQL, ataques XSS, vulnerabilidades de CSRF, entre otras. Debido al gran prestigio en el mundo de la seguridad por parte de la organización OWASP se ha decidido implementar esta herramienta en el proyecto. [OWASP, 2010]
- **Nessus API**: herramienta de escaneo de vulnerabilidades que permite realizar pruebas de seguridad en redes, sistemas y aplicaciones. La herramienta cuenta con una API RESTful, que ejecuta escaneos de puertos, detección de servicios, identificación de vulnerabilidades y un sondeo final que revisa que no hay falsos positivos dentro de la investigación. [KeepCoding, 2023]
- **OpenVAS**: Es un completo escáner de vulnerabilidades que cuenta con más de 50.000 test y datos de vulnerabilidades conocidas. Además, surge a raíz de Nessus mencionado anteriormente. [Vera, 2020]
- **Probely API**: Es la única opción de pago de la lista, debido a que ofrece una buena relación calidad precio. Es un escáner de vulnerabilidades que muestra posibles soluciones a los diferentes problemas encontrados durante el escaneo. No se planea utilizar esta alternativa en este proyecto, al menos que sea estrictamente necesario. [Probely, 2023]

Se ha optado por utilizar solo un escáner para buscar las vulnerabilidades, debido a que utilizar más escáneres afectaría el rendimiento de la aplicación y dificultaría la interpretación

de los resultados por parte del usuario. Se implementará aquel que se ajuste mejor a las funcionalidades de la aplicación y ofrezca mayores facilidades en su implementación.

10.4.4. Herramientas de testeo

Para comprobar el correcto funcionamiento de las distintas funcionalidades de la aplicación una vez haya sido desarrollada, se han seleccionado dos herramientas o páginas web diferentes que presentan vulnerabilidades y permiten realizar escaneos en ellas.

- **Gin and Juice Shop**: es una página web de testeo diseñada para simular una tienda web vulnerable. Está construida utilizando el framework Go llamado "Gin". proporciona una serie de vulnerabilidades comunes que se encuentran en las aplicaciones web. Esta herramienta permite a los desarrolladores probar la seguridad de sus aplicaciones al explorar y explotar las vulnerabilidades. [Atkinson, 2022]
- **Damn Vulnerable Web Application**: es una herramienta de testeo ampliamente utilizada para evaluar la seguridad de aplicaciones web. DVWA está especialmente diseñada para ser vulnerable a una amplia gama de ataques web comunes, como inyección SQL, cross-site scripting (XSS), inclusiones de archivos no seguras, entre otros. [digininja, 2023]

Con estas herramientas obtenemos un entorno controlado y seguro donde realizar las pruebas, para comprobar que los resultados obtenidos sean válidos.

10.5. Esquema general

En el siguiente apartado se explican los diferentes sistemas que actúan en el uso de la aplicación móvil y sus conexiones:

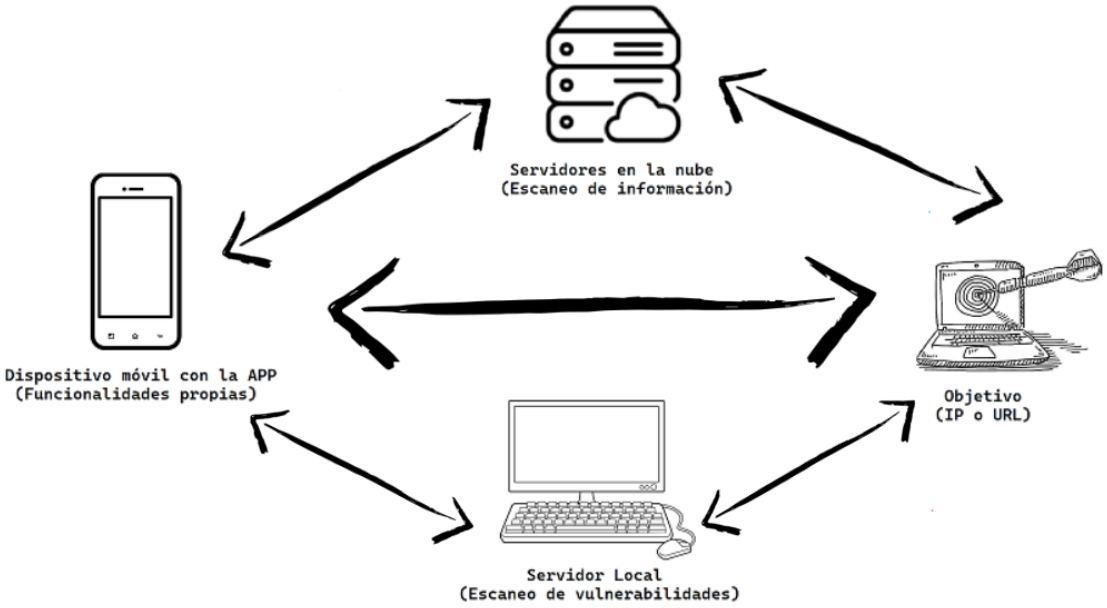

Figura 3: Arquitectura general

Primero que todo tenemos los 4 sistemas:

- **Dispositivo móvil**: Dispositivo en el cual estará instalada la aplicación. En esta se podrán ajustar diferentes configuraciones sobre el escaneo de vulnerabilidades y elegir entre escanear una IP o una URL. Además de realizar escaneos de red para obtener información del objetivo.
- **Servidor Local**: Servidor donde estarán instaladas aquellas herramientas que se enfocarán en el escaneo de vulnerabilidades. La instalación y configuración de estas herramientas tendrá que ser llevada a cabo manualmente.
- **Servidores en la nube**: Son los diferentes Software as a service (SaaS) gratuitos que serán utilizados para obtener información que no necesariamente corresponde a vulnerabilidades.
- **Objetivo**: Es la IP o URL marcada para realizar todos los escaneos.

En cuanto a las comunicaciones, se han establecido diferentes flujos en un orden específico.

En primer lugar, el flujo de comunicación 1 se encarga de que el dispositivo móvil verifique la disponibilidad del objetivo y realice pequeños escaneos de red para recopilar información. Una vez finalizada esta comunicación, el dispositivo móvil enviará peticiones de escaneo a los servidores en la nube que administran los SaaS gratuitos a través del flujo de comunicación 2. A su vez, estos servidores llevarán a cabo los escaneos sobre el objetivo utilizando el flujo de comunicación 3, transmitiendo los resultados de vuelta al dispositivo móvil a través del flujo 2.

Posteriormente, si el usuario desea realizar un escaneo de vulnerabilidades utilizando las diferentes herramientas configuradas en el servidor local, se establecerá la conexión 4 para enviar las solicitudes. Luego, el servidor local utilizará estas herramientas para llevar a cabo el escaneo de vulnerabilidades sobre el objetivo mediante el flujo de comunicación 5. Una vez que el servidor local haya obtenido todos los resultados, los reportará al dispositivo móvil a través de la conexión 4.

Además, a diferencia de la comunicación con los servidores de los SaaS, las herramientas de escaneo de vulnerabilidades requieren más tiempo para obtener los resultados. Por lo tanto, el servidor local mantendrá actualizada la aplicación móvil sobre el estado de los escaneos a través de la conexión 4, permitiendo al usuario seguir el progreso.

10.6. Presentación del Mock-up de la Aplicación Móvil

En el siguiente apartado, se presentan las diversas pantallas que forman parte del flujo de funcionamiento, sus diferentes componentes y la función que realizan. Para su diseño, se ha utilizado la plataforma gratuita NinjaMock [nin, 2012].

El enfoque del diseño se ha centrado en conseguir una facilidad de uso por parte del usuario, además de intentar mantener una estética atractiva y respetar el mapa de calor de accesibilidad:

Figura 4: Mapa de calor

Este mapa de calor representa la facilidad de acceso a diferentes partes de la pantalla, donde el color verde indica la mayor accesibilidad, el naranja representa un nivel intermedio y el rojo indica las áreas que pueden requerir un mayor esfuerzo o atención por parte del usuario.

10.6.1. Pantalla de carga

Esta pantalla inicial solo se presenta en el inicio de la aplicación durante un corto periodo de tiempo, mostrando una barra de carga junto a una imagen que representa el logo de la aplicación móvil.

Figura 5: Pantalla inicial de carga de la aplicación

10.6.2. Pantalla principal

Esta pantalla es el eje central de la aplicación donde se introduce el objetivo a escanear. Ofrece diferentes funcionalidades:

- Un editor de texto para introducir mediante teclado el objetivo a escanear y el botón scan que iniciará el proceso.
- Un icono de ajustes, que al ser pulsado por el usuario se cambiara a la pantalla de ajustes.
- Por último, un apartado de selección abajo a la derecha donde se seleccionara si el objetivo se trata de una URL o una IP.

Figura 6: Pantalla principal

10.6.3. Ajustes

En esta pantalla se mostrarán las diferentes configuraciones que ofrece la aplicación respecto al escaneo. Las funcionalidades que ofrece son:

- Permite seleccionar el modo de la aplicación, si el usuario quiere utilizar la versión de pago o la estándar.
- En la sección de la versión estándar se puede limitar el rango de puertos a escanear.
- En la sección de la versión de pago, se puede activar o desactivar las diferentes herramientas para el escáner de vulnerabilidades. Además, se puede definir el riesgo de las vulnerabilidades que quieren ser obtenidas en el informe final.
- Finalmente, se ofrece un botón para restaurar los ajustes originales y otro para guardarlos.

Figura 7: Pantalla de ajustes

10.6.4. Resultados del escaneo de red

Una vez seleccionado el objetivo, se muestra la pantalla de resultados que proporciona información relevante. Esta pantalla muestra la dirección IP del objetivo, su hostname, el sistema operativo detectado, así como detalles adicionales como la ubicación geográfica y una lista de puertos abiertos con los servicios correspondientes que se encuentran en funcionamiento.

Después de la visualización de los resultados, se presenta un botón que permite llevar a cabo el escaneo de vulnerabilidades.

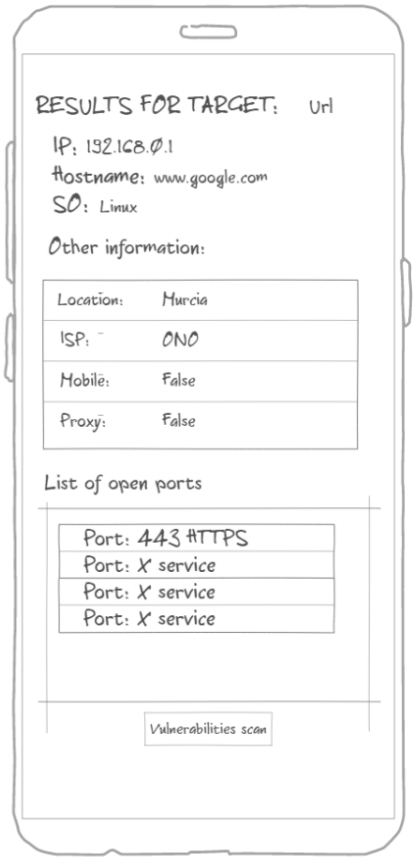

Figura 8: Pantalla de resultados del escaneo a la red

10.6.5. Pantalla de carga en los escaneos

La utilidad de esta pantalla es mostrar información al usuario en tiempo real sobre el progreso del escaneo de vulnerabilidades, mostrando mensajes de información y un porcentaje de avance.

