ANÁLISIS DE SEGURIDAD DE UNA APLICACIÓN MÓVIL PARA SISTEMAS ANDROID VERSIÓN 9, UTILIZANDO OWASP MOBILE 2016

JIMMY OSWALDO MIRANDA AGUIRRE

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTA
2021

ANÁLISIS DE SEGURIDAD DE UNA APLICACIÓN MÓVIL PARA SISTEMAS ANDROID VERSIÓN 9, UTILIZANDO OWASP MOBILE 2016

JIMMY OSWALDO MIRANDA AGUIRRE

Trabajo aplicado como requisito para optar al título de:
Especialista en Seguridad Informática

Director:
Ing. JOHN F. QUINTERO

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTA
2021

NOTA DE ACEPTACIÓN

Firma de jurado

Firma de jurado

Bogotá, 07 de abril de 2022

DEDICATORIA

Dedicó este trabajo, primero que todo a Dios, que es nuestra guía y fortaleza en los tiempos difíciles que se han vivido, por darnos el conocimiento y sabiduría para actuar.

A mi familia quien es mi motor de vida, mi apoyo constante y porque siempre están ahí para mí en cualquier propósito.

A mis profesores por brindarme nuevos conocimientos para mi futuro.

Y a todos aquellos que en algún momento compartieron y apoyaron mi deseo de llegar tener un título como especialista.

Jimmy Oswaldo Miranda Aguirre

AGRADECIMIENTOS

Agradezco profundamente a Dios, por guiarme en el sendero correcto de la vida, Porque ha iluminado el transcurso de mi camino y porque siempre guía lo que realizo y mi diario convivir.

Me encuentro agradecido con mis tutores**,** a quien consideró unas personas muy profesionales y que me dieron el conocimiento para llegar a ser un mejor profesional.

A la empresa Financiera, por permitirme y darme la oportunidad de trabajar para ellos y de brindarme la confianza para utilizar uno de sus programas para mi proyecto de grado.

Por último, gracias a todas las personas con las que he compartido en este largo camino, soportando y comprendiendo con paciencia, todos los problemas causados.

Muchas gracias a todos.

CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

RESUMEN

La aplicación móvil siendo una aplicación hibrida que permite a los clientes consultar saldos, realizar compra de boletería para eventos, pagar servicios, etc. Se asume que esta aplicación guarda, manipula o contiene información sensible de cada cliente, por lo que es necesario garantizar la seguridad de esta información, ya que presentaría un riesgo legal y de reputación para la empresa.

Es por ello, que se ha solicitado el permiso de realizar un análisis de seguridad de la aplicación móvil y así poder presentar un informe del estado actual en el que se encuentra, ya que en una revisión rápida con la herramienta Mobsf, Se evidenció que la aplicación no cuenta con todos los controles necesarios para que esta aplicación sea publicada en tiendas móviles (Play Store y AppGallery)

Por tanto, este trabajo tiene como objetivo realizar un análisis de vulnerabilidades, con el fin de conocer el estado de seguridad actual de la aplicación. Para ello se utilizará como referente el proyecto Mobile top 10 de OWASP (Open Web Application Security Project) en la versión 2016, el cual identifica los riesgos de seguridad que pueden surgir de acuerdo a su nivel de criticidad y con esto dar prioridad a los riesgos más críticos.

Para encontrar las vulnerabilidades o principales riesgos se realizarán 2 tipos de análisis, uno estático y otro dinámico de la aplicación, con el primero se podrá realizar una búsqueda profunda de su estructura, su código y seguridad enfocada en el top 10 de OSWAP Mobile, esto se hará con herramientas como Dex2jar, JD-Gui y APkTool, Frida, Burpsuite, etc., que permiten conocer el código fuente de la aplicación y su nivel de seguridad.

Y para el análisis dinámico se utilizarán herramientas como Genymotion, Docker, ADB, Tamer, etc., que permiten emular la aplicación como si se tratara de un

teléfono móvil real, pero en el que será posible analizar y capturar todas las acciones que se llevan a cabo dentro de la aplicación y así poder validar si presenta algún comportamiento anormal.

Una vez realizadas las pruebas de seguridad se presentará un informe a la empresa, para que tengan el conocimiento de los riesgos y/o vulnerabilidades presentes que tiene la aplicación y a su vez se les presentaran una serie de recomendaciones necesarias para poder solucionarlas.

PALABRAS CLAVES
Seguridad, Vulnerabilidades, Aplicativos Móviles, Análisis Estático Móvil, Análisis Dinámico Móvil.

ABSTRACT

The mobile application. It’s a hybrid application allow to the costumer check balance, buy tickets, pay home services. This application save information sensitive of each client, for this reason it’s necessary guarantee the security of this information, as it would generate legal risk and reputation for the company

For this project, permission was requested to carry out a vulnerability test on the mobile application and thus be able to present a report of its current status, it was found that the application does not have all the necessary controls for them to be published in the store

The goal for this work is carry out assessment security app mobile and to present the actual status. for this I will use to help me the project OWASP top 10 mobile, this project identifies the higher risk and give the options for priority for solves them

To find the vulnerabilities or main risks, 2 types of analysis will be carried out, one static and the other dynamic of the application, with the first one can perform a deep search of its structure, code and security focused on the top 10 of OSWAP Mobile, this is It will do with tools such as Dex2jar, JD-Gui and APkTool, Frida, Burpsuite, etc., which allow to know the source code of the application and its security level.

And for the dynamic analysis, tools such as Genymotion, Docker, ADB, Tamer, etc. will be used, which allow emulating the application as if it were a real mobile phone, but in which it will be possible to analyze and capture all the actions that are carried out. done within the application and thus be able to validate if it presents any abnormal behavior.

Once the security tests have been carried out, a report will be presented to the company, so that they are aware of the risks and / or vulnerabilities present in the

application and once a series of necessary recommendations are presented to be able to solve them.

KEYWORDS
Security, Vulnerabilities, Mobile Applications, Mobile Static Analysis, Mobile Dynamic Analysis.

INTRODUCCIÓN

El presente documento tiene como propósito presentar el proyecto aplicado que tiene como requisito para obtener el título de Especialista en Seguridad Informática en la Universidad Nacional Abierta y A Distancia – UNAD. El trabajo presenta el desarrollo del análisis de seguridad para la aplicación móvil, en esta aplicación de puede realizar consulta de saldos, compra de boletas, solicitudes de crédito y consultar información de las cuentas, entre otros.

En el desarrollo del trabajo se realizarán pruebas de la aplicación mediante el análisis de código estático y dinámico, utilizando diversas herramientas que permiten realizar estos procesos. Para posteriormente presentar el estado de seguridad actual de la aplicación y así validar si cumple con los parámetros de seguridad estándar establecidos por OWASP y con la Ley Colombiana de Protección de Datos Personales.

Para posteriormente, presentar un documento con la evidencia correspondiente, informando su estado actual y entregando una serie de recomendaciones y sugerencias que se pueden aplicar para corregir las vulnerabilidades encontradas.

La empresa financiera, día a día, busca hacer la vida más fácil a sus usuarios o asociados y cómo asegurarse de que no pasen horas en una oficina para realizar una transacción bancaria, una consulta o hacer efectivos los beneficios que tienen, por eso han hecho el relanzamiento de su aplicación móvil y la han puesto en las tiendas móviles Play Store y AppGallery. Dado que esta aplicación guarda, manipula o contiene información sensible de cada cliente, es necesario garantizar la seguridad de esta información, ya que presentaría un riesgo legal y reputacional para la empresa.

Ahora, para realizar el análisis de seguridad se solicitó la aprobación y permiso de la empresa, quien amablemente permitió que se realizara este trámite, para dejar constancia del permiso, se realizó la firma de documentos escritos, los cuales se adjuntan en el Anexos.

JUSTIFICACIÓN

Para la empresa uno de sus pilares más importantes es tener comunicación con los clientes y poder brindarles opciones que les facilite realizar operaciones, pagos, compras o tramites de manera rápida y oportuna y que adicionalmente pueda estar alineado a las nuevas tecnologías, Es por esto que se realizó el relanzamiento de la aplicación móvil.

Es importante que una empresa pueda vincular, alinear o establecer nuevas tecnologías para que pueda sobre vivir en el medio comercial, pero también es importante que la empresa y el área de tecnología, adopten nuevas estrategias con base a la seguridad informática y de la información. Ya que al no tener o aplicar estas estrategias es posible que este dañando uno de los tres pilares (confidencialidad integridad y disponibilidad) que lo componen y que coloque en riesgo la compañía y a los clientes o asociados que la componen.

Es por esto que mediante el análisis de seguridad que se realizará al aplicativo móvil se podrá medir el nivel de seguridad que se tiene y el manejo que se le da a la información que se captura y en caso de que presente un fallo, riesgo u evento grave, poder brindar las pautas que permitan mitigar el riesgo latente. Por tal razón, se desarrollará el proyecto con el fin de presentar un informe de seguridad que contenga los riesgos latentes y sus opciones de mejora, para esto se utilizaran herramientas open source como Dex2jar, JD-Gui, ApkTool, ADB, Genimotion, entre otras, que permitan conocer el estado de seguridad actual de la aplicación.

Con este proyecto se espera, más que dar las pautas para que se pueda crear una política, procedimiento o control de cambios, es que el aplicativo se agregue o vincule en la matriz de riesgos y se valorada según el riesgo que pueda presentar para la empresa, ya que, con los cambios realizados en su nueva versión, esta

aplicación paso a ser un activo valioso y que debe ser tratado con un alto nivel de seguridad.

Y desea que esto permita mejorar la seguridad no solo del aplicativo sino de la infraestructura que la soporta, debido a que la infraestructura hibrida que tiene el aplicativo hace que la interacción con los sistemas internos sea constante y se pueda llegar a generar alguna brecha para ataques o denegaciones de servicio, ya que manipular o guardar información de los clientes se vuelve un tema importante para cualquier empresa y más cuando se trata de una empresa que manipula información de miles de personas.

1 OBJETIVOS

1.1 OBJETIVO GENERAL

Evaluar el nivel de seguridad de una aplicación Móvil, basado en la guía OWASP Top 10 Mobile versión 2016 y utilizando el sistema operativo Android versión 9

1.2 OBJETIVOS ESPECÍFICOS

- Identificar vectores de ataque definidos en OWASP teniendo en cuenta M1 a M10 para validación de posibles vulnerabilidades en la app.

- Consultar posibles herramientas a usar en el proceso de análisis de seguridad para llevar a cabo pruebas concretas a partir de la metodología propuesta.

- Esquematizar las vulnerabilidades presentes en la aplicación móvil, su nivel de riesgo y posible solución expuesta mediante CVE.

- Elaborar informe final con conclusiones y buenas prácticas para el aseguramiento de la app.

2 PLANTEAMIENTO DEL PROBLEMA

La aplicación móvil está desarrollada para una entidad financiera que ejerce su función de cooperativa de crédito y ahorro y que su ente de control es la súper solidaria el cual enfoca su control en la parte financiera y de asociados, es por esto, que la entidad financiera se alinea a las reglamentaciones establecidas por la Superintendencia Financiera, tales como las circulares externas 007 en la que indica, "Se expidió teniendo en cuenta el auge de la digitalización de los servicios financieros, la mayor interconectividad de los agentes y la masificación en el uso de canales electrónicos, entre otros, y complementa las normas existentes con relación a la administración de los riesgos operativos y la seguridad de la información."[1]

Y la circular 008 que informa sobre, "Mecanismos de protección de la información de los consumidores financieros al realizar operaciones monetarias usando los servicios de las pasarelas de pago."[2]

Para el año 2019, se buscaba la manera de integrar las herramientas 4.0 y agilidad en los procesos para permitir que los clientes tengan un mejor y fácil acceso a los servicios de la empresa, por lo que se realizó el relanzamiento de la aplicación móvil en la Play Store y AppGallery. Incluyendo nuevas opciones que permiten realizar consultas de saldos, realizar compras de boleterías, pago de servicios, manejo de cuentas y ahorros.

---

[1] **SUPERINTENDENCIA FINANCIERA DE COLOMBIA.** Comunicados de prensa. [sitio web] Bogotá: 05 de junio de 2018. [Consultado: 19 de abril de 2020]. Disponible en: https://www.superfinanciera.gov.co/jsp/Publicaciones/publicaciones/loadContenidoPublicacion/id/10097769/f/0/c/00

[2] Ibíd.

Al ser una aplicación que manipula información sensible es importante que esta tenga la suficiente seguridad, ya que el decreto 1377 de 2013[3] habla sobre el uso adecuado de la información y si es mal utilizada puede generar efectos negativos en la sociedad.

El grupo de desarrollo junto con el área de innovación de la empresa, no tomaron las mejores prácticas de seguridad para hacer la implementación y puesta en marcha del aplicativo, y esto puede generar un problema en la compañía si se llegan a presentar un incidente con base al aplicativo. Adicionalmente, esto que hace posible que la aplicación tenga un alto riesgo de ser vulnerada o que se le pueda capturar información sensible.

Debido a esto surge la siguiente pregunta: ¿Cómo la metodología OWASP TOP 10 Mobile v2016 podría mejorar la seguridad de la aplicación móvil?

La manera más práctica para dar respuesta a este interrogante es que la entidad financiera no tiene un área o personal que tenga un control o metodología enfocada a la seguridad de móvil o nuevas tecnologías. Es por esto que, se presenta la solicitud a la gerencia, para que autoricen el permiso de realizar el análisis de seguridad al aplicativo móvil, con el fin de determinar si el aplicativo es seguro y no expone la información de los usuarios y para que el área de desarrollo tenga un enfoque metodológico para ir creando los parámetros, controles y/o políticas iniciales que permitan garantizar que la aplicación está cumpliendo a cabalidad las circulares y que está publicando una aplicación móvil segura para los asociados, además de ir dejando documentado todo el procedimiento que se debe tener para cumplir los requerimientos mínimos y evitar posibles sanciones.

---

[3] **COLOMBIA, MINISTERIO DE COMERCIO, INDUSTRIA Y TURISMO**. Decreto Número 1377 de 2013. 2013. [En línea] [Consultado: 19 de abril de 2020]. Disponible en: https://www.mintic.gov.co/portal/604/articles-4274_documento.pdf

3 MARCO REFERENCIAL

3.1 MARCO TEÓRICO

Con el propósito de presentar un referente que permita a los ingenieros de financiera tener en cuenta los problemas de seguridad que se presentan en los aplicativos que se desarrollan y los efectos que se pueden dar si no se corrigen a tiempo.

Ceballos y Marulanda exponen que. “Las aplicaciones móviles tienen una funcionalidad específica para lo cual fueron creados, pero todas tienen como patrón común que sus desarrollos dependen de la calidad del software, lo que permitirá además que los clientes reciban productos seguros, ya sean desarrollados por terceros o propios.”[4]

Y como lo indica MIFSUD, “La seguridad informática consiste en la implantación de un conjunto de medidas técnicas destinadas a preservar la confidencialidad, la integridad y la disponibilidad de la información, pudiendo, además, abarcar otras propiedades, como la autenticidad, la responsabilidad, la fiabilidad y el no repudio.”[5]

---

[4] **CEBALLOS, Julián y MARULANDA, Cesar.** Una Revisión de metodologías seguras en cada fase del ciclo de desarrollo de desarrollo de software. 2012. [En línea] Articulo. Medellín: Universidad de San Buenaventura Medellín. [consulta: marzo de 2020]. Disponible en internet: http://web.usbmed.edu.co/usbmed/fing/v3n1/v3n1a2.pdf

[5] **MIFSUD, Elvira**. Introducción a la seguridad informática - Seguridad de la información/Seguridad informática. España. 2012 [En línea]. [consulta: marzo de 2020]. Disponible en: http://recursostic.educacion.es/observatorio/web/ca/software/softwaregeneral/1040-introduccion-a-la-seguridadinformatica?start=1.

Por lo tanto, cada desarrollador debe garantizar un mínimo de seguridad, debido a que en cada aplicación expuesta se manipulara una cantidad indefinida de información confidencial. Tal como lo indica Guzmán y Forero “la seguridad de un sistema adquiere importancia directamente proporcional al valor de los datos que contiene.”[6] Por lo que cada vez es de mayor importancia mantenerla segura.

Mediante el uso del guía OWASP se pretende presentar documentado las vulnerabilidades encontradas en la aplicación que pueden afectar tanto su funcionamiento, como llegar a generar perdida de información. Puesto que estas aplicaciones y/o dispositivos son capaces de almacenar datos y como lo indican Guzmán y Forero “es importante tener presente que estas aplicaciones manejan información sensible que puede ser expuesta por un delincuente informático.”[7]

Saltzer y Schroeder, han encontrado útil ubicar posibles violaciones de seguridad en tres categorías. Las cuales son: “Lanzamiento de información no autorizada, Modificación de información no autorizada, Denegación de uso no autorizada” [8] que pueden colocar en riesgo la información de la empresa y de las personas que consumen los servicios de la aplicación.

---

[6] **GUZMÁN, jeyson y FORERO, Leonel**. Análisis de vulnerabilidades y seguridad de dispositivos móviles con sistema operativo iOS 6.1.4 [En línea]. Proyecto de Grado. Bogotá: Universidad Piloto De Colombia. 2013 [consulta: marzo de 2020]. Disponible en: http://polux.unipiloto.edu.co:8080/00001524.pdf

[7] Ibíd. 4. p. 23.

[8] **SALTZER, Jerome y SCHROEDER, Michael**. The Protection of Information in Computer Systems [En línea]. Invited Paper; Reino Unido: 1975 [consulta: marzo de 2020]. Disponible en: https://www.cl.cam.ac.uk/teaching/1011/R01/75-protection.pdf

Pero no solamente son las aplicaciones las que pueden tener fallos de seguridad ya que gran parte de los dispositivos móviles están con sistema operativo Android que es más vulnerable que el sistema operativo iOS, para el caso de Colombia en el año 2019 el 70.05% de los equipos son Android.[9] En el mismo 2019 para Android se reportaron 514 fallos de seguridad tal como lo reporta Giusto[10] con información del CVE y un total de 368 para iOS.

Por eso es importante proteger los aplicativos e infraestructura que maneja información sensible, por que como lo demuestra Zhang en su libro: "Los médicos o las familias de personas mayores pueden usar computadoras de escritorio y teléfonos inteligentes para acceder de forma remota a estos registros de salud a través de MHN. En caso de cualquier emergencia, como caídas y problemas cardíacos, los dispositivos portátiles pueden informar automáticamente estado de salud del paciente a sus médicos y familiares a través de MHN."[11]

Adicionalmente Zhang[12] en su libro Security and Privacy for Mobile Healthcare Networks, indica que el sistema medico de EEUU en el 2011 registró cerca de 150 exabytes de información médica de pacientes, los cuales si llegan a ser accedidos por un delincuente informático puede llegar a perjudicar los miles de pacientes.

---

[9] **JKIELTY**, Android v iOS market share 2019. [en línea]. DeviceAtlas. Ireland. 9 de septiembre del 2019. [Consultado: 15 de marzo de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share

[10] **GIUSTO**, **Denise**. Descienden las detecciones de malware en Android y crecen en iOS [en línea]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-android-crecen-ios/

[11] **ZHANG, Kuan.** Security and Privacy for Mobile Healthcare Networks. 2 ed. New York: Springer. 2015, 2 p. ISBN: 978-3-319-24715-1.

[12] Ibíd. P 81.

Y el resultado de esto es poder guiar la empresa a que adopte unos mejores controles de seguridad al momento de realizar un despliegue de cualquier aplicativo.

3.2 MARCO CONCEPTUAL

En este proyecto se pretende validar la **app móvil**[13] de la entidad financiera, como es definida por debitoor que es un programa que funciona para ejecutar una o varias tareas en equipos móviles y tabletas. Mediante uno de los programas llamados **Android Debug Bridge (adb)** [14], que en un programa que permite comunicarse con un dispositivo móvil. este programa permite, como instalar y depurar apps y proporciona accesos elevados, a los cuales se les llama **Root**[15] estos derechos y privilegios necesarios para leer y escribir a todos los archivos y programas en el **Sistema operativo**[16] que es el programa en el cual se ejecutan procesos básicos o avanzados y que permite a un usuario interactuar con el sistema.

