ANÁLISIS DINÁMICO DE SEGURIDAD EN APLICACIONES ANDROID CON EL PROYECTO DE SEGURIDAD MÓVIL OWASP

ING. YEISSON VALENTINO JARAMILLO QUIRAMA

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA – UNAD
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
MEDELLIN
2019

ANÁLISIS DINÁMICO DE SEGURIDAD EN APLICACIONES ANDROID CON EL PROYECTO DE SEGURIDAD MÓVIL OWASP

ING. YEISSON VALENTINO JARAMILLO QUIRAMA

Trabajo de grado para optar el título de
Especialista en Seguridad Informática

Asesor
Ing. HERNANDO JOSE PENA HIDALGO

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA – UNAD
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
MEDELLIN
2019

Nota de aceptación:

_____
_____
_____
_____
_____

Firma de los jurados

Ciudad, Fecha

Contenido

pág.

LISTA DE IMÁGENES

pág.

LISTA DE TABLAS

pág.

LISTA DE ANEXOS

pág.

RESUMEN

El propósito de este estudio es la comprensión de la seguridad móvil haciendo énfasis en el sistema operativo Android para lo cual se analiza la historia y su evolución, malware y formas de ataque, arquitectura y funcionamiento en pro de una mejor comprensión de la tecnología. Posteriormente se ejecutarán herramientas de análisis disponibles en el mercado que ayudaran en la evaluación de posibles fallos de seguridad y como la metodología OWASP Mobile hace recomendaciones basadas en los juicios de expertos a nivel mundial.

Los resultados obtenidos de este estudio servirán como guía para futuros ambientes de pruebas mejorando la calidad del desarrollo y evitando la exposición de datos no cifrados y la comunicación insegura http.

Palabras claves: OWASP, Android, seguridad, análisis estático, análisis dinámico

NOTA ACLARATORIA SOBRE RESPONSABILIDAD

Las pruebas de análisis de resultados son de tipo académico y no viola la privacidad e integridad de las aplicaciones ya que se observa su funcionamiento y comportamiento en un laboratorio seguro y controlado.

Así también ni el autor ni la UNAD se hacen responsables del uso que le puedan dar a la información contenida en este documento.

INTRODUCCION

Es cierto que el mundo de hoy se mueve basado en la movilidad, cada año las empresas tecnológicas lanzan al mercado nuevos modelos móviles cada vez más inteligentes para saciar un mercado que ya entiende que los dispositivos hacen parte de una nueva revolución digital y en síntesis la tecnología móvil se ha convertido en una herramienta tecnológica que impulsa un nuevo cambio en la forma en que se accede a la información.

En la actualidad el sistema operativo más popular del mundo es Android el cual cuenta en su AppStore de Google con más de 3 millones de aplicaciones lo que hace que se vuelva un ambiente propicio para ataques cibernéticos con fines de secuestro de la información donde las estadísticas demuestran un crecimiento en los ataques de hasta un 40% con un promedio de 1,2 millones por [1]mes.

Generalmente los usuarios no tienen presente la peligrosidad de estos ataques, pues confían sus datos a las aplicaciones entregando información personal de suma importancia. Este estudio trata riesgos enfocados en OWASP Mobile para el sistema operativo Android en base a almacenamiento de datos e información en trasporte seguro. Se hablará de los lineamientos mínimos y los planes de pruebas que deben tener en cuenta los desarrolladores de aplicaciones.

[1] Más allá de Google Play y App Store. Los usuarios recurren a alternativas a las tiendas oficiales para instalar aplicaciones que aún no han sido lanzadas en su país o que ofrecen promociones. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://elpais.com/tecnologia/2017/05/31/actualidad/1496242186_229624.html

1 PLANTEAMIENTO DEL PROBLEMA

1.1 ANTECEDENTES DEL PROBLEMA

La App Store, tienda de aplicaciones presenta una herramienta de acceso a desarrollos de gran utilidad para las necesidades de cada uno, sin embargo, es cierto que los desarrollos allí dispuestos a veces no tienen el análisis necesario antes de ser liberada al público. [2]Una de las críticas de muchos especialistas es la libertad de instalar Apps de terceros que no están en la plataforma de Google Play. Esto por supuesto es un riesgo innecesario que abre la puerta a todo tipo de vulnerabilidad.

Un filtro establecido en la instalación de Apps es dar conocimiento al usuario de los permisos que requiere, acceso a datos y características de conexión. Pues bien, un usuario puede saber por ejemplo si la aplicación puede enviar mensajes de texto, pero, aunque sea claro estos detalles no ayudan mucho pues muchas veces hay acciones en segundo plano que son invisibles para el usuario.

1.2 PLANTEAMIENTO DEL PROBLEMA

¿Cómo puede un equipo de desarrollo construir y mantener aplicaciones móviles seguras según la metodología OWASP MOBILE?

El incremento del uso de aplicaciones móviles viene en aumento considerable, [3]tanto así que la venta de equipos de cómputo ha caído vertiginosamente. Son más las organizaciones como bancos, juegos, redes sociales y gobiernos que ponen sus funcionalidades en aplicaciones de tipo móvil haciendo de esta variedad un blanco potencial para los ciberdelincuentes con diferentes fines.

Google indico que el aumento de aplicaciones maliciosas va en aumento ya que de las 8.5 millones un 77% son consideradas malware con ataques tipo Ransomware

[2] Nueva vulnerabilidad crítica en Android: si descargas de la Play Store, tranquilo, [en línea] [citado el 4 de julio, 2013]. Disponible en internet: https://www.xatakamovil.com/seguridad/nueva-vulnerabilidad-critica-en-android-si-descargas-de-la-play-store-tranquilo

[3] IMás allá del PC. Este es el futuro de la computación. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: http://www.elmundo.es/economia/innovadores/2017/06/19/5947920f468aebac028b4612.html

en aumento hasta en un [4]50%. Pero la movilidad es un servicio que da demasiadas ventajas a sus usuarios y por eso se debe mirar con mucho cuidado las nuevas formas de ataques y las formas de implementación para tener las aplicaciones blindadas.

Son tantas las utilidades y funciones de los móviles y las aplicaciones que existe la posibilidad de manejar datos de todo tipo, videos, fotos, conversaciones, claves e información bancaria, todo esto hace que la vulnerabilidad crezca con ataques de malware, software espía entre muchos más pues el abanico es muy grande. También juega el mal uso de los usuarios respecto a las aplicaciones y el manejo que requieren ya que es un activo critico volviéndose en bancos de información personal teniendo así un riesgo muy alto. Ahora bien, ese es el tema de datos, pero hay factores contrarios a la voluntad del usuario que hace que sea crucial blindar la información como por ejemplo cuando se pierde el móvil o es robado sin saber el uso que le darán a la información allí almacenada.

Android como sistema operativo predominante permiten que los desarrolladores piensen más en este tipo de aplicaciones y generalmente publican desarrollos solo con las mínimas medidas de seguridad y lo que es peor cuando se instalan aquellas Apps que ni siquiera están en la App Store hace que la vulnerabilidad alcance niveles críticos. Los móviles se han vuelto en bancos de información personal siendo así un riesgo muy alto de seguridad.

1.3 FORMULACION DEL PROBLEMA

¿Qué nivel de seguridad tienen las aplicaciones nativas para el sistema operativo Android según los riesgos OWASP Mobile M2 y M3?

[4] Ransomware en Android subió más del 50% durante 2016. [en línea] [citado el 5 mayo, 2018]. Disponible en internet: https://www.dinero.com/empresas/confidencias-on-line/articulo/ransomware-en-android-subio-mas-del-50-durante-2016/242231

2 JUSTIFICACION

Gran parte de la información personal reposa en dispositivos móviles, los Smartphone para todo tipo de eventos, desde llamadas, envió de textos, transferencias bancarias, fotografías sociales y personales, envió de correo electrónico entre muchas más, ya se puede saber la ubicación exacta de la persona lo que para muchos es una violación a la privacidad. Pues como la mayoría de los usuarios desconocen estos temas y de una manera inocente confían sus datos a las aplicaciones, nace entonces la necesidad de conocer en qué nivel se encuentran algunas aplicaciones con sus debilidades y riesgos existentes.

A nivel mundial las estadísticas muestran un incremento considerable en los ataques a este tipo de [5]tecnologías, las principales casas de antivirus cuentan los ataques por miles principalmente en Android con ataques de tipo malware, donde la finalidad principal era la de cobrar dinero por el secuestro de datos. De tal modo y basados en las [6]estadísticas donde se habla de un aumento de Ransomware en aplicativos móviles, este trabajo pretende demostrar algunos de los riesgos más comunes en este tipo de aplicaciones móviles. Basados en estadísticas de algunas casas de antivirus como Kaspersky Lab[7], las prácticas de seguridad que tienen los desarrolladores son muy arcaicas, pues se realizó el estudio de muchas aplicaciones donde se evidenciaba que solicitaban permisos de acceso de datos del usuario y otras no comunicaban transparentemente al usuario las intenciones reales de la App. Es necesario fomentar una correcta practica de seguridad de la información en este tipo de desarrollos para dispositivos móviles

[5] Un reciente análisis de Avast sobre 160 millones de dispositivos móviles demuestra que el cibercrimen móvil está en aumento. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://blog.avast.com/es/nueva-investigacion-revela-el-incremento-de-amenazas-moviles

[6] El Ransomware móvil se triplicó en el primer trimestre de 2017. El número de archivos detectados llegó a 218.625, según informe de Kaspersky Lab. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: http://www.eltiempo.com/tecnosfera/novedades-tecnologia/estadisticas-del-ransomware-movil-en-primer-trimestre-de-2017-91760

[7] KASPERSKY LAB. Kaspersky Security Bulletin 2013. Overall statistics for 2013. December de 2013. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://www.securelist.com/en/analysis/204792318/Kaspersky_Security_Bulletin_2013_Overall_statistics_for_2013

3 OBJETIVOS

3.1 OBJETIVO GENERAL

Realizar un estudio del análisis dinámico de la seguridad en el desarrollo de las aplicaciones móviles a partir de la metodología Owasp Mobile

3.2 OBJETIVOS ESPECÍFICOS

- Realizar un estudio del esquema de la arquitectura Android y su seguridad.
- Identificar riesgos de almacenamiento y comunicación inseguros según OWASP
- Realizar un laboratorio controlado que permita evaluar la seguridad en aplicaciones móviles
- Realizar un informe de diagnóstico de las aplicaciones evaluadas, donde se evidencie aspectos claves de seguridad respecto a almacenamiento y transporte de datos inseguro.

4 MARCO REFERENCIAL

4.1 MARCO TEÓRICO

4.1.1 La movilidad. Es innegable el papel que juega las tecnologías de información en la vida de las personas. El software es un pilar fundamental entre el entorno y el mundo móvil por lo cual la arquitectura del software debe ser de calidad en relación con los dispositivos móviles. A continuación, en la figura 1, un ejemplo de conectividad móvil en diferentes escenarios cotidianos.

Figura 1. Computación móvil

Fuente: https://labmoviles.virtual.uniandes.edu.co/

La movilidad se presenta como la tecnología que abre el camino a la vanguardia de la sociedad actual. Es bueno entender que esta tecnología ha tenido un largo camino para alcanzar los niveles a los que ha llegado todo en pro de brindar soluciones prácticas en el creciente mundo de la movilidad.

Entre las tecnologías de este tipo se encuentran implementadas en computadores portátiles, telefonía celular, dispositivos electrónicos conexiones vía Bluetooth, wifi, satelital entre otras. Hoy en día se habla con mucha claridad sobre la tecnología, pero fue gracias a los inventos de grandes matemáticos que se probó que las ondas de radio podían propagarse a la velocidad de la luz[8]. Claro esta que esta comprobación demoró mucho en ser aplicada.

4.1.2 Tecnología móvil. La tecnología móvil en el presente ha tenido un crecimiento exponencial y no parece desacelerarse, esto explica la aceptación a nivel mundial que ha tenido. Esta tecnología ha llegado para quedarse. Cabe anotar que fue a inicios del año 2000 cuando los diseños industriales en específico la industria de software y todos sus componentes crecen sin picos de detenimiento, hoy en día gran parte de la información se ha digitalizado. La tecnología móvil ha hecho parte de estos avances y aunque hace parte de muchas formas de comunicación la que mejor entendida es la forma en que los dispositivos móviles se comunican. Los grandes computadores se han convertido en teléfonos de bolsillos con funcionalidades que satisfacen las necesidades de la mayoría de los usuarios.

Hoy en día las tecnologías móviles y velocidades variables como anchos de banda. Tecnología 3G, 4G, Wifi y ahora la tan esperada 5G que expresa velocidades de 100 Gb por segundo. El tema hoy en día es de suma importancia, muchas aplicaciones web se están acomodando a los diseños móviles, otras han visto como hay más interacción en sus Apps que en sus páginas web, la explicación es sencilla, es más fácil iniciar un teléfono móvil que un pc y la portabilidad hace que el uso de equipos de mesa este en picada.

Las primeras formas de comunicación inalámbrica empezaron con los sistemas analógicos [9]1G y a finales de la década de los años 90s la tecnología celular se convirtió en algo más sofisticado añadiendo características que hoy son básicas como las llamadas de voz o análogas. Pero este avance permitió que las personas empezaran a comunicarse de forma portátil lo cual generó un uso general.

[8] Descubrimiento de las ondas de Radio: la confirmación de la Teoría Electromagnética, [en línea] [citado el 28 de abril, 2009]. Disponible en internet: https://www.investigacionyciencia.es/blogs/fisica-y-quimica/10/posts/descubrimiento-de-las-ondas-de-radio-la-confirmacin-de-la-teora-electromagntica-10186

[9] El teléfono celular. Historia y evolución de los celulares, [en línea] [citado el 26 de febrero, 2019]. Disponible en internet: https://tecnologia-informatica.com/telefono-celular-historia-evolucion-celulares/

4.1.3 Tecnología 2g. Afortunadamente la tecnología no se detiene y a mediados de la época de los años 90s se empezó a desarrollar nuevas redes de comunicación conocidas como GSM o 2G, desde aquí ya se empezó a formar los primeros pre-Smartphone permitiendo la comunicación de forma digital a la analógica, esto abrió la posibilidad de nuevas formas de funcionamiento tecnológicos para los aparatos móviles siendo la base para la tecnología y capacidad moderna como enviar mensajería de texto, descargar información, acceder a la web, ver videos. En este punto de la historia se inició la expansión móvil, pues las redes 2G cambiaron la forma de comunicación de línea fija. Es un avance en los inicios de la movilidad, aunque aún tenían muchas características básicas respecto a lo que se conoce hoy.

4.1.4 Tecnología 3g. Esta tecnología vio sus inicios a principios de la década del año 2000 y en ese momento se puede decir que empezó la era de los teléfonos inteligentes. El acceso a esta tecnología dio capacidades de comunicación inalámbrica muy potentes donde ya había paquetes de datos como las vistas en la Web diferente a los mecanismos de conmutación de las redes 2G.

Con la llegada 3G, los teléfonos se volvieron inteligentes desde el punto de vista portable y cuando se tuvo acceso a los paquetes de enlace de alta velocidad conocidos como HSDPA, las posibilidades de transmisión dieron acceso a todas las formas de medios en línea a la que hoy se puede acceder. Nacen las aplicaciones móviles, capaces de realizar las mismas funciones de los sitios web y su crecimiento ha sido acelerado. La tecnología de hardware también se vio obligada a rediseñar su arquitectura para poder mostrar las capacidades digitales que se acostumbraban a ver en la web y no solo eso, sino que nacieron mejores formas de interacción con los usuarios, todo gracias a las tecnologías móviles.

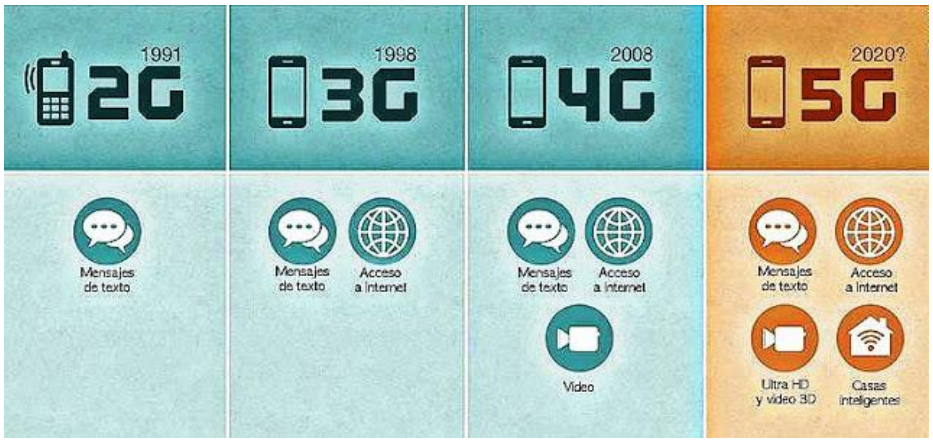
4.1.5 Tecnología 4g. Esta tecnología nace como reemplazo de la 3G pues es mucho más potente ya que conmuta y optimiza los datos eficazmente. Significa que es diez veces en capacidad y potencia que su antecesora lo que abre la posibilidad del crecimiento móvil y la demanda de Apps para realizar todas las tareas que se hacían en la web

4.1.6 Tecnología 5g. Esta tecnología ha venido pensándose hace más de 10 años sin embargo hasta ahora se está volviendo una realidad teniendo como principal avance un aumento de rendimiento en la transferencia de datos lo que permitirá la conexión de todas las cosas en el mundo. Efectivamente los operadores poco a poco están empezando a implementar estas tecnologías y se espera su masificación en el año 2020.

¿Pero qué significa esta nueva tecnología que viene como reemplazo de la 4G?. Aún hay muchas preguntas respecto a esta nueva tecnología sin embargo antes de explicar cómo funciona 5G, hay muchos detalles que requerirían una explicación extensa, pero esta próxima generación aumentara las velocidades de carga exponencialmente esto significa que las comunicaciones de las redes inalámbricas serán muy rápidas.

Su funcionamiento difiere del LTE, ya que 5G opera en tres bandas de espectro diferentes, puede sonar algo sin importancia, pero en la aplicación de la tecnología móvil su efecto será inmenso. Lo importante a saber es que la banda ancha será mejorada y eso significa mejores procesos para la transmisión de datos. Es una promesa esperada y lograr la conexión en todo tiempo y en cualquier parte gracias a sus capacidades de ser versátil flexible y sobre todo ayudar en lo que se conoce como tecnología ubicua, conexión de humanos y objetos. A continuación, en la figura 2, se observa las capacidades de cada banda de conexión móvil desde los inicios con 2G hasta la 5G que promete la conectividad de todas las cosas.