Además, se incluye un botón de cancelar que brinda al usuario la opción de regresar a la pantalla principal en cualquier momento, cancelando el escaneo.

Figura 9: Pantalla de carga

10.6.6. Resultados del escaneo de vulnerabilidades

Una vez finalizado el escaneo de vulnerabilidades, se presenta una lista que contiene las vulnerabilidades detectadas por el escáner utilizado, junto con su correspondiente nivel de riesgo. Cada elemento de esta lista puede ser seleccionado para obtener información adicional sobre esa vulnerabilidad en particular.

Además, se encuentra disponible el botón "Finish", el cual permite al usuario regresar a la pantalla principal, completando el proceso de escaneo.

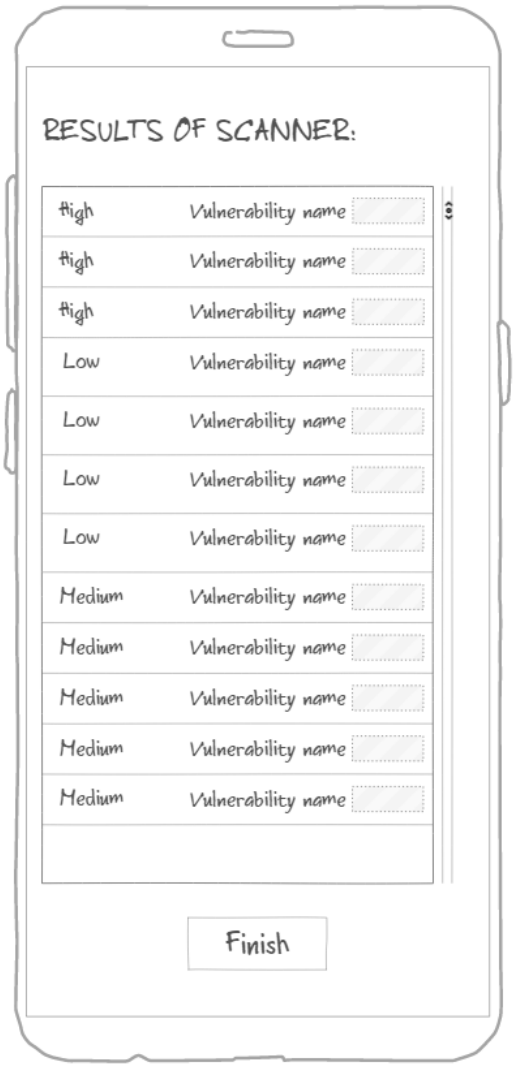

Figura 10: Pantalla de resultados del escaneo de vulnerabilidades

10.6.7. Información vulnerabilidad

Cuando el usuario selecciona una vulnerabilidad específica, se muestra esta pantalla que proporciona información detallada sobre dicha vulnerabilidad dentro de los resultados del escaneo.

En esta pantalla, se muestra el nombre de la vulnerabilidad, el nivel de riesgo asociado y otros detalles relevantes, como una explicación de la vulnerabilidad y sus identificadores (CVE), en caso de existir.

Para cerrar esta pantalla y volver a la vista general de todas las vulnerabilidades, se encuentra disponible un botón con forma de "X" en la parte inferior del cuadro de diálogo.

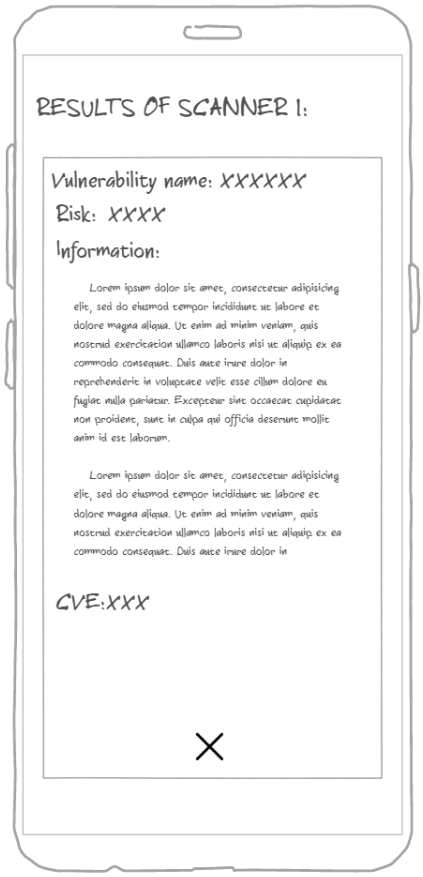

Figura 11: Pantalla de información de una vulnerabilidad

10.7. Flujo de navegación:

En la siguiente imagen se muestran los diferentes flujos que existen dentro de la aplicación y que han sido mencionados durante la explicación de las diferentes pantallas:

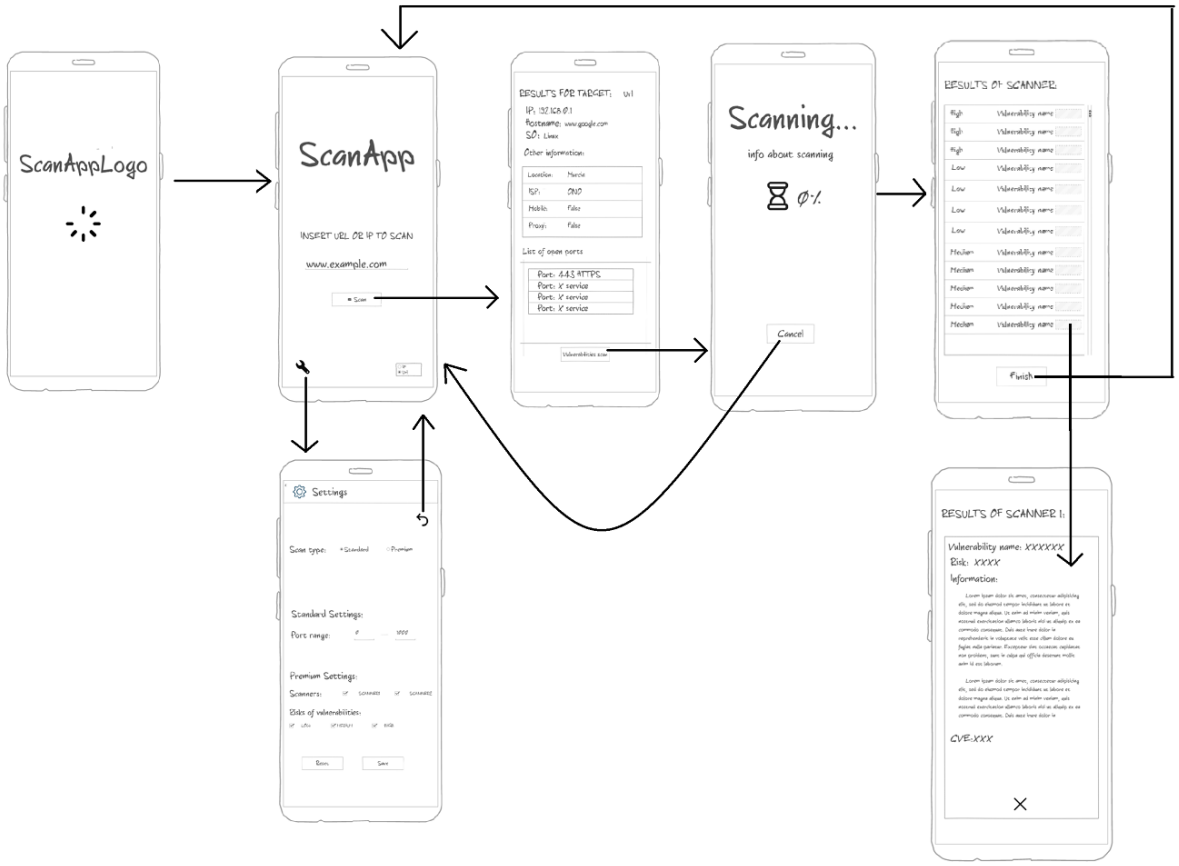

Figura 12: Flujo de navegación

11. Implementación

En esta sección se proporcionará toda la información importante sobre el desarrollo de la aplicación móvil, siguiendo las bases establecidas anteriormente. Se describirán las herramientas seleccionadas para la implementación y se explicará cómo funcionan. Además, se abordarán los riesgos que surgieron durante el proceso y cómo se han corregido. Por último, se mostrará la estructura final del proyecto, y se incluirá un informe de pruebas.

11.1. Implementación de la parte Interna

Inicialmente, se obtiene la dirección IP y/o el hostname correspondiente usando el input introducido por el usuario, ya sea una IP o una URL. Esta funcionalidad usa la librería de InetAddress, la cual representa una dirección IP o un nombre de host, y proporciona métodos para realizar operaciones de resolución de nombres y verificación de conectividad de red. [Java Platform, 2021]

```
InetAddress address = InetAddress.getByName(finalUrl1);
DataHolder.hostAddress = address.getHostAddress();
DataHolder.hostName = address.getHostName();
```

11.1.1. Escáner de puertos

Para realizar esta funcionalidad se utiliza la función runPortScan, que realiza un escaneo de puertos en un rango delimitado por dos variables definidas por el usuario en la pantalla de ajustes. Utiliza la librería java.net.Socket para establecer conexiones con cada puerto y la clase java.util.concurrent para gestionar los hilos y la concurrencia. [Baeldung, 2023]

Los diferentes pasos que sigue son:

1. Crea una lista openPorts para almacenar los puertos abiertos.
2. Crea un ExecutorService con un número fijo de hilos (por defecto 10) para ejecutar las tareas de escaneo.
3. Inicia un bucle que itera desde nbrPortMinToScan hasta nbrPortMaxToScan.

4. En cada iteración, se intenta establecer una conexión con la dirección IP y el puerto actual utilizando un Socket con un tiempo de espera (timeOut).