---

[13] **DEBITOOR**. App móvil - ¿Qué es una app móvil? [en línea]. Debitoor. España. (15 de marzo de 2017). [Consultado: 15 de marzo de 2020]. Disponible en: https://debitoor.es/glosario/app-movil

[14] **DEVELOPERS**. Android Debug Bridge (adb) [en línea]. Developers. España. (2018). [Consultado: 20 de marzo de 2020]. Disponible en: https://developer.android.com/studio/command-line/adb?hl=es-419

[15] **VENTURINI**, **Guillermo**. ¿Qué es rootear? [en línea]. Tecnología Fácil. Esperanza, Santa Fe. (2016). [Consultado: 20 de marzo de 2020]. Disponible en: https://tecnologia-facil.com/que-es/que-es-rootear/

[16] **REAL ACADEMIA ESPAÑOLA.** Sistema [en línea]. RAE. España. (2020). [Consultado: 18 de abril de 2020]. Disponible en:http://buscon.rae.es/draeI/SrvltObtenerHtml?origen=RAE&LEMA=sistema&SUPIND=0&CAREXT=10000&NE DIC=No#sistema_operativo.

3.2.1 Sistema operativo android

Como es bien conocido, Android está basado en código abierto de Linux orientado a un sinfín de dispositivos y múltiples plataformas.

La historia de Android se remonta hasta el 2005 así como lo comenta Castillo: "Android era un sistema operativo que estaba desarrollando una compañía de nombre Android Inc. y que fue adquirida por Google en julio del año 2005. Desde entonces siempre que se habla de este sistema operativo se hace referencia a la compañía del buscador, aunque es la Open Handset Alliance."[17]

Pero solo hasta el 2008, como lo indica Castillo[18] se pudo ver el sistema operativo funcionando en un equipo móvil el cual era un HTC modelo T-Mobile G1, y fue desarrollado junto con Google en su momento.

A través del tiempo se han presentado diferentes versiones de Android las cuales se desarrollaron así:

[17] **CASTILLO, Lucho**. Historia de Android [en línea]. EEUU. (5 de noviembre de 2012). [Consultado: 15 de abril de 2020]. Disponible en: https://github.com/LuchoCastillo/AndroidOS/blob/master/data/historia.rst

[18] Ibíd.

Tabla 1. Versiones del sistema operativo Android

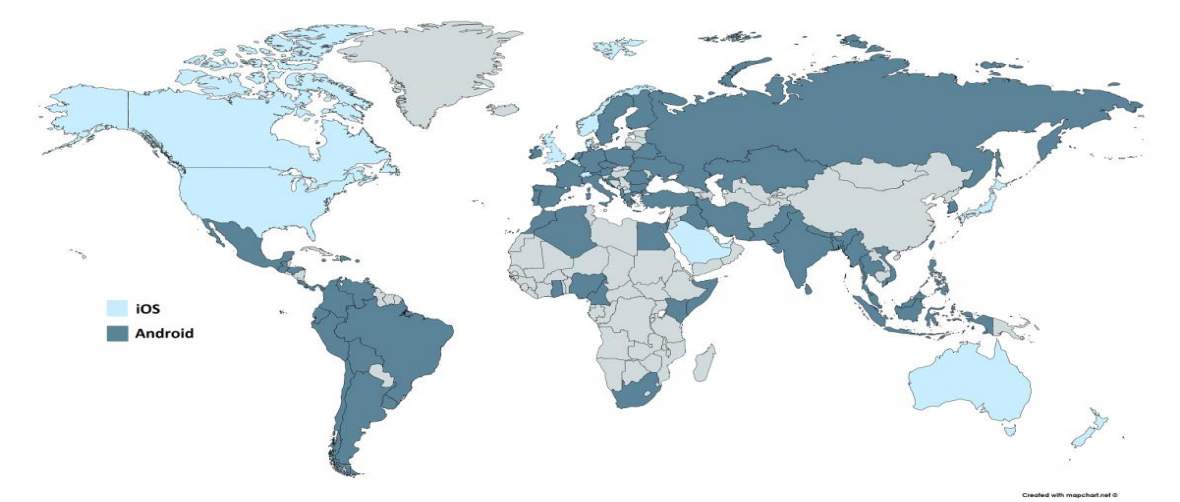
Versiones de Android	
Android Cupcake (Android 1.5) – abril 2009	Android Lollipop (Android 5.0 – 5.1) – noviembre 2014
Android Donut (Android 1.6) – septiembre 2009	Android Marshmallow (Android 6.0) – octubre 2015
Android Éclair (Android 2.0 – 2.1) – octubre 2009	Android Nougat (Android 7.0) - agosto 2016
Android Froyo (Android 2.2) – mayo 2010	Android Oreo (Android 8.0) - agosto 2017
Android Gingerbread (Android 2.3) – diciembre 2010	Android Pie (Android 9.0) - agosto 2018
Android Jelly Bean (Android 4.1 – 4.3) – octubre 2011	Android 10 (conocido anteriormente como Android Q) - marzo de 2019
Android Ice Cream Sandwich (Android 4.0) – Julio 2012	Android 11 - febrero de 2020
Android KitKat (Android 4.4) – octubre 2013	

Fuente: GARZON**,** Juan**.** Android 11 a Android 1.5: Cada versión de Android y sus novedades [en línea]. EEUU. (19 de febrero de 2020). [Consultado: 1 de abril de 2020]. Disponible en: https://www.cnet.com/es/imagenes/android-11-novedades-actualizacion-android-historia-google/

3.2.2 Sistema operativo iOS vs android

En una comparativa mundial que presenta Device Atlas, muestra el gusto que se tiene hacia un sistema operativo. Este comparativo se evidencia en la Figura 2.

Figura 1. Mapa mundial de la elección de un sistema operativo

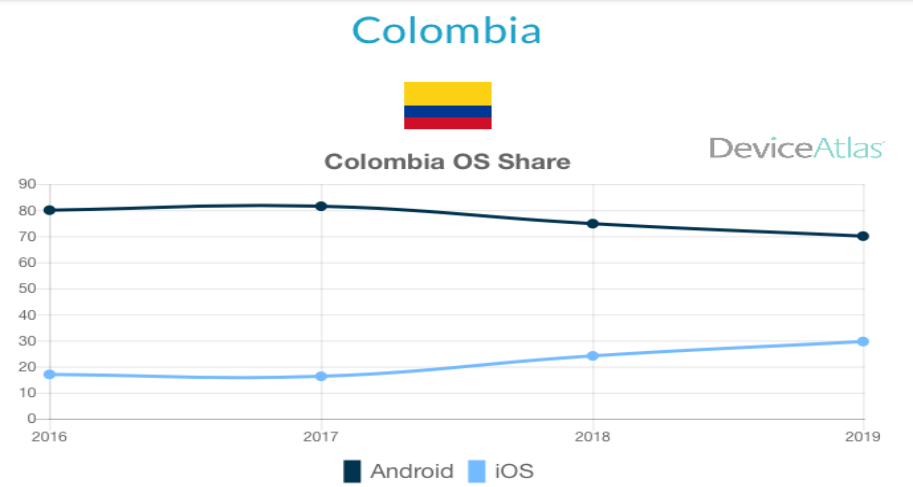


Fuente**:** DEVICE ATLAS, Android v iOS market share 2019 [imagen]. deviceatlas [Consultado: 3 de abril de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share

Ahora en un plan más limitado, el informe presentado por Device Atlas[19] muestra que para el 2019, el 70.05% de dispositivos móviles tienen sistema Android, frente a un 29.62% de iOS y con un 0.33% perteneciente a otros sistemas, tal como se presenta en la Figura 1.

[19] **DEVICE ATLAS**, Android v iOS market share 2019 [En línea]. Device3atlas, EEUU [Consultado: 4 de abril de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share

Figura 2. Porcentaje de uso de Android, frente a iOS



Fuente**:** DEVICE ATLAS, Android v iOS market share 2019 [imagen]. deviceatlas [Consultado: 3 de abril de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share

3.2.3 Comparaciones técnicas

La figura 3 está basada en la revisión de seguridad, las amenazas y malwares que realizó Fattoh AI-Qershi, Muhammad AI-Qurishi, Sk Md Mizanur Rahman, y AtifAI-Amri.

Figura 3. Comparación de seguridad iOS vs Android

Comparison Criteria		iOS	Android
Model Defense Techniques/ Mechanisms	*Code Signing*	*YES*	*YES*
	Application Permissions	*NO*	*YES*
	Sandboxing	More effective	Less effective
	Data Encryption	*YES*	*NO*
	Memory Protection	*YES* ASLR	*YES* MMU + ASLR (Android V4)
	Code Protection	*YES* DEP	*YES* Java Type Safety + DEP (Android V2.3)
	Availability of Antivirus tools	*YES* Few	*YES* Many
	Component Protections	Not a component based	*YES*
Vulnerabilities [8]	*Development Tools Availability*	Partial	*YES*
	Development Friendliness	*NO*	*YES*
	Installation Vectors	Restricted	Multiple
	Application Portability	*YES*	*YES*
	Unofficial Repositories	*NO*	*YES*
Strengths [8], [3]	*Application Testing*	*YES*	*NO*
	Application Remote Removal	*YES*	*YES*
	Distribution Cost	*YES*	*NO*
	API Restrictions	*NO*	*YES*
	Application Signing	*YES*	*YES*
	Keychain	*YES*	*NO*
	Authentication	*YES*	*YES*
	Device Wipe	*YES* Locally and Remotely	*YES* Remotely only
	Device Firewall	*NO*	*NO*
	Corporate Managed Email	*YES*	*NO*
	Virtualization	*YES* Virtual native OS	*YES* Virtual native Apps
	Security Certifications (FIPS 140-2)	*YES*	*YES*
Security Applications [20]	*Number*	1	6
	Examples	Lookout Mobile Security	Norton Mobile Security Lite, Kaspersky Security 9, iCareMobile
Implemented Solutions [20]	*Number*	17	24
Rating	[22] *out of 5*	3.4	3 (Android V 3,4)
	[2] *out of 5*	1.70 (iOS V5)	1.37 (Android V2.3)

Fuente**:** Fattoh Al-Qershi, Muhammad Al-Qurishi, Sk Md Mizanur Rahman, and AtifAl-Amri. Android vs. iOS: The Security Battle [en línea]. Riyadh, Kingdom of Saudi Arabia [Consultado: 5-de mayo de 2020]. Disponible en: https://www.researchgate.net/publication/268744667_ios_vs_android_the_security_battle#pf5

En la Figura 3, se puede ver que los autores tomaron diferentes puntos de referencia que impactan directamente tanto el uso del sistema operativo como la seguridad que tiene cada uno de ellos.

Y que al final entregan un resultado que en una parte ya se había expuesto el cual es la cantidad de dispositivos móviles Android que hay. “Android alcanzó el liderazgo y ganó las dos carreras en las ventas sobre iOS. Esto podría explicarse por aspectos diferentes a la seguridad. Android no se limita a un teléfono inteligente específico, sino que es el sistema operativo para diferentes dispositivos y el primero en las ventas (dispositivos Samsung)”[20]

Por otra parte, Fattoh AI-Qershi, Muhammad AI-Qurishi, Sk Md Mizanur Rahman, and AtifAI-Amri, muestran que el sistema operativo iOS es mucho más seguro y estricto en sus permisos y esto se puede ver no solo en sus celulares sino también en sus equipos de cómputo Mac, Mientras que Android al tener una filosofía más flexible hace o permite que sus Os sea más vulnerable, pero su constante actualización genera una capa de defensa o así lo exponen:

> Al buscar los modelos de seguridad de ambos sistemas operativos, podríamos concluir que usan diferentes políticas de defensa de seguridad además de algunos mecanismos similares. El modelo iOS es originalmente un modelo de seguridad estricto, en capas y multifacético. Por otro lado, el modelo de Android se basa en gran medida en el mecanismo basado en permisos más otros mecanismos que se heredan de los componentes de Android, como el lenguaje Java y su sistema operativo central, es decir, Linux. iOS parece ser más seguro que Android, pero este último intenta atraparlo mediante la actualización continua.[21]

---

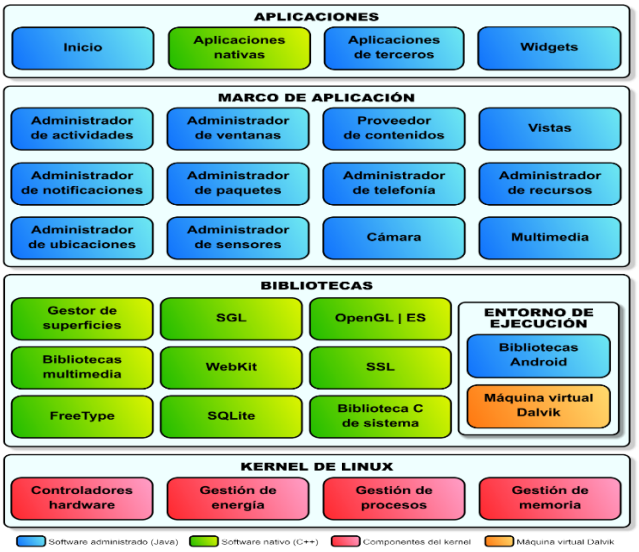
[20] **Fattoh AI-Qershi, Muhammad AI-Qurishi, Sk Md Mizanur Rahman, and AtifAI-Amri**. Android vs. iOS: The Security Battle [en línea]. (11 de enero de 2014) Riyadh, Kingdom of Saudi Arabia [Consultado: 5 de mayo de 2020]. Disponible en: https://www.researchgate.net/publication/268744667_ios_vs_android_the_security_battle#pf5 [Traducción: propia].

[21] Ibíd.

3.2.4 Arquitectura de android

El sistema operativo Android de estructura en 4 capas o base que cumplen con una función específica, “Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez los suyos propios a las capas de niveles superiores”[22]

Figura 4. Arquitectura de Android



Fuente**:** VICO, Ángel J. La columna 80, El blog técnico-personal de Ángel J. Vico… en español [imagen]. Arquitectura de Android [Consultado: 1 de abril de 2020]. Disponible en: https://columna80.wordpress.com/2011/02/17/arquitectura-de-android/

[22] **UNIVERSIDAD CARLOS III DE MADRID.** Software de Comunicaciones [en línea]. España. (1 de febrero de 2018). [Consultado: 1 de abril de 2020]. Disponible en: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android

3.2.4.1 Aplicación

En la primera capa se encuentran las todas las aplicaciones que posee el sistema Android y Vico expone más a fondo como están compuestas las aplicaciones “se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, tanto las nativas (programadas en C o C++) como las administradas (programadas en Java), tanto las que vienen de serie con el dispositivo como las instaladas por el usuario.”[23]

3.2.4.2 Framework de aplicaciones

Son el conjunto de herramientas que se utilizan al ejecutar una aplicación que se encuentra en un nivel superior y que se apoya en las bibliotecas las cuales están en un nivel inferior, aunque la Universidad Carlos III expone que “Toda aplicación que se desarrolle para Android, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo "framework", representado por este nivel.”[24]

---

[23] **VICO, Ángel J**. La columna 80, El blog técnico-personal de Ángel J. Vico… en español [en Línea]. Arquitectura de Android (17 de febrero de 2011). [Consultado: 1 de abril de 2020]. Disponible en: https://columna80.wordpress.com/2011/02/17/arquitectura-de-android/

[24] **UNIVERSIDAD CARLOS III DE MADRID.** Op. cit. 34

3.2.4.3 Bibliotecas

Las bibliotecas son la siguiente capa de la arquitectura de Android, la cual está definida por Muñoz como: “La capa que está escrita en lenguajes C y C++, las cuales tienen como función entablar comunicación entre la capa de abstracción de hardware con las API (Application Programming Interface - Interfaz de Programación de Aplicaciones) y las aplicaciones mismas. Se caracterizan por tener la extensión **.so**, y funcionan de la misma forma a los archivos **.dll** en Windows.”[25]

3.2.4.4 Kernel de linux

Ya que el núcleo es la parte principal del sistema Android Vico presenta una definición que permite entender su función:

> El núcleo del sistema operativo Android es un kernel Linux versión 2.6, similar al que puede incluir cualquier distribución de Linux, como Ubuntu, solo que adaptado a las características del hardware en el que se ejecutará Android (normalmente, un smartphone).
>
> Proporciona una capa de abstracción para los elementos hardware a los que tienen que acceder las aplicaciones. Esto permite que se pueda acceder a esos componentes sin necesidad de conocer el modelo o características precisas de los que están instalados en cada teléfono. De esta forma, si una aplicación necesita, por ejemplo, la brújula, podrá utilizar la que incluya el teléfono, sea cual sea. Para cada elemento hardware del teléfono existe un controlador (o *driver*) dentro del kernel que permite utilizarlo desde el software.[26]

---

[25] **VANEGAS, Carlos Alberto.** Citado por: **MUÑOZ, Yamir.** Estado del arte vulnerabilidades de seguridad en sistemas operativos móviles andrioid y iOS [en línea]. trabajo de monografía como requisito de grado para optar el título de especialista en seguridad informática. Bucaramanga. universidad nacional abierta y a distancia – unad escuela de ciencias básicas, tecnologías e ingenierías, 2019. 41 p. [Consultado: 2 de abril de 2020]. Disponible en: Repositorio Educativo Digital UNAD. https://repository.unad.edu.co/bitstream/handle/10596/25396/%20%09yamunozca.pdf?sequence=1

[26] **VICO, Ángel J**. Op. cit. 35

3.2.4.5 Entorno de ejecución

El entorno de ejecución como se visualiza en la Figura 4. Se encuentra en la misma capa de bibliotecas, pero estas se basan en una máquina virtual, pero se ejecutan sobre java.

Como indica González[27] en el que explica cómo funciona Dlavik y su sucesor ART, que a partir de Android 2.2 Froyo, el sistema operativo utilizaba JIT (Just-In-Time) pero google realizó el cambio al nuevo sistema ART el cual crea un archivo de compilación tras la instalación de una aplicación, lo que permite hacer una sola compilación del programa que se está ejecutando.

3.2.5 Taxonomía de las apps

El sistema operativo Android, posee 3 tipos de aplicaciones y cada una tiene una composición y método de desarrollo.

3.2.5.1 Aplicaciones nativas

Las aplicaciones nativas son aquellas que están desarrolladas específicamente para el dispositivo, ya que estas aplicaciones tienen la función de tener un rendimiento más rápido y son más confiables en comparación a las otras apps, adicionalmente que estas aplicaciones, por su entorno pueden tener acceso de manera directa a varios componentes del celular.

Ahora y como lo indica el grupo GitHub, “La desventaja más obvia de las aplicaciones nativas es que solo se dirigen a una plataforma específica. Para

---

[27] **GONZÁLEZ, Carlos**. Android Ayuda, ¿Qué es ART? Android Run time, el sucesor de Dalvik [en Línea]. Android Ayuda (07 de agosto de 2019). [Consultado: 1 de abril de 2020]. Disponible en: https://androidayuda.com/android/que-es/art-android-runtime/

construir la misma aplicación para Android y iOS, uno necesita mantener dos bases de código independientes o introducir herramientas de desarrollo a menudo complejas para portar una base de código única a dos plataformas”[28]

Tabla 2. Ventajas y Desventajas de Aplicaciones Nativas

Ventajas	Desventajas
- Acceso completo al dispositivo - Mejor IU y UX - Facilidad al cargarlas a los stores - Avisos y mensajes Push	- + Costos - Codigo poco reutilizable - Poca flexibilidad

Fuente: propia

3.2.5.2 Aplicaciones basadas en web

Las aplicaciones basadas en web, son sitios web diseñados específicamente para tener una experiencia de uso similar a una aplicación normal. Estas aplicaciones generalmente se desarrollan en HTML5, estas aplicaciones presentan limitaciones ya que están ligadas al navegador y su rendimiento es mucho menor.