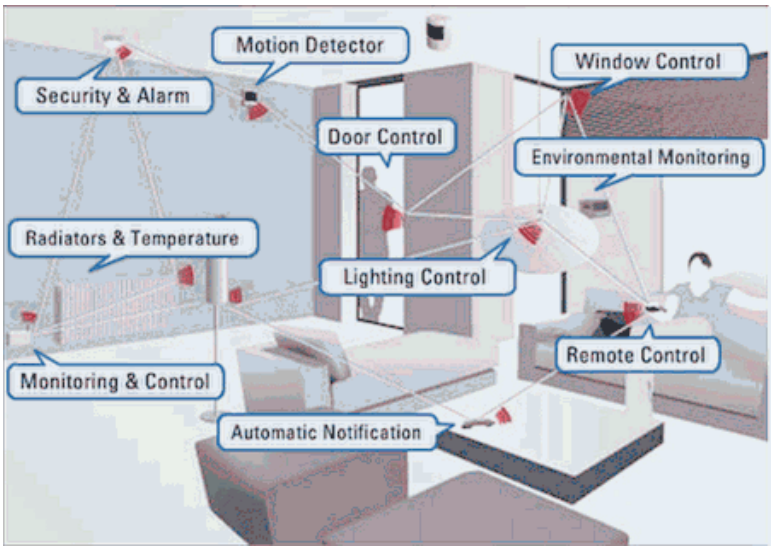
Figura 2. La tecnología 5G será el alma de la nueva economía mundial



Fuente: https://www.panoramadirecto.com/2018/01/la-tecnologia-5g-sera-el-alma-de-la.html

4.1.7 Tecnología ubicua. Se define como la capacidad de integrar la conectividad de los usuarios sin darse cuenta de su uso. Es una tecnología pensada para facilitar la interacción con lo digital, se piensa en integraciones entre personas y autos autónomos, o llevar dispositivos que paguen las cuentas sin necesidad de dinero entre muchas cosas más. A continuación, en la figura 3 se muestra una representación de ubicuidad tecnológica donde la conexión de los diferentes componentes de la casa está sincronizados.

Figura 3. Tecnología ubicua en la sociedad

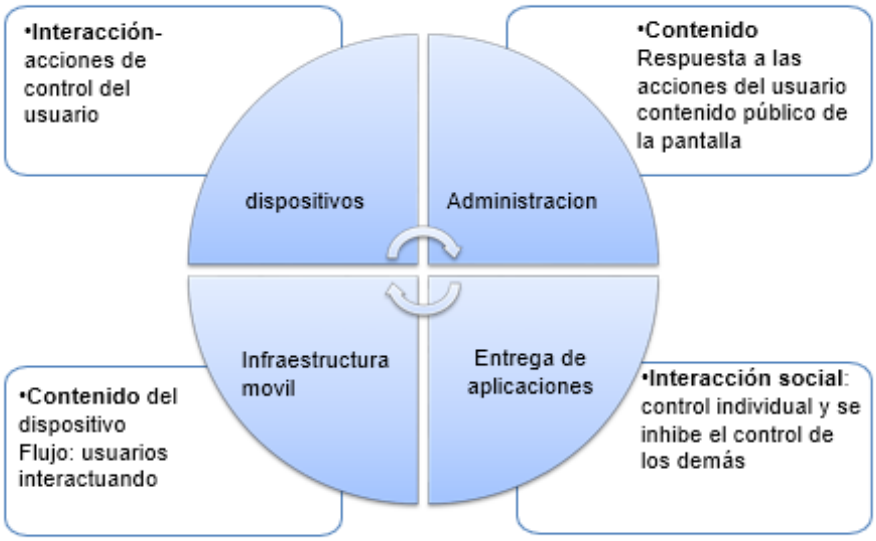


Fuente:
http://1.bp.blogspot.com/_8uS0WapllXA/TVMfgYTyT6I/AAAAAAAAAC4/PyP83kMnAhc/s1600/Ubicua.gif

4.1.8 Historia de los Smartphone. Antes del nacimiento de los teléfonos inteligentes el mundo tenía otro matiz, las personas tenían otra mentalidad y también otras formas de pasar el tiempo, si, era otra época y su visión la dará cada uno, según su perspectiva, hay quienes dicen que era mejor, otros que era peor, en fin, lo cierto es que los teléfonos inteligentes han llegado para quedarse y eso es inevitable. La vida ha cambiado, ahora un Smartphone es una extensión de las personas, el acceso móvil a la tecnología, a las aplicaciones y a la cantidad de temas da cuenta de la nueva era digital pasando por lo web a lo móvil.

En sus inicios quizá no se tenía un panorama tan amplio sobre el alcance de la tecnología móvil, se ha pasado desde simple llamadas y mensajes de texto a un completo mundo de enriquecimiento en cada dispositivo móvil. La sociedad poco a poco ha venido dándose cuenta del cambio de tecnología que se ha implantado, es una revolución digital como se muestra a continuación. A continuación, en la figura 4 se explica conceptualmente el funcionamiento móvil, pensado en la interacción de usuario, contenido que se presenta en pantalla y privacidad de las aplicaciones.

Figura 4. Marco conceptual de los dispositivos móviles



Fuente este estudio

4.1.9 Inicio llamado móvil. Un hombre llamado Martin Cooper realizo la primera llamada[10] en un laboratorio con el teléfono Motorola DynaTAC 8000x (como se puede observar en la figura 2). Este producto que no llego a las manos de los usuarios durante 10 años más fue el comienzo de la revolución móvil. Luego en Japón surgió la compañía Telephone NTT y empezó el primer servicio analógico conocido como 1G. A continuación, Martin Cooper

4.1.10 Década de 1980. Fue a partir de los años 80 cuando se dio el primero acceso de telefonía móvil y pasaron muchos años luego de que fuera una corriente mundial. Ya en el 83 la compañía Ameritech Mobile introdujo en Estados Unidos la primera red 1G del país empezando en Chicago. A continuación, un teléfono de esta década.

[10] Así fue la primera llamada por celular 45 años atrás, [en línea] [citado el 3 de abril, 2018]. Disponible en internet: https://mundo.sputniknews.com/tecnologia/201804031077570710-celular-primera-llamada/

4.1.11 Década de 1990. Nace afortunadamente la tecnología GSM y se establece en Europa la red común donde los usuarios tenía el servicio telefónico incluso desde países diferentes, el precursor de esta tecnología fue el Nokia 1011. Por su parte IBM en el año 1994 presento su teléfono de pantalla táctil Simon y aunque se adelantaba a su tiempo podían enviarse correos electrónicos y fax. Era el inicio de la revolución digital ya que vendría la internet y la adaptación de las personas a lo digital[11]. A continuación, un teléfono de la década.

4.1.12 Años 2000. Se puede decir que a partir de esta fecha empezó la verdadera revolución del Smartphone y fue cuando Steve Jobs reveló el primer iPhone, puesto que aquí se abrió un abanico de posibilidades ya que antes los teléfonos solo ofrecían teclados y una pésima interacción con la internet, el iPhone brindó la posibilidad de navegación como si se viera una computadora incluyendo nuevos diseños y teniendo todo a un dedo.

4.1.13 2010 y presente. Corre el año 2019 y el mundo supera los cinco mil millones de personas y la gran mayoría usan teléfonos inteligentes. Desde búsqueda de empleos, lecturas de libros, juegos digitales y tantas posibilidades que da estos dispositivos lo que indica que han llegado para quedarse. Actualmente hay nuevas capacidades de almacenamiento[12], procesadores muchos más potentes que computadoras sencillas y con las redes 5G pensadas para el 2020 la tecnología móvil promete telemetrías, autos autónomos, realidad virtual, ciudades inteligentes. El abanico es muy grande y estos dispositivos hacen parte de la revolución digital móvil.

4.1.14 Sistema operativo Android. Con el crecimiento de los teléfonos inteligentes el sistema operativo base va tomando mucha fuerza e importancia a nivel mundial. Android es el sistema operativo que parece funciona muy bien es estos dispositivos que contienen mucho hardware y funcionalidades operativas que usan el hardware de manera óptima. Aunque existen varios sistemas operativos para móviles como lo son iOS de Apple, WebOS de Palm o Symbian, Android cuenta con la máquina virtual Dalvik que ejecuta su propio código de bytes, se puede decir que Dalvik es un componente central y que las aplicaciones escritas en Java se ejecutan en máquinas virtuales. Es una plataforma creada por Android Inc., empresa adquirida por Google y que funciona como proyecto de código abierto.

[11] La historia del teléfono móvil: Origen, pasado y presente, [en línea] [citado el 28 de febrero, 2019]. Disponible en internet: http://culturacion.com/la-historia-del-telefono-movil-origen-pasado-y-presente/

[12] La tecnología móvil y el internet de las cosas, [en línea] [citado el 7 de febrero, 2018]. Disponible en internet: http://noticias.universia.es/ciencia-tecnologia/noticia/2018/02/07/1157844/tecnologia-movil-internet-cosas.html

Si se habla de código abierto es bueno entender que hay un grupo de 78 compañías Open Handset Alliance (OHA) que se encargan del desarrollo y distribución de Android. Es un software que se puede descargar desde un repositorio y se puede modificar de acuerdo con los términos de licencia BSD y Apache. El sistema operativo es sin duda el más usado en las terminales móviles en el mundo, como sistema operativo sirve para cualquier tipo de sistema inteligente. Fue desarrollado por la empresa originalmente llamada [13]Android Inc. en el año 2000 pero luego fue adquirida por Google quien lanzo su primera edición en el año 2008, en su momento Android 1.0. De eso queda poco pues ha habido más de [14]20 lanzamientos donde se ha mejorado las versiones respecto a compatibilidad y experiencia de usuario. Android ha evolucionado con nuevas funciones y mejoras respecto a software por las casas de tecnología. Cada lanzamiento se caracteriza por tener un número de versión, ejemplo 4.4, su nivel de API 3 y un nombre de algún dulce ejemplo KitKat. A continuación, en la figura 5 se visualiza un recuento de las versiones que se ha tenido Android desde Apple Pie hasta Oreo, versiones con nombres de dulces.

Figura 5. Versiones Android

Fuente: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android

Android se toma muy enserio su calidad y aproximadamente cada 6 meses, lanzan actualizaciones del API. Se puede decir que Android utiliza 4 capas de software donde la capa kernel de Linux proporciona los servicios del sistema operativo (SO), además de todos los controladores que acceden al hardware con código de bajo nivel con las capas de ejecución de Android y sus bibliotecas nativas, este tiempo

[13] Android Inc. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://es.wikipedia.org/wiki/Android_Inc.

[14] versiones de Android [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://actualizar-android.com/versiones

presenta la máquina virtual llamada Dalvik (DVM) que se encarga de correr el código tipo byte en las aplicaciones del sistema operativo además de la máquina virtual Java (JVM) que por su parte ejecuta los bytecode de Java y es así como hay una comunicación de bibliotecas nativas que permite que la CPU ejecute directamente y sea más rápida en dar soluciones a tareas de larga ejecución. A continuación, en la figura 6 se muestra el esquema de la arquitectura del sistema operativo Android donde se resume la interacción entre componentes seguros pues su base Kernel está restringido. Su arquitectura se define de la siguiente manera según el sitio oficial de desarrolladores para Android[15]:

Figura 6. Arquitectura Android

Fuente: *https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android*

[15] Arquitectura de la plataforma, [en línea] [citado el 8 de enero, 2019]. Disponible en internet: https://developer.android.com/guide/platform/?hl=es-419

4.1.15 Capa de aplicación. Se conoce como la capa más superior de la arquitectura. Las aplicaciones se ejecutan aquí con el usuario final. Aunque el sistema operativo trae aplicaciones preinstaladas en el dispositivo estas aun funcionan con datos de usuarios y como base de información para otras aplicaciones

4.1.16 Marco de aplicación. Las aplicaciones que están en desarrollo hacen uso de esta capa de clases y servicios. Esta capa de puede extender con componentes propios de la arquitectura, también habilitar el acceso a datos. Algunas funciones de esta capa.

Administrador de actividades: se encarga de gestionar los ciclos de vida de las aplicaciones y su correcta función.

Administrador de recursos: se encarga de proporcionar los recursos para que el desarrollo se ejecute correctamente.

Administrador de notificaciones: aquí las aplicaciones personalizan sus alertas en la barra de estado.

Administrador de ubicación: este recurso se encarga de gestionar la ubicación geográfica según la aplicación.

Administrador de paquetes: este recurso permite recuperar los paquetes que se han instalado en el dispositivo.

Administrador de ventanas: este recurso gestiona correctamente las vistas y diseños de las aplicaciones.

Administrador de telefonía: este recurso es muy importante debido al manejo que le da al dispositivo para configurar las conexiones de red y servicios del teléfono.

Tiempo de ejecución de Android: este recurso permite la ejecución de las aplicaciones, Android tiene su máquina virtual DVM (Dalvik Virtual Machine).

4.1.17 Librerías. Estas bibliotecas del sistema operativo escritas en C y C++ [16]son inaccesibles al usuario, pero el marco de aplicaciones resuelve el acceso para ser usadas en los diferentes desarrollos**.**

4.1.18 Kernel de Linux. Es el núcleo central del sistema operativo ya que proporciona lo más esencial para su funcionamiento, como lo es la [17]administración de energía, memoria seguridad, comunicación entre otros. Es muy interesante la formación del sistema operativo base con sus capas funcionales para dar el resultado que se conoce en Android, pues Linux tiene muchas distribuciones y todas con resultados diferentes**.**

4.1.19 Aplicaciones móviles. Generalmente todas las aplicaciones se escriben en el lenguaje de programación JAVA con los Apis (interfaz de programación) de programación tradicionales. El proceso es el natural para un programador, pero lo que interesa es que esta programación se transforma en bytecode de Dalvik y luego quedara un único archivo para Android (APK). También es posible encontrar implementaciones o partes en C o C ++. Y como resultado los archivos y su estructura en el empaquetado tienen el archivo AndroidManifest.xml que servirá para pruebas estáticas que no es el caso en este estudio. A continuación, la tabla 1 muestra los archivos y carpetas de un APK.

Tabla 1. Estructura APK

Estructura	Descripción
AndroidManifest.xml	Declaración de la aplicación, componentes, servicios etc.
classes.dex	Bytecode Dalvik, generado a partir del código Java
resources.arsc	Archivos del recurso comprimido
META-INF/	Metadata relacionado con el contenido del APK
res/	Directorio de recursos, almacenando archivos como imagen, diseño, etc.
assets/	Directorio de datos, almacenando archivos que se compilarán en el archivo APK

Fuente: http://huawei.forosactivos.net/t261-estructura-de-un-apk

El archivo AndroidManifest.xml es de suma importancia ya que define, nombre, versión, los permisos necesarios para su ejecución y los componentes que posee la aplicación. Este archivo, aunque está codificado en XML binario, puede ser leído

[16] Ibid., p 24.
[17] Ibid., p 24.

con herramientas como apk-tool6 entre otras. Google agiliza los procesos de desarrollo de Apps proporcionando herramientas conocidas como (ADT). Este se encarga de convertir los archivos. dex en apk.

4.1.20 Componentes de una app. De acuerdo con muchas universidades[18] que enseñan los temas relacionados con Android y su desarrollo dicen que son 4 componentes esenciales para cualquier aplicación. Bloques de construcción de una aplicación de Android. Componentes que están poco acoplados y se vinculan por el archivo de manifiesto que debe exponer toda aplicación donde reposa la descripción de cada componente y su interacción, metadatos e información de hardware y requisitos de la plataforma, además de los permisos necesarios para funcionar. Estos componentes son: [19]Actividades, Servicios, Proveedores de contenido y receptores de transmisión.

4.1.21 Actividades. Una actividad se refiere a una información mostrada en la pantalla de usuario se puede ver por ejemplo cuando se envía un correo electrónico puede existir una actividad que muestre correos disponibles, otra que permite leerlos y otra que permite escribirlos. Muchos hablan de las actividades como la capa de presentación o interfaz de usuario de la aplicación. Estas actividades usan vistas y establecen el diseño además de entregar las salidas para la interacción del usuario.

4.1.22 Servicios (servicies). El servicio es aquel que puede ejecutarse en segundo plano, un ejemplo claro es cuando se escucha música mientras se lee algo, o se descarga una actualización mientras esta en las redes sociales. Estos son como trabajadores invisibles de las aplicaciones. Los programadores de aplicaciones saben que son servicios que se ejecutan en el back-end, que actualizan fuentes de datos y dan notificaciones. Cuando las aplicaciones se ejecutan en segundo plano son estos servicios lo que tienen dicha tarea.

---

[18] Master en desarrollo de aplicaciones Android. Componentes de una aplicación. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/149-componentes-de-una-aplicacion

[19] portfolio juan José Cánovas Bustamante. Elementos de una App de Android. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://juanjosecanbus.wordpress.com/2014/09/28/practica-1-elementos-de-una-App-de-android-pmm/

4.1.23 Proveedores de contenido (content provider). Como su nombre lo dice este proveedor se encarga de administrar los datos de las aplicaciones para presentarlos a cualquier petición, ejemplo de ello puede ser el lenguaje para una aplicación, este se guarda en la configuración y el proveedor de contenido tiene la tarea de hacerlo funcionar. Esta tarea administra y conserva datos de la aplicación, interactúa con la base de datos y son responsables de compartir datos más allá de los límites de la aplicación.

4.1.24 Receptor de anuncios (broadcast receiver). Este componente es importantísimo pues se encarga de avisarnos de los elementos esenciales de Android tales como la carga de las baterías, SMS recibidos, llamadas perdidas, en fin, es de suma importancia para conocer el estado del dispositivo. Se sabe que son oyentes atentos, ya que permiten que su aplicación escuche los intentos que satisfacen los criterios de coincidencia especificados por nosotros. Los broadcast hacen que nuestra aplicación reaccione a cualquier intención recibida, lo que los hace perfectos para crear aplicaciones basadas en eventos.

4.1.25 Seguridad en dispositivos móviles. La seguridad siempre es un tema de alta importancia en estos días, las empresas y los mismos consumidores ya tienen más conciencia sobre los riesgos informáticos que existen. Casos de piratería, robo de información, denegación de servicio, inyección de virus, son el pan de cada día para los expertos en seguridad. Antes la preocupación era la web, los parches de los sistemas operativos como Windows, sin embargo, ya el panorama pasó a la tecnología móvil.