5. Si la conexión tiene éxito, se agrega el puerto a la lista openPorts.

6. Se cierra el Socket.

7. Al finalizar el escaneo, se espera a que todas las tareas se completen, se recopilan los puertos abiertos en una lista y esta se devuelve.

```
public static List runPortScan(String ip, int nbrPortMinToScan, int nbrPortMaxToScan) {


    ConcurrentLinkedQueue openPorts = new ConcurrentLinkedQueue<>();
    ExecutorService executorService = Executors.newFixedThreadPool(poolSize);
    AtomicInteger port = new AtomicInteger(nbrPortMinToScan);

    while (port.get() < nbrPortMaxToScan) {
        final int currentPort = port.getAndIncrement();
        executorService.submit(() -> {
            try {
                Socket socket = new Socket();
                socket.connect(new InetSocketAddress(ip, currentPort), timeOut);
                socket.close();
                openPorts.add(currentPort);
            } catch (IOException e) {
                //System.err.println(e + " Puerto:" + currentPort);
            }
        });

    }
    executorService.shutdown();
    try {
        executorService.awaitTermination( timeout: 10, TimeUnit.MINUTES);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
    List openPortList = new ArrayList<>();
    while (!openPorts.isEmpty()) {
        openPortList.add(openPorts.poll());
    }

    return openPortList;
}
```

Figura 13: Código del escáner de puertos

11.1.2. Detección del sistema operativo

Para diseñar esta funcionalidad se ha realizado el método getSo, el cual devuelve el sistema operativo aproximado para un nombre de host introducido por parámetro. Utiliza una petición ICMP ("ping") para obtener el valor del campo TTL en la respuesta del host. Luego, se compara el valor obtenido con ciertos valores predefinidos para determinar el sistema operativo aproximado. [Iñigo, 2018]

Los pasos son:

1. Se obtiene la dirección IP correspondiente al nombre de host.
2. Se ejecuta el ping hacia la dirección IP y se obtienen los resultados de la petición.
3. Si una línea de los resultados contiene "ttl=", se extrae el valor TTL de esa línea y se almacena en ttlValue.
4. Se realiza una comparación de ttlValue con diferentes valores predefinidos para determinar el sistema operativo, devolviendo aquel con el cual coincida el valor.

```
public static String getSo(String hostname) {
    int ttlValue = 0;
    String so;
    try {
        InetAddress inetAddress = InetAddress.getByName(hostname);
        Process pingProcess = Runtime.getRuntime().exec( command: "ping -c 1 " + inetAddress.getHostAddress());
        BufferedReader pingOutput = new BufferedReader(new InputStreamReader(pingProcess.getInputStream()));
        String line;

        while ((line = pingOutput.readLine()) != null) {
            if (line.contains("ttl=")) {
                Log.d( tag: "TTL:", line);
                int ttlIndex = line.indexOf("ttl=") + 4;
                int endIndex = line.indexOf( str: " ", ttlIndex);
                ttlValue = Integer.parseInt(line.substring(ttlIndex, endIndex));
                Log.d( tag: "TTL: ", msg: "" + ttlValue);
            }
        }

    } catch (Exception e) {
        ttlValue = 0;
        e.printStackTrace();
    }
    switch (ttlValue) {
        case 64: so = "Linux/FreeBSD/MacOs"; break;
        case 128: so = "Windows"; break;
        case 254: so = "Cisco"; break;
        case 255: so = "Solaris/Aix"; break;
        default: so = "Unknown"; break;
    }
    return so;
}
```

Figura 14: Código del escáner de sistema operativo

11.2. Desafíos de Bloqueo de Respuesta en la Consulta de APIs

Durante el desarrollo de la aplicación, han surgido algunos problemas inesperados que requerían solución. Uno de ellos fue la comunicación con las APIs sin bloquear el hilo principal de la aplicación, lo cual daba como resultado una pantalla en negro hasta obtener una respuesta. Este enfoque no era funcional, ya que impedía al usuario seguir el proceso en pantalla y aumentaba la posibilidad de errores. Para solucionarlo, se utilizó la librería Android Volley, que simplifica la gestión de solicitudes HTTP al proporcionar clases eficientes para realizar peticiones, manejar respuestas y gestionar imágenes en línea. De esta forma, se pueden enviar peticiones de forma asíncrona utilizando otros hilos secundarios y sin bloquear al principal.

Una vez que se logró realizar correctamente las peticiones a las APIs, surgió otro problema que requería esperar el resultado de una solicitud antes de realizar la siguiente. Para abordar esta situación, se implementaron llamadas a las APIs utilizando el objeto StringRequest de la clase Volley. Esta clase nos permite insertar código para manejar tanto la respuesta exitosa como los errores en la solicitud. Para ejecutar estas peticiones en orden se ha utilizado una cola que va ejecutando estas peticiones en orden de entrada. [Develou, 2023]

A continuación se muestra un ejemplo de creación de una StringRequest para una solicitud GET con una URL específica (urlApi). En el código se puede observar cómo se maneja la respuesta de la petición y los errores. Por último, se muestra cómo se agrega esta solicitud a la cola:

```
StringRequest stringRequest = new StringRequest(Request.Method.GET, urlApi,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                textLoading.setText("Received vulnerabilities");
                DataHolder.response = response;
                runOnUiThread(new Runnable() {...});
            }
        }, new Response.ErrorListener() {
    2 usages
    @Override
    public void onErrorResponse(VolleyError error) {
        textLoading.setText("Error getting the results");
    }
});

// Add the request to the RequestQueue.
queue.add(stringRequest);
```

Figura 15: Ejemplo de petición a una Api

11.3. Implementación de las APIs de VirusTotal y CensysSearch

En este apartado, se detallarán las peticiones efectuadas a las APIs y los resultados que estas devuelven en su funcionamiento.

11.3.1. VirusTotal API

Para esta API, se envía una petición POST con la URL que se quiere escanear, insertada en el cuerpo de la petición, y la API Key en los encabezados. Esta petición se efectúa a la URL:

`https://www.virustotal.com/api/v3/urls`

Esta petición inicia el proceso de escaneo de la URL por parte de VirusTotal. Posteriormente, se obtendrán los resultados mediante una petición GET con la API Key en los encabezados y especificando el ID que se ha recibido como resultado de la petición POST.

`https://www.virustotal.com/api/v3/analyses/+ID`

Entre los resultados obtenidos se encuentra el hash SHA256 de la respuesta y la ID de la petición de escaneo. Aunque los resultados de interés son los siguientes:


```
{
  "data": {
    "attributes": {
        "date": 1684421490,
        "status": "completed",
        "stats": {
            "harmless": 70,
            "malicious": 0,
            "suspicious": 0,
            "undetected": 19,
            "timeout": 0
        },
        "results": {
            "Bkav": {
                "category": "undetected",
                "result": "unrated",
                "method": "blacklist",
                "engine_name": "Bkav"
            },
            "CMC Threat Intelligence": {
                "category": "harmless",
                "result": "clean",
                "method": "blacklist",
                "engine_name": "CMC Threat Intelligence"
            },
            .....
        }
    }
  }
}
```


Figura 16: Campos de respuesta de VirusTotal API

En la imagen se pueden ver los resultados de los escaneos de las diferentes entidades de seguridad sobre el objetivo. Teniendo contadores comunes para todos estos. Seguidamente, en

la sección de "results", se puede observar una lista de las diferentes entidades y el resultado del análisis para cada entidad de seguridad.

11.3.2. CensysSearch API

Para realizar peticiones a esta API también se añade la API Key en la cabecera y se envía una petición GET indicando la IP del objetivo a la URL:

`https://search.censys.io/api/v2/hosts/+IP`

Como resultados de esta petición se visualiza la sección "result", que tiene una sección servicios donde se expone toda la información sobre cada servicio ejecutado en el objetivo (En este caso hay 2). También se brindan datos referentes a la geolocalización y a los diferentes dominios que pertenecen a ese objetivo. Incluso de ser posible muestra información sobre el sistema operativo.

Figura 17: Campos de respuesta de CensysSearch API

11.4. Implementación de OWASP ZAP

Esta herramienta se ha tenido que instalar en un ordenador que actuara como servidor privado para ejecutar los escáneres de vulnerabilidades que la propia aplicación ordene. Se ha configurado la aplicación para que permita las peticiones por parte de dispositivos externos, como el teléfono móvil.

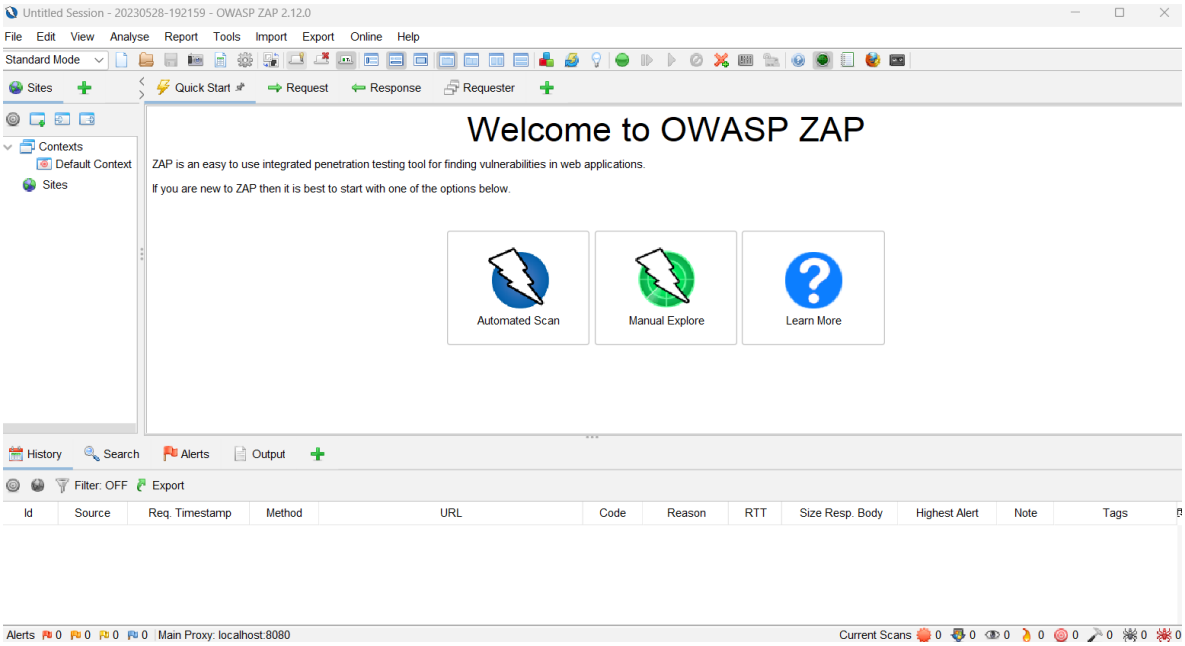

Figura 18: Interfaz de usuario de OWASP ZAP

Esta herramienta ofrece 2 componentes principales:

- **Spider (Araña):** que se encarga de explorar una web de forma sistemática, siguiendo enlaces y recopilando información sobre las páginas y funcionalidades disponibles.
- **Vulnerability Scan (Escaneo de vulnerabilidades)**: que se realiza después de que el spider ha recopilado información sobre la aplicación, con el objetivo de identificar posibles vulnerabilidades en la aplicación web. Para ello utiliza una base de conocimientos de vulnerabilidades conocidas como inyecciones SQL, XSS, CSRF y desbordamiento de búfer, entre otros. Además de las pruebas de seguridad en el código, el escaneo también verifica las configuraciones en busca de debilidades de seguridad.