Una de “La mayor ventaja es la reducción de los costos de desarrollo y mantenimiento asociados con una base de código único, así como también permitir a los desarrolladores distribuir actualizaciones sin comprometer las tiendas de aplicaciones específicas de la plataforma”[29]

[28] **GITHUB**. Mobile App Taxonomy. [en Línea]. Mobile Security Testing Guide (s.f). [Consultado: 11 de noviembre de 2020]. Disponible en: https://mobile-security.gitbook.io/mobile-security-testing-guide/overview/0x04a-mobile-app-taxonomy

[29] **Ibíd.** Par. 8

Tabla 3. Ventajas y Desventajas de Aplicaciones Web

Ventajas	Desventajas
- Económico - Disponibilización rápida - Update - Código reutilizable	- Conectividad 100% - Acceso a características del equipo - No disponible en stores.

Fuente: propia

3.2.5.3 Aplicaciones hibridas

Las aplicaciones hibridas son las utilizadas en muchas empresas ya que estas aplicaciones se ejecutan y trabajan como una aplicación nativa y a su vez estas aplicaciones adquieren todo el desarrollo que traen las apps nativas y web. Por otra parte, realizar el desarrollo en modo hibrido permite tener "una base de código puede dar como resultado múltiples aplicaciones que se dirigen a diferentes plataformas, con una IU muy parecida a la de la plataforma original para la que se desarrolló la aplicación."[30]

Tabla 4. Ventajas y Desventajas de Aplicaciones Hibridas

Ventajas	Desventajas
- Acceso al dispositivo - Reutilización de contenido - Posibilidad de subir al play store	- Acceso a internet - Condicionada a revisión - Experiencia de usuario.

Fuente: propia

[30] **Ibíd.** Par. 10

3.3 MARCO LEGAL

El presente trabajo se encuentra en el marco de las leyes colombianas las cuales establecen el tratamiento de los datos personales y los delitos informáticos. Estas normas y leyes que se establecen son:

La LEY 1273 DE 2009 la cual indica que, “Por medio del cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado – denominado “De la Protección de la información y de los datos”- y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones.”[31]

Esta ley indica y hace relevancia a los delitos informáticos que se pueden presentar con el fin de tener acceso no permitido y/o conseguir información de manera ilegal. Por otro lado, se encuentra como ya se había mencionado antes, la ley estatutaria 1581 del 2012, Habeas Data, que se enfoca en el tratamiento de los datos que se manejan en las empresas y los delitos que se pueden incurrir al violar alguno de sus artículos citados.

> Los principios sobre protección de datos serán aplicables a todas las bases de datos, incluidas las exceptuadas en el presente artículo, con los límites dispuestos en la presente ley y sin reñir con los datos que tienen características de estar amparados por la reserva legal. En el evento que la normatividad especial que regule las bases de datos exceptuadas prevea principios que tengan en consideración la naturaleza especial de datos, los mismos aplicarán de manera concurrente a los previstos en la presente ley.[32]

---

[31] **COLOMBIA. CONGRESO DE LA REPÚBLICA. Ley 1273** (05, 5 de enero de 2009). Por medio de la cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado [en línea]. Santa Fe de Bogotá, D.C.: Diario Oficial No. 47.223. [Consultado: mayo 6 de 2020]. Disponible en: http://www.secretariasenado.gov.co/senado/basedoc/ley_1273_2009.html

[32] **COLOMBIA. CONGRESO DE LA REPÚBLICA. LEY ESTATUTARIA 1581**(18 de octubre de 2012). Por la cual se dictan disposiciones generales para la protección de datos personales. [en línea]. Santa Fe de Bogotá,

3.4 MARCO ESPACIAL

El proyecto está desarrollado para la empresa financiera, la cual tiene su oficina principal en la ciudad de Bogotá y la ejecución del proyecto fue realizado en los laboratorios y espacios que se tienen para toda el área de tecnología.

Esto permitió mantener y prevalecer la seguridad de los datos encontrados, además de evitar que las pruebas realizadas fueran modificadas y alterar los resultados de las pruebas.

D.C.: Diario Oficial No. 48.587 [Consultado: mayo 6 de 2020]. Disponible en: http://www.secretariasenado.gov.co/senado/basedoc/ley_1581_2012.html

3.5 MARCO METODOLÓGICO

El proyecto que se desarrollara tiene como objetivo realizar la búsqueda de fallos, backdoors y/o vectores de ataque que se puedan presentar en la aplicación, lo que permitirá tanto a los desarrolladores y a los proveedores, tener un mejor conocimiento de la arquitectura que posee la aplicación haciendo que puedan modificar el código, métodos compilar y la aplicación de controles de seguridad.

Para ejecutar este análisis se procederá a dividirlo en fases, las cuales permitirán solicitar los permisos necesarios de manipulación de la aplicación, listar las herramientas a utilizar en el desarrollo del análisis, validar permisos de la aplicación, obtener información relevante y revisar las vulnerabilidades.

Siguiendo las buenas prácticas para desarrollar el proyecto se desglosan las fases a ejecutar.

Documentación de la aplicación y el sistema operativo Android, el cual está enfocado el conocer acerca de la aplicación, sus versiones y como está compuesta, además de tener un mayor conocimiento del sistema operativo Android.

Preparación de los entornos de prueba, una vez documentada la aplicación y teniendo conocimiento de cual versión de Android se trabajará, se procederá configurar los equipos y herramientas necesarias para realizar el análisis de seguridad, estas se irán configurando sobre máquinas virtuales que hospedarán los emuladores del Sistema Operativo Android y las herramientas a utilizar.

Posteriormente se procederá con el análisis funcional y de su comportamiento de la aplicación en el cual se utilizarán herramientas de captura de información, con esto se procederá a realizar las pruebas de penetración y búsqueda de vulnerabilidades

Por último, se realizará la captura y documentación de evidencias, pruebas realizadas, para luego crear un documento final en el que se evidencie lo realizado y se agreguen las recomendaciones para mitigar lo evidenciado.

Un punto importante, es que la aplicación se ejecuta sobre las versiones más recientes de Android y para esta prueba se utilizará la versión 9 (Android Pie) ya que es una de las versiones que más se puede encontrar en los móviles de gama baja. Y que el desarrollador ha puesto un limitante a las versiones lo que hace un poco más segura la aplicación.

3.5.1 Recursos necesarios

Los recursos que se presentan a continuación están unificados tanto el costo que se utilizará en un recurso humano y en materiales o desplazamientos a lugares.

Tabla 5. Recursos necesarios para el desarrollo del proyecto

RECURSO	DESCRIPCIÓN	PRESUPUESTO
Equipo Humano	**1 persona, que realizará la investigación y las pruebas**	**$ 2.500.000**
Equipos y Software	**1 computador con los requerimientos necesarios, se utilizará en todo el proyecto.**	**$ 2.000.000**
	1 sistema Operativo Kali, se instalará a partir del mes 5, en una máquina virtual	**$ 0.00**
	1 máquina virtual con el sistema operativo Genymtion, para el sistema Android	**$ 0.00**
	Máquina virtual con el sistema operativo Tamer, para emular los sistemas	**$ 0.00**
	Se utilizarán programas open source para el Pentesting entre los cuales están, APKTool, Dex2jar, ADB y otras herramientas que se utilizan desde internet.	**S 0.00**
Viajes y Salidas de Campo	**Consultorías y Asesorías, visita a bibliotecas y búsqueda de información adicional**	**$ 60.000**
Materiales y suministros	**1 cable UTP para conexión a internet del equipo**	**$ 7.000**
	1 cable para conectar celular Android	**$ 15.000**
	Energía, no tiene costo ya que se utilizará el de la empresa	**$ 0.0**
	Internet, no tiene costo ya que se utilizará el de la empresa	**$ 0.0**
	Papelería, se comprará una resma de papel para entregar el informe	**$ 20.000**
	Libros Guía, libros que se deban comprar o manuales para ejecutar las pruebas	**$ 200.000**
Bibliografía	**Libros Guía, libros que se deban comprar o manuales para ejecutar las pruebas**	**$ 200.000**

Fuente**:** El Autor, MIRANDA, Jimmy.

4 DESARROLLO DE LOS OBJETIVOS

4.1 METODOLOGÍA OWASP MOBILE TOP 10 VERSION 2016

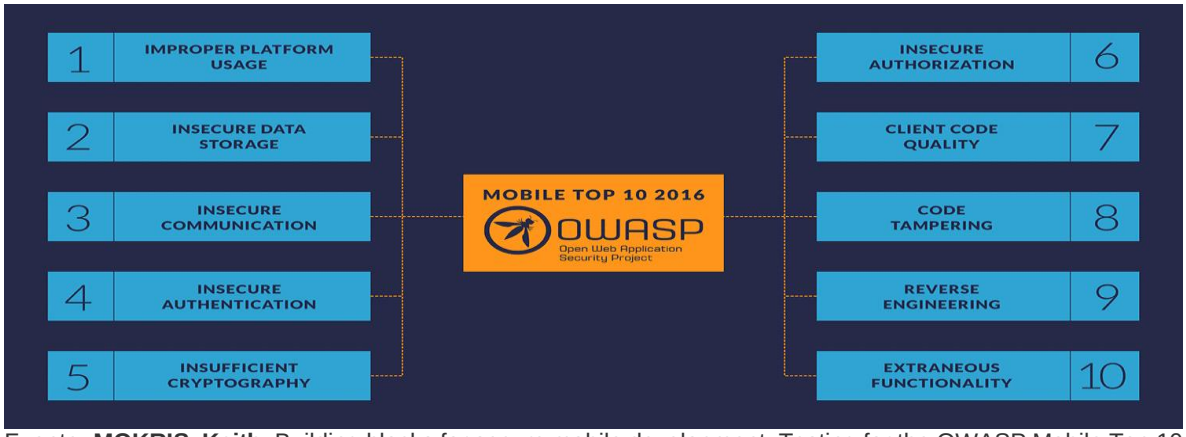
OWASP es un proyecto creado por una comunidad de ingenieros de sistemas o informática, en el cual desarrollan, comparten herramientas, documentos, aplicativos. En OWASP se tienen foros en los que de manera abierta y sin restricciones entregan e invitan a usar, probar y comentar las aplicaciones creadas para todo el público, Ahora y debido al creciente auge de las aplicaciones móviles, OWASP creó el proyecto Mobile top 10.

> El Proyecto de seguridad móvil de OWASP es un recurso centralizado destinado a brindar a los desarrolladores y equipos de seguridad los recursos que necesitan para construir y mantener aplicaciones móviles seguras. A través del proyecto, nuestro objetivo es clasificar los riesgos de seguridad móvil y proporcionar controles de desarrollo para reducir su impacto o probabilidad de explotación. El proyecto es un punto de partida para muchos proyectos diferentes de seguridad móvil dentro de OWASP. En este momento, puede encontrar los siguientes proyectos activos de seguridad móvil de OWASP[33]

Esta metodología como su nombre lo indica tiene 10 pasos los cuales están enfocados en analizar y detectar fallos o problemas que se presentan en la metodología.

[33] **OWASP Foundation**. OWASP Mobile Security Project [en línea]. OWASP Mobile segurito (8 de enero 2019.) [Consultado: 15 de marzo de 2020]. Disponible en: https://owasp.org/www-project-mobile-security/

Figura 5. OWASP top 10 Mobile



Fuente**: MOKRIS**, **Keith**. Building blocks for secure mobile development: Testing for the OWASP Mobile Top 10 [Imagen]. Now Secure. 13 de octubre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.nowsecure.com/blog/2016/10/13/secure-mobile-development-testing-owasp-mobile-top-10/

4.1.1 M1 – improper platform usage[34]

En el primer riesgo que se expone es el uso inadecuado de la plataforma o usos en la aplicación de controles en seguridad, y Saavedra lo expone como, "Esta categoría cubre el uso indebido de características de la plataforma o el no uso de los controles de seguridad, permisos de plataforma, uso indebido de TouchID, servicios de contraseñas (KeyChain IOS) o algún otro control de seguridad que sea parte del sistema operativo móvil. Lo que se valora en el ataque es la seguridad de cualquier API expuesta."[35] Esto debido al uso de algunos SDK en versiones muy viejas o incompatibles con el desarrollo.

[34] **MOKRIS**, **Keith**. Building blocks for secure mobile development: Testing for the OWASP Mobile Top 10 [Imagen]. Now Secure. 13 de octubre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.nowsecure.com/blog/2016/10/13/secure-mobile-development-testing-owasp-mobile-top-10/

[35] **SAAVEDRA, Fernando.** OWASP Top Ten Mobile Risks [en línea]. Audea. Madrid. España. (15 de octubre de 2019). [Consultado: 1 de abril de 2020]. Disponible en: https://www.audea.com/owasp-top-ten-mobile-risks/

4.1.2 M2 – insecure data storage[36]

Este riesgo que se ha identificado es uno de los más comprometidos ya que muchas veces, para poder acceder a una página o sitio específico, hacemos clic o aceptamos sin darnos cuenta de lo que aceptamos. Saavedra defina este riesgo como, “Desde aplicaciones de terceros que utilizan caché, cookies y otra información para recopilar datos protegidos, hasta que un adversario pueda obtener físicamente el dispositivo y ver información, debe manejar el almacenamiento de datos correctamente de varias maneras. Esto incluye la autenticación, el cifrado y el manejo adecuado de todas las funciones de almacenamiento en caché.”[37]

4.1.3 M3 – insecure communication[38]

Para la capa de sesión y aplicación del modelo OSI este riesgo es uno de los más importantes ya que Saavedra afirma que, “Los protocolos inseguros de enlace, versiones SSL incorrectas, negociación débil, la comunicación sin cifrar de datos sensibles, etc. Por lo que hay que comprobar que los desarrolladores (a menudo muy preocupados en cuanto a la protección del procedimiento de autenticación y los datos en reposo) hayan implementado un cifrado de los datos que se transmiten correctamente.”[39] Para que no se exponga la data y sea capturada en su proceso de transmisión.

---

[36] **MOKRIS**, **Keith.** Op. cit. 75

[37] **SAAVEDRA, Fernando** Op. cit. 76

[38] **MOKRIS**, **Keith.** Op. cit. 75

[39] **SAAVEDRA, Fernando** Op. cit. 76

4.1.4 M4 – insecure authentication[40]

Esta falla se encuentra comúnmente en aplicaciones, ya que la mayoría de apps no realiza autenticación biométrica o de dos factores, lo que le permite ser una aplicación vulnerable y como asegura Saavedra “La captura las nociones de autenticación del usuario final o gestión de sesión incorrecta. Esto puede incluir: 1 - No identificar al usuario en absoluto cuando sea necesario. 2 - No mantener la identidad del usuario cuando se requiere 3 - Debilidades en el manejo de sesiones”[41]

4.1.5 M5 – insufficient cryptography[42]

Este se presenta debido al mal uso de los cifrados de la aplicación o a un nivel bajo que es fácil de romper, Saavedra asegura que. “El uso incorrecto del cifrado es extremadamente común en las aplicaciones móviles. Los algoritmos de cifradas débiles, así como el procedimiento de cifrado / descifrado defectuoso, pueden ser fácilmente explotados y por ende lo que se valorara son los problemas donde se intentó la criptografía”[43] y en muchas aplicaciones no se tiene una manera exacta de saber si la aplicación tiene un certificado o llave que permita encriptar la información que se maneja en ella.

---

[40] **MOKRIS**, **Keith.** Op. cit. 75

[41] **SAAVEDRA, Fernando** Op. cit. 76

[42] **MOKRIS**, **Keith.** Op. cit. 75

[43] **SAAVEDRA, Fernando** Op. cit. 76

4.1.6 M6 – insecure authorization[44]

Una aplicación o sitio web que realice lo que argumenta Saavedra es una aplicación que debe ser analizada ya que genera un fallo de seguridad “capturar cualquier fallo en la autorización (por ejemplo, decisiones de autorización en el lado del cliente, navegación forzada, etc.). Es distinto de los problemas de autenticación (por ejemplo, inscripción de dispositivos, identificación de usuarios, etc.).”[45] hace fácil el ingreso de los atacantes y permite que se tenga información sensible.

4.1.7 M7 – client code quality[46]

Saavedra define muy bien este fallo ya que en general “Los problemas de implementación a nivel de código en el cliente móvil. Esto es distinto de los errores de codificación del servidor. Entrarían dentro de ella, cosas como desbordamientos de búfer, vulnerabilidades de cadena de formato y varios otros errores de nivel de código donde la solución es reescribir algún código que se esté ejecutando en el dispositivo móvil. Es decir, se centra en las vulnerabilidades creadas debido a errores de codificación.”[47] . Por lo que acá es importante que el desarrollador o su equipo aplique las mejores técnicas o métodos que hay a nivel de normas y tengan un DevSecoOps

[44] **MOKRIS**, **Keith.** Op. cit. 75

[45] **SAAVEDRA, Fernando** Op. cit. 76

[46] **MOKRIS**, **Keith.** Op. cit. 75

[47] **SAAVEDRA, Fernando** Op. cit. 76

4.1.8 M8 – code tampering[48]

Este riesgo generaliza en si a todas las aplicaciones las cuales sacan una versión y no corrigen los fallos o quedan mal, es por esto que Saavedra indica que, “Los parches binarios, modificación de recursos locales, incorrecta utilización de métodos y modificación de memoria dinámica. Por ende, esta categoría cubre cualquier modificación que el adversario pueda realizar en el código de la aplicación.”[49]

4.1.9 M9 – reverse engineering[50]

Muchas veces en las pruebas de vulnerabilidad los mismos programas pueden ejecutar esta ingeniería y facilitar los ataques y es algo que Saavedra lo asegura. “El análisis del núcleo binario final para determinar su código fuente, bibliotecas, algoritmos y otros activos. Esto puede utilizarse para explotar otras vulnerabilidades nacientes en la aplicación, así como para revelar información sobre los servidores backend, las claves criptográficas y la propiedad intelectual.”[51]

---

[48] **MOKRIS**, **Keith.** Op. cit. 75

[49] **SAAVEDRA, Fernando** Op. cit. 76

[50] **MOKRIS**, **Keith.** Op. cit. 75

[51] **SAAVEDRA, Fernando** Op. cit. 76

4.1.10 M10 – extraneous functionality[52]

Saavedra expone un de los errores que pueden cometer los desarrolladores al finalizar una aplicación, y casi siempre ocurre cuando se es presionado por el cliente. "Cuando los desarrolladores no eliminan funciones adicionales, creadas durante el proceso de desarrollo para facilitar la prueba de la aplicación y que pueden ser explotadas para realizar ataques contra la aplicación."[53]

4.2 VECTORES DE ATAQUE A DISPOSITIVOS MÓVILES

En el 2015 se presentaba un informe por parte de la empresa Tech Crunch[54] el cual indicaba que para el 2020 habría más de 6 mil millones de celulares en el mundo y de acuerdo con Ericsson Mobility[55], se estima que hay más de 9,5 mil millones de equipos móviles.

---

[52] **MOKRIS**, **Keith.** Op. cit. 75

[53] **SAAVEDRA, Fernando** Op. cit. 76

[54] **TECH CRUNCH**. 6.1B Smartphone Users Globally by 2020, Overtaking Basic Fixed Phone Subscriptions [en línea]. (2 de junio de 2015) Ingrid Lunden TechCrunch [Consultado: 5 de mayo de 2020]. Disponible en: https://techcrunch.com/2015/06/02/6-1b-smartphone-users-globally-by-2020-overtaking-basic-fixed-phone-subscriptions/#.n7ibu3d:RPIH [Traducción: propia].

[55] **ERICSSON MOBILITY**. The Ericsson Mobility Report [en línea]. (1 de octubre de 2019) Telefonaktiebolaget LM Ericsson [Consultado: 5 de mayo de 2020]. Disponible en: http://www.ericsson.com/news/141118-90percent-to-have-mobile-phones-by-2020-predicts-ericsson-mobility-report_244099435_c [Traducción: propia].