Los sistemas operativos móviles están siendo víctimas de ataques por ciberdelincuentes maliciosos y en la mayoría de los casos los usuarios no tienen precaución de los mínimos requisitos de seguridad que deben tener para evitar estas amenazas. Android se está convirtiendo en el principal objetivo de los piratas[20] y una de las principales causas es que es de código abierto y es el más popular a nivel mundial.

El dispositivo móvil hace parte de toda persona y por eso ya las agencias estatales[21] han optado por revisar los recursos de los empleados para evitar los riesgos que se derivan del acceso que tienen los dispositivos móviles a la información. Esto sin mencionar que muchos usuarios tienen versiones de Android muy antigua, lo que

[20] Android fue el sistema operativo más atacado durante 2017. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://elcomercio.pe/tecnologia/empresas/youtube-android-sistema-operativo-atacado-2017-video-noticia-486089

[21] Huawei: por qué Estados Unidos considera al gigante tecnológico chino una amenaza a la seguridad nacional, Redacción, BBC News Mundo. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://www.bbc.com/mundo/noticias-46475391

hace más vulnerable el tema de seguridad ya que las últimas versiones han parchado agujeros de seguridad a lo largo de la historia del sistema operativo.

4.1.26 Ataques cibernéticos. Una de las formas más comunes de ataque es por medio de malware, un software malicioso peligrosísimo pues está desarrollado para obtener acceso al dispositivo y dañarlo. En síntesis, está pensado para violar la integridad de datos. Pues bien, este tipo de virus no avisa y solo se instala. Google se toma muy enserio este tema y ha detectado toda clase de virus como exploit de root para privilegios o envió de datos sin consentimiento del usuario entre otras violaciones de seguridad que resalta este tipo de virus. No se puede negar que los dispositivos están siendo atacados por este tipo. [22]Kaspersky dice que el 99% del malware detectado tenía como objetivo la plataforma Android y esto debido al alto uso de este sistema pues ya está considerado como el sistema operativo que más se usa en el mundo además de la facilidad con que se pueden hacer aplicaciones y la ingeniería social para instalar apps fuera de la app store[23]**.**

La casa de antivirus ESET ha dicho que se han detectado una familia de Malware que es capaz de saltarse los sistemas de seguridad y que estas se han encontrado en la app store como aplicaciones, el tema es muy complejo pues ya existen formas sofisticadas de virus que ocultan su detección pues utilizan una arquitectura de capas que es capaz de engañar los servicios de seguridad de Google pudiendo estar en la app store. A continuación, en la figura 7 se observa ataques registrados desde el 2012 hasta 2017 presentando un crecimiento particularmente en la plataforma Android.

[22] Kaspersky lab. Virus y Malware en Móviles Android. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://www.kaspersky.es/resource-center/threats/mobile

[23] Portaltic Europa press. Detectadas ocho aplicaciones con 'malware' en Google Play Store capaces de saltarse sus sistemas de seguridad. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: http://www.europapress.es/portaltic/ciberseguridad/noticia-detectadas-ocho-aplicaciones-malware-google-play-store-capaces-saltarse-sistemas-seguridad-20171116144529.html

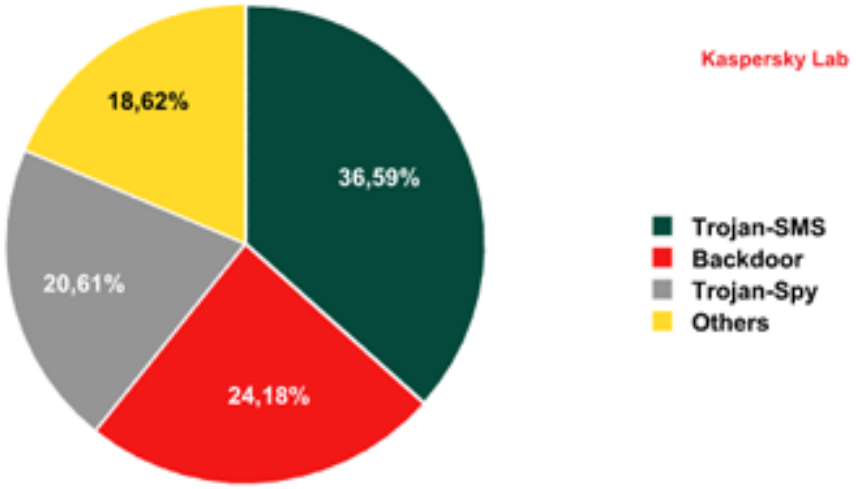
Figura 7. Ataques registrados en diferentes plataformas

sistema operativo	2012	2013	2014	2017
Android	497.082	860.937	1.069.503	1.468.619
Windows	346.457	354.410	397.533	570.937
ios/MacOs	212.899	293.428	359.483	504.147
RIM	34.722	31.253	27.150	24.121
Otros	1.122.213	871.718	702.786	396.959
Total	2.213.373	2.411.746	2.556.455	2.964.783

Fuente: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTUwU5EFtWwEYu9QKAt0PKEvxKt8awpWrTuo4jg3yalUJcyZeyP

A continuación, la figura 8 muestra el porcentaje de los diferentes tipos de virus que se han encontrado en los dispositivos móviles que tienen productos Kaspersky.

Figura 8. Tipos de Malware en dispositivos móviles



Fuente: https://www.kaspersky.es/blog/ataques-contra-android/296/

4.1.27 Riesgos en la tecnología móvil. Cuando se habla de los riesgos que puede tener una aplicación móvil, se encuentran muchas situaciones que ayudan a que las vulnerabilidades sean más fuertes. Entender que los usuarios de las apps depositan la confianza en las aplicaciones es un motivo para garantizar que sus datos estén protegidos correctamente. Algunos expertos en seguridad dicen que el 90%[24] de las aplicaciones presentan problemas de seguridad además las empresas caminan a pasos cortos sobre el tema de seguridad móvil, por lo cual el rango de ataque es mayor para cualquier pirata informático.

Debilidad en servidores: Es común que no exista grandes presupuestos para el cuidado y la implementación de seguridad en la comunicación de un servidor y un aplicativo móvil, es muy común encontrar poca configuración se seguridad en el [25]servidor lo que deja a las aplicaciones móviles con problemas de vulnerabilidad informática

Protección binaria: la protección binaria permite que no pueda aplicarse ingeniería inversa en el código y dar como resultado Apps falsas generalmente para robo de datos. [26]Para esto se recomienda técnica de endurecimiento binario, bifurcaciones, técnicas de codificación

Almacenamiento de datos inseguros: Es un problema muy común, debido a que el teléfono es la bodega de datos por lo cual si es extraviado podrán acceder a los datos privados.

Capa de transporte con problemas: Para este caso que es muy común, los hackers utilizan redes públicas para acceder a los datos que por allí pasan, una de las mejores formas es el uso de transporte seguro SST

Fuga de información involuntariamente: Este es un tema muy diferente respecto al ítem anterior y se da como violación de la privacidad del usuario pues a veces los datos quedan guardados en la cache o registro de aplicaciones, tema que debe ser controlado por el desarrollador

---

[24] Cómo solucionar los problemas más comunes de Android, [en línea] [citado el 15 mayo, 2018]. Disponible en internet: https://elandroidelibre.elespanol.com/2015/02/como-solucionar-los-problemas-mas-comunes-de-android.html

[25] Seguridad en el desarrollo de aplicaciones móviles: los 5 mayores riesgos, de Alejandro Caballero, marzo 8, 2018, Noticias mundo móvil, [en línea] [citado el 15 mayo, 2018]. Disponible en internet: https://kingofApp.es/blog/seguridad-desarrollo-aplicaciones-moviles-mayores-riesgos/

[26] Ingeniería inversa de malware protegido [en línea] [citado el 15 mayo, 2018]. Disponible en internet: http://s3lab.deusto.es/ingenieria-inversa-malware-protegido/

Autorización y autenticación: El tema de autenticación deficiente es muy común en las aplicaciones, a veces los usuarios seleccionan contraseñas de 4 dígitos y los creadores de apps no obligan a que guarde estándares adecuados. Otro problema es las sesiones abiertas que guardan las apps, lo que se traduce como vacíos de seguridad ya que por medio de fuerza bruta de inicio de sesión un hacker puede realizar operaciones con las apps por lo que se recomienda no iniciar sesión si no está en línea el usuario.

Criptografía: es un problema común en las aplicaciones móviles, pues no se implementa correctamente creando vulnerabilidades que pueden llevar al descifrar los datos sensibles del usuario.

Inyección de código malicioso: por lo general los hackers incorporan este tipo de código malicioso cuando logran cambiar el marco e interpretación de la app o también con un ataque de fuerza bruta haciendo que se interprete en el celular. La mejor forma de evitar esto es validando los datos de entrada del usuario.

Configuración de seguridad no confiable: muchas veces en los desarrollos hay configuraciones ocultas que distinguen los niveles de usuarios, comunicación entre servidores y clientes, o comunicación entre diferentes fuentes lo que permite al hacker interceptar este tipo de datos si no tienen una configuración de seguridad correcta. Para esto se usa listas blancas, interacción de usuarios con parámetros de entrada, evitar pasar información sensible entre apps.

Sección abierta: configurar incorrectamente la sesión de usuario puede acarrear problemas de seguridad muy grandes puesto que, si el teléfono es robado y logran desbloquearlo, las sesiones pueden quedar abiertas dejando los datos de usuario desprotegidos.

4.1.28 Proyecto de seguridad móvil Owasp. Todo desarrollo móvil debe implementar las mejores pautas de seguridad y analizar su arquitectura teniendo presente los 10 riesgos de seguridad que propone OWASP Mobile, metodología de techical hacking con miras a buscar vulnerabilidades en las aplicaciones. Es una metodología muy conocida entre expertos de la computación. A continuación, en la figura 9 el logo que representa la metodología de análisis de riegos Owasp.

Figura 9. Owasp Mobile

Fuente: https://www.waratek.com/owasp-top-10-application-security-risks

OWASP es una aplicación de seguridad conocida a nivel mundial, alimentada y discutida por expertos de todas partes que fortalecen las técnicas de [27]testing que pueden realizar a una aplicación antes de sacarla al mercado productivo. OWASP fue inicialmente para análisis de aplicaciones web, sin embargo, con el auge de la tecnología móvil en el año 2014 inicia el análisis de seguridad donde describe los principales riesgos detectados. Tiene como principal enfoque el análisis de la capa de aplicación y la comunicación entre el dispositivo y el servidor donde se presenta autenticaciones y almacenamiento de datos en la nube.

4.2 TOP 10 DE RIESGOS MOBILE DEFINIDOS POR OWASP[28]

4.2.1 M1 uso inapropiado de la plataforma. Este riesgo se presenta cuando los desarrolladores no usan correctamente las funciones que le brinda la plataforma, lo que hace que las aplicaciones desarrolladas queden expuestas a vulnerabilidades de seguridad. Esta vulnerabilidad es muy común y fácilmente explotable. Tiene como característica los problemas de seguridad de los mismos sistemas operativos ya que a veces los desarrolladores no tienen control sobre estos riesgos[29].

Algunas características de este riesgo pueden ser, pasar por alto las recomendaciones de seguridad del sistema operativo donde se desplegará el desarrollo, un ejemplo de ello es asegurar los servicios expuestos en Android. No tener principios de desarrollo de calidad que fortalezcan la capa de seguridad de los desarrollos. No tener claridad de las acciones de las Apps cuando son desarrolladas, esto significa que puede haber una buena intención pero que se puede convertir en una función incorrecta del llamado del API. Cuando una

[27] Proyecto de seguridad móvil OWASP, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Home

[28] Mobile Top 10 2016-Top 10, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

[29] Ibid., p 35.

aplicación solicita demasiados permisos de facto está incurriendo en el uso inadecuado de la plataforma. Muchas veces los usuarios no tienen en cuentan las precauciones que deben tomar frente a actividades sospechosas de las Apps que descargan, incluso de han presentado bloqueos de pantalla y solicitud de códigos como una clase de secuestro de los móviles.

4.2.2 M2 almacenamiento de datos inapropiado. Este riesgo varía mucho en su ataque debido a que muchas aplicaciones ajenas a la desarrollada pueden usar cache, cookies o cualquier otra forma de recopilar datos que le permita acceder al equipo. Para evitar este tipo de riesgo se recomienda tener muy presente la autenticación y cifrado de todo el almacenamiento de cache y sobre todo revisar exhaustivamente los datos perdidos que deriven en violaciones de privacidad. Este riesgo se considera muy severo. Algunas características de este riesgo se presentan en la fuga de datos involuntarias, datos no cifrados para bases de datos SQL, datos abiertos en archivos de registro, datos XML que se almacenan en archivos de manifiesto, datos almacenados en binario, vulnerabilidad de almacenamiento en tarjeta SD[30].

4.2.3 M3 comunicación insegura. Esta es una vulnerabilidad extremadamente común presente en la mayoría de las aplicaciones con estructura cliente-servidor. Si bien los desarrolladores a menudo son diligentes en cuanto a la protección del procedimiento de autenticación y los datos en reposo, rara vez se molestan en cifrar los datos en movimiento. Cuando no hay cifrado la aplicación automáticamente entra en la vulnerabilidad de hombre en medio ahí la información se vuelve altamente vulnerable. Los ataques para este tipo de vulnerabilidad se pueden dar en dispositivos de red o un malware que este en el dispositivo. Se recomienda ante todo utilizar cifrado y verificación de datos, protocolos SSL y TLS y aceptando solo comunicación de certificados de confianza. Es una vulnerabilidad de alto impacto. Algunas características de este riesgo se presentan en **c**omunicaciones TCP/IP, comunicaciones Wi-Fi, comunicaciones Bluetooth, comunicaciones infrarrojas, GSM, 3G, 4G, SMS[*31*]

[30] Ibid., p.35.
[31] Ibid., p.35.

4.2.4 M4 autenticación insegura. Este riesgo está relacionado con la autenticación y manejo de sesión. En las aplicaciones móviles, muchos ataques se dan con herramientas que envían solicitudes al servidor saltándose la aplicación e inicio de sesión, por eso es muy importante tener todos los cuidados de autenticación en las apps móviles ya que estas cuentan con opciones de trabajar sin conexión lo que permite la explotación de autenticación y envió de solicitudes al server. Cuando un ataque es logrado saltándose la autenticación se obtiene acceso al sistema, datos e inserción de comandos causando la falla del sistema. Este riesgo es muy importante ya que se ha visto muchas debilidades que se explotan fácilmente. Se recomienda el uso de autenticación verificada en el servidor y verificaciones de código modificado en las Apps[32].

4.2.5 M5 criptografía insuficiente. Este riesgo a veces se pasa por alto, pero es de alta importancia debido a que un ciberdelincuente puede obtener la lectura de información incorrectamente cifrada. Esto deriva en problemas de seguridad de la información, obteniendo información personal y demandas contra la dueña de la App por el no cuidado de los datos personales, en Colombia este delito está tipificado y es por eso por lo que se deben aplicar mecanismos muy fuertes de cifrado y descifrado utilizando estándares de criptografía[33].

4.2.6 M6 autorización insegura. Este riesgo puede parecerse al M4, sin embargo, actúan diferente pues la vulnerabilidad que se presentan en el lado del servidor cuando falla el proceso de autenticación se considera crítico. Cuando una App presenta problemas de autenticación se puede decir que tiene las puertas abiertas al ataque, a veces estos ataques son difíciles de detectar y su resultado deriva en perdida de confiabilidad en la organización dueña del App. Se recomienda en muchos casos la autenticación y roles permitidos por usuario en el lado del servidor, tener muy claro los niveles de permiso que otorga la aplicación[34].

4.2.7 M7 calidad del código pobre. Este riesgo analiza las vulnerabilidades que se encuentran en la codificación de un App. Es cierto que los niveles de calidad en el desarrollo pueden variar según la madurez en el ciclo del software, pero eso no significa que dichos errores puedan derivarse en vulnerabilidades a ser explotadas. Hay ejemplos claros que se dan en este tipo de riesgos como lo son el desbordamiento de memoria que termina en robos de información y control del App. No es fácil detectar estos errores debido a la capa de abstracción que tiene la codificación y esto es un punto a favor para la seguridad, pero mientras exista la vulnerabilidad existe el riesgo de ataque. Se recomienda ante todo seguir

[32] Ibid., p.35.
[33] Ibid., p.35.
[34] Ibid., p 35.

estándares de calidad en el código para que existan correctas implementaciones y captura de excepciones que blinden el código ante problemas de desarrollo[35].

4.2.8 M8 código alterado. Este riesgo se presenta cuando el atacante logra entrar al código base y modificar su estructura. Aunque parezca complejo hay muchas formas de hacerlo por medio de manipulación de métodos, clases y parches incrustado en el código. La principal motivación es saltarse los niveles de pago y ser premium, así violan los derechos de autor y la cadena de distribución de la aplicación. Aunque también se puede incluir malware por medio de recursos y distribuir las modificaciones por parte de terceros. Muchas veces el usuario no se percata de los riesgos de instalar Apps fuera de la Google Play. El punto es que logrado este ataque se puede decir que la cadena de seguridad se rompe por lo que hay que quitar o actualizar urgentemente el desarrollo, tema que no es tan sencillo si es un desarrollo complejo y de muchos meses de trabajo. Se recomienda usar técnicas que detecten la manipulación ya que es muy difícil de detectar[36].

4.2.9 M9 ingeniería inversa. Este riesgo no siempre tiene intenciones de daño en las Apps, sino que se da para realizar estudios de patrones y gustos para luego venderlos a los clientes que comercializan productos. Es lo que se conoce como las sugerencias en las publicidades en la web. También se puede presentar ataques por medio de telemetrías pues el estudio de la información y patrones de uso derivan en conocimiento de errores de la app donde se puede aprovechar para tratar de explotarla. Se recomienda siempre el cifrado de la información para que no sea accesible por medio de vulnerabilidades que muestren la información. Es un riesgo muy complejo pues casi siempre se da de manera no invasiva pero no debe tenerse en poco y es bueno aplicar metodologías que combatan la ingeniería inversa[37].