Una vez comprendido el funcionamiento de la herramienta, al iniciar un escaneo de vulnerabilidades con OWASP API, se realiza primero una petición GET para ejecutar un Spider sobre la URL designada. Esta solicitud incluye la API Key, la URL y una opción para realizar un escaneo recursivo, lo que significa que la herramienta analizará todas las páginas que encuentre o solo la designada.

```
http://192.168.0.12:8080/JSON/spider/action/scan/?apikey="+ZAP_API_KEY+"&url="+url+"&recurse="+recursivo
```

Esta petición inicia el escaneo del Spider y devuelve un ID para tratar de monitorizar el proceso u obtener los resultados. Dicho ID se usa en la ejecución de la segunda petición que sirve para comprobar el estado de la ejecución en el servidor, mostrándolo al usuario por pantalla:

```
http://192.168.0.12:8080JSON/spider/view/status/?apikey="+ZAP_API_KEY+"&scanId="+scanId
```

Esta petición GET se va efectuando cada 2 segundos hasta la finalización del proceso del spider para mantener al usuario informado.

Una vez haya acabado el spider, iniciará el escaneo realizando una petición GET indicando la URL y la API Key, a la siguiente URL:

```
http://192.168.0.12:8080/JSON/ascan/action/scan/?url="+url+"&apikey="+ZAP_API_KEY
```

De la misma forma que con spider, esta petición nos devolverá el ID de escaneo que utilizaremos para comprobar el estado del escaneo:

```
http://192.168.0.12:8080/JSON/ascan/view/status/?apikey="+ZAP_API_KEY+"&scanId="+scanId
```

Para finalizar, solo hay que realizar una petición GET indicando la URL que hemos escaneado y la API Key a la URL:

```
http://192.168.0.12:8080/JSON/core/view/alerts/?baseurl="+url+"&apikey="+ZAP_API_KEY
```

Una vez realizada la última petición para obtener los resultados, se recibe un objeto JSON que enumera todas las vulnerabilidades encontradas, junto con sus diversas características, como se muestra en la siguiente imagen:


```
"alerts": [
    {
        "sourceid": "3",
        "other": "",
        "method": "GET",
        "evidence": "Set-Cookie: AWSALB",
        "pluginId": "10010",
        "cweid": "1004",
        "confidence": "Medium",
        "wascid": "13",
        "description": "A cookie has been set without the HttpOnly flag, which means
            that the cookie can be accessed by JavaScript. If a malicious script can
            be run on this page then the cookie will be accessible and can be
            transmitted to another site. If this is a session cookie then session
            hijacking may be possible.",
        "messageId": "8",
        "inputVector": "",
        "url": "https://ginandjuice.shop/sitemap.xml",
        "tags": {
            "OWASP_2021_A05": "https://owasp.org/Top10/
                A05_2021-Security_Misconfiguration/",
            "WSTG-v42-SESS-02": "https://owasp.org/
                www-project-web-security-testing-guide/v42/
                4-Web_Application_Security_Testing/06-Session_Management_Testing/
                02-Testing_for_Cookies_Attributes",
            "OWASP_2017_A06": "https://owasp.org/www-project-top-ten/2017/
                A6_2017-Security_Misconfiguration.html"
        },
        "reference": "https://owasp.org/www-community/HttpOnly",
        "solution": "Ensure that the HttpOnly flag is set for all cookies.",
        "alert": "Cookie No HttpOnly Flag",
        "param": "AWSALB",
        "attack": "",
        "name": "Cookie No HttpOnly Flag",
        "risk": "Low",
        "id": "0",
        "alertRef": "10010"
    },
```


Figura 19: Características de una vulnerabilidad encontrada con OWASP ZAP

Se han seleccionado como las características más importantes para informar al usuario en la aplicación: el nombre de la vulnerabilidad, el riesgo que representa, la URL donde se ha encontrado, la descripción correspondiente, la solución recomendada y, por último, el vector de ataque.

11.4.1. Desafíos de Cancelación de Escaneo

Si el usuario decide cancelar el escaneo de vulnerabilidades después de haber enviado la petición al servidor, se debe realizar una solicitud GET a la siguiente URL, incluyendo la API Key y el ID del escaneo:

```
http://192.168.0.12:8080/JSON/ascan/action/removeScan/?apikey="+ZAP_API_KEY+"&scanId="+scanId
```

De esta manera, se evita realizar escaneos innecesarios que podrían sobrecargar el servidor, asegurando que cada usuario tenga únicamente un escaneo activo.

11.5. Cambios efectuados durante el desarrollo

Durante el desarrollo de la aplicación, surgieron diversas modificaciones que no habían sido contempladas en la planificación inicial del proyecto.

11.5.1. Cambios en los ajustes

Como se mencionó anteriormente, durante la implementación, se identificaron varias configuraciones que debían agregarse en la pantalla de ajustes para que el usuario pudiera personalizarlas según sus necesidades. A continuación, se describen las funcionalidades finales implementadas y su propósito.

- **Tipo de escaneo**: Esta opción define el modo de uso de la aplicación, ya sea premium o estándar.
- **Rango de puertos**: Permite al usuario definir el rango de puertos a escanear en el escaneo manual.
- **Recursividad**: Un nuevo ajuste que permite al usuario especificar si el escaneo realizado por el spider de OwaspZap debe realizarse de forma recursiva o no.
- **Https**: Otro nuevo ajuste que define si el escaneo de la página se realizará utilizando una conexión con TLS o sin ella.
- **Escáneres**: Muestra una lista de los escáneres utilizados en la aplicación para que el usuario pueda activarlos o desactivarlos según sus preferencias.
- **Riesgos de las vulnerabilidades**: Esta opción permite seleccionar el tipo de riesgo que se desea recibir en las alertas de seguridad generadas por OwaspZap. Se agregó el riesgo "informador" como opción adicional.

11.5.2. Funcionalidad de exportación añadida

Se ha implementado una forma sencilla de compartir los resultados generados por OwaspZap. Para ello, se ha creado un archivo en formato CSV que enumera todas las vulnerabilidades encontradas y sus respectivas características. A continuación, se permite al usuario seleccionar cómo desea compartir este archivo utilizando los medios disponibles en su dispositivo.

A continuación, se muestra una imagen de los resultados presentados en el archivo una vez compartido:

	A	B	C	D	E	F
1	NAME	RISK	URL	DESCRIPTION	SOLUTION	ATTACK
2	Cross Site Scripting (Reflected)	High	https://ginandjuice.shop/login	Cross-site Scripting (XSS) is an attack techr	Phase: Architecture and Design Use a vetted library or fram	':alert(1);'
3	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/robots.txt	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
4	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/sitemap.xml	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
5	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
6	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
7	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
8	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/blog	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
9	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/blog	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
10	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
11	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/blog/post?postId=2	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
12	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/blog/post?postId=6	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
13	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=4	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
14	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=4	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
15	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=4	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
16	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/about	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
17	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=3	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
18	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=2	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
19	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=2	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
20	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=3	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
21	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=2	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
22	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=3	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
23	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/blog/post?postId=3	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
24	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/cart	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
25	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/blog/post?postId=4	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
26	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=1	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
27	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=7	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
28	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=8	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
29	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=7	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
30	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=1	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
31	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=7	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
32	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=1	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
33	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=8	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
34	Absence of Anti-CSRF Tokens	Medium	https://ginandjuice.shop/catalog/product?productId=8	No Anti-CSRF tokens were found in a HTM	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides co	
35	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=9	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
36	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=10	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
37	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=11	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	
38	Content Security Policy (CSP) Header Not Set	Medium	https://ginandjuice.shop/catalog/product?productId=12	Content Security Policy (CSP) is an added l	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.	

Figura 20: Ejemplo de los resultados en el archivo CSV

11.6. Estructura del proyecto

Este apartado presenta la estructura del proyecto, mostrando sus relaciones y principales funcionalidades. Los diferentes componentes se clasifican por colores según su funcionalidad. Los elementos de color azul representan actividades, lo que implica que muestran una nueva pantalla al usuario con su respectivo diseño, ofreciendo diversas funcionalidades. Los elementos de color amarillo son clases utilizadas para generar objetos con los atributos mencionados. Por último, se incluyen clases de color verde, que se utilizan exclusivamente para gestionar métodos de manera más clara y eficiente.

A continuación se puede ver la imagen general de como está montado el proyecto:

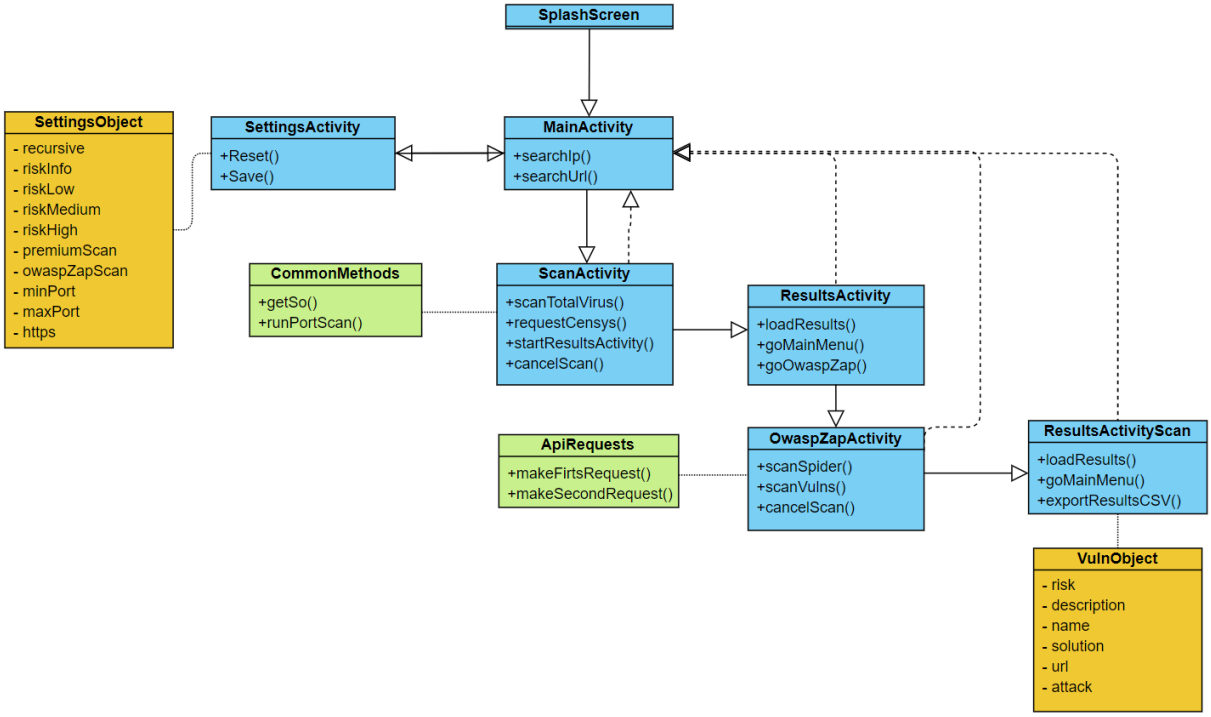

Figura 21: Diagrama de clases

Esta estructura es similar al diagrama de flujo propuesto en la sección de investigación. Comenzamos con una actividad de carga que muestra únicamente el logo de la aplicación y carga la MainActivity, que es el eje principal de la aplicación. Desde esta actividad, podemos pasar a la actividad encargada de los ajustes (SettingsActivity), que tiene un singleton con

un objeto de tipo SettingsObject que se modifica mediante la actividad, restableciendo sus valores o guardando los cambios.