En un informe presentado por Giusto[56] a principios de enero, indica que, para finales del 2019, los reportes de vulnerabilidades de Android en CVE bajaron un 16% lo que significa que, de 613 reportes en el 2018, se pasaron a 514 en el 2019 tal como de evidencia en la Figura 5. Mientras que iOS tuvo 368 reportes en el 2019, un 194% frente al 2018, como de evidencia en la Figura 6.

Figura 6. Reportes de Android

Fuente**:** GIUSTO, Denise. Descienden las detecciones de malware en Android y crecen en iOS [Imagen]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-android-crecen-ios/

Figura 7. Reportes de iOS

Fuente: GIUSTO, Denise. Descienden las detecciones de malware en Android y crecen en iOS [Imagen]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-android-crecen-ios/

---

[56] **GIUSTO**, **Denise**. Descienden las detecciones de malware en Android y crecen en iOS [en línea]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-android-crecen-ios/

Estas vulnerabilidades reportadas, permiten que diferentes tipos de malware puedan explotarlas. Además de otros tipos de engaños que los atacantes puedan realizar para robar información, unos de estos tipos de malware o ataques son los que se presentan a continuación**.**

4.2.1 Filtración de datos

Como lo explica Kaspersky a continuación en la pérdida de datos que es una de las maneras más sencillas de ataque.

> Las aplicaciones móviles a menudo son la causa de la filtración involuntaria de datos. Por ejemplo, como señaló eSecurity Planet, las aplicaciones "riskware" plantean un problema real para los usuarios móviles, que les otorgan amplios permisos, pero no siempre comprueban la seguridad. El malware móvil utiliza un código de distribución nativo en sistemas operativos móviles populares, como iOS y Android, para difundir datos valiosos en redes corporativas sin levantar sospechas.[57]

[57] **KASPERSKY LAB**. Las siete amenazas principales para la seguridad móvil, teléfonos, Tablet y dispositivos de Internet móvil: qué nos deparará el futuro [en línea]. Kaspersky Lab. (8 de enero 2015.) [Consultado: 15 de marzo de 2020]. Disponible en: https://latam.kaspersky.com/resource-center/threats/top-seven-mobile-security-threats-smart-phones-tablets-and-mobile-internet-devices-what-the-future-has-in-store

4.2.2 Accesos wi-fi

Este es un de los puntos en los que miles de personas caen, debido a que muchos quieren mantener el mínimo consumo de datos del plan, entonces muchos son los pecadores, y pues, sin embargo, las redes Wi-Fi gratuitas generalmente son inseguras. Y con el ejemplo que presenta Kaspersky: en el que demuestra cómo se pierde información fácilmente: “De hecho, según V3, tres policías británicos que accedieron a participar en un experimento de seguridad asociado a redes inalámbricas gratuitas fueron fácilmente presa de ataques realizados por expertos en tecnología y vieron comprometidas sus cuentas de redes sociales, PayPal e, incluso, conversaciones a través de VoIP.”[58]

4.2.3 Suplantación de red

Otro de los fallos de seguridad que los atacantes aprovechan con regularidad, ya que muchas personas, le gusta tener al mínimo el consumo de datos, Kaspersky afirma que: “en ubicaciones públicas concurridas, como cafeterías, bibliotecas y aeropuertos. Los cibercriminales asignan a estos puntos nombres comunes, como "Wi-Fi gratuita del aeropuerto" o "Cafetería", lo que anima a los usuarios a conectarse. En algunos casos, los atacantes exigen a los usuarios crear una "cuenta" para acceder a estos servicios gratuitos”[59] y como se indica, solicitan crear contraseñas o permitir acceso mediante las redes sociales.

---

[58] **Ibíd**.

[59] **Ibíd**.

4.2.4 Ataques de phishing

El phishing es el ataque más común y en el que mayor gente cae, independiente como se ejecute, y como lo indica Kaspersky frente a este ataque: “Los usuarios móviles son más vulnerables porque a menudo son los primeros en recibir correos electrónicos aparentemente legítimos y caer en la trampa. El monitoreo del correo electrónico es fundamental. No hagas nunca clic en enlaces de correo electrónico desconocidos. Pueden resultar aún más difíciles de verificar en la pantalla más pequeña de un dispositivo móvil.”[60] Por esto siempre se debe abrir los correos en un computador, que ofrece más opciones de visualizar.

4.2.5 Spyware

Aunque muchos de los usuarios móviles descargan aplicaciones porque si y porque no, en estas descargas se corre el riesgo de quedar con un programa malicioso, pero lo que expone Kaspersky, es un punto que sube el nivel de ataque. “En numerosos casos, no es el malware lo que debe preocuparles, sino el spyware instalado por cónyuges, compañeros de trabajo o empleadores para rastrear sus desplazamientos y patrones de uso.”[61]

4.2.6 Criptografía quebrada

Infosec Institute en su guía de criptografía explica que este punto es uno de los más sensibles, ya que los ataque son efectuados sobre la aplicación o los servicios que se presta debido a que.

---

[60] **Ibíd**.

[61] **Ibíd**.

Se pueden producir casos de criptografía quebrada si los desarrolladores de aplicaciones utilizan algoritmos de cifrado débiles o cifrado seguro sin una implementación adecuada. Los desarrolladores utilizan algoritmos de cifrado que ya poseen vulnerabilidades conocidas para acelerar el proceso de desarrollo de aplicaciones y el resultado es que cualquier atacante decidido puede descifrar las contraseñas y obtener acceso. Por eso es responsabilidad de los desarrolladores y de las organizaciones aplicar estándares de cifrado antes de implementar las aplicaciones.[62]

4.2.7 Gestión inadecuada de las sesiones

Muchas de las aplicaciones bancarias realizan estos tipos de configuraciones en los portales y aplicaciones transaccionales. Y como lo afirma Kaspersky**:** "Con el fin de facilitar el acceso a transacciones en dispositivos móviles, numerosas aplicaciones usan "tokens", que permiten a los usuarios ejecutar varias acciones sin necesidad de volver a autenticar su identidad. La gestión inadecuada de las sesiones se produce cuando las aplicaciones comparten involuntariamente tokens de sesión con entidades maliciosas, lo que les permite hacerse pasar por usuarios legítimos."[63]

---

[62] **INFOSEC INSTITUTE**. ANDROID HACKING & SECURITY - PART 14, Broken Cryptography [en línea]. Infosec Institute (8 de enero 2019.) [Consultado: 15 de marzo de 2020]. Disponible en: https://resources.infosecinstitute.com/android-hacking-security-part-16-broken-cryptography/

[63] **KASPERSKY LAB**. Op. cit., 65

4.3 SELECCIÓN DE LAS HERRAMIENTAS PARA EL ANÁLISIS DE LA APP

Las herramientas que se presentan a continuación, todas son open source o están incluidas en el Sistema Operativo Kali Linux, adicionalmente de otras que se han visto en lo largo de la especialización.

- **ADB:** Es una herramienta de línea de comandos versátil que te permite comunicarte con un dispositivo. El comando ADB permite realizar una variedad de acciones en el dispositivo, como instalar y depurar apps, y proporciona acceso a un Shell de Unix que puedes usar para ejecutar distintos comandos en un dispositivo.[64]

- **Apk Combo:** Es una página que permite buscar y descargar cualquier tipo de APK que se encuentra en la Play Store.

- **Apk Downloader:** Es una página que permite buscar y descargar cualquier tipo de APK que se encuentra en la Play Store.

- **Apktool**: Permite realizar ingeniería inversa a partir archivos APK.[65]

- **Burp Suite**: Es una plataforma integrada para realizar pruebas Free de penetración de aplicaciones web y móviles. Entre las funcionalidades más importantes está el actuar como proxy para la interceptación de datos transmitidos por la red[66]

---

[64] **DEVELOPERS.** Android Debug Bridge (adb) [En línea] Android Studio [Consultado: 22 de noviembre de 2020]. Disponible en internet: https://developer.android.com/studio/command-line/adb?hl=es

[65]**APACHE 2.0 License**. Apktool. [En línea] GitHub, 2018. [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://github.com/androguard/androguard [Traducción: Propia]

[66]**PORTSWIGGER**. Burp Suite. The Burp Suite family [En línea] Burp Suite 2018. [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://portswigger.net/burp/ [Traducción: Propia]

- **Dex2Jar**: Es una herramienta que convierte archivos del tipo Free. DEX a código. Jar[67]

- **Docker:** Es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.[68]

- **Frida:** Es un conjunto de herramientas de instrumentación potente y extensible; entre sus muchas fortalezas, es ideal para probar y evaluar aplicaciones nativas de Android. Frida le permite interceptar datos recibidos y enviados por aplicaciones e inyectar su propio código en el proceso.[69]

- **Genymotion:** Es un software que emula los diversos sistemas Android, con diferentes versiones de Android y múltiples equipos.

- **JD**-**Gui:** es una herramienta gráfica (GUI: Graphic User Interface o Interfaz Gráfica de usuario) que nos permite visualizar la representación de las clases Java de los archivos .jar. De esta manera, podremos revisar la composición de la aplicación a analizar de forma organizada, entendiendo sus dependencias, interrelaciones, llamadas a recursos**.**

---

[67] **SOURCEFORGE.** Dex2jar**.** [En línea] dex2jar 2014. [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://sourceforge.net/projects/dex2jar/ [Traducción: Propia]

[68] **REDHAT.** Contenedores de Software, ¿Qué es DOCKER? [En línea] RedHat. s.f [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://www.redhat.com/es/topics/containers/what-is-docker

[69] **VALENTINE, Rob.** Pruebas de penetración, Serie de tutoriales de herramientas de penetración de Android: Frida. [En línea] INFOSEC 2018 [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://resources.infosecinstitute.com/topic/frida/

- **MobSF Mobile Security Framework:** es una herramienta Free que permite realizar análisis estáticos y dinámicos de aplicaciones Android y iOS[70]

- **Nmap** Es una utilidad que permite descubrir y auditar redes. Nmap permite levantar información de redes, servicios o servidores, para lo cual envía paquetes y analiza las respuestas a dichos paquetes[71]

- **Tamer "**Es una plataforma virtual para profesionales de seguridad de Android. Android Tamer es un entorno de máquina virtual basado en Debian 8 y precargado con herramientas y *scripts* para realizar test de penetración en Android."[72]

- **Virtual Box: "**Es una aplicación que sirve para hacer máquinas virtuales con instalaciones de sistemas operativos. Esto quiere decir que, si tienes un ordenador con Windows, GNU/Linux o incluso macOS, puedes crear una máquina virtual con cualquier otro sistema operativo para utilizarlo dentro del que estés usando."[73]

---

[70] **GITHUB. Mobile Security Framework.** [En línea] Mobile Security Framework (MobSF) 2019. [Consultado: 24 de marzo de 2020]. Disponible en internet: https://github.com/ajinabraham/Mobile-Security-Framework-MobSF [Traducción: Propia]

[71] **NMAP ORG. Nmap (Network mapper).** [En línea] Nmap (Network mapper) 2019. [Consultado: 24 de marzo de 2020]. Disponible en internet: https://nmap.org/

[72] **BORGHELLO, Cristian**, Segu.Info, Noticias sobre seguridad de la Información [En línea] Android Tamer: Penetration Test en Android. 2017 [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://blog.segu-info.com.ar/2017/01/android-tamer-penetration-test-en.html

[73] **FERNANDEZ, Yubal.** Virtual Box: qué es y cómo usarlo para crear una máquina virtual con Windows u otro sistema operativo [En línea] Xataka Basic 2020 2017 [Consultado: 24 de noviembre de 2020]. Disponible en internet: https://www.xataka.com/basics/virtualbox-que-como-usarlo-para-crear-maquina-virtual-windows-u-otro-sistema-operativo

- **W3af**: Es un Framework desarrollado en Python que permite hacer ataques y auditoría a aplicaciones web. W3af permite hacer un escaneo dinámico de una aplicación web, enfocado en un grupo de objetivos especifico [74]

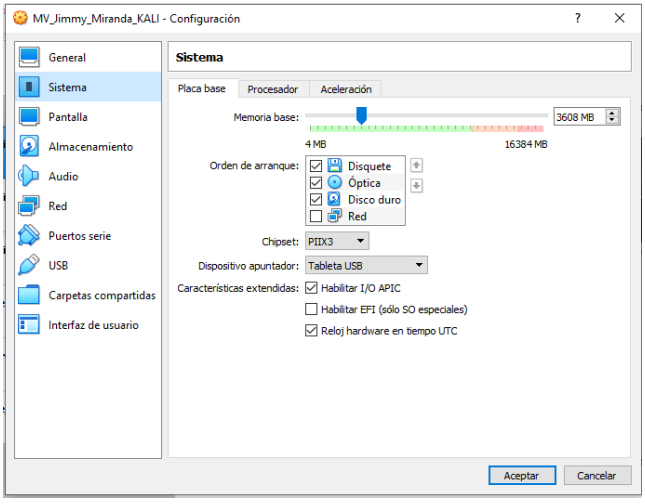
4.4 PRUEBAS CON LAS HERRAMIENTAS SELECCIONADAS

4.4.1 Instalación del programa kali linux

En el desarrollo del ambiente de pruebas se realizará en máquinas virtuales, en que la principal será Kali Linux, Kali Linux es una distribución de Linux basada en Ubuntu, que contiene herramientas preinstaladas para la realizar de pruebas de penetración para sitios web y aplicaciones.

Kali Linux fue descargado desde la url: https://www.kali.org/downloads/ en su versión 2020.2 el proceso de creación de la maquina se realizó así:

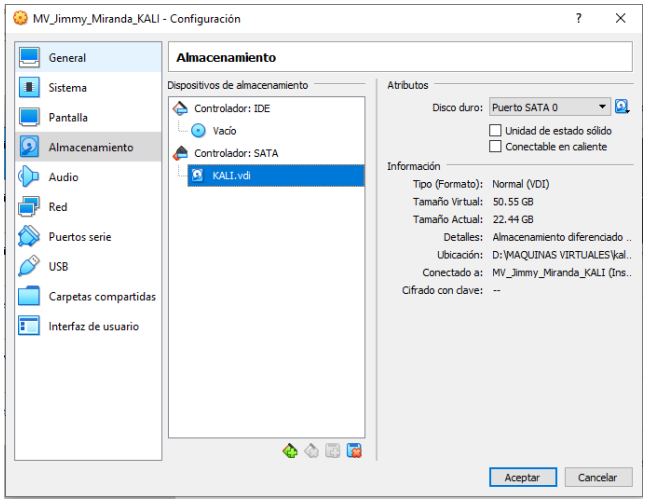
Figura 8. Tamaño de la memoria RAM



Fuente: Propia

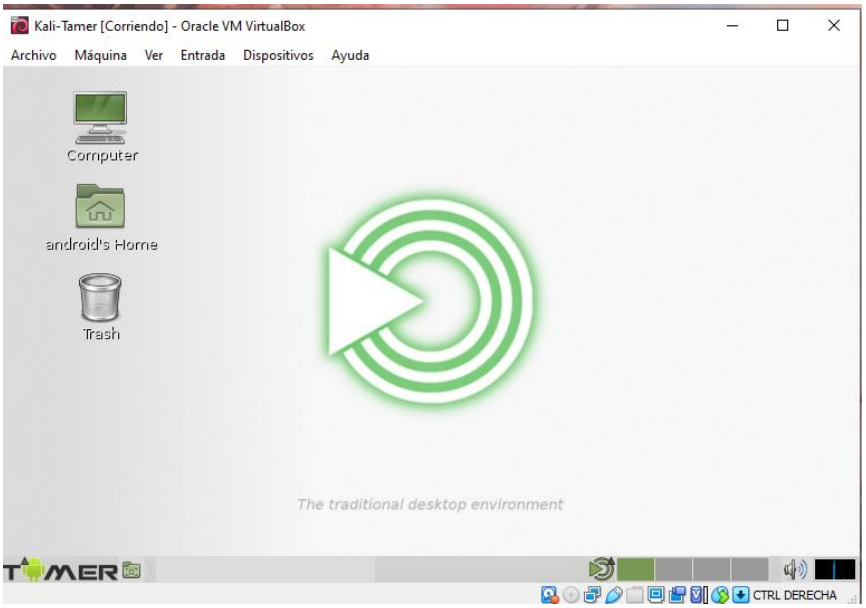
[74] **WA3F.** [En línea] Web Application Attack and Audit Framework 2013. [Consultado: 25 de marzo de 2020]. Disponible en: http://w3af.org/

Figura 9. Tamaño de disco asignado



Fuente: Propia

Figura 10. Maquina finalizada e iniciada



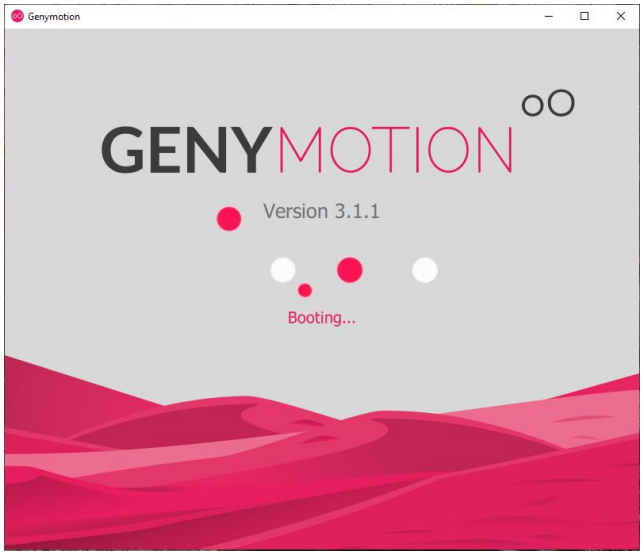
Fuente: Propia

4.4.2 Instalación del programa genymotion

Una vez configurado el sistema Kali, se procede a realizar la instalación del programa Genymotion que permitirá emular el dispositivo móvil, este programa se descarga de la página https://www.genymotion.com/

Al instalar el aplicativo se descarga la última versión que tiene la página.

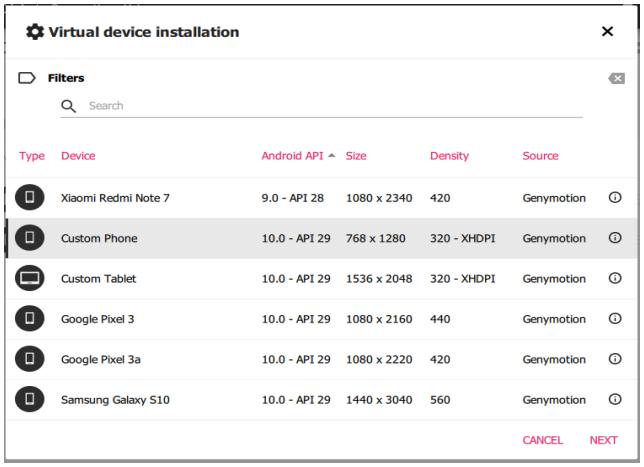
Figura 11. Instalación de Genymotion



Fuente: Propia

Para poder tener un mayor rendimiento y que los recursos no presenten problema, Se realiza la instalación Custom Phone 9.0 API 28 que no consume muchos recursos o que coloca los básicos de funcionamiento.

Figura 12. Creación del Equipo Móvil



Fuente: Propia

Finalizada la instalación del equipo móvil, se ejecuta tal y como se observa en la Figura 12.

Figura 13. Emulando el Dispositivo Móvil

Fuente: Propia

Con el emulador encendido se procede a instalar el APK y se procede a ejecutarla.

4.4.3 Revisión externa del aplicativo

Al validar el aplicativo se evidencia que el aplicativo solicita 4 permisos:

Accesos de cámara;
Accesos de ubicación;
Accesos de almacenamiento;
Otros accesos tales como:

- *Uso de la linterna*
- *Acceso total a la red*
- *Uso del hardware de huellas digitales*

Figura 14. Permisos de la aplicación

Fuente: Propia

Al ejecutar el aplicativo, en la vista principal se puede observar que este solicita un usuario y una contraseña, además de la opción de ingresar con la huella, por lo que se valida la solicitud de uno de sus permisos solicitados.