4.2.10 M10 funcionalidad superflua. Este riesgo nació en el año 2016 cuando se evidencio una vulnerabilidad muy grave la cual surge cuando en el desarrollo no se eliminan las pruebas de aplicación que se dan como métodos adicionales. Un buen ejemplo es cuando se ejecutan pruebas con todos los privilegios activos y no se eliminan cuando sale a ambientes productivos, este es un riesgo de puerta trasera y dará control total al atacante. Es una vulnerabilidad difícil de explotar, pero no significa que no se pueda explotar, por eso se recomienda eliminar dichos métodos de prueba en la compilación final[38].

[35] Ibid., p 35.
[36] Ibid., p 35.
[37] Ibid., p 35.
[38] Ibíd., p 35.

4.3 FUTURO DE OWASP MOBILE

Es muy interesante entender que este análisis de riesgos que ofrece OWASP es muy completa y profesional. De deben tener en cuenta en cualquier proceso de análisis de pruebas debido a que el panorama de la seguridad informática está teniendo niveles de ataques muy altos y esto en el ámbito móvil. OWASP está en constante cambio, se espera para finales de las 2018 y 2019 nuevas publicaciones según el análisis de los múltiples profesionales que participan en su elaboración.

4.4 M2. ALMACENAMIENTO DE DATOS INSEGURO

Las vulnerabilidades de información es uno de los riesgos más críticos de cualquier desarrollo. [39]Las organizaciones tienen la obligación de velar por la seguridad de la información ya que un ataque puede dejar los datos personales de un usuario al descubierto lo cual significa la perdida de confiablidad en cualquier organización. La información debe ser confidencial cuando se trata de datos personales o críticos para el negocio. En Android el almacenamiento inseguro se puede dar de muchas formas, sobre todo cifrado inseguro y fácil de descifrar por medio de tantas técnicas que existen para este fin.

4.4.1 Base de datos. Las aplicaciones generalmente realizan almacenamiento local, muchas usan SQLite y allí hay cadenas en texto simple sin cifrar. Esto significa el acceso a datos de usuario por lo que siempre es recomendable y casi obligatorio el no almacenamiento de credenciales en el dispositivo o el envío de estas de forma visible entre aplicación y servidor para que el cumplimiento de este riesgo sea efectivo.

4.4.2 Cifrado. El cifrado de datos es de gran utilidad para mitigar el riesgo de acceso a la información, cuando se da un hash correcto la información que se comparte entre aplicación y servidor será de difícil acceso al atacante. Es necesario que en los procesos de desarrollo se conozcan las técnicas de cifrado fuerte y la aplicación de metodologías de valores hash que eviten el almacenamiento de contraseñas. También se recomienda el cifrado de toda la base de datos, aunque SQLite no tiene esta opción se puede valer de extensiones tipo AES de 256 bytes**.**

[39] M2-Almacenamiento de datos inseguros, [en línea] [citado el 1 marzo, 2019]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-M2-Insecure_Data_Storage

4.4.3 Ubicación de archivos. Hay muchas opciones de almacenamiento que Android proporciona y los datos confidenciales también se pueden almacenar. Almacenes de datos binarios, archivos de registro, archivos de manifiesto XML o almacenes de cookies. Los datos binarios son muy utilizados por los desarrolladores para ocultar los datos de los usuarios y la aplicación, pero son muy fáciles de acceder y leer.

Los datos puestos en el archivo de manifiesto XML muchas veces es utilizado para incluir información del hardware y otros lo han usado para guardar información del usuario lo cual se constituye en una pésima practica ya que puede ser accedida fácilmente por el atacante. También cuando el desarrollador deja abierto el modo de depuración de la aplicación y lo pone en ambiente productivo de esta forma la información confidencial suele pasarse también constituyéndose en un grave error de seguridad.

4.4.4 Permisos de archivos. Debe evitarse a toda costa el uso del modo "MODE_WORLD_READABLE" y "MODE_WORLD_WRITEABLE" a no ser que sea de estricta necesidad. Se recomienda siempre el accedo al almacenamiento interno desde la aplicación. Estas opciones dan mayor acceso al storage del dispositivo, pero eso significa vulnerabilidades de seguridad. También se recomienda eliminar en lo posible los permisos de compartir información con otras Apps.

4.4.5 Prácticas de prevención para Android. Siempre se debe tener presente las recomendaciones que brindan los expertos en seguridad. Se recomienda el uso de herramientas de cifrado para almacenamiento local y almacenes de archivos. Si se almacena en tarjetas SD el cifrado de datos con AES de 128 bytes y el uso de una contraseña maestra[40].Limitar el modo de compartir información con otras aplicaciones de desarrollo, No establecer la confianza absoluta en cifrado de información provisto por el sistema operativo, se recomienda la utilización de cifrados adicionales. Considere proporcionar una capa adicional de cifrado más allá de cualquier mecanismo de cifrado predeterminado provisto por el sistema operativo.

4.5 M3. PROTECCIÓN INSUFICIENTE EN LA CAPA DE TRANSPORTE[41]

Las vulnerabilidades que se presentan para este tipo de riesgo generalmente están dadas por la falta de protocolos seguros y eliminación de transferencias de datos

[40] Ibid., p 40.
[41] Ibid., p 40.

con certificados SSL. Pues bien es un riesgo critico debido a la exposición de datos, audios, llamadas y ataques de hombre en medio conocida como (MITM). Cabe anotar que muchas aplicaciones usan seguridad con protocolos SSL, pero básicamente en comunicaciones de autenticación olvidando otros canales y formas de comunicación de la aplicación por lo que los datos quedan expuestos. Generalmente las aplicaciones se diseñan para conmutar erróneamente comunicación por el canal http cuando no encuentran los certificados seguros o hubo un problema de comunicación. También hay problemas cuando las aplicaciones no lograr cifrar adecuadamente los datos y hay comunicación en las redes facilitando así la interceptación de la información.

La aseguración de la información cuando se comunica entre aplicación y servidor es una tarea de estricto cumplimiento y para eso los desarrolladores y arquitectos deben proveer correctas implementaciones que garanticen datos privados todo en pro de garantizar la integridad y confidencialidad de los datos y la comunicación. Como este riesgo se basa en la correcta implementación de certificados se recomienda usar claves privadas y largas de 2048 bits además de que el certificado sea válido y no esté vencido, además que esta emitida por una entidad certificadora. No es sencillo interceptar tráfico sin embargo hay condiciones que lo permite como, por ejemplo, la instalación de certificados en el proxy del dispositivo, la aceptación de confianza de la aplicación de este tipo de certificados, dispositivo funcionando como enrutador, omisión de protocolos SSL

Recomendaciones para comunicación segura: Certificados aceptados de entidades certificadoras, longitud de claves de al menos 2048 bits de longitud, protocolos seguros TLS v1.2, aceptación de cifrados seguros como lo son algoritmos AES256 y SHA2

4.6 SEGURIDAD APLICADA EN SISTEMA OPERATIVO ANDROID

Android ofrece una capacidad de seguridad estable, tiene modelos de permisos y su kernel de Linux ofrece las capas de seguridad del sistema operativo como aislamiento de procesos y protección de datos del usuario entre otras. Android además es líder en la industria del desarrollo y ecosistemas seguros sujeto rigurosamente a la seguridad pues, aunque está diseñado para ser abierto su software y hardware es avanzado en cuanto a datos y servicios, su fuerte es la confidencialidad, integridad y disponibilidad para las acciones de los usuarios.

Android también ofrece una gama de soporte y actualizaciones de seguridad permanente pues el equipo de desarrollo está en constante búsqueda de

vulnerabilidades. En su Play Services se pueden encontrar bibliotecas de seguridad críticas como es OpenSSL para el tema de comunicaciones. Android piensa en sus usuarios ya que cuando se instala cualquier aplicación esta muestra obligatoriamente los permisos a los que necesita acceder y ya es cuestión de cada usuario el dar ese aval. Esto permite que los ataques sean más difíciles de realizar por lo cual se recurre a la ingeniería social para lograr que los usuarios caigan en aplicaciones poco seguras, pero Android está pensado para reducir esta posibilidad de ataques debido a su negación de permisos sobre los dispositivos. Algo importante para recalcar es que los componentes de Android están asegurados y todo el código que esta sobre el kernel de Linux está restringido por Application Sandbox.

Anqué de igual manera se enseña a los usuarios a tomar precauciones como las actualizaciones, evitar redes abiertas, aplicaciones fuera de la Play store, revisar los permisos de acceso de las aplicaciones entre otros consejos[42]. La vulnerabilidad en Android según Security Project OWASP[43] es una debilidad o bug de una aplicación y esto puede ser aprovechado por un hacker para hacer cambios en la configuración de la aplicación o acceder a los datos del usuario, OWASP recomienda validar entradas, tener escritura de logs y cerrar conexiones a base de datos. La amenaza en Android se puede definir como acciones que puedan poner en peligro las aplicaciones y estas amenazas existen debido a las vulnerabilidades que se puedan encontrar[44] .

El riesgo en Android es una alta probabilidad de que una amenaza se ejecute. Existe un proyecto de investigación llamado Project Zero[45] de Google, que analizan e investigan fallas encontradas en diferentes terminales y han dado con 11 problemas de seguridad consideradas de alto impacto en muchos terminales y referencias. Un ataque puede suceder aprovechándose de las debilidades del sistema, esto es de alto impacto ya que pone en entredicho la seguridad de la que debe disponer cualquier aplicación. Esto siempre generara desconfianza en los usuarios si los ataques son perpetuados[46] .

---

[42] Seguridad para Android: cinco consejos fundamentales. Seguridad en Internet. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://latam.kaspersky.com/resource-center/preemptive-safety/android-security-tips

[43] OWASP Open Web Application Security Project. Category: Vulnerability. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://www.owasp.org/index.php/Category:Vulnerability

[44] Universidad Nacional de Lujan. Departamento de Seguridad Informática. Análisis a la seguridad de la información. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: http://www.seguridadinformatica.unlu.edu.ar/?q=node/12

[45] ABC tecnología. Android: el riesgo de los «smartphones» inseguros. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: http://www.abc.es/tecnologia/moviles/telefonia/abci-android-riesgo-smartphones-inseguros-201511070317_noticia.html

[46] MIERES, Jorge. Ataques informáticos. Debilidades de Seguridad comúnmente explotadas. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://www.evilfingers.com/publications/white_AR/01_Ataques_informaticos.pdf

Generalmente un ataque consta de los siguientes pasos. Reconocimiento, exploración, acceso o elevación de permisos y por último tratar de no dejar registro de su ataque. Existen muchos problemas de seguridad actualmente, por eso los expertos recomiendan utilizar técnicas criptográficas para cifrar información y esta debe aplicarse en la autenticación, integridad de datos firmas digitales y controles de acceso. La norma ISO 27000 contiene requisitos específicos para la seguridad de sistemas de información.

4.7 APLICACIONES SANDBOX

Android, dispone de un aislamiento entre el Kernel del sistema operativo y las aplicaciones, de modo que cada aplicación tiene un usuario UID único que se ejecuta por separado exceptuando las que están firmadas por el mismo certificado en cuanto a desarrollo. Esto significa que las aplicaciones solo pueden acceder a recursos si se les da el permiso requerido. Esto significa que el kernel se impone entre toda aplicación y el sistema del sistema operativo, esto ayuda por ejemplo a que las aplicaciones no puedan expiarse entre si ya que no hay permisos necesarios para este tipo de ejecuciones por lo que su seguridad está basada en separación de procesos y permisos de estilo UNIX. Hablar de este tipo de aplicaciones Sandbox es entender un modelo de seguridad que separa el Kernel como lo es las bibliotecas del sistema, tiempo de ejecución y otros procesos críticos que hace que los desarrolladores no puedan desarrollar sino hasta cierto punto o mejor dicho están limitados a un marco de desarrollo especifico, un API definido en base a seguridad, en otras palabras, para ejecutar el código arbitrario hay que romper la seguridad del Kernel Linux[47].

4.8 PROBLEMAS DE VERSIONAMIENTO

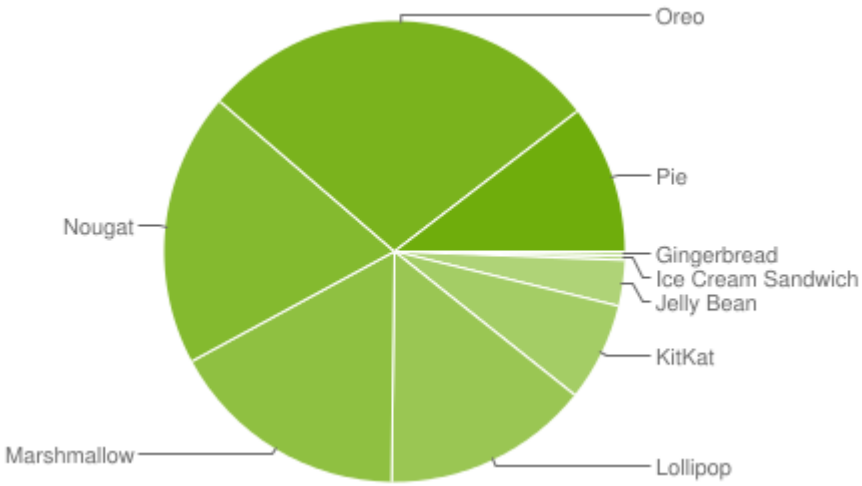
Este tema se refiere a las diferentes versiones de la plataforma Android y es que son necesario las actualizaciones debido a las mejoras de seguridad con las que viene cada versión. Android trato en 2012 de actualizar los diferentes sistemas existentes e hizo un sondeo respecto a la cantidad de dispositivos con diferente versión de sistema, pero al analizar no fue posible o salía muy costoso debido a los fabricantes de teléfonos y dispositivos que agregaban en la memoria y espacio diferentes desarrollos personales lo que hacía imposible estas actualizaciones y es por eso por lo que los fabricantes prefieren vender teléfonos nuevos con las últimas versiones de Android[48]. A continuación, la figura 10 muestra el número de

[47] Seguridad para aplicaciones móviles. ¿Cómo funciona el Sandbox Linux en Android? [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: http://seguridadparaaplicaciones.com/como-funciona-el-sandbox-linux-en-android/

[48] Paneles de control. Versiones de la plataforma. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://developer.android.com/about/dashboards/

distribuciones de Android en la actualidad y como va creciendo su participación en el mercado.

Figura 10. Distribución de versiones plataforma Android



Fuente: https://developer.android.com/about/dashboards/

4.9 MARCO CONCEPTUAL

4.9.1 La seguridad móvil. La seguridad móvil se presenta hoy como una prioridad en temas de arquitectura móvil, la tendencia presentada por las compañías de seguridad demuestra un incremento sustancial en los ataques de todo tipo en esta nueva tecnología, incluso los ataques a nivel desktop han caído y es porque la sociedad usa más los dispositivos móviles que los equipos de escritorio[49].

La comunicación y planificación de tareas cotidianas ya se presenta de modo móvil por lo que las organizaciones están cambiando sus modelos de negocio a la movilidad. Todo ataque va en pro de explotar debilidades a veces de hardware, pero la mayoría de las veces a nivel de aplicaciones, cualquier cuidado adicional que ayude a proteger la forma de comunicación y almacenamiento de datos es necesario, además del papel fundamental que deben hacer los proveedores de tecnología para que los usuarios tengan los cuidados básicos frente a posibles exploit de vulnerabilidades.

[49] Razones que explican las menores ventas de computadores, [en línea] [citado el 2 de marzo, 2019]. Disponible en internet: https://www.dinero.com/economia/articulo/razones-para-caida-venta-computadores-nivel-mundial/208282

Las amenazas en los dispositivos móviles han venido creciendo de forma exponencial de tal modo que en los últimos trimestres ha habido un crecimiento de las de 200%[50]. Las amenazas pueden variar desde la interrupción del funcionamiento del celular hasta el acceso indebido a los datos del usuario. Es casi estricto que las aplicaciones que están en la Play Store garanticen la privacidad e información. Según la casa de antivirus Kaspersky[51] Android ha sido el sistema operativo más atacando en los últimos años. Se han realizado estudios estadísticos y alcances según las regiones de los equipos que tienen este software como contramedida de los ataques tipo malware. A continuación, en la figura 11 se muestra la forma en que se opera los ataques a los dispositivos móviles. Esta información ha sido compartida por la casa de antivirus Kaspesrky lab. Los ataques durante el último año fue el más alto jamás visto[52], a lo largo de todo el tiempo se observaron nuevas técnicas de ataques incluso secuestro de DNS y aumento de spam de SMS.

Figura 11. Esquema de las principales amenazas móviles

Fuente: https://www.le-vpn.com/es/wp-content/uploads/2017/11/Top-Mobile-Apps-Security-Threats-1200x628-sp.jpg

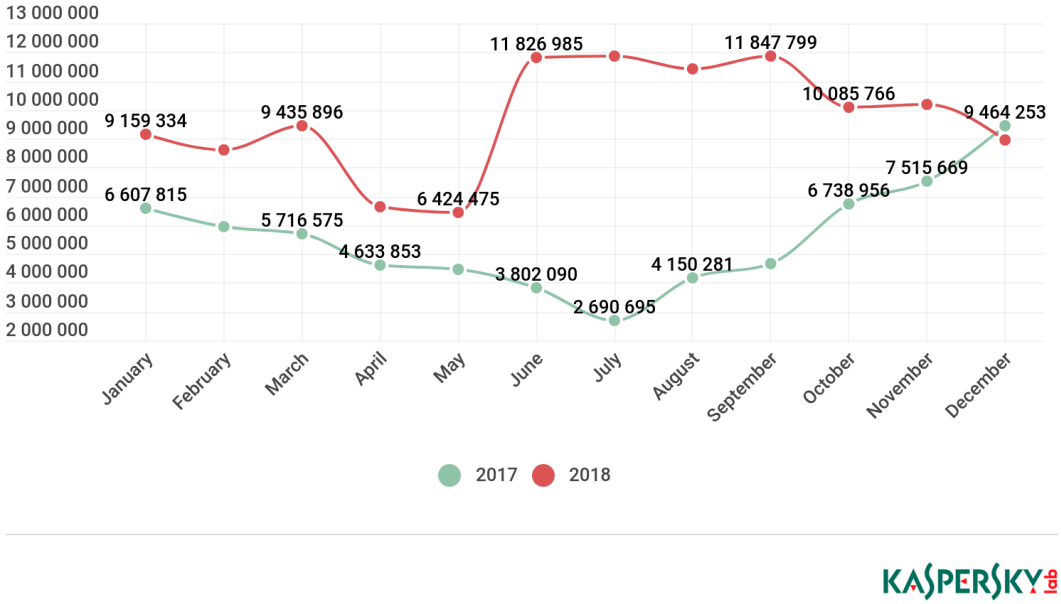
A continuación, en la figura 12 se observa el crecimiento de ataques cada mes entre el año 2017 y 2018 en dispositivos que tienen antivirus de la casa Kaspersky de donde se toman las estadisticas

[50] Las amenazas móviles. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://www.symantec.com/connect/blogs/las-amenazas-moviles

[51] MOBILE MALWARE EVOLUTION 2016, [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180734/Mobile_report_2016.pdf

[52] Los ataques cibernéticos se incrementaron este año, [en línea] [citado el 2 de marzo, 2019]. Disponible en internet: https://www.dinero.com/internacional/articulo/incremento-de-ataques-ciberneticos-en-el-2018/264180

Figura 12. Número de ataques detectados por Kaspersky Lab, 2018



Fuente: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2019/03/04082525/en-attacks-total.png

4.9.2 Ataque *droppers*. Es una forma de ataque que viene en aumento, se ha convertido en un arma preferida por los hackers ya que se generan aplicaciones que son troyanos en busca de información bancaria y visualización de anuncios sin consentimiento, también ejecución del explorador en segundo plano en busca de url´s que pagan por clics. La capacidad de estos troyanos es evitar su detección cambiando su hash de archivos aleatoriamente mientras su malware interior sigue intacto. Se han detectado un crecimiento muy alto en la clase Trojan-Dropper.AndroidOS.Piom. El problema de estos virus es que a veces no se detectan lo que significa que el mapa estadístico de ataques no mostrara su ataque.