La MainActivity proporciona la funcionalidad de escanear una dirección IP o una URL y envía la entrada del usuario a ScanActivity, que muestra una pantalla de carga al usuario. Esta actividad ejecuta las funciones mencionadas anteriormente: getSO() y runPortScan() de CommonMethods. También realiza solicitudes a las APIs de VirusTotal y CensysSearch antes de pasar a ResultsActivity, que muestra los resultados.

A continuación, el usuario puede decidir si volver a MainActivity o ir a OwaspZapActivity, que también muestra la pantalla de carga y realiza la funcionalidad de escaneo utilizando la API de OwaspZap instalada en el servidor propio. Para esto, utiliza funciones de la clase ApiRequests que facilitan el manejo de las respuestas.

Una vez obtenidos los resultados de las vulnerabilidades, se pasa a la actividad ResultsActivityScan, que utiliza la clase VulnObject para crear una lista de todas las vulnerabilidades y mostrarlas a los usuarios. Finalmente, esta actividad permite volver a MainActivity y exportar los resultados como un archivo CSV.

11.7. Interfaz final de la aplicación

En esta sección se presentan los resultados visibles al usuario durante la navegación en la aplicación:

Figura 22: Diseño de la pantalla inicial de carga de la aplicación.

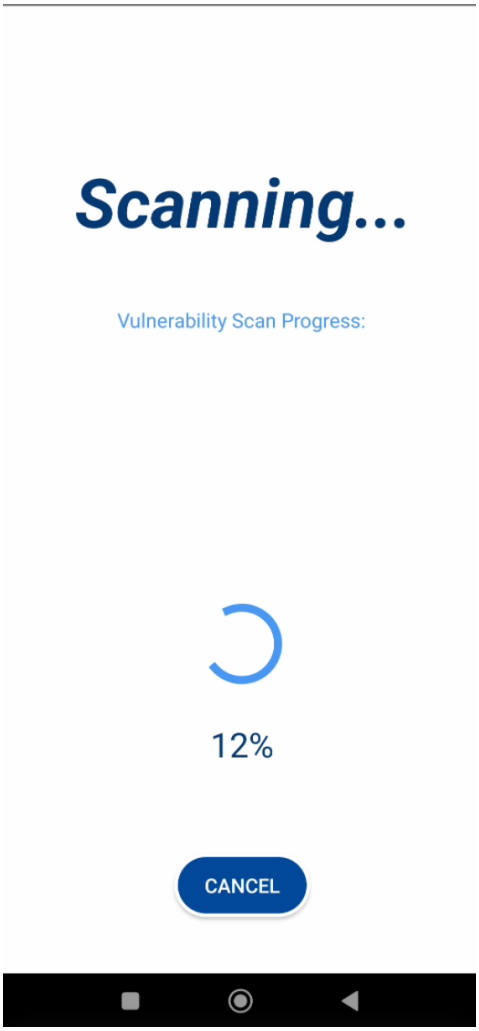

Figura 23: Diseño de la pantalla de carga para escaneos.

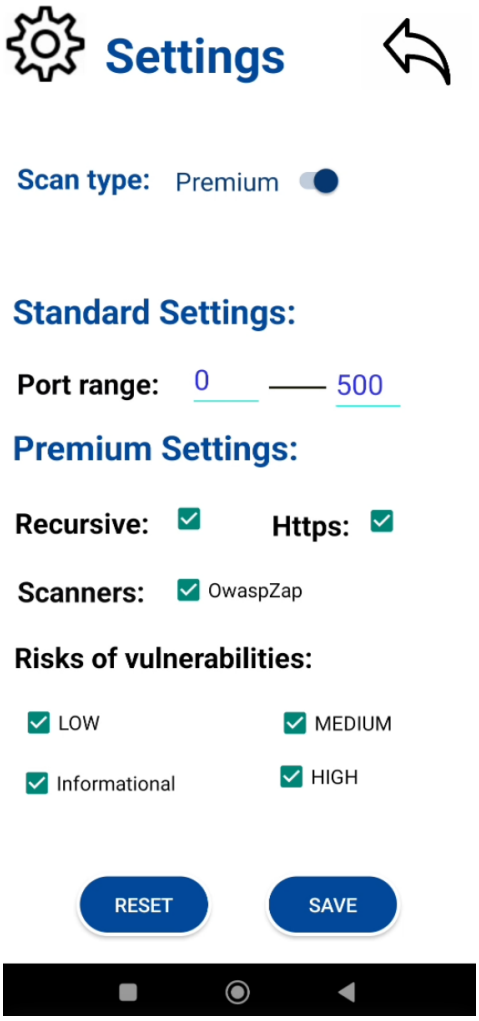

Figura 24: Diseño de la pantalla de ajustes.

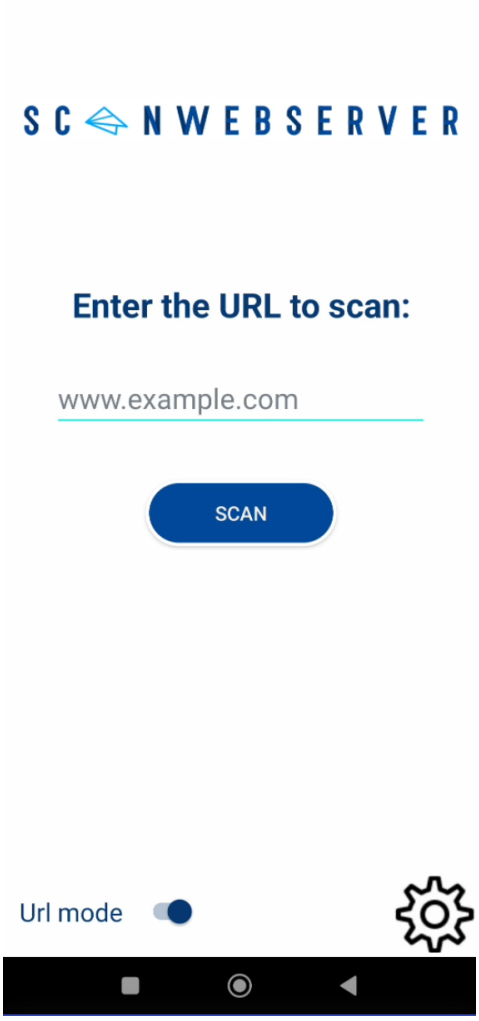

Figura 25: Diseño de la pantalla principal.

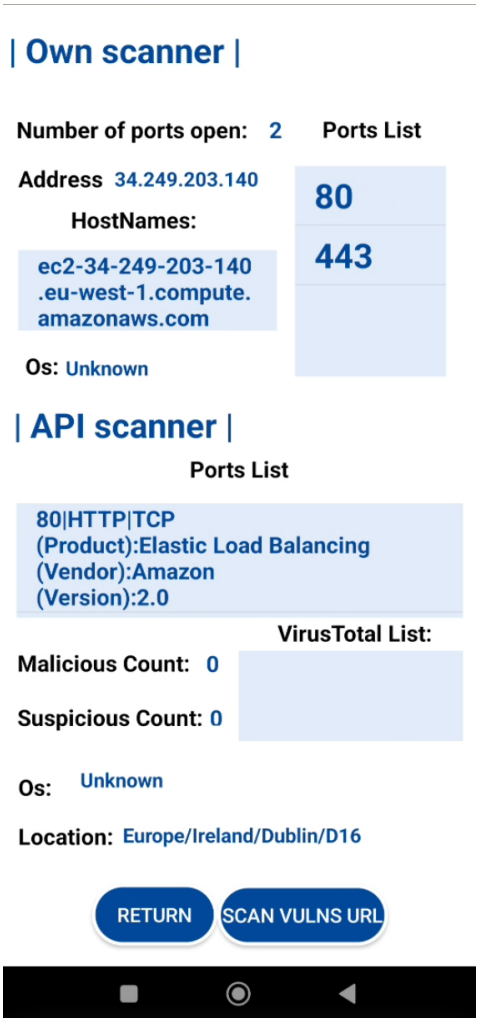

Figura 26: Diseño de la pantalla de resultados del escaneo a la red (Censys-Search y VirusTotal).

Figura 27: Diseño de la pantalla de resultados del escaneo de vulnerabilidades.

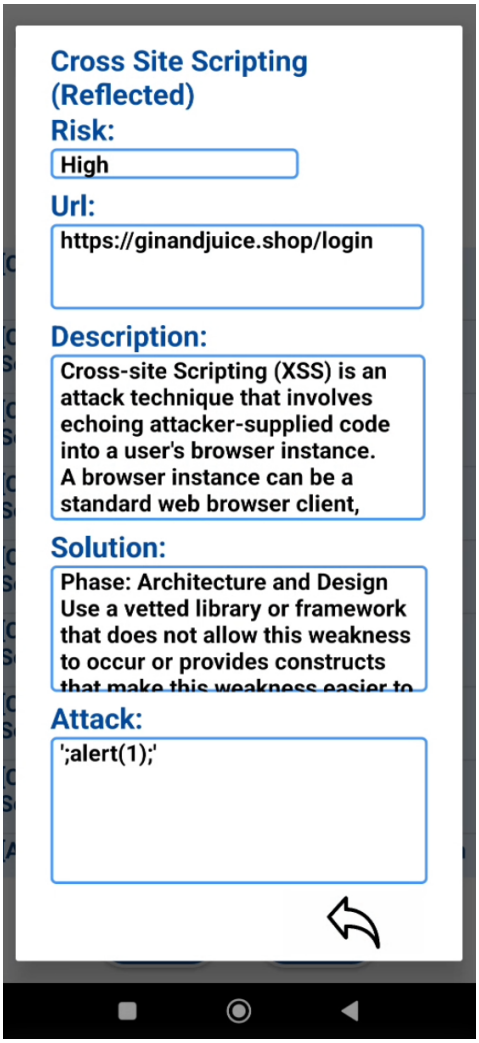

Figura 28: Diseño de la pantalla de información de una vulnerabilidad

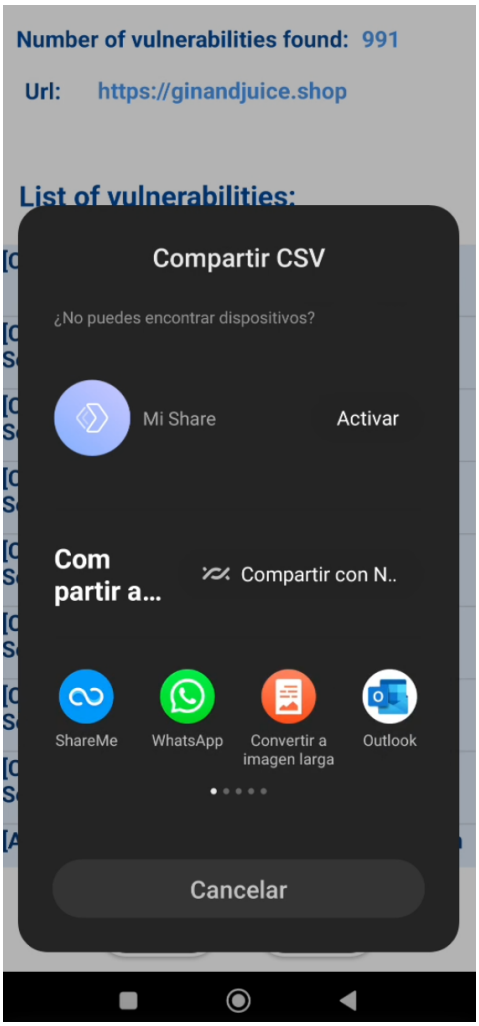

Figura 29: Visualización de la funcionalidad de exportación de resultados

11.8. Laboratorio de Pruebas y Resultados

En esta sección se verificará la precisión de los resultados generados por las diversas funcionalidades implementadas en la aplicación. Se evaluarán tanto las funcionalidades de obtención de información de la red, como las de detección de vulnerabilidades para asegurar su exactitud.