Figura 15. Vista principal de la aplicación

Fuente: Propia

En la parte inferior, se pueden visualizar unas opciones que no necesitan ingreso de credenciales, pero si realizan captura de información, tal como se ve en la Figura 16.

Figura 16. Opciones adicionales

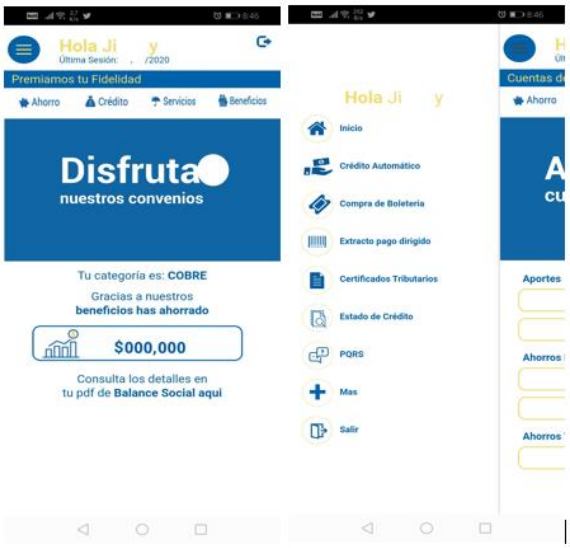
Fuente: Propia

Figura 17. Opciones que solicitan información

Fuente: Propia

Al ingresar en el aplicativo ya se puede ver una interfaz que permite desplazarse en varios menús, confirmando que esta es una aplicación hibrida.

Figura 18. Ingreso en la aplicación

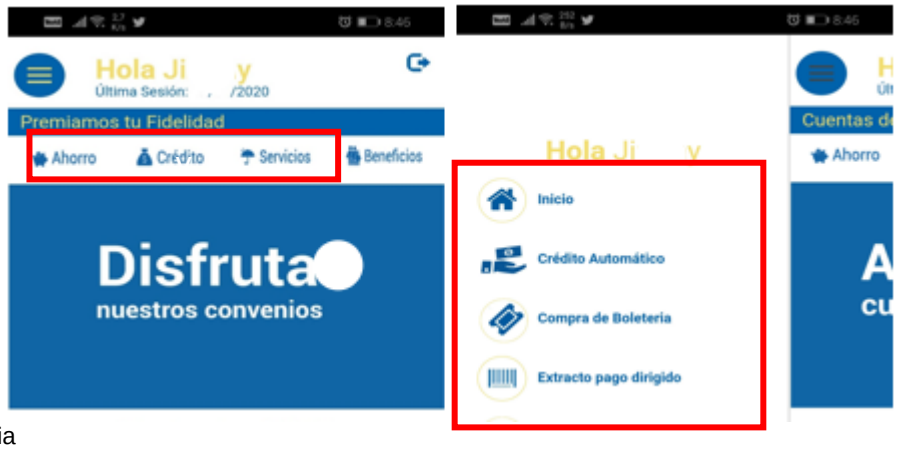


Fuente: Propia

Al revisar la aplicación se puede observar que hay unas opciones que permiten visualizar la información que se tiene en las cuentas y ahorros. Por otro lado, las

opciones que están en el panel izquierdo, esto si permiten una interacción cliente-servidor o cliente – servicios.

Figura 19. Opciones de visualización Vs Opciones de Interacción



Fuente: Propia

Al revisar los sitios que y vínculos con los que interactúan se encuentra que este tiene conexión con los siguientes sitios:

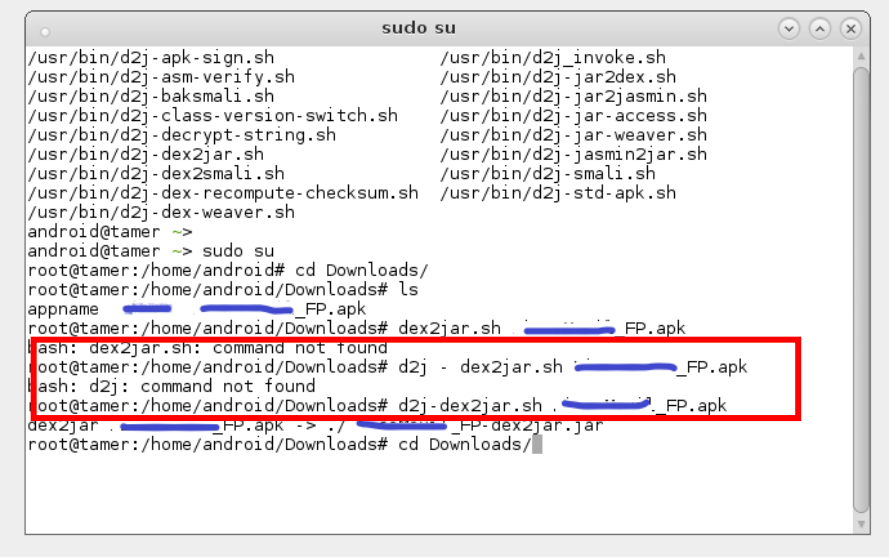
- https://www.transfiya.com.co/
- https://www.pse.com.co/persona
- https://www.cinecolombia.com/
- https://mundoaventura.com.co/
- https://www.playland.com.co/
- https://www.salitremagico.com.co/

y cuando se realiza clic en la opción de beneficios, la aplicación abre una página externa o fuera de la aplicación para interactuar con los convenios que se tienen.

4.4.4 Descompilar la aplicación

Una vez visualizada la aplicación, se procede a realizar la extracción de los archivos que se revisaran y se presentaran en el informe.

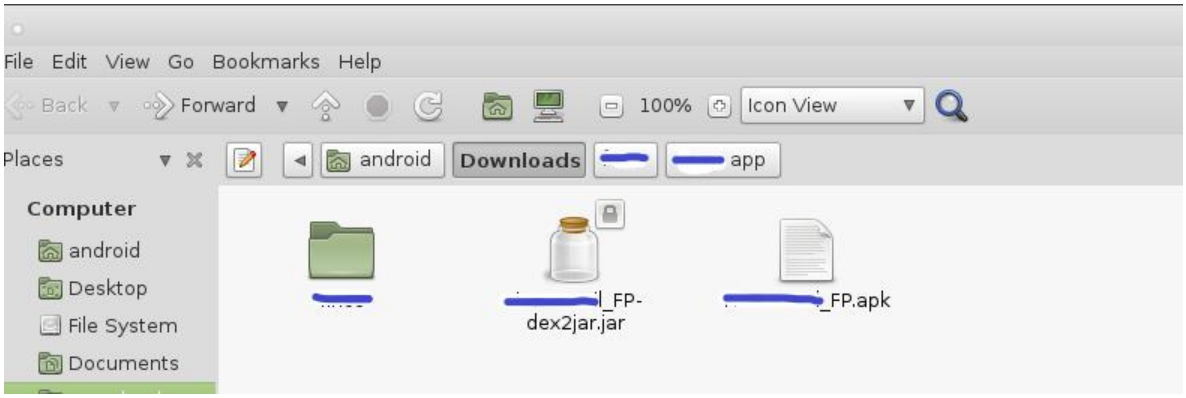
Figura 20. Cambio de formato con Dex2jar



Fuente: Propia

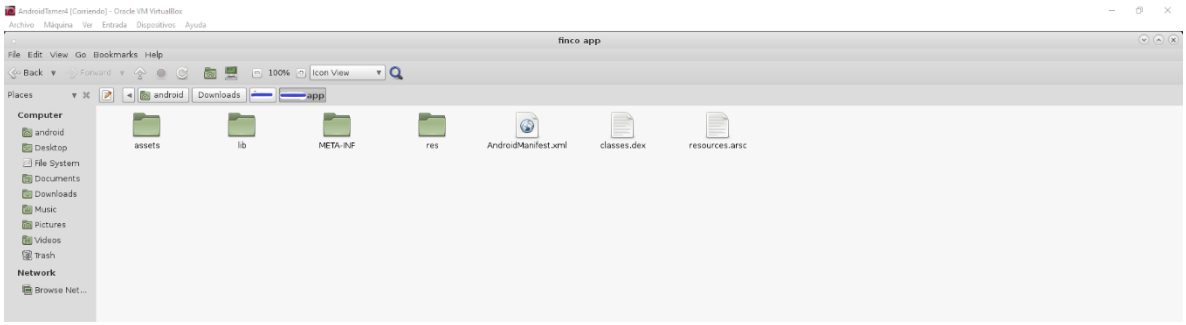
Con la herramienta Dex2jar, se realiza el cambio de formato de la aplicación, el cual cambia de APK a JAR.

Figura 21. Aplicación con el formato JAR



Fuente: Propia

Figura 22. Archivos Internos de la App

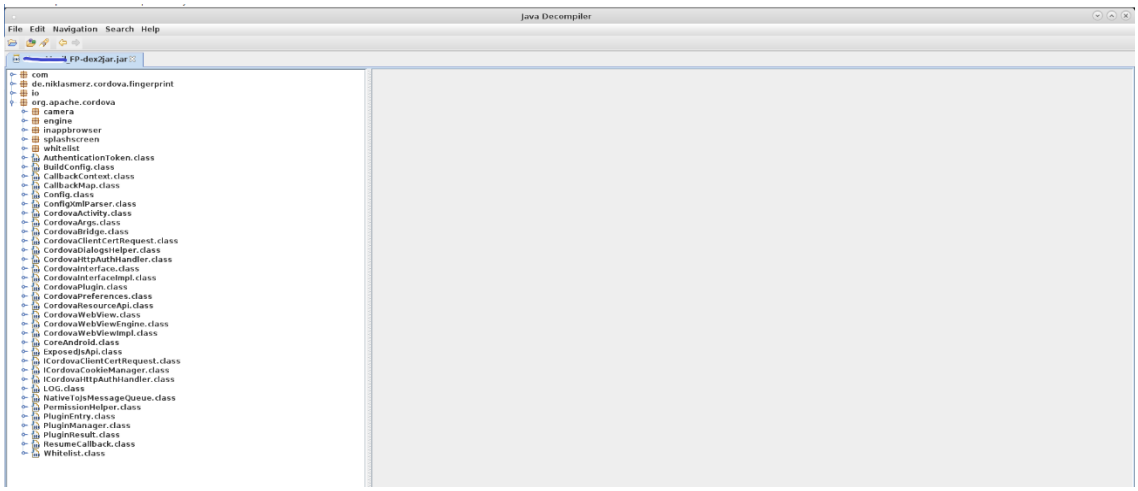


Fuente: Propia

Ahora, con la herramienta JD-Gui se abre el archivo del aplicativo movil_FP-dex2jar.jar y el AndroidManifest.xml para revisar el código fuente de la aplicación.

4.4.5 Revisión del android manifest

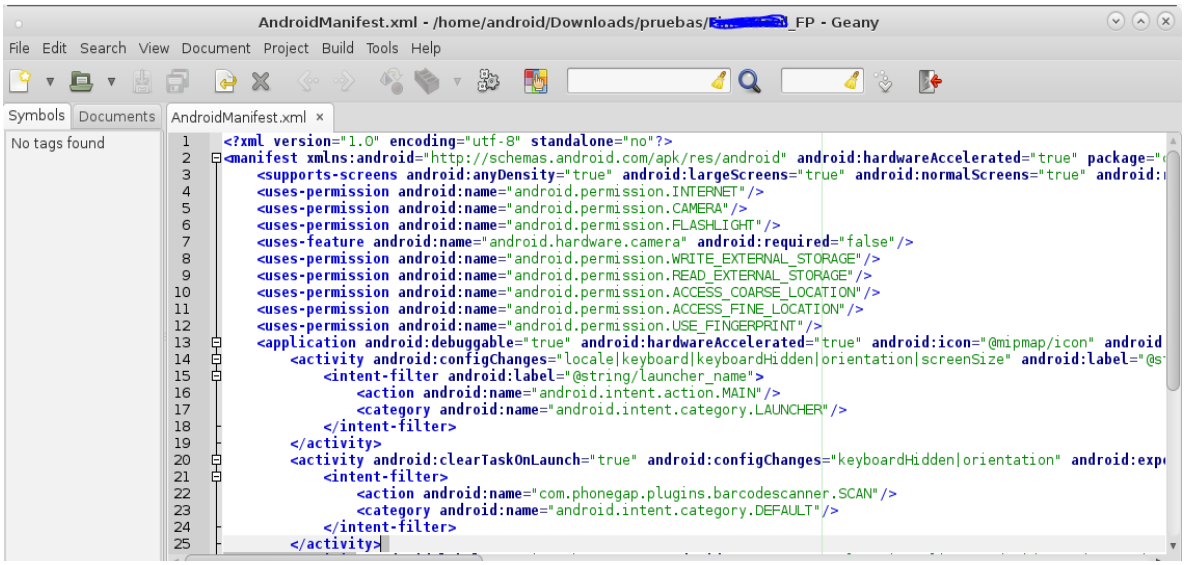
Figura 23. Aplicación en código estático



Fuente: Propia

Con la herramienta ApkTool extraemos la información que se encuentra en el archivo Android Manifest, en el cual se puede evidenciar varios errores, de los cuales se evidencian los permisos que se necesitan para los diferentes aplicativos, adicionalmente se evidencia que tiene activo el modo debbugable, el cual permitiré que se realicen modificaciones en el aplicativo.

Figura 24. Android Manifest



Fuente: Propia

En la primera validación se ha comprobado la posibilidad de inyectar código en tiempo de ejecución sobre los métodos y funciones de la aplicación.

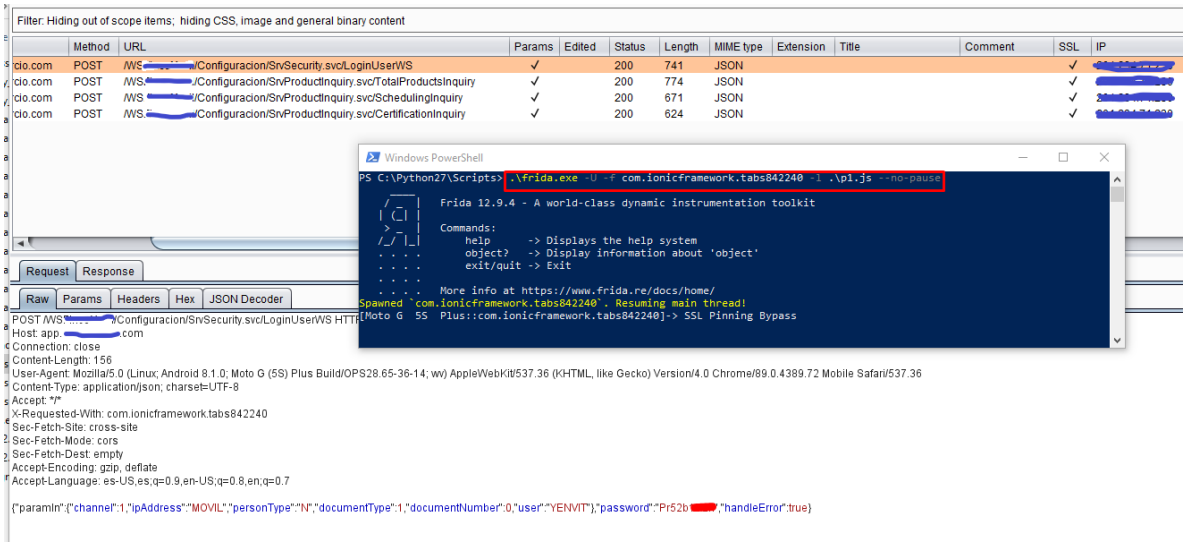
Esto puede permitir a un atacante, modificar la aplicación en run time para evadir diferentes controles, como por ejemplo SSL Pinning, Root Detection, Memory Injection, etc. En otros casos, puede facilitar la materialización de fraude sobre los clientes de la app ya que puede captura información sensible, por medio de Malware instalado en el dispositivo.

4.4.6 Revisión estática del aplicativo

Para este punto se procede a realizar la revisión del aplicativo con las herramientas seleccionadas anteriormente.

En primera instancia se configura el burpsuite para que haga de hombre en el medio y capture toda la información que se necesita, entra primera opción se realizan pruebas con el aplicativo Frida.

Figura 25. Inyección de código con Frida



Fuente: Propia

Al tener los resultados podemos ver con esta información, se evidencia que la aplicación puede ser atacada mediante los siguientes CVE:

Tabla 6. CVE fallo Inyección de código

CVE-2018-6018	Los tamaños fijos de las respuestas HTTPS en la aplicación Tinder para iOS y la aplicación Tinder para Android permiten que un atacante extraiga información privada sensible al rastrear el tráfico de la red.
CVE-2017-3759	La aplicación de Android Lenovo Service Framework acepta algunas respuestas del servidor sin la validación adecuada. Esto expone la aplicación a ataques man-in-the-middle que conducen a una posible ejecución remota de código.
CVE-2017-14487	La aplicación OhMiBod Remote para Android e iOS permite a los atacantes remotos hacerse pasar por usuarios al rastrear el tráfico de la red en busca de respuestas de búsqueda del servidor de API OhMiBod y luego editar el nombre de usuario, user_id y los campos de token en data / data / com.ohmibod.remote2 / shared_prefs / OMB.xml.

CVE-2016-10398	Android 6.0 tiene un desvío de autenticación para atacantes con acceso físico y raíz. Los tokens de autenticación criptográfica (AuthTokens) utilizados por Trusted Execution Environment (TEE) están protegidos por un desafío débil. Esto permite a los adversarios reproducir respuestas capturadas previamente y usar el TEE sin autenticarse. Todas las aplicaciones que utilizan criptografía activada por autenticación son vulnerables a este ataque, que se confirmó en el LG Nexus 5X.

Fuente: Resultado del Análisis

Y para este tipo de vulnerabilidad, puede llegar a explotarse mediante el exploit encontrado como Google Android - Bypass de seguridad remota del código 'APK'

Tabla 7. Notificación de EDB-ID

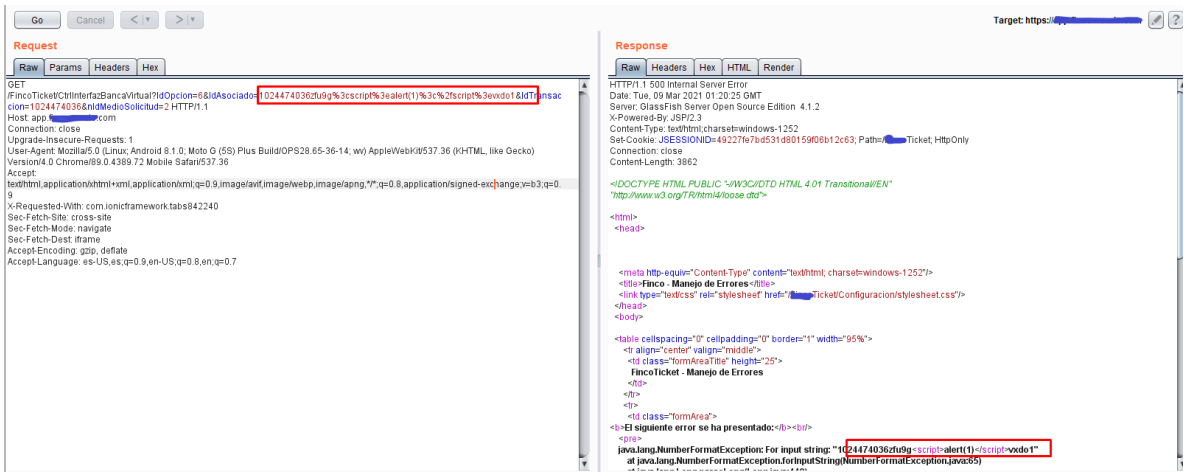
Edb-Id:	38627
Cve:	2013-4787
Autor:	Seguridad Bluebox
Tipo:	Remoto
Plataforma:	Androide
Fecha:	3/7/2013
Url	Https://Www.Exploit-Db.Com/Exploits/38627

Fuente: Resultado del Análisis

Para el siguiente hallazgo se ha detectado una debilidad en la validación de parámetros de entrada de la aplicación móvil dado que es posible inyectar código JavaScript que se refleja en la respuesta del servidor.