4.9.3 Ataque troyano bancario. Este tipo de virus han tenido un crecimiento en el último año, esto debido al paso de la banca a la movilidad. Al principio era una amenaza fácilmente detectable ya que pretendía el Phishing por lo que los usuarios los detectaban fácilmente, sin embargo, en el último semestre los ataques han crecido y son de la familia Asacub[53] y Hqwar. Estas variantes no son nuevas para los expertos de seguridad pues han sido troyanos que han evolucionado en la forma de ataque como el paso de SMS detectados como Spam y la ingeniería social donde llega información que parece confiable, pero es totalmente falsa.

Es un problema en continuo crecimiento ya que muchas variantes de estos virus no se dejan desinstalar y secuestran aplicaciones legitimas llegando incluso a ejecutarlas de modo que se realicen transferencias bancarias sin supervisión del usuario. Android en su lucha contra el Phishing ha evitado la superposición de ventanas en las apps bancarias pero los hackers han sido capaces de ejecutar métodos de ofuscación y técnicas anti dinámicas infiltrando así los troyanos en la tienda de Android. Es por eso por lo que las pruebas de caja de arena deben aplicarse a las aplicaciones pues cada vez es más sofisticado las formas de evasión de virus activos.

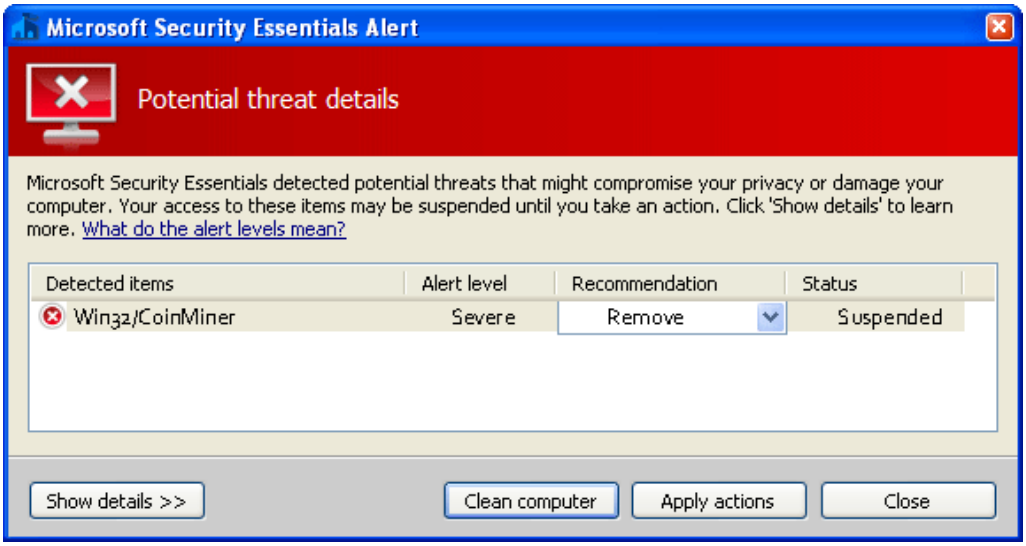
4.9.4 Ataque adware y software peligroso. Es una forma vieja de infección que se encontraba en la web pero que ha migrado a la tecnología móvil. Básicamente los clics que pagan los anunciantes son una forma de dinero seguro y los atacantes lo saben muy bien. Este tipo de troyanos no dañan el dispositivo a excepción de algunos modelos que se recalientan por los procesos en segundo plano.

Este tipo de troyanos piden acceso al root infectando el celular con boots. Un ejemplo claro fue una app que hacía funcionar la linterna del teléfono y había anuncios aleatorios incluso que sé que se ejecutaban fuera de la interfaz comercial que dispone Android. Es un problema ya que convierte el dispositivo en un adware zombi. El tema es complejo ya que estos virus están logrando tener acceso al directorio del sistema incluso los scripts de restauración de fábrica lo que hace casi imposible la eliminación de este.

[53] The rise of mobile banker Asacub, By Tatyana Shishkova on August 28, 2018. 10:00 am. [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://securelist.com/the-rise-of-mobile-banker-asacub/87591/

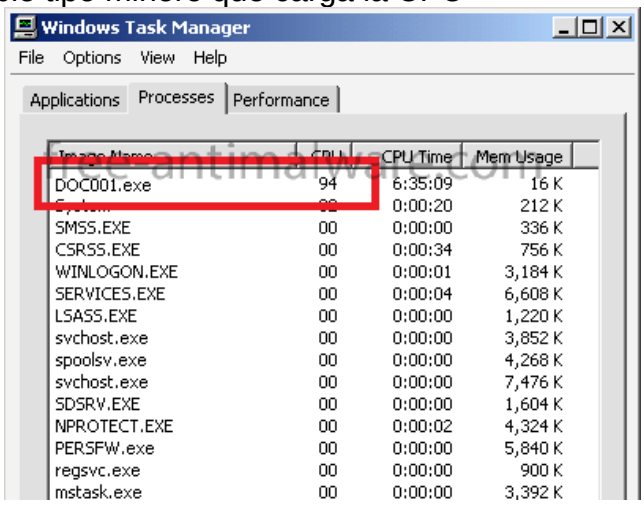
4.9.5 Troyanos mineros. Este tipo de virus también está en crecimiento y se debe a varios factores. Los dispositivos están siendo cada vez mejor equipados con tarjetas de procesadores gráficos, material necesario para el procesamiento de las criptomonedas. Es un virus fácil de instalar y fácil de detectar ya que su comportamiento carga el dispositivo y se empiece a sospechar de actividad maliciosa. A continuación, en la figura 13 y 14, la detección por parte del antivirus de Windows Defender y como sus procesos llevan CPU a niveles que bloquean las actividades de otros procesos.

Figura 13. Detección de tipo virus mineros



Fuente: este estudio

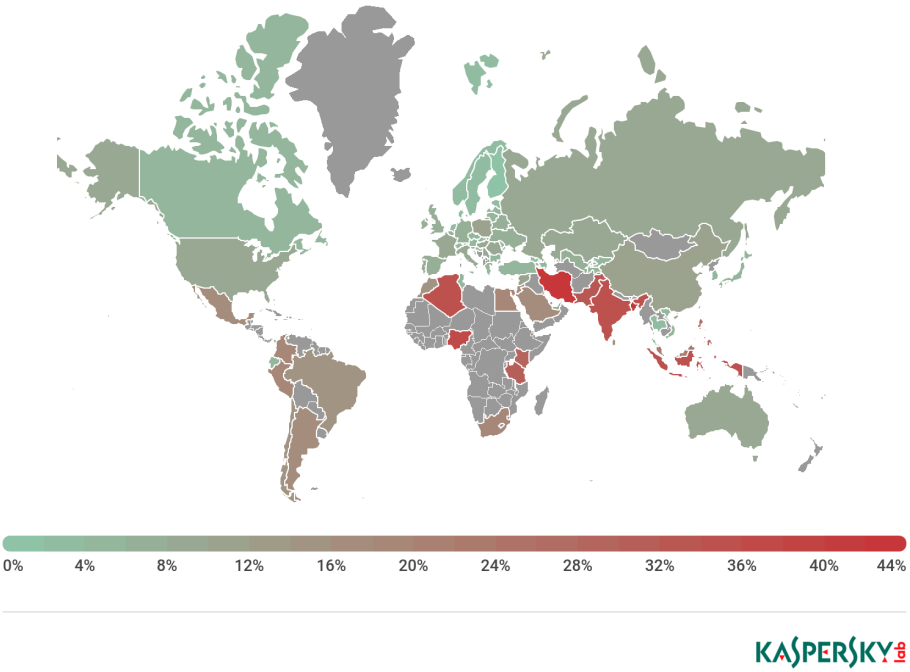
Figura 14. Ejecutable tipo minero que carga la CPU



Fuente: este estudio

A continuación, en la figura 15 se observa un panorama mundial de ataques por país en dispositivos que tienen software de la casa Kaspersky.

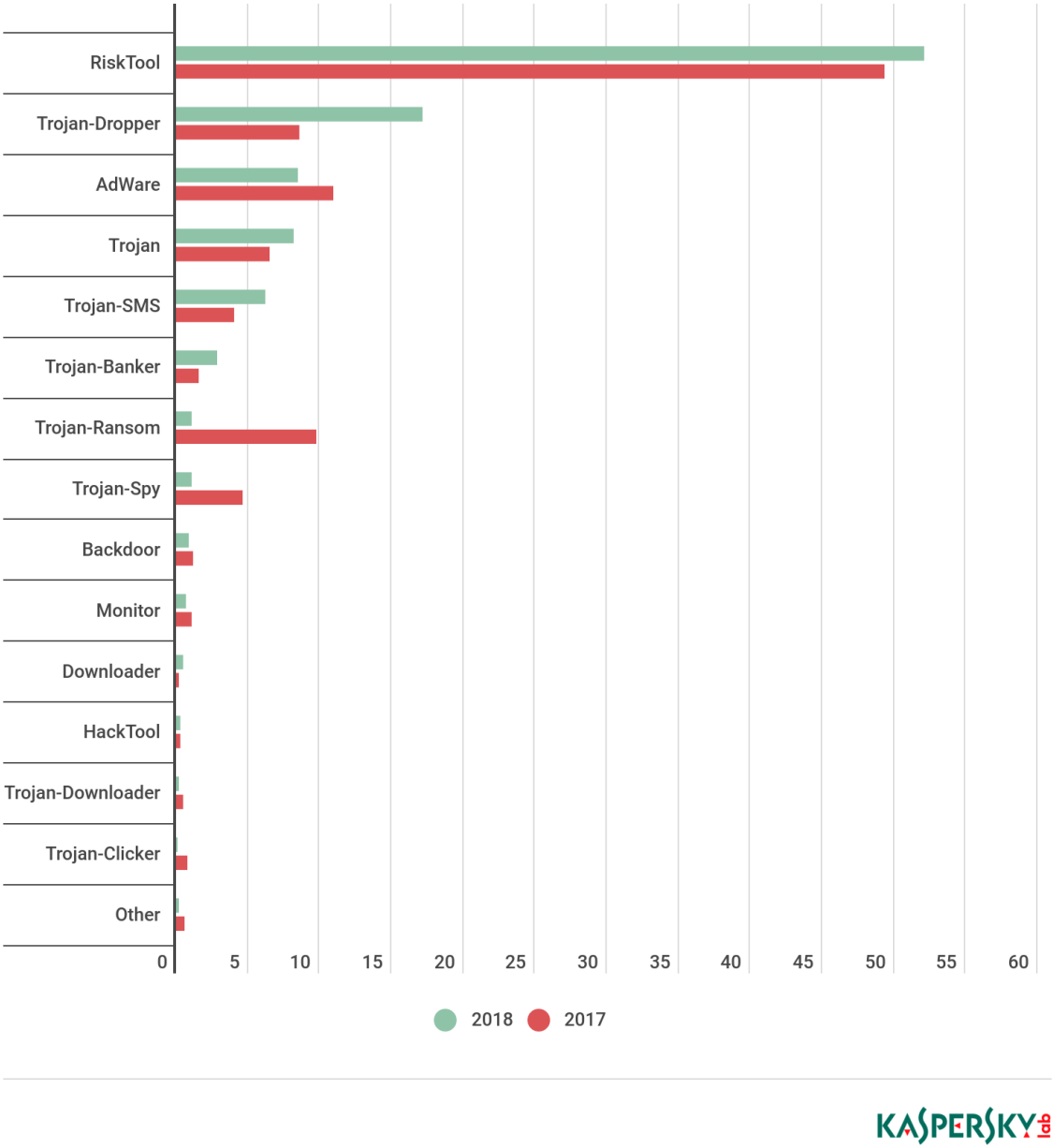
Figura 15. Geografía de los usuarios atacados, 2018



Fuente: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2019/03/04082504/en-geography-users.png

Según la casa de antivirus Kaspersky se ha realizado un análisis de los virus en cantidad de ataques para los años 2017 y 2018. A continuación, en la figura 16 hay una representación de las amenazas más comunes y su distribución en dispositivos móviles con Android

Figura 16. Distribución de nuevas amenazas móviles por tipo, 2017 y 2018



Fuente: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2019/03/04082658/en-mob-malwares-types.png

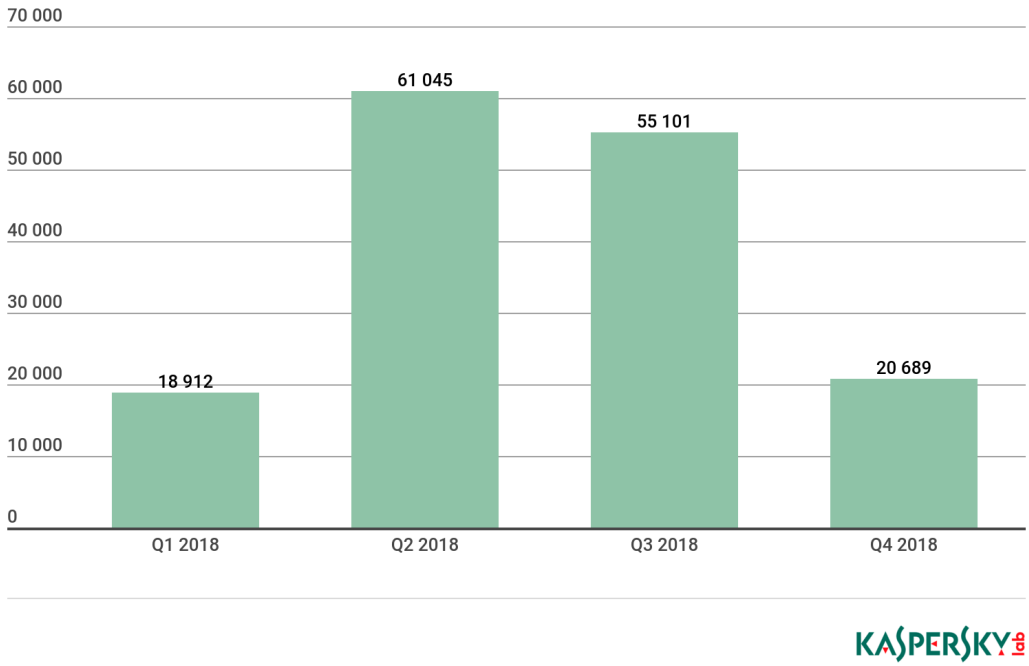
El cuadro 1 muestra la clasificación según el daño que puedan causar en los dispositivos Android analizados por las herramientas antivirus de la casa Kaspersky.

Cuadro 1. Top 20 de malware para móviles

numero	tipo	% *
1	DangerousObject.Multi.Generic	68.28
2	Trojan.AndroidOS.Boogr.gsh	10.67
3	Trojan-Banker.AndroidOS.Asacub.a	6.55
4	Trojan-Banker.AndroidOS.Asacub.snt	5.19
5	Trojan-Dropper.AndroidOS.Hqwar.ba	3.78
6	Trojan-Dropper.AndroidOS.Lezok.p	3.06
7	Trojan-Banker.AndroidOS.Asacub.ce	2.98
8	Trojan-Dropper.AndroidOS.Hqwar.gen	2,96
9	Trojan-Banker.AndroidOS.Asacub.ci	2,95
10	Trojan-Banker.AndroidOS.Svpeng.q	2,87
11	Trojan-Dropper.AndroidOS.Hqwar.bb	2,77
12	Trojan-Banker.AndroidOS.Asacub.cg	2,31
13	Trojan.AndroidOS.Triada.dl	1.99
14	Trojan-Dropper.AndroidOS.Hqwar.i	1.84
15	Trojan-Dropper.AndroidOS.Piom.kc	1.61
16	Exploit.AndroidOS.Lotoor.be	1,39
17	Trojan.AndroidOS.Agent.rx	1,32
18	Trojan-Banker.AndroidOS.Agent.dq	1.31
19	Trojan-Dropper.AndroidOS.Lezok.b	1.22
20	Trojan.AndroidOS.Dvmap.a	1.14
Fuente: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180734/Mobile_report_2016.pdf		

A continuación, en la figura 17 se observa una imagen tipo barras donde se muestra el crecimiento de virus que simulan banca móvil para el año 2018. Todo con el propósito de capturar información financiera.

Figura 17. Troyanos de banca móvil detectados por Kaspersky Lab, 2018



Fuente: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2019/03/04082751/banker-packages.png

La empresa NowSecure ha hecho una anatomía de un ataque móvil donde hay diferentes capas para atacar. Para que un ataque sea exitoso se necesita encontrar algún tipo de vulnerabilidad ya sea de tipo hardware o software o incluso que las personas entreguen sus datos personales engañados por medio de ingeniería social. La cadena de tecnología donde se ejecutan los diferentes ataques es para dispositivos móviles, redes de comunicación y recepción de datos. A continuación, en la figura 18 se esquematiza la forma más común de ataque en un dispositivo móvil.