11.8.1. Pruebas en el escaneo de redes

La **primera prueba** realizada ha consistido en escanear la URL **ginandjuice.shop** utilizando los ajustes predeterminados. Los resultados y el proceso se pueden observar en las siguientes imágenes:

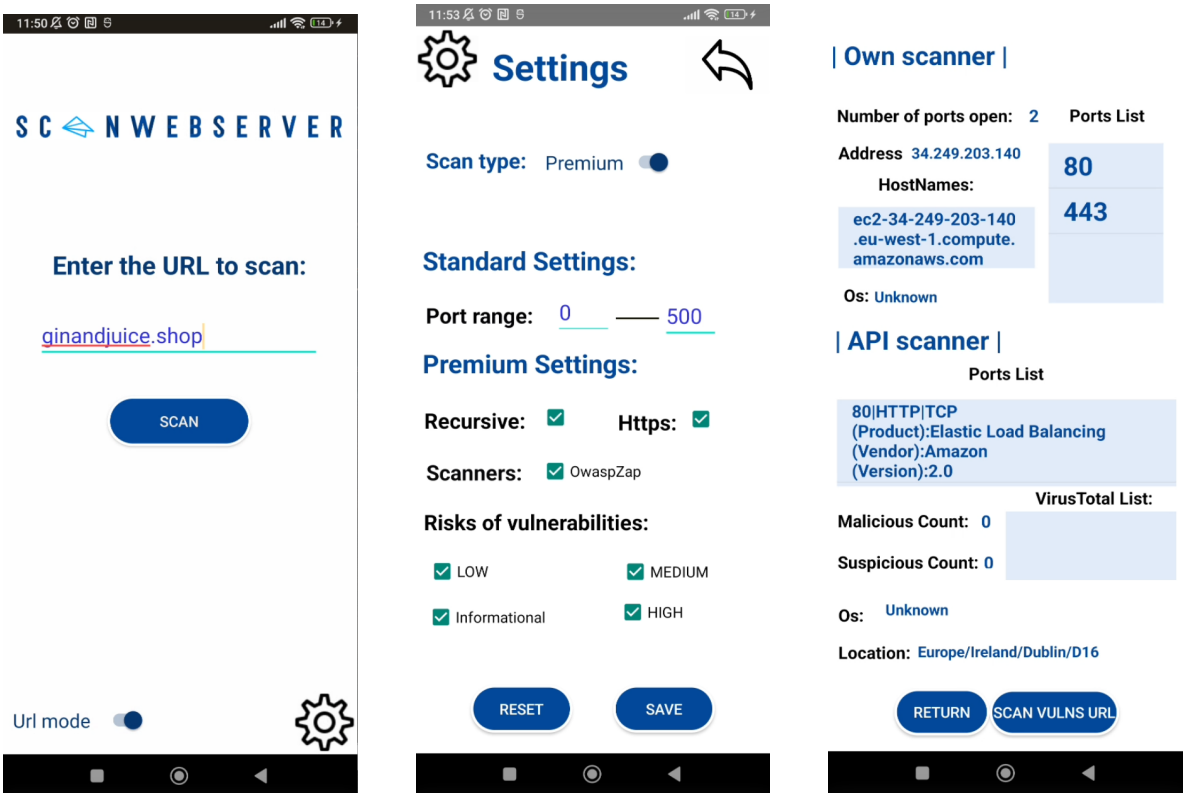

Figura 30: Prueba del escáner de redes 1

En los resultados obtenidos, se ha observado que los puertos 80 y 443 se encuentran abiertos en el sistema. Además, se ha obtenido la dirección IP y una lista de HostNames asociados a esta.

En cuanto a los resultados de las API, al utilizar VirusTotal no se ha encontrado ningún elemento sospechoso relacionado con la dirección IP analizada. Por otro lado, al utilizar Censys-Search, se ha obtenido nuevamente la lista de puertos abiertos, pero con información adicional sobre cada uno de ellos y datos sobre la localización geográfica, incluyendo el código postal.

En la **segunda prueba** se ha vuelto a escanear el mismo objetivo, pero limitando el rango de escaneo de puertos a 81 para el escáner manual, de tal forma que esta vez solo tendría que localizar el puerto 80 pero no el 443. Los resultados obtenidos han sido:

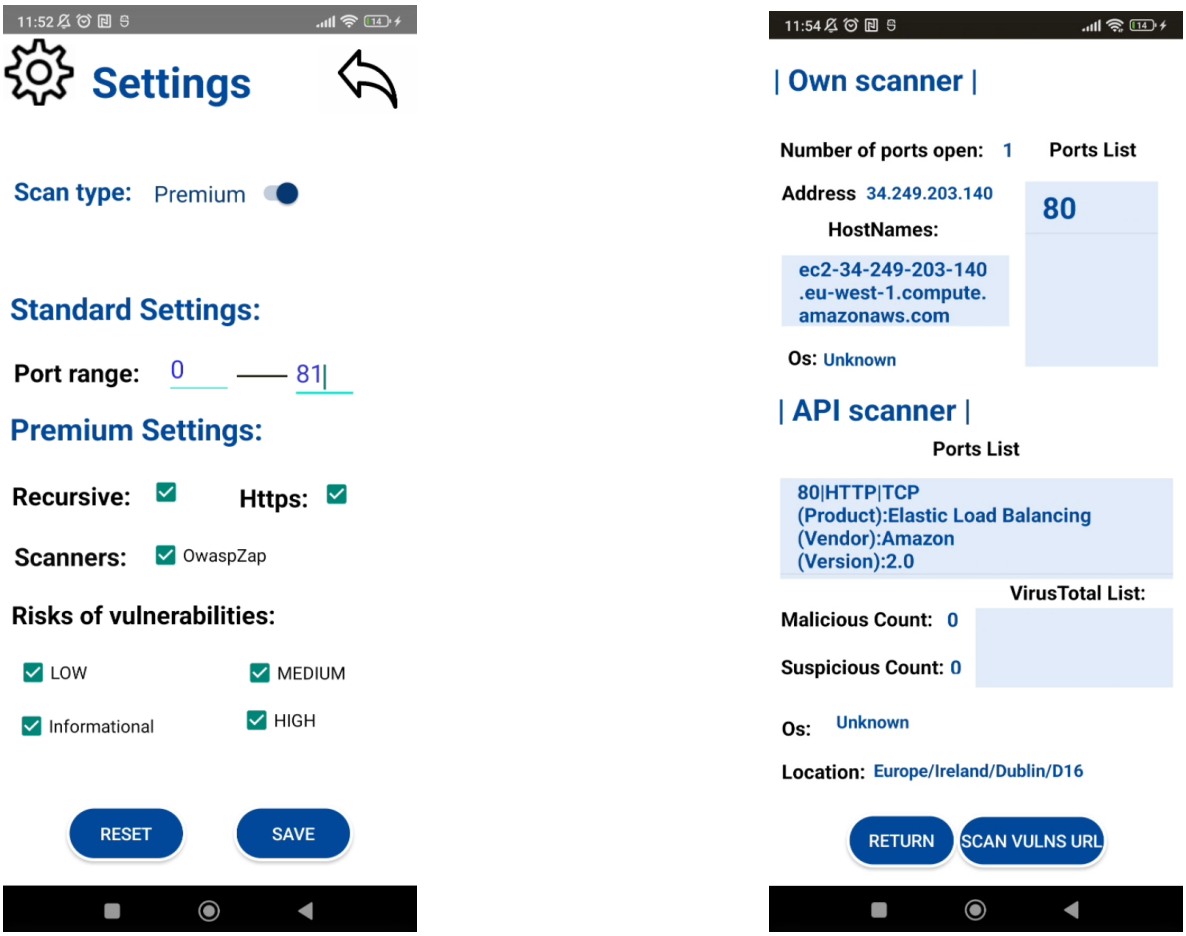

Figura 31: Prueba del escáner de redes 2

Para comprobar la funcionalidad de realizar un escaneo a una IP concreta se ha realizado la **tercer prueba**, la cual ha consistido en utilizar la IP obtenida anteriormente como input para la aplicación para comprobar que se obtienen los mismos resultados:

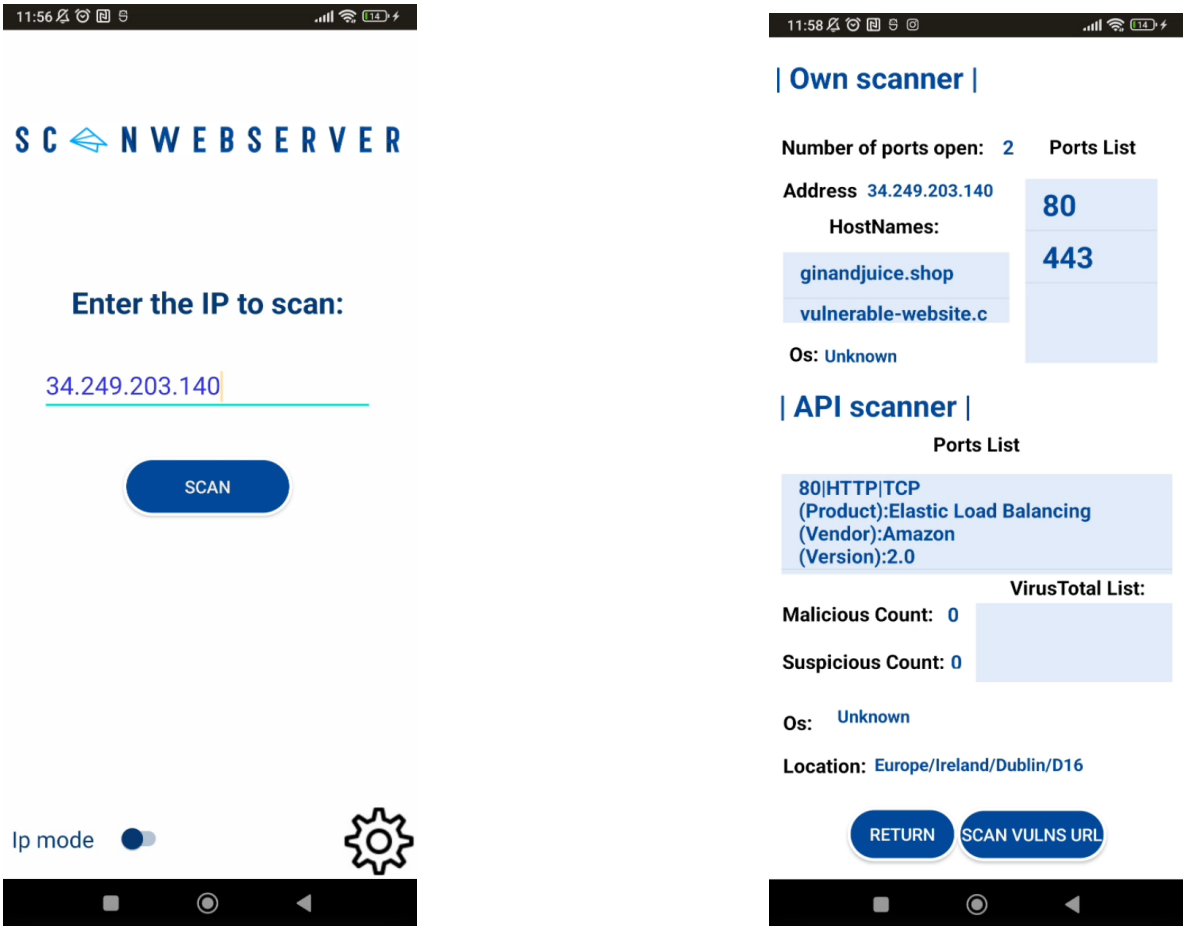

Figura 32: Prueba del escáner de redes 3

Para terminar, se ha realizado la **cuarta prueba** en la que se ha escaneado una página maliciosa **aladel.net** la cual servirá para comprobar el funcionamiento de la API de VirusTotal:

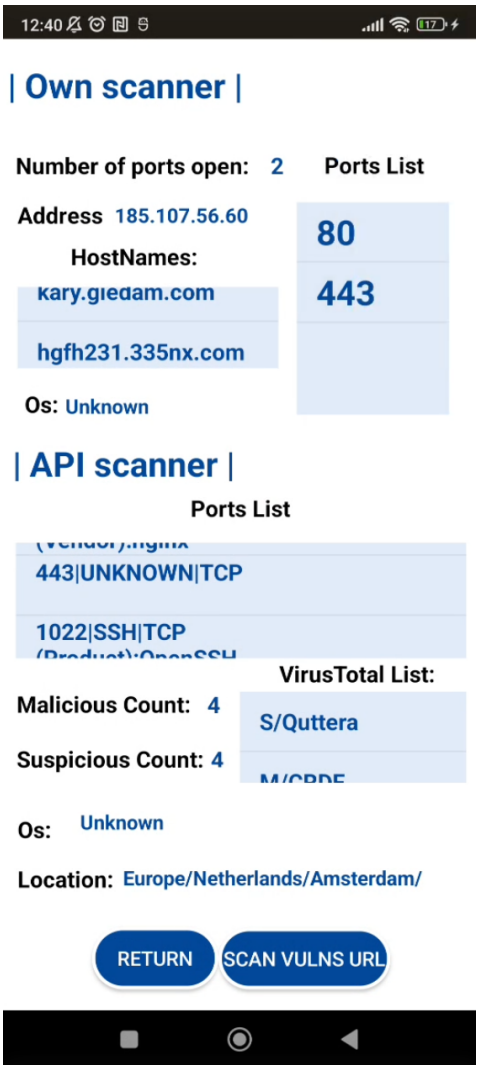

Figura 33: Prueba del escáner de redes 4

Los resultados obtenidos, tal como se puede apreciar en las imágenes, concuerdan plenamente con lo esperado.

11.8.2. Pruebas en el escaneo de vulnerabilidades

Con el fin de evaluar el funcionamiento del escaneo de vulnerabilidades, se llevarán a cabo dos pruebas. La primera consistirá en escanear la URL previamente mencionada, **ginandjuice.shop**, mientras que la segunda se realizará en la aplicación DVWA, y se mostrarán los resultados obtenidos en ambos casos.

A continuación se puede observar la lista de vulnerabilidades resultante de realizar la primera prueba y un ejemplo de las características que muestra de una de las vulnerabilidades en concreto:

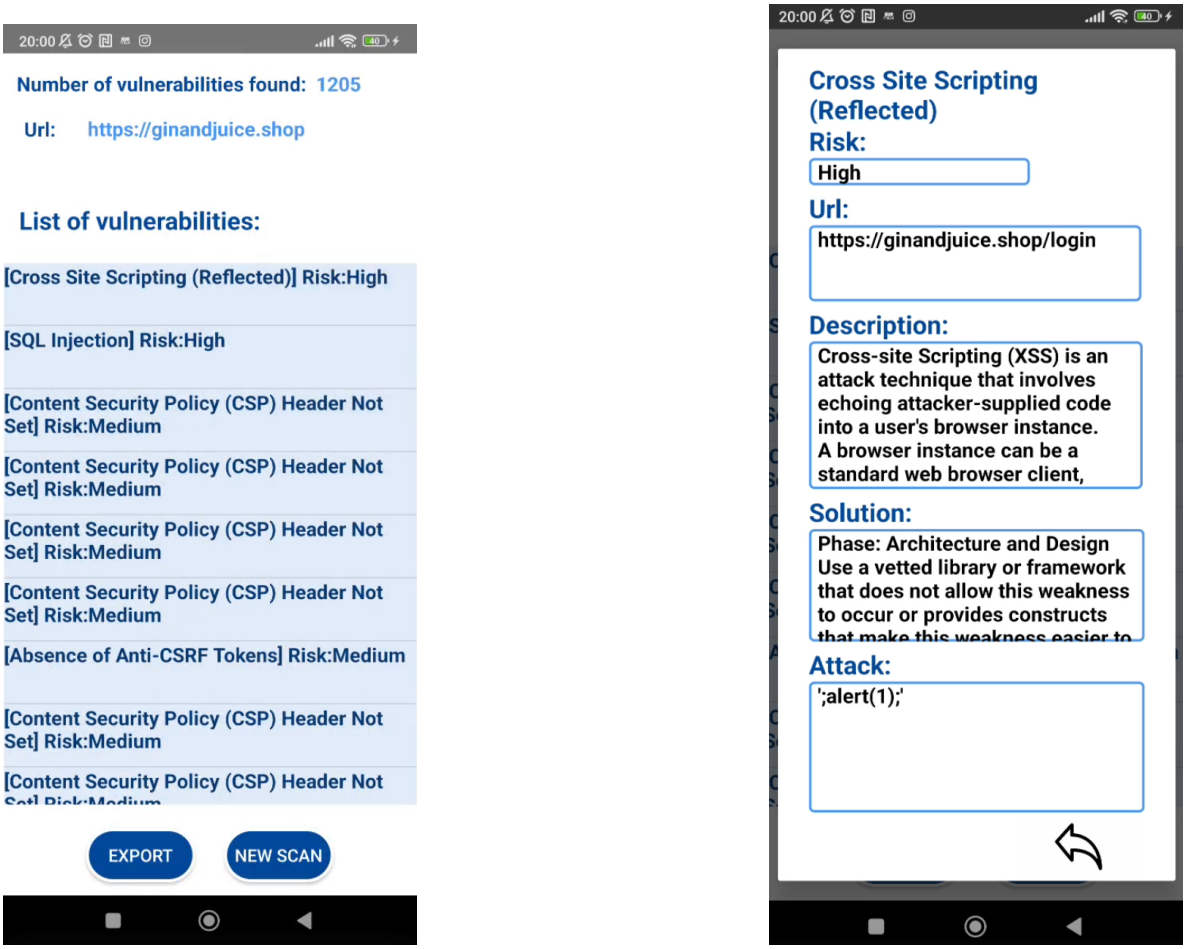

Figura 34: Prueba de escaneo de vulnerabilidades 1

Seguidamente, se muestran los resultados obtenidos en la segunda prueba:

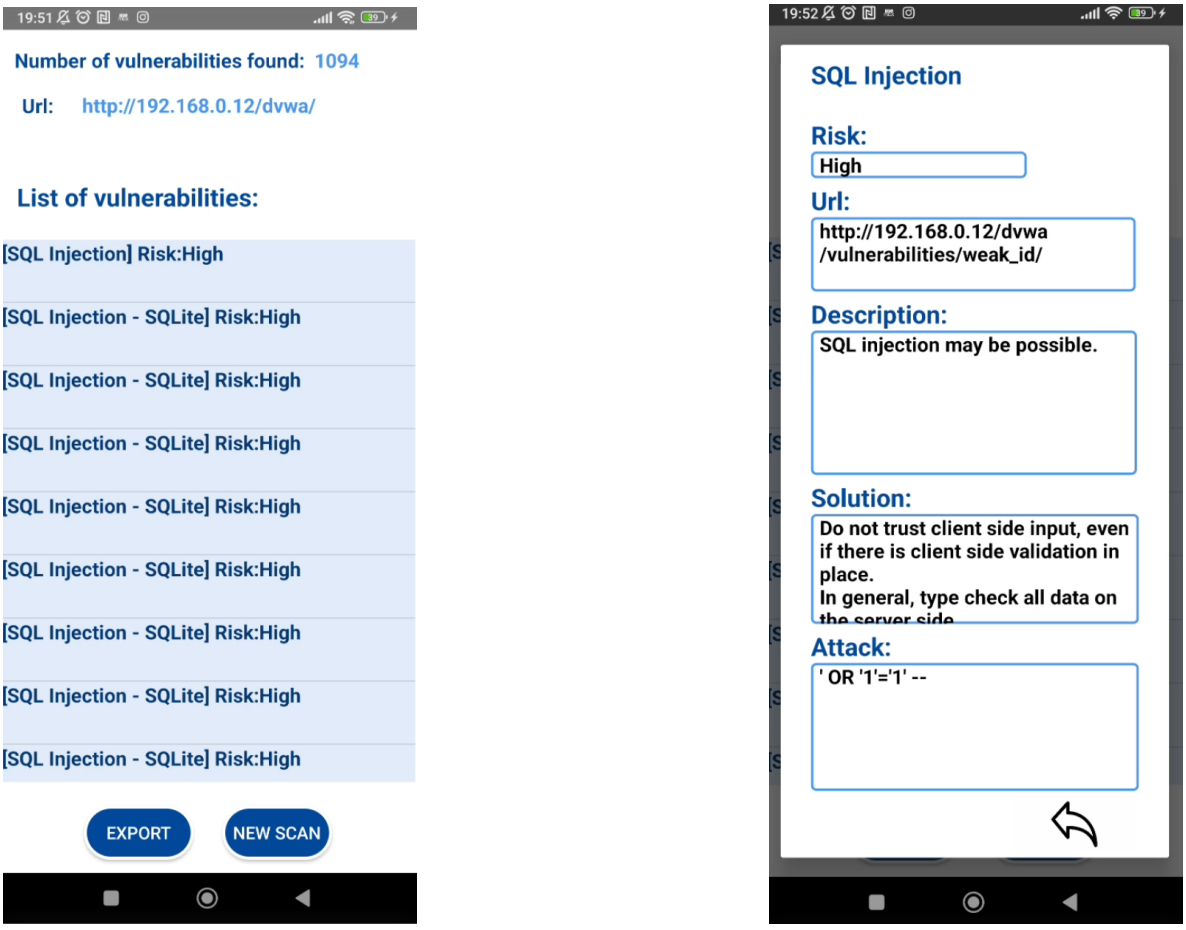

Figura 35: Prueba de escaneo de vulnerabilidades 2

Finalmente, como prueba adicional se cambian algunas opciones de los ajustes para verificar su funcionalidad. En este caso se desactivan las alertas de vulnerabilidades que no sean altas:

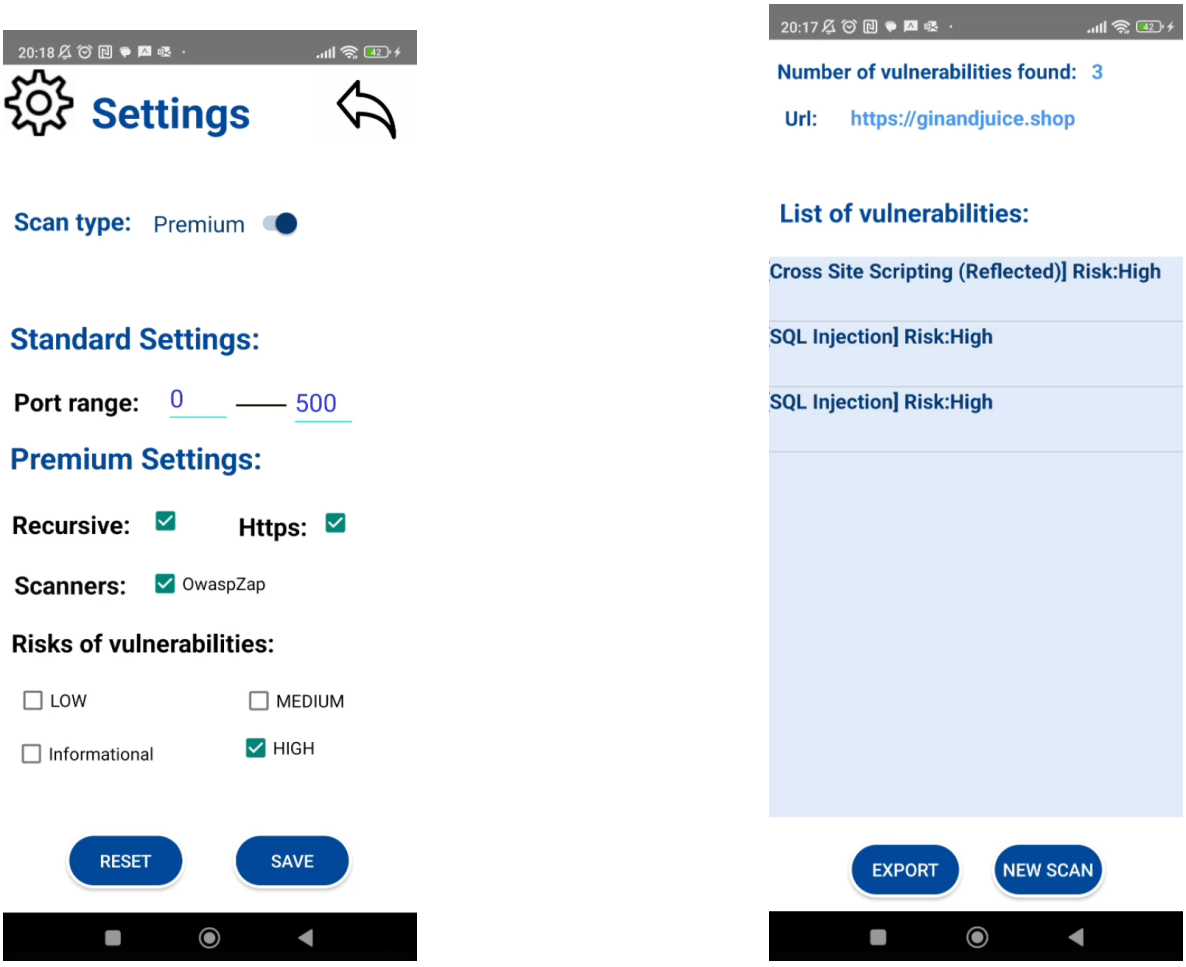

Figura 36: Prueba de escaneo de vulnerabilidades 3

12. Conclusiones

En este proyecto, se ha logrado desarrollar una aplicación móvil modular y fácil de usar, que permite realizar análisis de vulnerabilidades en servidores a partir de su dirección IP o una URL concreta. A través de la integración de las APIs de CensysSearch, VirusTotal y la herramienta OWASP ZAP, se ha obtenido información valiosa sobre los objetivos, lo que puede contribuir a mejorar la seguridad de los sistemas informáticos.

La investigación y exploración de herramientas open-source existentes fue fundamental para maximizar la funcionalidad de la aplicación y evitar reinventar la rueda. Se han encontrado muchas alternativas, pero se ha usado la que más se adaptaba al objetivo del proyecto para el escaneo de vulnerabilidades sobre un aplicativo web.

Sobre la creación de herramientas propias para proporcionar modularidad a la aplicación y no tener que depender de consultas a fuentes externas, se han desarrollado el escáner de puertos y el identificador de sistema operativo, los cuales ofrecen una mínima funcionalidad al usuario sin tener que depender de recursos externos. Además, también identifica todos los nombres de dominio pertenecientes a una dicha IP.

Por la parte de la interfaz de usuario, es fácil de utilizar y no presenta muchas opciones que puedan confundir al usuario. Además, los resultados de los análisis se presentan de manera clara y concisa en una pantalla dedicada para ello.

Estas pantallas podrían contemplarse como informes detallados sobre todas las características extraídas del objetivo. Aunque también existe la funcionalidad de compartir los resultados del escáner de vulnerabilidades en formato CSV a cualquier aplicación del teléfono móvil.

En resumen, se han logrado alcanzar los objetivos principales establecidos para este proyecto. La combinación de herramientas open-source, herramientas propias y la integración de APIs externas ha permitido ofrecer información relevante para el análisis de vulnerabilidades. La interfaz de usuario intuitiva ha hecho que sea fácil de usar, y los informes detallados han

mejorado la comprensión de los resultados obtenidos.

12.1. Trabajo futuro

A continuación, se presentan algunas ideas y áreas en las que se puede continuar trabajando para seguir mejorando la funcionalidad, seguridad y rendimiento de la aplicación:

- **Aumento de la seguridad en las comunicaciones**: Este era el objetivo de menor importancia presentado en el proyecto, que no se ha podido realizar, ya que ha quedado fuera del alcance del proyecto. Se trata de conseguir realizar comunicaciones con el exterior empleando protocolos HTTPS para cifrar y proteger los datos transmitidos, fortaleciendo la confidencialidad e integridad de la aplicación.
- **Integración de más herramientas**: Se pueden explorar nuevas herramientas y técnicas avanzadas en el campo de la seguridad informática para mejorar la funcionalidad y el rendimiento de la aplicación.
- **Escalabilidad**: Debido a la utilización de la herramienta gratuita de OWASP ZAP, se reduce en gran medida su rendimiento al recibir varias solicitudes de escaneo. Se planea usar una gran cantidad de servidores para poder manejar la demanda de peticiones o utilizar otras herramientas de pago que puedan ofrecer un rendimiento óptimo.
- **Reducción del tiempo de espera durante el escaneo**: Se debe trabajar en reducir el tiempo de espera durante los escaneos de vulnerabilidades. Una solución es implementar un sistema que permita al usuario salir de la aplicación mientras el escaneo está en curso, sin interrumpir el proceso. Ofreciendo mayor flexibilidad al usuario y optimizando el tiempo de espera.

13. Acrónimos

API Interfaz de programación de aplicaciones

CSRF Falsificación de petición en sitios cruzados

CVE Vulnerabilidades y exposiciones comunes

DVWA Damn Vulnerable Web App

ODS Objetivos de Desarrollo Sostenible

OWASP Proyecto de seguridad de aplicaciones web abiertas

SaaS Software como servicio

TLS Seguridad de la capa de transporte

TTL Tiempo para vivir

URL Localizador de Recursos Uniforme

VPS Servidor virtual privado

XSS Script entre sitios (Cross-site scripting)

Bibliografía

[nin, 2012] (2012). Ninjamock. https://ninjamock.com/es/home/index. Online.

[Atkinson, 2022] Atkinson, M. (2022). Gin and juice shop: put your scanner to the test. https://portswigger.net/blog/gin-and-juice-shop-put-your-scanner-to-the-test. Online.

[Baeldung, 2023] Baeldung (2023). ¿port scanning with java. https://www.baeldung.com/java-port-scanning. Online.

[Develou, 2023] Develou (2023). Realizar peticiones http con la librería volley en android. https://www.develou.com/android-volley-peticiones-http/. Online.

[digininja, 2023] digininja (2023). Damn vulnerable web application. https://github.com/digininja/DVWA. Online.

[Iñigo, 2018] Iñigo (2018). Tiempos ttl de los distintos sistemas operativos más usados. https://cronicasdeuninformatico.com/2018/08/tiempos-ttl-de-los-distintos-sistemas.html. Online.

[Java Platform, 2021] Java Platform, S. E. . A. S. (2021). Inetadress. https://docs.oracle.com/javase/8/docs/api/java/net/InetAddress.html. Online.

[KeepCoding, 2023] KeepCoding (2023). ¿qué es nessus? https://keepcoding.io/blog/que-es-nessus/. Online.

[OWASP, 2010] OWASP (2010). Zed attack proxy. https://www.zaproxy.org/. Online.

[OWASP, 2023] OWASP (2023). Vulnerability scanning tools. https://owasp.org/www-community/Vulnerability_Scanning_Tools. Online.

[Petrosyan, 2023] Petrosyan, A. (2023). Estimated cost of cybercrime worldwide from 2016 to 2027. https://www.statista.com/statistics/1280009/cost-cybercrime-worldwide/#statisticContainer. Online.

[Probely, 2023] Probely (2023). Web application and api vulnerability scanner. https://probely.com/. Online.

[Roa, 2023] Roa, M. M. (2023). El mapa mundial de android e ios. Recuperado de: https://es.statista.com/grafico/29620/sistema-operativo-movil-con-la-mayor-cuota-de-mercado-por-pais. Online.

[Vera, 2020] Vera, R. A. (2020). Qué es openvas. https://openwebinars.net/blog/que-es-openvas/. Online.

[Vázquez, 2022] Vázquez, I. (2022). ¿qué diferencia hay entre una app híbrida y una nativa? https://appmarketingnews.io/que-diferencia-hay-entre-una-app-hibrida-y-una-nativa/. Online.