Un atacante puede preparar enlaces inyectados para extraer cookies, modificar el comportamiento del navegador embebido en la app móvil, o incluso secuestrar la sesión de los usuarios.

Figura 26. Posibilidad de Cross Site Scripting



Fuente: Propia

Con esta información se evidencia que la aplicación puede ser atacada mediante los siguientes CVE

Tabla 8. CVE fallo XSS

CVE-2021-33562	“Una vulnerabilidad reflejada de secuencias de comandos entre sitios (XSS) en Shopizer antes de 2.17.0 permite a los atacantes remotos inyectar secuencias de comandos web arbitrarias o HTML a través del parámetro ref en una página sobre un producto arbitrario, por ejemplo, un producto / insertar-nombre-producto-aquí .html / ref = URL.”[75]
CVE-2019-19370	“Una vulnerabilidad de cross-site scripting (XSS) en el componente de conferencias web de la aplicación Mitel MiCollab antes de la versión 9.0.15 para Android podría permitir a un atacante no autenticado realizar un ataque de

[75]**NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2021-33562 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2021-33562

	cross-site scripting (XSS) reflejado debido a una validación insuficiente en la carga del archivo interfaz. Un exploit exitoso podría permitir a un atacante ejecutar scripts arbitrarios."[76]
CVE-2019-17573	"De forma predeterminada, Apache CXF crea una página / services que contiene una lista de los nombres y direcciones de los puntos finales disponibles. Esta página web es vulnerable a un ataque de Cross-Site Scripting (XSS) reflejado, que permite a un actor malintencionado inyectar javascript en la página web. Tenga en cuenta que el ataque explota una función que normalmente no está presente en los navegadores modernos, que eliminan los segmentos de puntos antes de enviar la solicitud. Sin embargo, las aplicaciones móviles pueden ser vulnerables."[77]
CVE-2013-7002	"La vulnerabilidad de secuencias de comandos entre sitios (XSS) en mobile / php / translation / index.php en LiveZilla antes de 5.1.1.0 permite a los atacantes remotos inyectar secuencias de comandos web arbitrarias o HTML a través del parámetro g_language."[78]

Fuente: Resultado del Análisis

[76] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2019-19370 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2019-19370

[77] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2019-17573 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2019-17573

[78] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2013-7002 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2013-7002

Y para este tipo de vulnerabilidad, puede llegar a explotarse mediante el exploit encontrado como Google Chrome para Android: com.android.browser.application_id Intent Extra Data Cross-Site Scripting.

Tabla 9. Notificación de EDB-ID

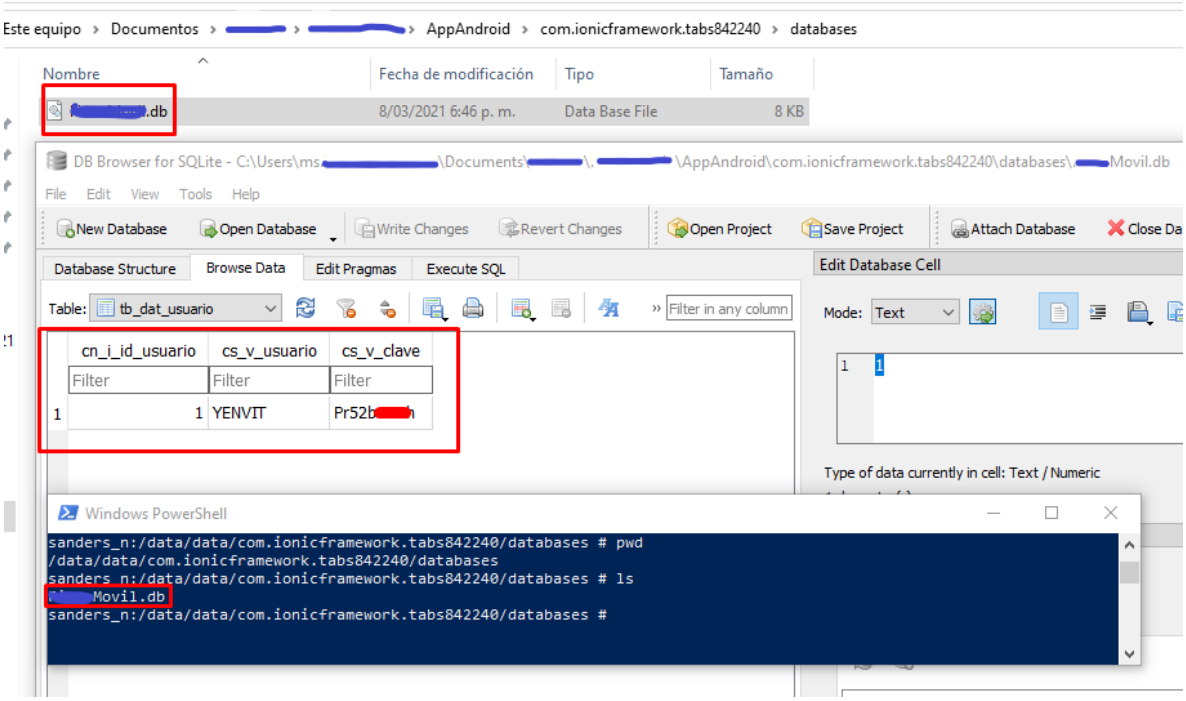
Edb-Id:	37792
Cve:	2012-4905
Autor:	Artem Chaykin
Tipo:	Remoto
Plataforma:	Androide
Fecha:	2012-09-12
Url	https://www.exploit-db.com/exploits/37792

Fuente: Resultado del Análisis

Al analizar los archivos estáticos generados en la aplicación se ha evidenciado una base de datos en texto plano que contiene el nombre de usuario y la contraseña de acceso.

El módulo se utiliza para autenticar al usuario por medio de la huella digital.
Un atacante que lograse tener acceso al dispositivo de su víctima, puede extraer estas credenciales y materializar fraude sobre el usuario de la aplicación.

Figura 27. Almacenamiento de credenciales en texto plano



Fuente: Propia

Con esta información se evidencia que la aplicación puede ser atacada mediante los siguientes CVE

Tabla 10. Manejo de credenciales

CVE-2012-1923	"RealNetworks Helix Server y Helix Mobile Server 14.x antes de 14.3.x almacenan contraseñas en texto sin cifrar en adm_b_db \ users \, lo que permite a los usuarios locales obtener información confidencial mediante la lectura de una base de datos."[79]

Fuente: Resultado del Análisis

[79] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2012-1923 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2012-1923

Y para este tipo de vulnerabilidad, puede llegar a explotarse mediante el edb-id encontrado como Ftp Server 1.32 - Divulgación de credenciales

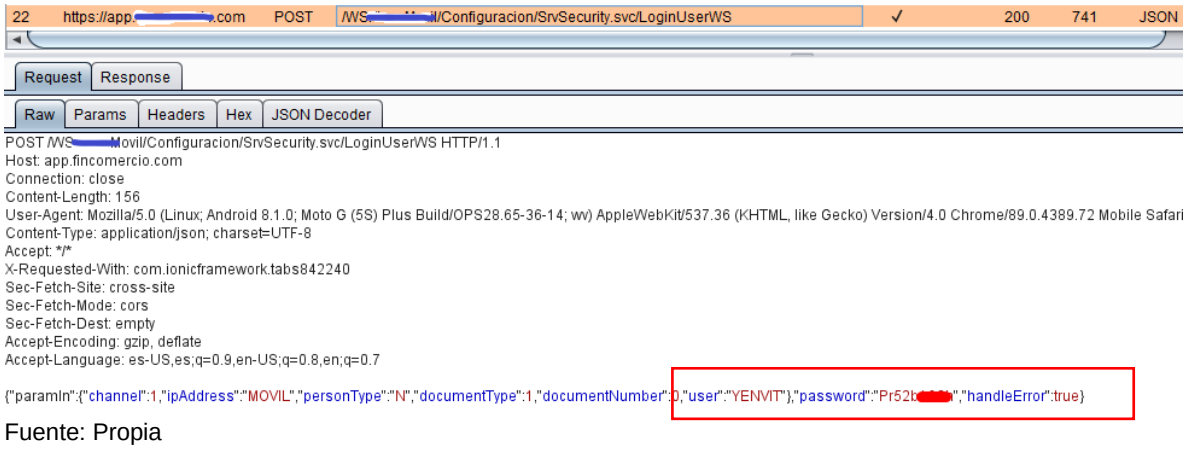
Tabla 11. Notificación de EDB-ID

Edb-Id:	44852
Cve:	N/A
Autor:	Manhnho
Tipo:	Local
Plataforma:	Androide
Fecha:	2018-06-07
Url	https://www.exploit-db.com/exploits/44852

Fuente: Resultado del Análisis

Para este hallazgo se ha evidenciado que algunas solicitudes HTTP transportan información sensible confiando únicamente en el canal cifrado. Esto le permitiría a un criminal que materialice un ataque de MITM (Man in the Middles) o un Proxy HTTPS, descifrar el tráfico y leer en texto plano información de los usuarios de la aplicación.

Figura 28. Envío de información en texto plano



Fuente: Propia

Por lo que se provee que este evento esté relacionado a los siguientes CVE

Tabla 12. Credenciales en texto plano

CVE-2021-21385	“La aplicación de Android Mifos-Mobile para MifosX es una aplicación de Android construida sobre la plataforma de autoservicio MifosX. Mifos-Mobile antes de confirmar e505f62 deshabilita la verificación del nombre de host HTTPS de su cliente HTTP. Además, aceptó como válido cualquier certificado autofirmado. La verificación del nombre de host es una parte importante cuando se usa HTTPS para garantizar que el certificado presentado sea válido para el host. Deshabilitarlo puede permitir ataques de intermediario. Aceptar cualquier certificado, incluso los autofirmados, permite ataques de intermediario. Este problema se solucionó en mifos-mobile commit e505f62.”[80]
CVE-2017-9601	“La aplicación "FNB Kemp Mobile Banking" de First National Bank of Kemp 3.0.2, también conocida como fnb-kemp-mobile-banking / id571448725 para iOS, no verifica los certificados X.509 de los servidores SSL, lo que permite atacantes intermedios para falsificar servidores y obtener información confidencial a través de un certificado elaborado.”[81]
CVE-2017-3190	“Flash Seats Mobile App for Android version 1.7.9 and earlier and for iOS version 1.9.51 and earlier fails to properly

[80] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2021-21385Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/ CVE-2021-21385

[81] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2017-9601Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/ CVE-2017-9601

	validate SSL certificates provided by HTTPS connections, which may enable an attacker to conduct man-in-the-middle (MITM) attacks."[82]

Fuente: Resultado del Análisis

Y su explotabilidad puede darse mediante los siguientes exploits encontrados como, Aplicación industrial SKILLS.com.au: ejecución remota de código Man In The Middle y Virtual Postage (VPA) - Man In The Middle Remote Code Execution respectivamente

Tabla 13. Notificación de EDB-ID

Edb-Id:	42349
Cve:	N/A
Autor:	Pasante
Tipo:	Remoto
Plataforma:	Androide
Fecha:	2017-07-20
Url	https://www.exploit-db.com/exploits/42349

Fuente: Resultado del Análisis

Tabla 14. Notificación de EDB-ID

Edb-Id:	42350
Cve:	N/A
Autor:	Intern0t
Tipo:	Remoto
Plataforma:	Androide
Fecha:	2017-07-20

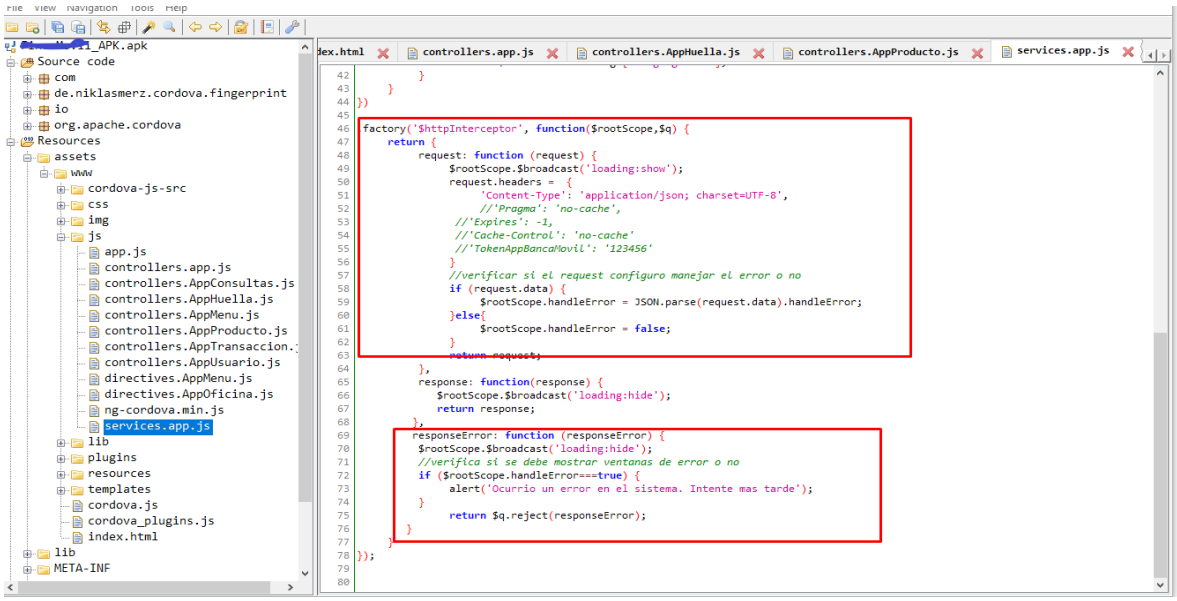
[82] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2017-3190 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2017-3190

Url	https://www.exploit-db.com/exploits/42350

Fuente: Resultado del Análisis

No se ha detectado ofuscación de los archivos del código fuente de la aplicación, pero al tratarse de un desarrollo híbrido, un atacante que conozca el lenguaje Javascript puede determinar toda la lógica del lado del cliente para materializar ataques más profundos.

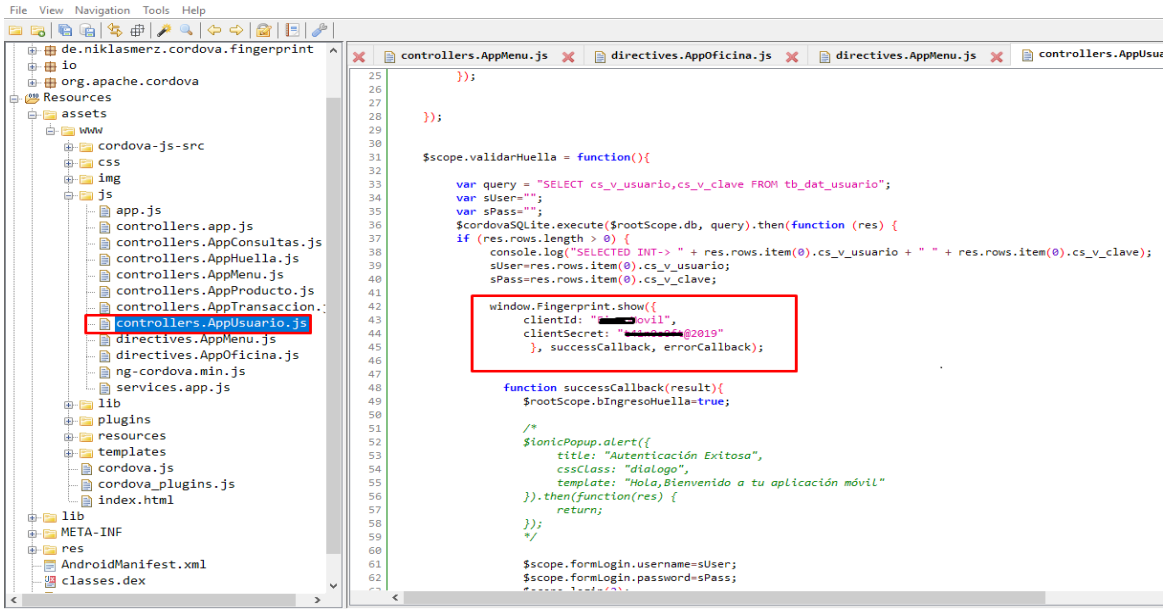
Figura 29. Presentación del código en plano



Fuente: Propia

Adicionalmente se puede evidenciar información del proveedor que desarrollo la aplicación

Figura 30. Información sensible expuesta



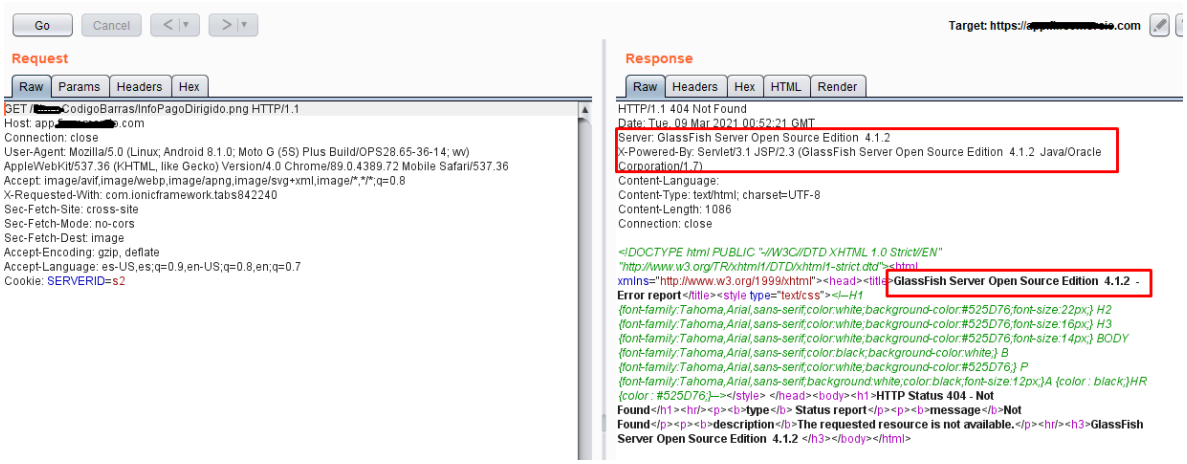
Fuente: Propia

Durante las pruebas se ha identificado que algunas peticiones de la aplicación a diferentes recursos de la entidad retornarn la versión del servicio, por ejemplo, el servidor web Microsoft IIS 8.5.

Y por otra parte el servidor GlassFish Server Open Sourcer Edition 4.1.2 y Servlet 3.1 y JSP 2.3

Un atacante puede aumentar su superficie de ataque contra la entidad identificando los componentes específicos de software que se utilizan en las soluciones telemáticas.