Figura 18. La anatomía de un ataque móvil

Fuente: https://www.nowsecure.com/wp-content/uploads/2016/08/anatomy_of_a_mobile_attack.jpg

4.10 MARCO CONTEXTUAL

4.10.1 Análisis de impacto del malware. Ali Feizollah a, Nor Badrul Anuar a, Rosli Salleh a, Guillermo Suarez-Tangil, Steven Furnell (2016). “AndroDialysis: análisis de la intención de Android, Eficacia en la detección de malware" (PDF). computers & security 65 (2017) 121–134. Es un artículo que explica como los malware intentan dañar los dispositivos y cual arquitectura usan, intentos de infección y categorías[54].

4.10.2 Evolución del malware en 2016. MOBILE MALWARE EVOLUTION 2016, Kaspersky lab. (PDF), explica la frecuencia de ataques de este tipo de virus y los diferentes tipos que existen mostrando algunas Apps que contiene variantes de Malware[55].

[54] Ali Feizollah a, Nor Badrul Anuar a, Rosli Salleh a, Guillermo Suarez-Tangil, Steven Furnell (2016). " AndroDialysis: análisis de la intención de Android, Eficacia en la detección de malware, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: http://www0.cs.ucl.ac.uk/staff/G.SuarezdeTangil/papers/2017cosec-androdialysis.pdf

[55] MOBILE MALWARE EVOLUTION 2016, Kaspersky lab. (PDF). [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180734/Mobile_report_2016.pdf

4.10.3 Ataques y defensas sobre virus en los móviles. Dunham, Ken; Abu Nimeh, Saeed; Becher, Michael (2008). Ataque y defensa maliciosos móviles. Syngress Media. ISBN 978-1-59749-298-0. Muestra las nuevas tendencias de los ataques y como han pasado de la web a los dispositivos móviles. Deja ver la necesidad de implementar seguridad debido a la amplia gama de virus que se han encontrado en los últimos años[56].

4.11 MARCO LEGAL

4.11.1 En Colombia. Colombia cuenta con una legislación sobre los delitos informáticos debido al crecimiento y amenazas que representan para la infraestructura tecnológica en el país. La ley 1273 de 2009 tipifica los delitos informáticos y agrupa los posibles ataques en nueve tipos que trata de abarcar la protección de los sistemas de información y los patrimonios empresariales.

En Colombia se ha incrementado los delitos[57] por lo que la policía nacional y la Dijin han hecho esfuerzos con la creación de la dirección de investigación criminal donde se asocia todas las actividades ilegales que violen el principio de integridad, confiabilidad y disponibilidad de los sistemas de información. Sin embargo, apenas se crea la conciencia de denunciar estos casos pues son muchísimos más que quedan sin ser denunciados por cuidar la reputación empresarial y la buena imagen ante los clientes.

El caso de los delitos informáticos en Colombia ha sido incremental pues en años como el 2005 Colombia país ocupaba puestos[58] bajos respecto a los demás países latinoamericanos en ataques que se generaban, sin embargo, el panorama ha cambiado y ya se tiene el 3 lugar luego de Brasil y México. Entre los delitos informáticos que se presentan con más frecuencia están los abusos de los sistemas de información por parte de los empleados, el desconocimiento de la ley y la aplicación de políticas de seguridad que concienticen a las personas a ser éticos. También ayuda a que hay sistemas muy frágiles que no tienen presenten la implementación de seguridad siendo vulnerables ante los piratas informáticos sin mencionar la falta de auditoria que revela los informes de riesgos encontrados en

[56] Dunham, Ken; Abu Nimeh, Saeed; Becher, Michael (2008). Ataque y defensa maliciosos móviles. Syngress Media. ISBN 978-1-59749-298-0, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://books.google.com.co/books?id=Nd1RcGWMKnEC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[57] A diario se registran 542.465 ataques informáticos en Colombia, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://www.eltiempo.com/tecnosfera/novedades-tecnologia/informe-sobre-ataques-informaticos-en-colombia-y-al-sector-financiero-135370

[58] Colombia: el sexto país con más ataques cibernéticos en a. latina, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: http://www.enter.co/chips-bits/seguridad/colombia-sexto-pais-ataques-ciberneticos/

los activos informáticos. El panorama no es muy alentador, aunque es cierto que se han realizado grandes esfuerzos en este tema y ya hay más conciencia respecto a la seguridad de la información. Cuando hay conciencia del costo que puede generar un ataque éxitos las empresas entonces fortalecerán sus sistemas de información.

4.11.2 Delito informático. Un delito puede ser entendido desde la perspectiva social y dependiendo en enfoque y el área donde se dé, se evalúa como atacarlo y como evitarlo. El hombre siempre ha tenido comportamientos antisociales y conductas que lo llevan a la transgresión de las normas es por que nace la sanción social y en el caso de la información la ley Colombiana tiene establecido las penas por violación de la ley[59].

4.11.3 Tipos de ataques más comunes en Colombia.

En Colombia los delitos cibernéticos que más se presentan son los siguientes:

Phising: Es un delito muy frecuente que busca la suplantación de entidades o personas. El objetivo es llevar a los usuarios a sitios que no son auténticos con el fin de obtener información sensible y así ser estafados. Su intención es generar confianza entre sus víctimas y es usado desde hace más de 2 décadas. Los países han hecho mucho énfasis en este tipo de ataques concientizando a los usuarios de las formas en que opera.

Skimming: Es un delito muy común en nuestro país y aunque en otros países se presenta en balances financieros, en Colombia se utiliza en la clonación de tarjetas de crédito o débito, instalando dispositivos electrónicos en los cajeros para obtener la información del plástico y realizar el robo en este caso de tico económico. Siempre es recomendable no perder el pastico de vista para evitar la clonación.

Estafa: Las estafas por internet están a la orden del día, en Colombia se ha conocido de múltiples casos de ciudadanos que denuncian en la línea de la policía este tipo de delito. Casos como apps de citas donde hay links falsos que roban información financiera, casos como ventas por internet donde él envió del producto comprado no es el solicitado o está en mal estado y la más grave que ha venido creciendo es la suplantación de identidad donde sujetos con intenciones delictivas

[59] Autocontrol y Ciberdelincuencia. Club Ciencias Forenses. [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://www.clubforenses.com/autocontrol-y-ciberdelincuencia-club-ciencias-forenses/

hablan en nombre de gerentes y propietarios haciendo pedidos a proveedores en nombre de empresas legales.

Carta nigeriana: Este tipo de delito ha cobrado mucha importancia en las investigaciones de los expertos de seguridad. Básicamente su forma de actuación es la manipulación psicológica donde los delincuentes se especializan en idear formas y maneras para obtener información reservada, se envía cartas con supuestos testamentos o números ganadores de lotería todo con el fin de generar en la victima la sorpresa de ganar.

Malware: Es un archivo tipo malicioso que viene camuflado en archivos que parecen inofensivos pero que tiene la capacidad de infectar y robar información de nuestros dispositivos electrónicos. Estos se ven mucho en archivos con terminación en .pdf o elementos que se instalen en los exploradores del cliente como lo es la tecnología de Adobe llamada Flash.

Smishing: SMiShing es un ataque de seguridad generalmente enviado por mensaje de texto con el fin de engañarlo para la descarga de un troyano, virus u otro malware. Su abreviatura viene de "SMS phishing". Hay aplicaciones que comparte informacion de los usuarios como es trucaller y así los mensajes o llamadas fraudulentas tienen una etiqueta delictiva.

Colombia cuenta con una [60]legislación sobre los delitos informáticos debido al crecimiento y amenazas que representan para la infraestructura tecnológica en el país. La ley 1273 de 2009 tipifica los delitos informáticos y agrupa los posibles ataques en nueve tipos que trata de abarcar la protección de los sistemas de información y los patrimonios empresariales.

Entre los delitos informáticos que se presentan con más frecuencia están los abusos de los sistemas de información por parte de los empleados, el desconocimiento de la ley y la aplicación de políticas de seguridad que concienticen a las personas a ser éticos. También ayuda a que hay sistemas muy frágiles que no tienen presenten la implementación de seguridad siendo vulnerables ante los piratas informáticos sin mencionar la falta de auditoria que revela los informes de riesgos encontrados en los activos informáticos. El panorama no es muy alentador, aunque es cierto que se han realizado grandes esfuerzos en este tema y ya hay más conciencia respecto a

[60] Microsoft Word - Ley 1273 de 2009 delitos informáticos DO 47223.doc. [en línea] [citado el 7 de agosto, 2019]. Disponible en internet: http://www.sic.gov.co/recursos_user/documentos/normatividad/Ley_1273_2009.pdf

la seguridad de la información. Cuando hay conciencia del costo que puede generar un ataque éxitos las empresas entonces fortalecerán sus sistemas de información.

4.11.4 En Android. Google ha aplicado muchas actualizaciones de seguridad en los últimos años, incluso en octubre del año 2018 actualizo sus políticas donde solo permiten que una aplicación tenga acceso a los registros de llamadas y SMS, esto porque los Malware se han convertido en una plaga y se está haciendo los correctivos para blindar mejor la seguridad de la Play Store. Google tiene muy buenos manuales de buenas prácticas para seguridad de la información.

El principio de la Play Store es la confianza entre el proveedor y el desarrollador[61] que quiere brindar a los usuarios una aplicación que le sirva para un fin común. Estos son algunos principios para tener en cuenta y no violar los acuerdos de distribución de desarrolladores. Contenido restringido, el cual incluye contenido inadecuado que violen las leyes de cualquier nación, suplantación de identidad y violación de propiedad intelectual lo cual hace referencia al respecto por los derechos de autor y autenticidad de identidad, privacidad, engaño y poca seguridad.

Una de las principales puertas de malware que sufre la Play Store, es cuando hay configuración de permisos que no tienen que ver con la aplicación o código que intenta violar la privacidad del usuario entran en la lista de restringidos, pero los hackers se las ingenian para pasar su Malware como troyanos sin ser detectados. Cumplimiento de acuerdos para publicidad, cancelaciones o devoluciones de dinero Aplicaciones que usen telemetría para obtener estadísticas de los usuarios también están prohibidas. Aplicaciones que se comporten como Spam, publicidad de instalación instantánea, especificar el tipo de público al que puede o debe mostrarse el App, aceptar las políticas de Android para Play store, aceptar las actualizaciones dispuesta por el sistema operativo.

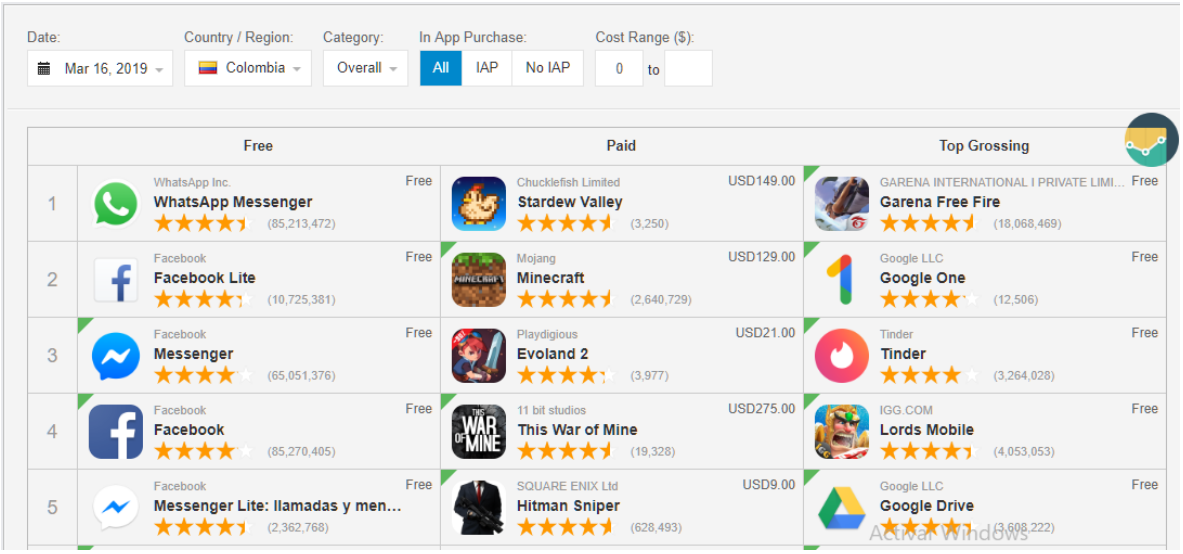
[61] Centro de políticas de desarrolladores. [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://play.google.com/about/developer-content-policy/?hl=ES#!?modal_active=none

5 DESARROLLO DEL ESTUDIO Y DIAGNOSTICO

5.1 APLICACIONES ANALIZADAS

El desarrollo de Apps sigue en crecimiento, en el primer trimestre del año 2018 se estima la descarga de 25.000 millones de aplicaciones aproximadamente a nivel mundial donde hubo un crecimiento respecto al año anterior de un 7.6% según la empresa consultora Sensor Tower[62]. A continuación, en la figura 19 el ranking de las Apps más descargadas en Colombia para marzo del 2019.

Figura 19. Ranking Apps Play Store, para compras IAP y no IAP



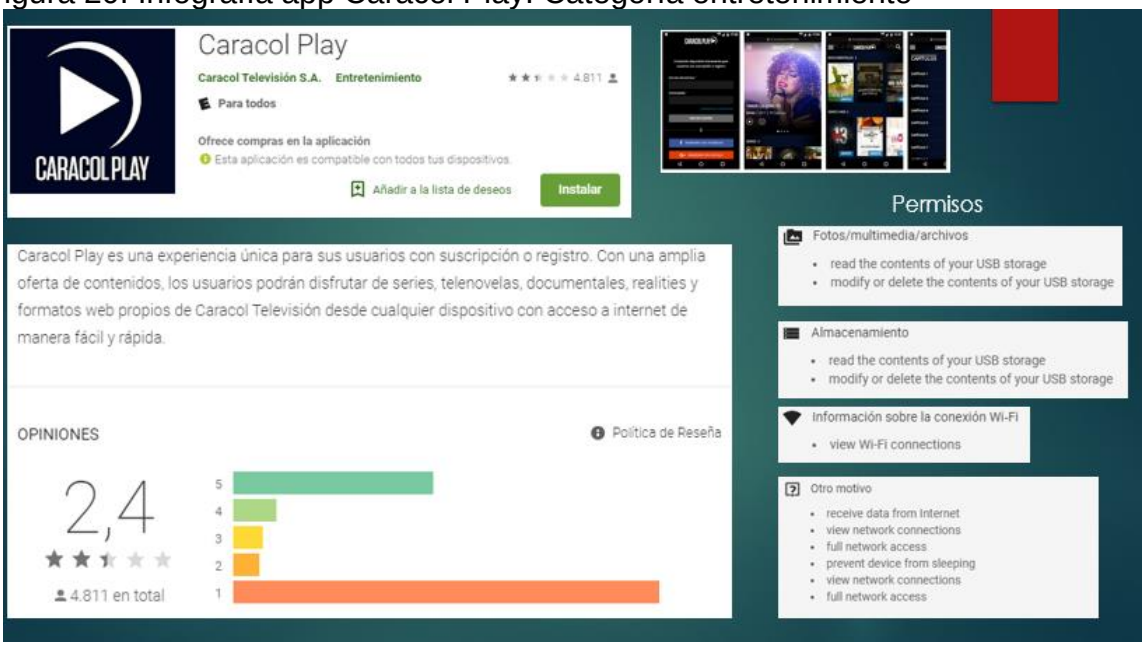
Fuente: https://sensortower.com/colombi/rankings/top/mobile/colombia/overall?date=2019-03-16

Investigando las categorías se escogió al azar un total de 5 Apps free para el análisis de riesgos M2 y M3 de la metodología OWASP Mobile.

[62] Las 'Apps' más descargadas en Android e iOS durante el 2018, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://www.eltiempo.com/tecnosfera/novedades-tecnologia/las-aplicaciones-mas-descargadas-en-el-primer-trimestre-de-2018-213556

Caracol Play: Es una aplicación de contenido de entretenimiento del país de Colombia donde se puede observar información del formato del canal Caracol. Su funcionalidad es presentar contenido e información de interés del canal para los usuarios. A continuación, en la figura 20 la infografía de la aplicación, donde se una breve explicación de su función, calificaciones de los usuarios, capturas de pantalla en miniatura y los permisos que solicita al ser instalada. Información muy valiosa para el análisis de seguridad.

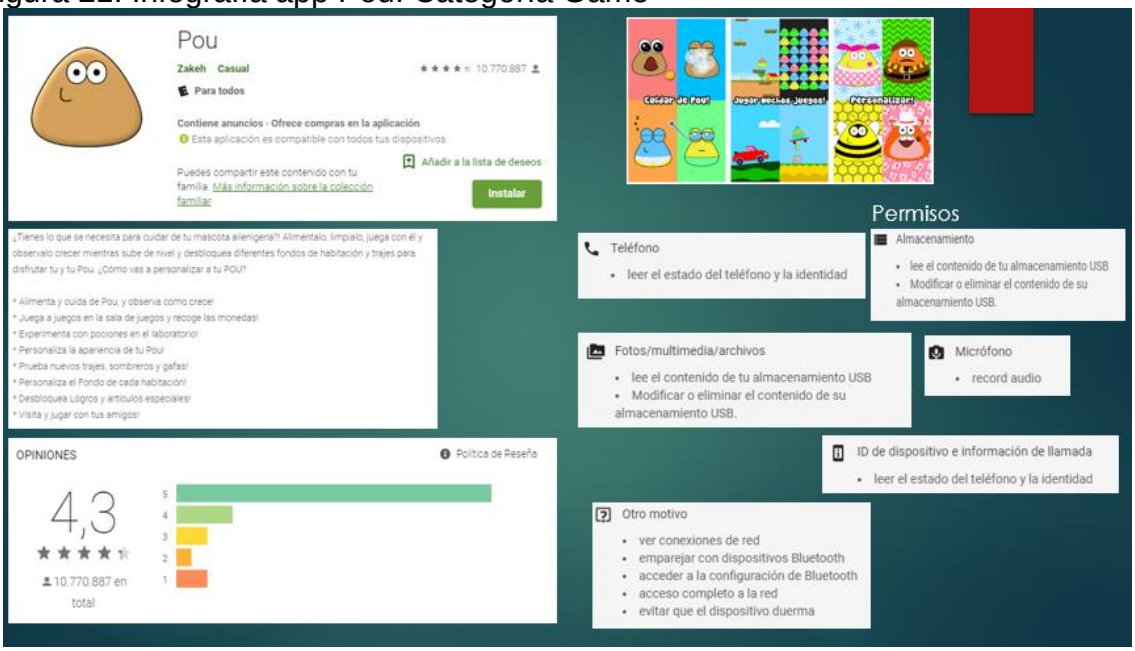
Figura 20. Infografía app Caracol Play. Categoría entretenimiento



Fuente este estudio

Pou: Esta es una aplicación de categoría Game presentando un personaje al cual hay que alimentarlo, jugar con él y ver cómo va creciendo de acuerdo con los cuidados que se le da por parte del usuario. Su funcionalidad principal es divertir a los usuarios simulando el cuidado de un personaje. A continuación, en la figura 21 infografía de la aplicación, donde se una breve explicación de su función, calificaciones de los usuarios, capturas de pantalla en miniatura y los permisos que solicita al ser instalada. Información muy valiosa para el análisis de seguridad.

Figura 21. Infografía app Pou. Categoría Game

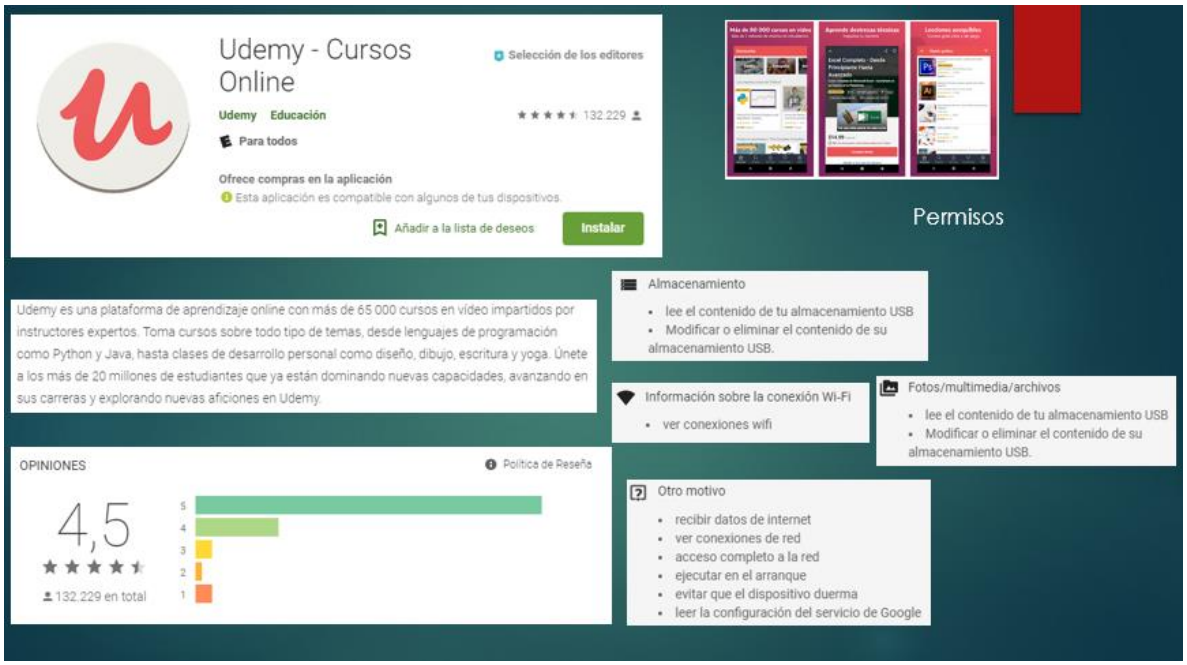


Fuente este estudio

Udemy: Es una aplicación de aprendizaje que presenta miles de cursos de forma interactiva contando con instructores calificados según lo que promete la plataforma. Tiene un crecimiento muy alto de usabilidad a nivel mundial y cada vez que liberan nuevas versiones se notan mejoras de diseño y seguridad ya que cuenta con un sistema de pago por medio de los servicios de Google.

Está catalogada como una aplicación de educación. A continuación, en la figura 22 la infografía de la aplicación, donde se una breve explicación de su función, calificaciones de los usuarios, capturas de pantalla en miniatura y los permisos que solicita al ser instalada. Información muy valiosa para el análisis de seguridad.

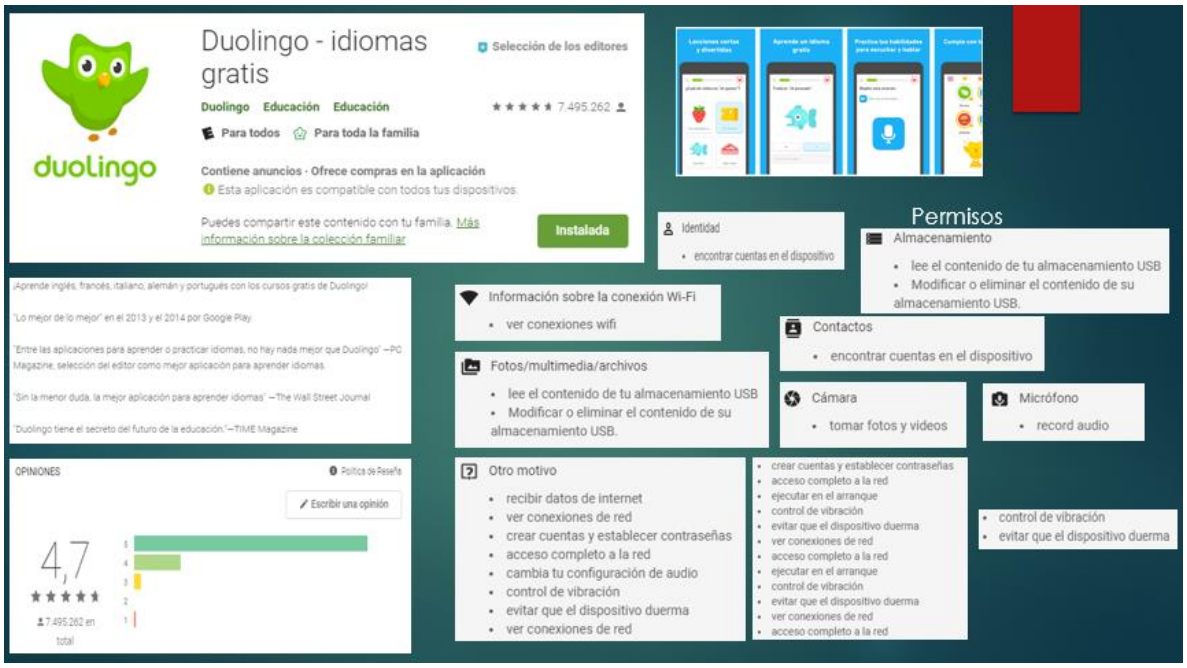
Figura 22. Infografía app Udemy. Categoría Educación



Fuente este estudio

Duolingo: Esta se especializa en brindar una interfaz interactiva para el aprendizaje de varios idiomas en los que se incluyen el inglés, francés, italiano, alemán y portugués. Esta aplicación tiene una valoración muy positiva por parte de los usuarios y es de tipo educativo. A continuación, en la figura 30 la infografía de la aplicación, donde se una breve explicación de su función, calificaciones de los usuarios, capturas de pantalla en miniatura y los permisos que solicita al ser instalada. Información muy valiosa para el análisis de seguridad.

Figura 23. Infografía app Duolingo. Categoría Educación

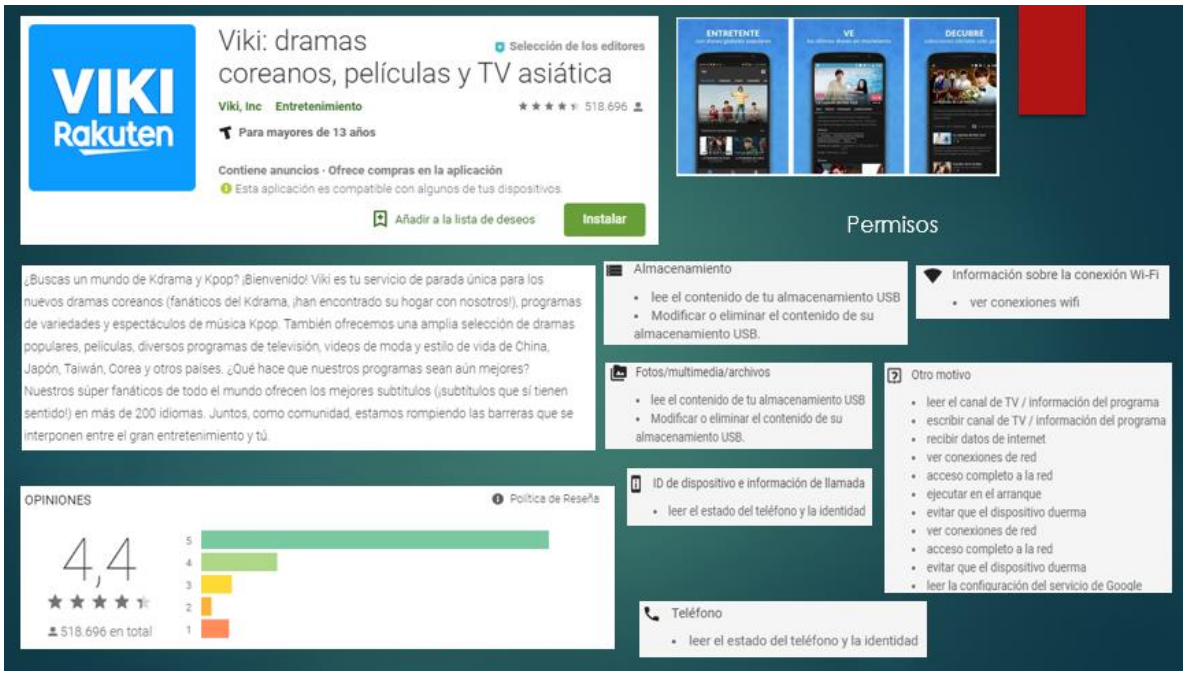


Fuente este estudio

Viki: Esta aplicación es muy conocida por especializarse en dramas del país de Corea del sur. Ofrece programas variados de espectáculos, películas e información de la vida cotidiana de países como China, Japón o Taiwán. Está enfocada en un público que se divierte con la forma de vida de los orientales. Esta aplicación es de tipo entretenimiento y promete no alojar contenido de páginas legales y así evitar descargas de sitios piratas fortaleciendo la seguridad de la aplicación.

A continuación, en la figura 24 la infografía de la aplicación, donde se una breve explicación de su función, calificaciones de los usuarios, capturas de pantalla en miniatura y los permisos que solicita al ser instalada. Información muy valiosa para el análisis de seguridad.

Figura 24. Infografía app Viki. Categoría Entretenimiento

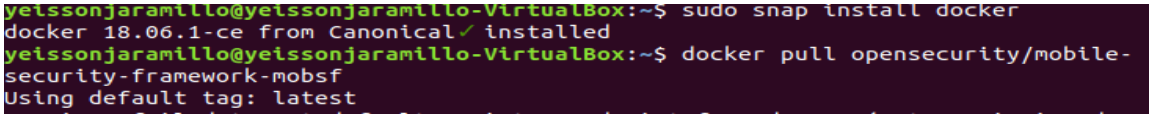


Fuente este estudio

5.2 CONFIGURACION DEL AMBIENTE DE PRUEBAS

En el proceso de analizar las Apps desde sus archivos APK, existen muchas herramientas que ayudan a realizar el diagnostico de seguridad. Para el caso de análisis estático se utiliza la versión estable de Mobile-Security-Framework-MobSF en ambiente Linux. El proceso de instalación de la herramienta se encuentra a continuación. En la figura 25 se ejecutan los comandos de instalación del contenedor Docker que nos servirá como repositorio de instalación y en la figura 26 la instalación del paquete preconfigurado en Docker que contiene la información preinstalada de framework mobsf

Figura 25. Instalación del ambiente Docker



Fuente este estudio

Figura 26. Instalación del framework Mobile Security Framework MobSF

```
yeissonjaramillo@yeissonjaramillo-VirtualBox:~$ sudo docker pull opensecurity/mo
bile-security-framework-mobsf
```

Fuente este estudio

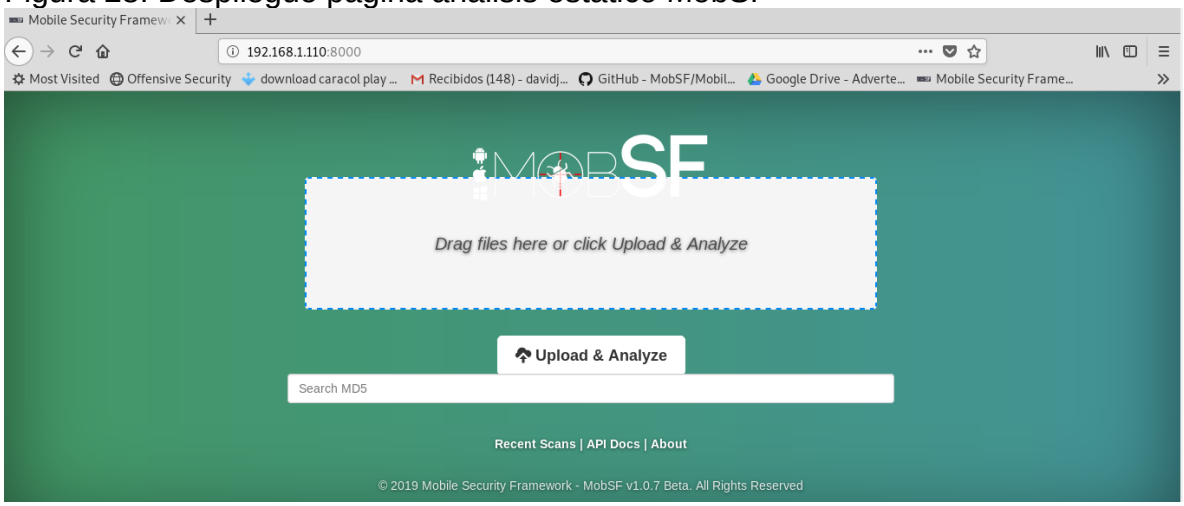
A continuación, en la figura 27 se ejecuta el comando de inicialización de Docker y su ambiente para MobSF. En la figura 28 se ejecuta la ip que sale por el puerto preconfigurado en el servidor Apache para este caso el 8000, aunque este puede variar según la configuración de cada equipo.

Figura 27. Inicio y ejecución de la herramienta de análisis móvil

```
yeissonjaramillo@yeissonjaramillo-VirtualBox:~$ sudo docker run -it -p 8000:8000
 opensecurity/mobile-security-framework-mobsf:latest
[2019-03-25 21:19:19 +0000] [1] [INFO] Starting gunicorn 19.9.0
[2019-03-25 21:19:19 +0000] [1] [INFO] Listening at: http://0.0.0.0:8000 (1)
[2019-03-25 21:19:19 +0000] [1] [INFO] Using worker: sync
[2019-03-25 21:19:19 +0000] [11] [INFO] Booting worker with pid: 11
[INFO] 25/Mar/2019 21:20:30 -

  __  __       _     ____  _____           _  __
 |  \/  | ___ | |__ / ___||  ___| __ _ _ _ \
 | |\/| |/ _ \| '_ \\___ \| |_   \ \ / / || | | |
 | |  | | (_) | |_) |__) |  _|    \ V /| || |_| |
 |_|  |_|\___/|_.__/____/|_|       \_/ |_(_)___/



[INFO] 25/Mar/2019 21:20:30 - Mobile Security Framework v1.0.7 Beta
REST API Key: 2a824d5c42d700654d35edff6b41eae67d74a8a774157a3466d5f596f54181f5
[INFO] 25/Mar/2019 21:20:30 - OS: Linux
[INFO] 25/Mar/2019 21:20:30 - Platform: Linux-4.18.0-16-generic-x86_64-with-Ubun
tu-18.04-bionic
[INFO] 25/Mar/2019 21:20:30 - Dist: ('Ubuntu', '18.04', 'bionic')
[INFO] 25/Mar/2019 21:20:30 - Finding JDK Location in Linux/MAC....
```

Fuente este estudio

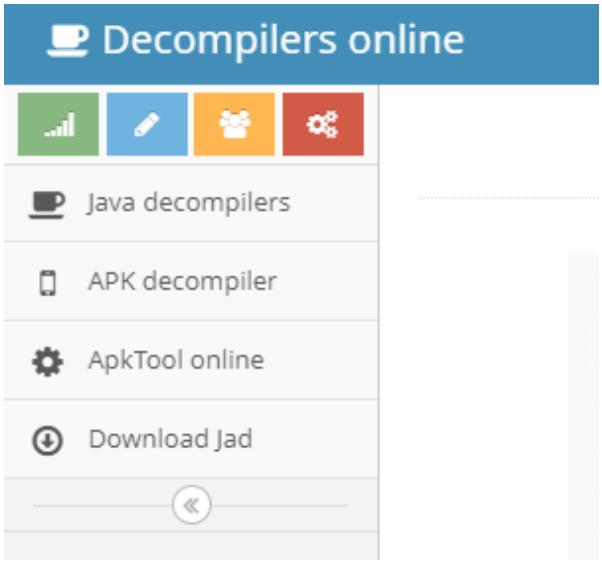
Figura 28. Despliegue página análisis estático MobSF



Fuente este estudio

Para el tema de revisión del código fuente y análisis de problemas de codificación se utilizó la página online http://www.javadecompilers.com/, a continuación, la figura 29 muestra su interfaz con las diferentes opciones de descompilador de Apks.