Figura 31. Entrega de información del software



Fuente: Propia

Para las vulnerabilidades presentadas en las figuras 29,30 y 31 se evidencia que la aplicación entrega fácilmente la información en los desarrollos y el lenguaje en que se está trabajando por lo que estos eventos encajarían fácilmente en los CVE

Tabla 15. Divulgación de información

CVE-2021-0404	"En mobile_log_d, existe una posible divulgación de información debido a una validación de entrada incorrecta. Esto podría dar lugar a la divulgación de información local con los privilegios de ejecución del sistema necesarios. La interacción del usuario no es necesaria para la explotación. Producto: Android; Versiones: Android-11; ID de parche: ALPS05457039."[83]
CVE-2017-9969	"Existe una vulnerabilidad de divulgación de información en la aplicación IGSS Mobile de Schneider Electric versión 3.01 y anteriores. Las contraseñas se almacenan en texto sin

[83] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2021-0404 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2021-0404

	cifrar en la configuración, lo que puede resultar en la exposición de información confidencial.”[84]

Fuente: Resultado del Análisis

Y su explotabilidad puede darse mediante los siguientes ejemplos encontrados con sus respectivos nombres:

- Google Android - RKP Information Disclosure via s2-remapping Physical Ranges
- Facebook for Android - 'LoginActivity' Information Disclosure
- Google Android 2.3.5 - PowerVR SGX Driver Information Disclosure

Tabla 16. Notificación de EDB-ID

Edb-Id:	41218
Cve:	N/A
Autor:	Google Security Research
Tipo:	DoS
Plataforma:	Androide
Fecha:	2017-02-01
Url	https://www.exploit-db.com/exploits/41218

Fuente: Resultado del Análisis

Tabla 17. Notificación de EDB-ID

Edb-Id:	38170
Cve:	N/A
Autor:	Takeshi Terada

[84] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2017-9969 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2017-9969

Tipo:	Remoto
Plataforma:	Androide
Fecha:	2013-01-07
Url	https://www.exploit-db.com/exploits/38170

Fuente: Resultado del Análisis

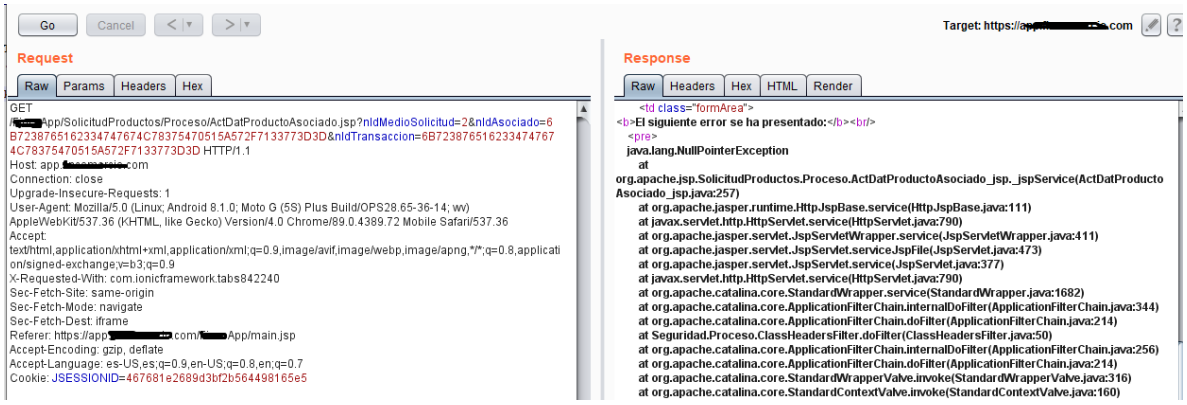
Tabla 18. Notificación de EDB-ID

Edb-Id:	38310
Cve:	2011-1350
Autor:	Geremy Condra
Tipo:	Remoto
Plataforma:	Androide
Fecha:	2011-11-03
Url	https://www.exploit-db.com/exploits/38310

Fuente: Resultado del Análisis

Al realizar diferentes ataques de fuzzing sobre los parámetros de las solicitudes HTTP se han identificado algunos errores que no están siendo tratados correctamente, debido a que revelan información interna de la app.

Figura 32. Manejo incorrecto de errores



Fuente: Propia

La aplicación no tiene configurada el siguiente encabezado de seguridad:

Strict-transport-security (HSTS): es un mecanismo de políticas de seguridad web que ayuda a proteger sitios web contra ataques de degradación (Down grade) de protocolo y secuestro de archivos de sesión (cookies). Permite a los servidores web declarar que los navegadores web (u otros agentes del lado del cliente) solo deben interactuar con él mediante conexiones HTTPS seguras, y nunca a través del protocolo HTTP inseguro.

Tabla 19. Error en las Cookies

CVE-2008-7298	“El navegador de Android en Android no puede restringir adecuadamente las modificaciones a las cookies establecidas en las sesiones HTTPS, lo que permite a los atacantes de intermediario sobrescribir o eliminar cookies arbitrarias a través de un encabezado Set-Cookie en una respuesta HTTP, relacionado con la falta de HTTP Strict. Transport Security (HSTS) incluye la función SubDomains, también conocida como un problema de "forzamiento de cookies".”[85]
CVE-2012-4909	“Google Chrome antes de 18.0.1025308 en Android permite a atacantes remotos obtener información de cookies a través de una aplicación diseñada.”[86]

Fuente: Resultado del Análisis

Y para este evento de encuentra un posible método de explotación el cual es: Google Chrome for Android - Local Application Handling Cookie Theft

Tabla 20. Notificación de EDB-ID

Edb-Id:	37794
Cve:	2012-04909
Autor:	Artem Chaykin
Tipo:	Remoto

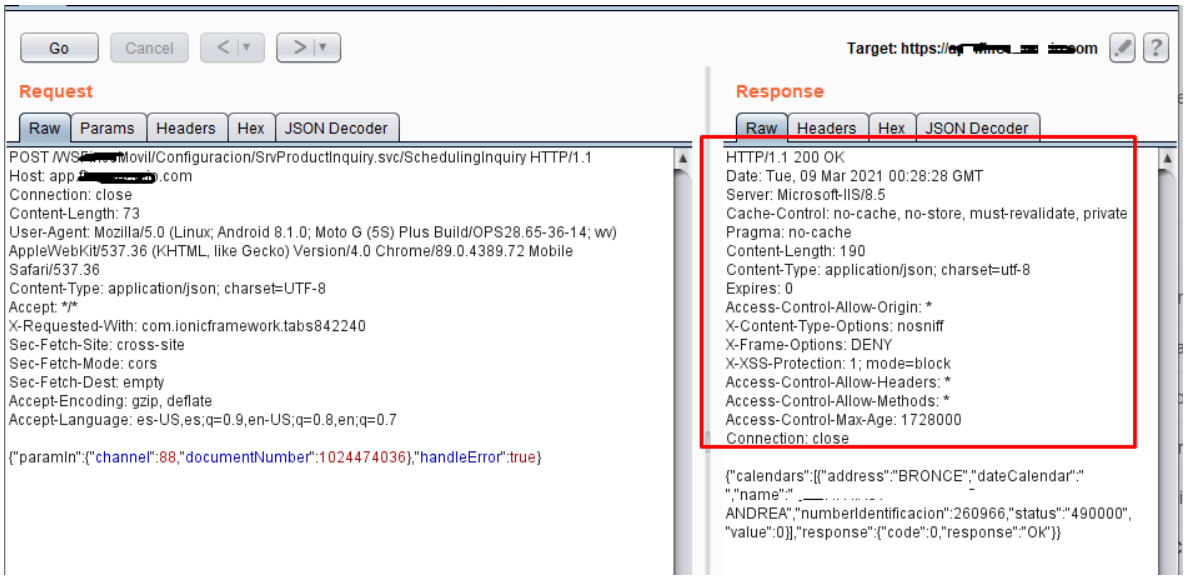
[85] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2008-7298 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2008-7298

[86] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2012-4909 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2012-4909

Plataforma:	Androide
Fecha:	2012-09-12
Url	https://www.exploit-db.com/exploits/37794

Fuente: Resultado del Análisis

Figura 33. Falta de seguridad en cabeceras



Fuente: Propia

Se ha identificado un incorrecto aseguramiento de las cookies, dado que no se encuentran protegidas a través de la bandera HTTP "SECURE". Lo que puede permitir a un usuario no autorizado el robo de la cookie y su posterior utilización para suplantar la identidad de un usuario valido.

Figura 34. Falta de seguridad en cookies

Fuente: Propia

Tabla 21. Cabeceras inseguras

CVE-2012-4906	“Google Chrome antes de la 18.0.1025308 en Android no restringe adecuadamente el acceso al archivo: URL, lo que permite a los atacantes remotos obtener información confidencial a través de vectores no especificados, como se demuestra al obtener datos de credenciales, una vulnerabilidad diferente a CVE-2012-4903.”[87]

Fuente: Resultado del Análisis

Y para estos eventos se relaciona el exploit que podría afectar el aplicativo o servicio relacionado, Google Chrome for Android - Multiple 'file::' URL Handler Local Downloaded Content Disclosure Vulnerabilities

Tabla 22. Notificación de EDB-ID

Edb-Id:	37793
Cve:	2012-4906
Autor:	Artem Chaykin
Tipo:	Remoto
Plataforma:	Androide
Fecha:	2012-09-12
Url	https://www.exploit-db.com/exploits/37793

Fuente: Resultado del Análisis

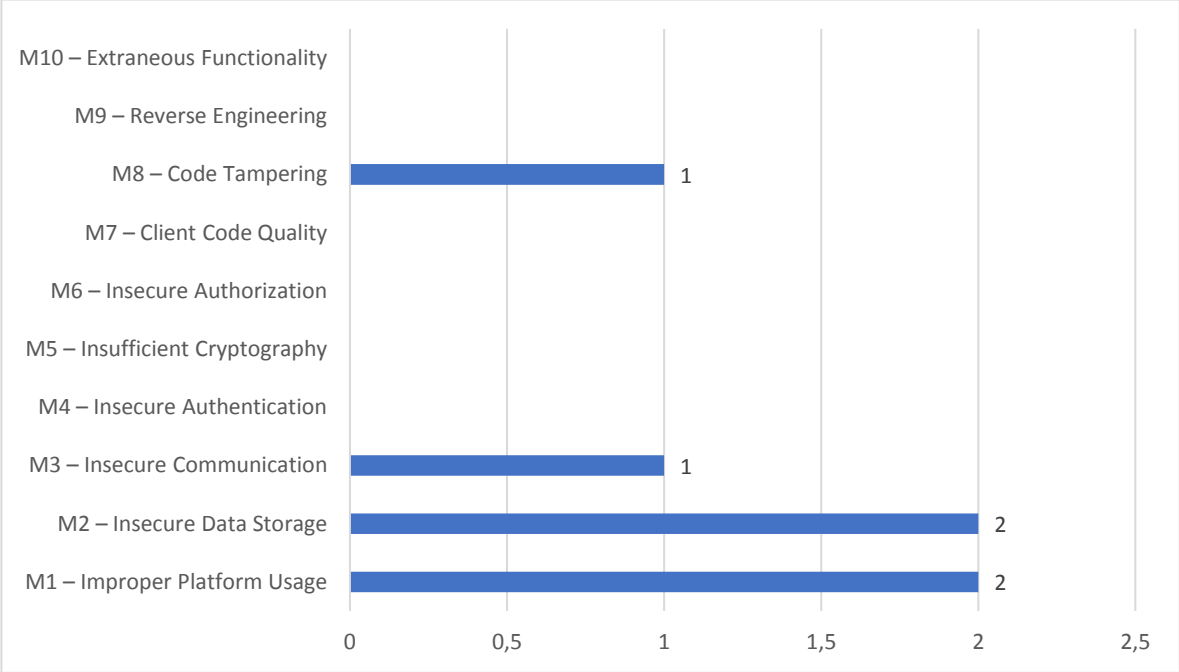
[87] **NATIONAL VULNERABILITY DATABASE**, Information Technology Laboratory [sitio web]. CVE-2012-4906 Detail, Gaithersburg, MD 20899 [Consultado: 18 de julio de 2021]. Disponible en: https://nvd.nist.gov/vuln/detail/CVE-2012-4906

4.5 INFORME PARA LA EMPRESA

De acuerdo con el trabajo realizado en la aplicación móvil, se realiza la búsqueda de fallos o posibles vulnerabilidades que afecten el aplicativo, El análisis se realizó de acuerdo a las recomendaciones que presenta la metodología OWASP top 10 Mobile versión 2016. En la revisión realizada se encontraron 6 vulnerabilidades que pueden poner en riesgo la seguridad de la información y la aplicación en sí.

Dando como resultado que 4 de las vulnerabilidades encontradas tienen una afectación alta, a de más de su explotación, que puede llegar a comprometer el aplicativo y abrir una puerta a los servicios internos de la empresa.

Tabla 23. Vulnerabilidades encontradas



Fuente: propia

De acuerdo a estos hallazgos se presenta la tabla en la que se describe cada vulnerabilidad encontrada, como afecta la aplicación y las recomendaciones para poder mitigar la aplicación, entre otros.

Tabla 24. Informe técnico para la entidad financiera

Dominio OWAPS	Vulnerabilidad	Descripción	Remediación	CVE	Información	Exploit
M1: uso inadecuado de la plataforma M9: Ingeniería inversa	Inyección de código en tiempo de ejecución	Se ha comprobado la posibilidad de inyectar código en tiempo de ejecución sobre los métodos y funciones de la aplicación. Esto puede permitir a un atacante, modificar la aplicación en run time para evadir diferentes controles, como por ejemplo SSL Pinning, Root Detection, Memory Injection, etc. En otros casos, puede facilitar la materialización de fraude sobre los clientes de la app, por medio de Malware instalado en el dispositivo.	Verificar la presencia de permisos de administrador en el dispositivo e informar al usuario. Ofuscar el código fuente de la aplicación. Implementar un SDK de protección sobre ataques de ingeniería inversa.	CVE-2018-6018	Fixed sizes of HTTPS responses in Tinder iOS app and Tinder Android app allow an attacker to extract private sensitive information by sniffing network traffic.	https://www.exploit-db.com/exploits/38627
M1: uso inadecuado de la plataforma	Cross Site Scripting reflejado	Se ha detectado una debilidad en la validación de parámetros de entrada de la aplicación móvil dado que es posible inyectar código JavaScript que se refleja en la respuesta del servidor. Un atacante puede preparar enlaces inyectados para extraer cookies, modificar el comportamiento del navegador embebido en la app móvil, o incluso secuestrar la sesión de los usuarios.	Implementar un módulo de sanitización de parámetros de entrada y salida para todas las peticiones HTTP que salen dese la aplicación al servidor.	CVE-2021-33562	A reflected cross-site scripting (XSS) vulnerability in Shopizer before 2.17.0 allows remote attackers to inject arbitrary web script or HTML via the ref parameter to a page about an arbitrary product, e.g., a product/insert-product-name-here.html/ref= URL.	https://www.exploit-db.com/exploits/37792

M2: almacenamiento de datos inseguro	Almacenamiento de credenciales en texto plano	Al analizar los archivos estáticos generados en la aplicación se ha evidenciado una base de datos en texto plano que contiene el nombre de usuario y la contraseña de acceso. El módulo se utiliza para autenticar al usuario por medio de la huella digital. Un atacante que lograse tener acceso al dispositivo de su víctima, puede extraer estas credenciales y materializar fraude sobre el usuario de la aplicación.	Utilizar la implementación de KeyStore en Android para el almacenamiento de datos sensibles y cifrar los mismos con algoritmos simétricos como por ejemplo AES256.	CVE-2012-1923	Real Networks Helix Server and Helix Mobile Server 14.x before 14.3.x store passwords in clear text under adm_b_db\users\, which allows local users to obtain sensitive information by reading a database.	https://www.exploit-db.com/exploits/44852
M3: Comunicación insegura		Se ha evidenciado que algunas solicitudes HTTP transportan información sensible confiando únicamente en el canal cifrado. Esto le permitiría a un criminal que materialice un ataque de MITM o un Proxy HTTPS, descifrar el tráfico y leer en texto plano información de los usuarios de la aplicación.	Cifrar o enmascarar todos los datos sensibles que se envían en las solicitudes HTTP, en especial los datos de autenticación de los usuarios. Para esto puede utilizarse cifrado asimétrico o alguna especie de algoritmo de hash más oscuridad.			
M8: manipulación de código	Ofuscación débil	No se ha detectado ofuscación de los archivos del código fuente de la aplicación, al tratarse de un desarrollo híbrido, un atacante que conozca el lenguaje JavaScript puede determinar toda la lógica del lado del cliente para materializar ataques más profundos.	Utilizar algoritmos de ofuscación robustos como DexGuard. También se pueden utilizar Strings para almacenar URLS.	CVE-2021-0404	In mobile_log_d, there is a possible information disclosure due to improper input validation. This could lead to local information disclosure with System execution privileges needed. User interaction is	https://www.exploit-db.com/exploits/41218 - https://www.exploit-db.com/exploits/38170 - https://www.exploit-db.com/exploits/38310

		Al realizar diferentes ataques de fuzzing sobre los parámetros de las solicitudes HTTP se han identificado algunos errores que no están siendo tratados correctamente, debido a que revelan información interna de la app.	Sanitizar todas las excepciones de todos los fragmentos del código y almacenar el error plano en un archivo de log. Finalmente retornar un error común que no revele información interna.		not needed for exploitation. Product: Android; Versions: Android-11; Patch ID: ALPS05457039.	
M2: almacenamiento de datos inseguro		Durante las pruebas se ha identificado que algunas peticiones de la aplicación a diferentes recursos de la entidad retornaran la versión del servicio, por ejemplo el servidor web Microsoft IIS 8.5. Y por otra parte el servidor GlassFish Server Open Sourcer Edition 4.1.2 y Servlet 3.1 y JSP 2.3 Un atacante puede aumentar su superficie de ataque contra la entidad identificando los componentes específicos de software que se utilizan en las soluciones telemáticas.	Eliminar de la respuesta del servidor aquella información que no sea procesada por el navegador o que revele detalles internos como versiones, nombres o modos de funcionamiento.			

Fuente: Propia

5 RECOMENDACIONES

Con el fin de dar solución a las vulnerabilidades encontradas se presentan las siguientes recomendaciones

- Verificar la presencia de permisos de administrador en el dispositivo e informar al usuario.

- Ofuscar el código fuente de la aplicación.

- Implementar un SDK de protección sobre ataques de ingeniería inversa.

- Implementar un módulo de sanitización de parámetros de entrada y salida para todas las peticiones HTTP que salen dese la aplicación al servidor.

- Utilizar la implementación de KeyStore en Android para el almacenamiento de datos sensibles y cifrar los mismos con algoritmos simétricos como por ejemplo AES256.

- Cifrar o enmascarar todos los datos sensibles que se envían en las solicitudes HTTP, en especial los datos de autenticación de los usuarios.

- Se recomienda aplicar mejoras en los certificados digitales, los cuales aplique para la APP, teniendo en cuenta los niveles mínimos de seguridad, tales como TLS 1.2 o 1.3, o instalar certificados con cifrado SHA256 o superior.

- Utilizar algoritmos de ofuscación robustos como DexGuard, También se pueden utilizar Strings para almacenar URLS.

- Crear una lista blanca de certificados válidos en el servidor e implementar una función propia que verifique la autenticidad de los mismos.

- Eliminar de la respuesta del servidor aquella información que no sea procesada por el navegador o que revele detalles internos como versiones, nombres o modos de funcionamiento.

- Se sugiere configurar los siguientes parámetros: * HSTS: La aplicación debe indicar a los navegadores web que sólo accedan a la aplicación mediante HTTPS. Para ello, se recomienda habilitar HTTP Strict Transport Security (HSTS) añadiendo un el header de respuesta ‘Strict-Transport-Security' y el valor 'max-age = expireTime', donde expireTime. Considerar la posibilidad de añadir la bandera "includeSubDomains" si es apropiado.

- Crear sólo las cookies necesarias para la aplicación y directorio correspondiente

- Limitar la información almacenada en las cookies a lo estrictamente necesario y bajo ningún punto de vista almacenar información sensible (contraseñas, IDs, tarjetas de crédito, etc.)

- Utilizar "HTTP-Only" para evitar el uso de ataques y scripts genéricos de XSS

- La empresa deberá tener presente las recomendaciones dadas en este proyecto, para que lo puedan utilizar en próximos proyectos o actualizaciones de la aplicación, además de empezar a incluir todo un ciclo de desarrollo enfocado o basado en SecDevOps, para que permita identificar una falla de seguridad en el software más temprano, más rápido y para que su mitigación sea más económica.

- Una vez presentado el informe a las áreas involucradas en el proceso de desarrollo de la aplicación, se debe establecer un plan de trabajo que permita mitigar las vulnerabilidades encontradas y en caso de ser necesario se deberá cambiar su plataforma de desarrollo.

6 CONCLUSIONES

Mediante el desarrollo de este proyecto de seguridad basado en la metodología OWASP, permitió evidenciar los problemas que actualmente tiene la aplicación analizada y a los cuales puede llegar a incurrir si se materializa alguno de los riesgos que se evidenciaron durante el desarrollo del proyecto.

Se pudo conocer los diferentes métodos que se pueden realizar para ejecutar o explotar una vulnerabilidad que pueda colocar en riesgo la seguridad de una aplicación y la forma en que se puede recopilar información sensible desde una aplicación.

Durante el proceso de análisis se pudo evidenciar que para cada tipo de ataque o de búsqueda de información se hace necesario conocer o utilizar diferentes herramientas que permitan o complementen la búsqueda de la información o análisis a realizar.

Las aplicaciones móviles que manejan información confidencial o que poseen pasarela de pagos, se les debe realizar un control más estricto que permita reforzar la seguridad de la información ya sea mediante análisis periódicos y creando un proceso que mejore la seguridad desde su inicio.

Como expertos en seguridad informática debemos conocer diferentes herramientas ya sean pagas u open sources, que permitan realizar los análisis y así tener resultados más acertados o que eviten generar falsos positivos.

Realizar este proyecto permitió adquirir un mayor conocimiento en el área de la seguridad informática y apalancar la necesidad de continuar generando conocimiento y capacitación con el fin de poder brindar una mayor seguridad y

menor riesgo a las empresas en la que pueda llegar aportar el conocimiento adquirido o apoyar en el mejoramiento del mismo.

Este proyecto también genera la necesidad de no quedarse con una sola información o resultado obtenido, sino que motiva a realizar diferentes pruebas que permitan asegurar el entorno sobre el que se trabaja.

BIBLIOGRAFÍA

ALONSO, Chema. Pentesting con FOCA. Mostoles: ZeroxWord Computing S.L, 2016. ISBN:978-84-616-6319-4.

ARROYO, Miguel. Pentesting de Aplicaciones iOS. En: CyberCamp [en línea]. España. 2016.[consulta: marzo de 2020]. Disponible en: https://cybercamp.es/cybercamp2016/sites/default/files/contenidos/material/cybercamp2016-pentesting_de_aplicaciones_ios-miguel.

BETANCUR, Oscar. ERASO, Sonia. Seguridad En Dispositivos Móviles Android. [en línea]. Proyecto De Trabajo De Grado. Bogotá: Universidad Nacional Abierta Y A Distancia 2015. [consulta: marzo de 2020]. Disponible en: https://stadium.unad.edu.co/preview/UNAD.

BORGHELLO, Cristian. OASAM: metodología para el análisis de seguridad de aplicaciones Android [En Linea]. Segu-info, 7 de enero 2014. [Consultado: 15 de marzo de 2020]. Disponible en: https://blog.segu-info.com.ar/2014/01/oasam-metodologia-para-el-analisi.

BORGHELLO, Cristian. OWASP Mobile Security (español). En blog.segu-info [en línea]. España 2015. [Consultado el 25 de marzo 2020] Disponible en: https://blog.segu-info.com.ar/2015/03/owasp-mobile-security-espanol.html.

CASTILLO, Lucho. Historia de Android [en línea]. EEUU. (5 de noviembre de 2012). [Consultado: 15 de abril de 2020]. Disponible en: https://github.com/LuchoCastillo/AndroidOS/blob/master/data/historia.rst.

CEBALLOS, Julián y MARULANDA, Cesar. Una Revisión de metodologías seguras en cada fase del ciclo de desarrollo de desarrollo de software. 2012. [En línea] [Revisado el 25 de mayo, 2016] Disponible en internet: http://web.usbmed.edu.co/usbmed/fing/v3n1/v3n. [En línea]

COLOMBIA, MINISTERIO DE COMERCIO, INDUSTRIA Y TURISMO. Decreto Número 1377 de 2013. 2013. [En línea] [Consultado: 19 de abril de 2020]. Disponible en: https://www.mintic.gov.co/portal/604/articles-4274_documento.pdf.

COLOMBIA. CONGRESO DE LA REPÚBLICA. LEY ESTATUTARIA 1581(18 de octubre de 2012). Por la cual se dictan disposiciones generales para la protección de datos personales.[en línea]. Santa Fe de Bogotá, D.C.: Diario Oficial No. 48.587 [Consultado: mayo 6 de 20.

COLORADO, Pedro. TORRES Inirida.[en línea]. Trabajo fin de grado. Villavicencio. Universidad Nacional Abierta Y A Distancia. 2015. [consulta: marzo de 2020]. Disponible en: https://repository.unad.edu.co/handle/10596/3412.

DEBITOOR. App móvil - ¿Qué es una app móvil? [en línea]. Debitoor. España. (15 de marzo de 2017). [Consultado: 15 de marzo de 2020]. Disponible en: https://debitoor.es/glosario/app-movil.

DEVELOPERS. Android Debug Bridge (adb) [en línea]. Developers. España. (2018). [Consultado: 20 de marzo de 2020]. Disponible en: https://developer.android.com/studio/command-line/adb?hl=es-419.

DEVICE ATLAS, Android v iOS market share 2019 [En línea]. Device3atlas, EEUU [Consultado: 4 de abril de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share.

DRAKE, Joshua. FORA, Pau. LANIER, Zach. MULLINER, Collin. Ridley, STEPHEN. Wicherski, Georg. Android™ Hacker's Handbook. Indianapolis. John Wiley & Sons, Inc. 2014. ISBN: 978-1-118-60864-7.

ERICSSON MOBILITY. The Ericsson Mobility Report [en línea]. (1 de octubre de 2019) Telefonaktiebolaget LM Ericsson [Consultado: 5 de mayo de 2020]. Disponible en: http://www.ericsson.com/news/141118-90percent-to-have-mobile-phones-by-2020-predicts-ericsso.

ESCOBAR, J. & QUINTO, L. Vulnerabilidad en dispositivos móviles con sistema operativo Android. (2015).Cuaderno Activa, 7, 55-65.

FAJARDO, Carmen. Análisis De Los Riesgos De Seguridad De La Información De Un Aplicativo De Gestión Documental Líder En El Mercado Colombiano. [en línea]. Trabajo fin de grado. Bogotá: INSTITUCIÓN UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO. 2017. [consulta:.

GARZON, Juan. Android 11 a Android 1.5: Cada versión de Android y sus novedades [en línea]. EEUU. (19 de febrero de 2020). [Consultado: 1 de abril de 2020]. Disponible en: https://www.cnet.com/es/imagenes/android-11-novedades-actualizacion-android-histori.

GIUSTO, Denise. Descienden las detecciones de malware en Android y crecen en iOS [en línea]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-.

GIUSTO, Denise. Descienden las detecciones de malware en Android y crecen en iOS [en línea]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-.

GIUSTO, Denise. Descienden las detecciones de malware en Android y crecen en iOS [en línea]. welivesecurity. 8 de enero 2020. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.welivesecurity.com/la-es/2020/01/08/descienden-detecciones-malware-.

GONDI, Tilakkumar. iOS – Architecture. [En Línea]. tilakgondi.wordpress (14 de enero de 2015). [Consultado: 3 de abril de 2020]. Disponible en: https://tilakgondi.wordpress.com/2015/01/14/ios-architecture/.

GONZÁLEZ, Alfonso. Seguridad en Smartphones Análisis de riesgos, de vulnerabilidades y auditorías de dispositivos [en línea]. Trabajo Final de Máster. Instituto Nacional de Ciberseguridad (INCIBE). España 2018. [consulta: marzo de 2020]. Disponible en:.

GONZÁLEZ, Carlos. Android Ayuda, ¿Qué es ART? Android Runtime, el sucesor de Dalvik [en Línea]. Android Ayuda (07 de agosto de 2019). [Consultado: 1 de abril de 2020]. Disponible en: https://androidayuda.com/android/que-es/art-android-runtime/.

GONZÁLEZ, Daycith. Análisis De Seguridad De La Aplicación Prism Para Sistemas Operativos Android, Propiedad De La Empresa Barcodeapps, Tomando Como Referencia El Top 10 De Riesgos Establecidos En El Proyecto De Seguridad Móvil -Owasp. [en línea]. Traba.

GUPTA, Aditya. Learning Pentesting for Android Devices. Packt Publishing. 2014. ISBN 978-1-78328-898-4.

GUZMÁN, jeyson y FORERO, Leonel. Análisis de vulnerabilidades y seguridad de dispositivos móviles con sistema operativo iOS 6.1.4 [En línea]. Proyecto de Grado. Bogotá: Universidad Piloto De Colombia. 2013 [consulta: marzo de 2020]. Disponible en: http.

GUZMÁN, jeyson y FORERO, Leonel. Análisis de vulnerabilidades y seguridad de dispositivos móviles con sistema operativo iOS 6.1.4 [En línea]. Proyecto de Grado. Bogotá: Universidad Piloto De Colombia. 2013 [consulta: marzo de 2020]. Disponible en: http://.

GUZMÁN, jeyson y FORERO, Leonel. Análisis de vulnerabilidades y seguridad de dispositivos móviles con sistema operativo iOS 6.1.4 [En línea]. Proyecto de Grado. Bogotá: Universidad Piloto De Colombia. 2013 [consulta: marzo de 2020]. Disponible en: http://.

HERRERA, Domingo. Estudio de Vulnerabilidades en transacciones bancarias para dispositivos móviles con Sistema Operativo ANDROID [en línea]. Trabajo fin de grado. Loja: Universidad Nacional De Loja. 2017. [Consultado el 15 de abril de 2020] Disponible en:.

INFOSEC INSTITUTE. ANDROID HACKING & SECURITY - PART 14, Broken Cryptography [en línea]. Infosec Institute (8 de enero 2019.) [Consultado: 15 de

marzo de 2020]. Disponible en: https://resources.infosecinstitute.com/android-hacking-security-part-16-broken.

JKIELTY, Android v iOS market share 2019. [en línea]. DeviceAtlas. Ireland. 9 de septiembre del 2019. [Consultado: 15 de marzo de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share.

JKIELTY, Akndroid v iOS market share 2019. [en línea]. DeviceAtlas. Ireland. 9 de septiembre del 2019. [Consultado: 15 de marzo de 2020]. Disponible en: https://deviceatlas.com/blog/android-v-ios-market-share.

KASPERSKY LAB. Las siete amenazas principales para la seguridad móvil, teléfonos, tablets y dispositivos de Internet móvil: qué nos deparará el futuro [en línea]. Kaspersky Lab. (8 de enero 2015.) [Consultado: 15 de marzo de 2020]. Disponible en: https://.

LOPEZ, Miguel. Ya disponible la beta 4 de iOS 13.5, tvOS 13.4.5, watchOS 6.2.5 y iPadOS 13.5 con la API de notificación de exposición [en Línea]. Applesfera (06 de mayo de 2020). [Consultado: 7 de abril de 2020]. Disponible en: https://www.applesfera.com/.

MAHALIK, Heather. TAMMA, Rohit. BOMMISETTY, Satish. Practical Mobile Forensics, Second Edition, Packt Publishing. 2016. ISBN 978-1-78646-420-0.

MANERA, Pablo. Los 10 errores más comunes de seguridad de aplicaciones móviles. En: CYBSEC [en línea]. España. 2016 [consulta: marzo de 2020]. Disponible en: http://www.cybsec.com/upload/Los_10_errores_mas_comunes_de_seguridad_de_aplicaciones_moviles_CYB.

MATÍAS, Laura. Gestión De Riesgo En Dispositivos Android Basada En Eliminación De Vulnerabilidades Y Detección De Contextos. [en línea]. Trabajo fin de grado. Leganés: Universidad Carlos III De Madrid. 2013. [consulta: abril de 2020]. Disponible en: https.

MEDIANERO, Daniel. OASAM, Open Android Security Assessment Methodology [En Linea]. Comunidad dragon jar. 13 de junio 2013. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.dragonjar.org/oasam.xhtml.

MIFSUD, Elvira. Introducción a la seguridad informática - Seguridad de la información/Seguridad informática. España. 2012 [En línea]. [consulta: marzo de 2020]. Disponible en: http://recursostic.educacion.es/observatorio/web/ca/software/softwaregeneral/1.

MIFSUD, Elvira. Introducción a la seguridad informática - Seguridad de la información/Seguridad informática. España. 2012 [En línea]. [consulta: marzo de 2020]. Disponible en: http://recursostic.educacion.es/observatorio/web/ca/software/softwaregeneral/1.

MOKRIS, Keith. Building blocks for secure mobile development: Testing for the OWASP Mobile Top 10 [Imagen]. Now Secure. 13 de octubre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://www.nowsecure.com/blog/2016/10/13/secure-mobile-developme.

NMAP ORG. Nmap (Network mapper). [En línea] Nmap (Network mapper) 2019. [Consultado: 24 de marzo de 2020].Disponible en internet: https://nmap.org/ .

OAKLEY, jacob. Professional Red Teaming: Conducting Successful Cybersecurity Engagements. Apress. 2019.ISBN-13 (pbk): 978-1-4842-4308-4.

OWASP Foundation. OWASP Mobile Security Project [en línea]. OWASP mobile security (8 de enero 2019.) [Consultado: 15 de marzo de 2020]. Disponible en: https://owasp.org/www-project-mobile-security/.

PACHECO Veliz, EXEQUIEL, Sebastián, PIAZZA. Orlando, DAMIÁN, Carlos. Estudio y análisis de seguridad en dispositivos móviles. BYOD y su impacto en las organizaciones. [en línea]. Trabajo fin de grado. La Plata, Universidad Nacional De La Plata. 2007. [con.

PAVA, Jonier. SARMIENTO, James. FORERO, Bryan. Diagnóstico De Seguridad Y Privacidad De La Información En La alcaldía Municipal De Icononzo Tolima. [en

línea]. Proyecto De Trabajo De Grado. Bogotá: Universidad Católica De Colombia 2018. [consulta: marz.

PORRAS, Eve. Ingeniería de sistemas. Sistemas operativos móviles: iOS [en línea]. Internet (11 de abril de 2012). [Consultado: 1 de abril de 2020]. Disponible en: http://eve-ingsistemas-u.blogspot.com/2012/04/sistemas-operativos-moviles-ios.html.

RAMNATH, Rajib y LOFFING, Cheyney. Beginning IOS Programming for Dummies [en línea]. New Jesey. 2014, p.14. [Consultado: 30 de abril de 2020]. Disponible en Internet: https://books.google.com.co/books?id=8tIsAwAAQBAJ&pg=PA14&lpg=PA14&dq=core+os+layer&sou.

RAY, Jhon. OS 8 Application Development in 24 Hours [en línea]. 2. Ed. Sams teach yourself. 2015 p 123. [Consultado: 2 de abril de 2020]. Disponible en: https://books.google.com.co/books?id=FS75BgAAQBAJ&printsec=frontcover&hl=es#v=onepage&q&f=false.

REAL ACADEMIA ESPAÑOLA. Sistema [en línea]. RAE. España. (2020). [Consultado: 18 de abril de 2020]. Disponible en:http://buscon.rae.es/draeI/SrvltObtenerHtml?origen=RAE&LEMA=sistema&SUPIND=0&CAREXT=1 0000&NE DIC=No#sistema_operativo.

ROJAS, Cristián, Evaluación De La Seguridad De Aplicaciones Móviles Bancarias [en línea]. Trabajo fin de máster. Chile. Universidad De Chile. 2016. [consulta: marzo de 2020]. Disponible en: https://pdfs.semanticscholar.org/d521/52537e84f6513b0dc94bc6ce5d9.

SAAVEDRA, Fernando. OWASP Top Ten Mobile Risks [en línea]. Audea. Madrid. España. (15 de octubre de 2019). [Consultado: 1 de abril de 2020]. Disponible en: https://www.audea.com/owasp-top-ten-mobile-risks/.

SALAZAR, Edgar. Pruebas de Seguridad en aplicaciones web según OWASP. [en línea]. Owasp. 2018. [consulta: marzo 2020]. Disponible en: https://owasp.org/www-pdf-archive/OWASP_SUSCERTE.pdf.

SALTZER, Jerome y SCHROEDER, Michael. The Protection of Information in Computer Systems [En línea]. Invited Paper; Reino Unido: 1975 [consulta: marzo de 2020]. Disponible en: https://www.cl.cam.ac.uk/teaching/1011/R01/75-protection.pdf.

SALTZER, Jerome y SCHROEDER, Michael. The Protection of Information in Computer Systems [En línea]. Invited Paper; Reino Unido: 1975 [consulta: marzo de 2020]. Disponible en: https://www.cl.cam.ac.uk/teaching/1011/R01/75-protection.pdf .

SUPERINTENDENCIA FINANCIERA DE COLOMBIA. Comunicados de prensa. [sitio web] Bogotá: 05 de junio de 2018. [Consultado: 19 de abril de 2020]. Disponible en: https://www.superfinanciera.gov.co/jsp/Publicaciones/publicaciones/loadContenidoPublicacion/id/10097.

Tahiri, Soufiane. 2016. Mastering Mobile Forensics. Birmingham B : Packt Publishing Ltd, 2016.

TECH CRUNCH. 6.1B Smartphone Users Globally By 2020, Overtaking Basic Fixed Phone Subscriptions [en línea]. (2 de junio de 2015) Ingrid Lunden TechCrunch [Consultado: 5 de mayo de 2020]. Disponible en: https://techcrunch.com/2015/06/02/6-1b-smartphone-use.

TECNO VILLANUEVA. ¿Qué es el sistema operativo iOS y cuando se inició? Historia y evolución [en Línea]. Tecnovn, Cordoba, Rep.Argentina (08 de julio de 2019). [Consultado: 5 de abril de 2020]. Disponible en: https://tecnovn.net/que-es-el-sistema-operativo.

Thiel, David. 2016. iOS Application security. USA : No strach press. Inc, 2016.

TORI. Carlos. hacking ético, Sea proactivo: descubra las vulnerabilidades antes de que otro lo haga. Mastroianni Impresiones. 2008. ISBN 978-987-05-4364-0.

TORRADO, Jacobo. OASAM: oasam-auth-authentication [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-auth-authentication [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-bl-business-logic [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-bl-business-logic [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-conf-conFiguration-and-deploy-management [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-conf-conFiguration-and-deploy-management [Tr.

TORRADO, Jacobo. OASAM: oasam-crypt-cryptography [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-crypt-cryptography [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-dv-data-validation [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-dv-data-validation [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-info-information-gathering [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-info-information-gathering [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-is-intent-spoofing [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en:

https://github.com/b66l/OASAM/tree/master/oasam-is-intent-spoofing [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-leak-information-leak [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-leak-information-leak [Traducción: Propia].

TORRADO, Jacobo. OASAM: oasam-uir-unauthorized-intent-receipt [En Linea]. Github, 29 de diciembre 2016. [Consultado: 15 de marzo de 2020]. Disponible en: https://github.com/b66l/OASAM/tree/master/oasam-uir-unauthorized-intent-receipt [Traducción: Propia].

UNIVERSIDAD CARLOS III DE MADRID. Software de Comunicaciones [en línea]. España. (1 de febrero de 2018). [Consultado: 1 de abril de 2020]. Disponible en: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android.

VANEGAS, Carlos Alberto. Citado por: MUÑOZ, Yamir. Estado del arte vulnerabilidades de seguridad en sistemas operativos móviles andrioid y iOS [en línea]. trabajo de monografía como requisito de grado para optar el título de especialista en seguridad i.

VENTURINI, Guillermo. ¿Qué es rootear? [en línea]. Tecnología Fácil. Esperanza, Santa Fe. (2016). [Consultado: 20 de marzo de 2020]. Disponible en: https://tecnologia-facil.com/que-es/que-es-rootear/.

VICO, Ángel J. La columna 80, El blog técnico-personal de Ángel J. Vico… en español [en Línea]. Arquitectura de Android (17 de febrero de 2011). [Consultado: 1 de abril de 2020]. Disponible en: https://columna80.wordpress.com/2011/02/17/arquitectura-de-an.

WA3F. [En línea] Web Application Attack and Audit Framework 2013. [Consultado: 25 de marzo de 2020].Disponible en: http://w3af.org/.

ZHANG, Kuan. Security and Privacy for Mobile Healthcare Networks. 2 ed. New York: Springer. 2015, 2 p. ISBN: 978-3-319-24715-1.