Figura 29. Descompilador web de apk



Fuente este estudio

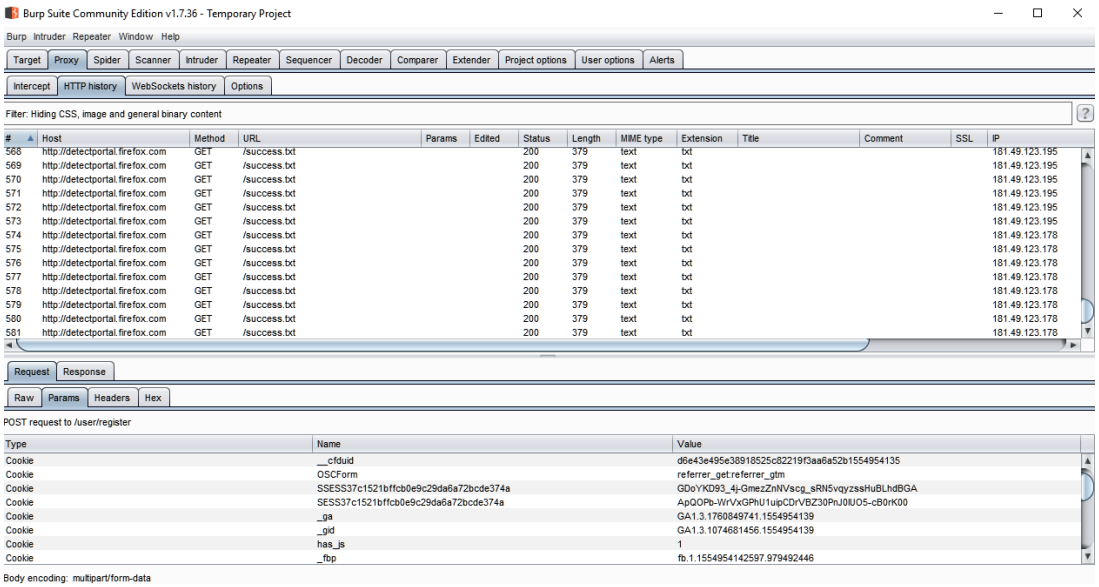
Para el análisis de tráfico y riesgo M3 se utilizará. MobSF Android 4.4.2 x86 VirtualBox VM el cual se puede implementar según la página oficial https://github.com/MobSF/Mobile-Security-Framework-MobSF/wiki/11.-Configuring-Dynamic-Analyzer-with-MobSF-Android-4.4.2-x86-VirtualBox-VM, a continuación en la figura 30 se observa la interfaz de la máquina virtual móvil para MobSF inicializada en VirtualBox siguiendo los pasos de instalación y puesta en marcha.

Figura 30. MobSF Android 4.4.2 x86 VirtualBox VM

Fuente este estudio

Para la interceptación de tráfico y paquetes desde las aplicaciones se utilizará la herramienta Burp Suite que se puede descargar desde la página: https://portswigger.net/burp/communitydownload. A continuación, en la figura 31 se muestra la interfaz de inicialización de ejecución luego de su correcta configuración.

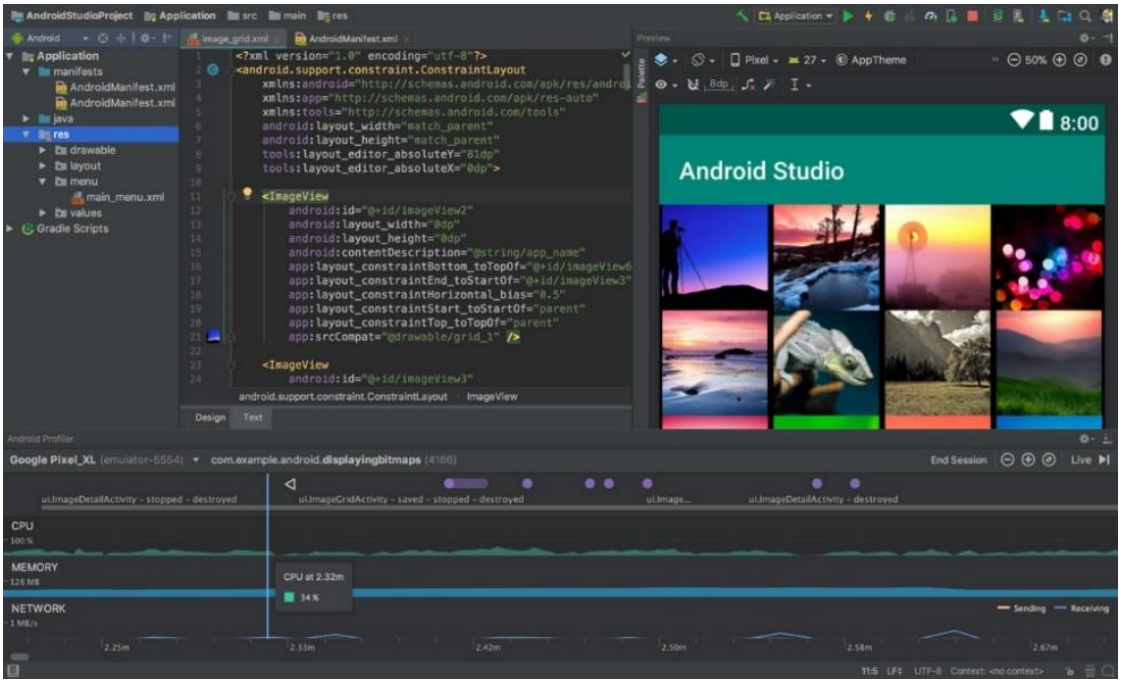
Figura 31. Burp Suite



Fuente este estudio

Para el análisis de las Apps se tendrá en cuenta el api de desarrollo de Android 3.3.2 que permite ver el funcionamiento y desarrollo de las apps que serán objeto de este estudio. A continuación, en la figura 32 la interfaz de la aplicación ejecutada

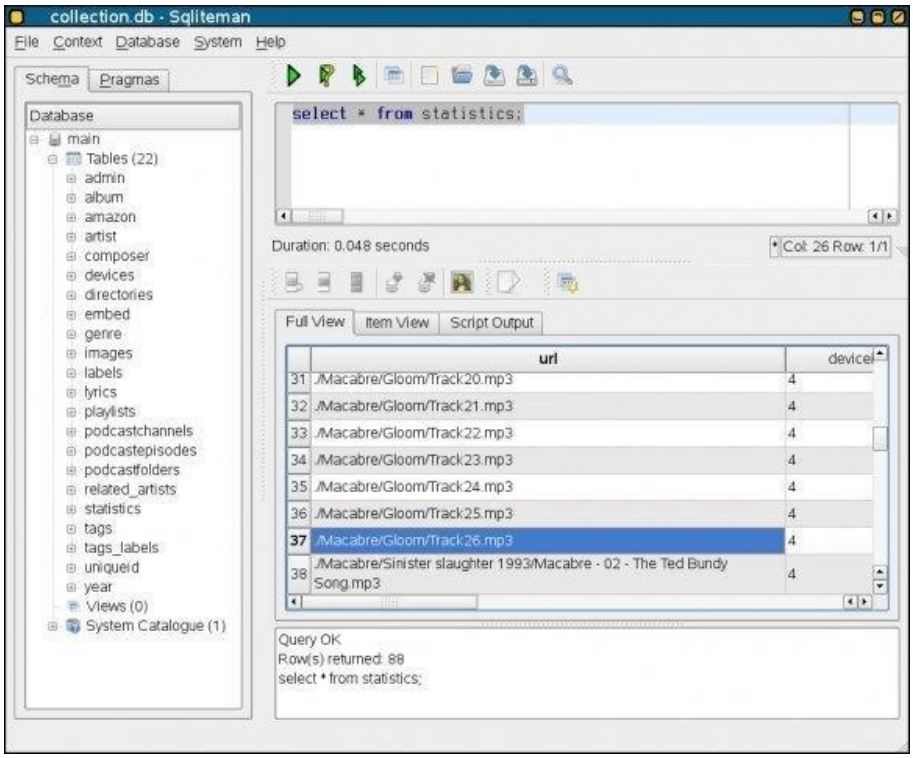
Figura 32. Api Android estudio



Fuente: https://developer.android.com/studio/images/studio-homepage-hero.jpg

Como herramienta de lectura de bases de datos usadas por Android, se utilizará SQLite. A continuación, en la figura 33 una interfaz sencilla cuando ejecutamos la herramienta.

Figura 33. SQLiteMan. Herramienta para visualizar bases de datos de SQLite



Fuente: https://a.fsdn.com/con/app/proj/sqliteman/screenshots/180790.jpg/max/max/1

5.3 PLAN DE PRUEBAS

El objetivo principal es hallar vulnerabilidades desde la fuente de la aplicación. Para esto se usarán las herramientas que arrojen información configurativa de cada aplicación móvil.

Se tendrá en cuenta los siguientes ítems de revisión:

Análisis del archivo AndroidManifest.xml

5.3.1 Riesgo estático M2 insecure data storage. Los siguientes son los riesgos que serán analizados. Base de datos SQLITE, tipo texto sin cifrar, datos codificados, exposición de datos potencialmente sensibles, almacenamiento de datos externos, copia de seguridad habilitada de la aplicación, exposición a la información.

5.3.2 Riesgo estático M3 insecure communication. Solo se analiza el siguiente riesgo. Uso de protocolo http sin cifrar

5.3.3 Riesgo dinámico M2 insecure data storage. Como análisis dinámico solo se tendrá en cuenta las bases de datos que la aplicación maneje en el dispositivo y su revisión de datos sensibles.

5.3.4 Riesgo dinámico M3 insecure communication Como análisis dinámico solo se tendrá en cuenta las peticiones Get y Post y sus datos no cifrados.

5.4 ANALISIS ESTATICO Y DINAMICO PARA RIESGOS MOBILES M2 Y M3

Los resultados obtenidos se explicarán detalladamente en el Anexo donde se describirán los procesos y ejecución de pruebas.

5.5 ANALISIS DE RESULTADOS

Los resultados aquí descriptos son producto del análisis hecho con herramientas de código estático donde se explican los hallazgos encontrados. Aquellos resultados del análisis dinámico serán expuestos en el anexo A. En este se explica detalladamente el laboratorio seguro que se implementó para sacar las conclusiones de seguridad según los riesgos de almacenamiento y comunicación insegura. … (Véase anexo A)

A continuación, se puede observar en el cuadro 2 los permisos solicitados por las aplicaciones y el nivel de vulnerabilidad con el que es calificado según la herramienta de análisis estático OWASP.

Rojo: Peligroso, Azul: Normal, Verde: Firma

Cuadro 2. Permisos solicitados por las Apps en el archivo manifiest.xml

PERMISOS	Caracol Play	pou	Udemy	duolingo	Viki	Rojo: Peligroso, Azul: Normal, Verde: Firma
android.permission.INTERNET	x	x	x	x	x	
com.android.vending.BILLING	x	x		x	x	
android.permission.ACCESS_NETWORK_STATE	x	x		x	x	
android.permission.ACCESS_WIFI_STATE	x	x	x	x	x	
android.permission.WAKE_LOCK	x	x	x	x	x	
com.google.android.c2dm.permission.RECEIVE	x		x		x	
net.icck.CaracolPlay.permission.C2D_MESSAGE	x		x		x	
android.permission.VIBRATE				x		
android.permission.WRITE_EXTERNAL_STORAGE		x	x	x	x	
android.permission.READ_PHONE_STATE		x			x	
com.udemy.android.gcm.permission.C2D_MESSAGE			x	x		
android.permission.RECEIVE_BOOT_COMPLETED			x			
com.udemy.android.permission.PUSHIO_MESSAGE			x			
android.permission.READ_EXTERNAL_STORAGE		x	x	x	x	
android.permission.INTERACT_ACROSS_USERS			x			
android.permission.RECORD_AUDIO		x		x		
android.permission.BLUETOOTH		x				
android.permission.BLUETOOTH_ADMIN		x				
android.permission.SYSTEM_ALERT_WINDOW		x				
android.permission.ACCESS_COARSE_LOCATION		x				
com.google.android.gms.permission.ACTIVITY_RECOGNITION		x				
android.permission.ACCESS_FINE_LOCATION		x				
com.google.android.providers.gsf.permission.READ_GSERVICES		x				
android.permission.GET_TASKS		x				
android.permission.FOREGROUND_SERVICE				x		

Fuente: este estudio.

Cuadro 3. (Continuación) Permisos solicitados por las Apps en el archivo manifiest.xml

android.permission.GET_ACCOUNTS				x		
android.permission.AUTHENTICATE_ACCOUNTS				x		
com.google.android.c2dm.permission.RECEIVE				x		
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE				x		
android.permission.MODIFY_AUDIO_SETTINGS				x		
com.sec.android.provider.badge.permission.READ				x		
com.sec.android.provider.badge.permission.WRITE				x		
com.htc.launcher.permission.READ_SETTINGS				x		
com.htc.launcher.permission.UPDATE_SHORTCUT				x		
com.sonyericsson.home.permission.BROADCAST_BADGE				x		
com.sonymobile.home.permission.PROVIDER_INSERT_BADGE				x		
com.anddoes.launcher.permission.UPDATE_COUNT				x		
com.majeur.launcher.permission.UPDATE_BADGE				x		
com.huawei.android.launcher.permission.CHANGE_BADGE				x		
com.huawei.android.launcher.permission.READ_SETTINGS				x		
com.huawei.android.launcher.permission.WRITE_SETTINGS				x		
android.permission.READ_APP_BADGE				x		
com.oppo.launcher.permission.READ_SETTINGS				x		
com.oppo.launcher.permission.READ_SETTINGS				x		
com.oppo.launcher.permission.WRITE_SETTINGS				x		
me.everything.badger.permission.BADGE_COUNT_READ				x		
me.everything.badger.permission.BADGE_COUNT_WRITE				x		

Fuente: este estudio.

5.5.1 Explicación de los permisos según auditoria MobSF

Permisos:

android.permission.VIBRATE: Se considera de tipo normal, pues solo tiene acceso al control del vibrador.

android.permission.INTERNET: Se considera de tipo peligroso debido a que la aplicación si es maliciosa puede crear puertas traseras en la red.

android.permission.ACCESS_NETWORK_STATE: se considera de tipo normal pues se utiliza para ver el estado de las redes.

android.permission.ACCESS_WIFI_STATE: Se considera de tipo normal pues se utiliza para conocer el estado de la red Wi-Fi

android.permission.WAKE_LOCK: Se considera de tipo peligroso pues evita que el teléfono entre en estado de reposo lo que determina un gasto mayor de energía.

com.google.android.providers.gsf.permission.READ_GSERVICES: se considera de tipo peligroso pues permite que servicios desconocidos analicen la referencia del dispositivo usado.

com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE: Se considera peligroso pues permite que servicios desconocidos tomen información del teléfono.

com.google.android.c2dm.permission.RECEIVE: Se considera peligroso pues permite que servicios desconocidos analicen la referencia del dispositivo usado.

com.android.vending.BILLING: Se considera peligroso pues permite que servicios desconocidos analicen la referencia del dispositivo usado.

net.icck.CaracolPlay.permission.C2D_MESSAGE: Se considera firma propia de la aplicación y se usa para el envío de mensajes y notificaciones automáticas.

android.permission.WRITE_EXTERNAL_STORAGE: Se considera peligroso ya que puede acceder a la tarjeta externa y realizar cambios la misma.

android.permission.READ_PHONE_STATE: Se considera peligrosa ya que puede acceder a las funciones del teléfono y revisar llamadas incluso a quien se está llamando.

android.permission.DISABLE_KEYGUARD: Se considera peligroso ya que puede deshabilitar el bloqueo de teclas y visualizar las contraseñas escritas por el mismo.

com.udemy.android.gcm.permission.C2D_MESSAGE: Se considera tipo firma ya que es utilizado para recibir notificaciones y mensajería de la misma aplicación.

android.permission.RECEIVE_BOOT_COMPLETED: se considera normal ya que permite que una aplicación se inicie por si misma de forma automática. Aunque también pueda implicar que el teléfono inicie más lento.

com.udemy.android.permission.PUSHIO_MESSAGE: se considera peligroso pues puede dar permisos en el dispositivo de forma oculta sobre algún recurso.

android.permission.READ_EXTERNAL_STORAGE: Se considera peligroso pues puede acceder a la tarjeta SD y realizar lecturas de lo que se encuentra allí.

android.permission.INTERACT_ACROSS_USERS: Se considera peligroso debido a los permisos desconocidos que puede brindar en la referencia del sistema operativo usado.

android.permission.RECORD_AUDIO: Se considera peligroso debido a que se puede acceder a las grabaciones del dispositivo

android.permission.BLUETOOTH: Se considera peligroso debido a que una aplicación puede ver la configuración del Bluetooth local, además de poderse emparejar con otros dispositivos.

android.permission.BLUETOOTH_ADMIN: Se considera peligroso debido a que una aplicación puede administrar la configuración del Bluetooth local, además de poderse emparejar con otros dispositivos.

android.permission.SYSTEM_ALERT_WINDOW: Se considera peligroso debido a que la aplicación puede administrar ventanas de alerta del sistema permitiendo que las que puedan ocupar toda la pantalla.

android.permission.ACCESS_COARSE_LOCATION: Se considera peligroso debido a que usan base de datos, para determinar una ubicación aproximada del teléfono disponible.

com.google.android.gms.permission.ACTIVITY_RECOGNITION: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

android.permission.ACCESS_FINE_LOCATION: Se considera peligroso debido a que la aplicación puede acceder a las fuentes de ubicación exacta como el posicionamiento global.

com.google.android.providers.gsf.permission.READ_GSERVICES: Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

android.permission.GET_TASKS: Se considera peligroso debido a que la aplicación puede recuperar información de las tareas ejecutadas lo cual permite que se supervise las tareas de otras aplicaciones.

android.permission.FOREGROUND_SERVICE : Se considera normal pues es un servicio usado para la ejecucion de la aplicación.

android.permission.GET_ACCOUNTS: Se considera normal debido a que permite que una aplicación acceda a la lista de cuentas conocidas por el teléfono.

android.permission.AUTHENTICATE_ACCOUNTS: Se considera peligroso debido a que la aplicación puede usar el autenticador de cuenta y las capacidades del administrador para creación.

com.google.android.c2dm.permission.RECEIVE : Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

android.permission.MODIFY_AUDIO_SETTINGS: Se considera peligroso debido a que la aplicación puede modificar la configuración de audio global, como volumen y enrutamiento.

com.sec.android.provider.badge.permission.READ: Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

com.sec.android.provider.badge.permission.WRITE : Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

com.htc.launcher.permission.READ_SETTINGS: Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

com.htc.launcher.permission.UPDATE_SHORTCUT: Se considera peligroso debido a que es un permiso desconocido de referencia de Android.

com.sonyericsson.home.permission.BROADCAST_BADGE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.sonymobile.home.permission.PROVIDER_INSERT_BADGE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.anddoes.launcher.permission.UPDATE_COUNT : Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.majeur.launcher.permission.UPDATE_BADGE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.huawei.android.launcher.permission.CHANGE_BADGE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.huawei.android.launcher.permission.READ_SETTINGS: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.huawei.android.launcher.permission.WRITE_SETTINGS: Se considera peligroso porque una aplicación puede modificar los datos de configuración del sistema esto puede significar corromper la configuración de su sistema.

android.permission.READ_APP_BADGE: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.oppo.launcher.permission.READ_SETTINGS: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

com.oppo.launcher.permission.WRITE_SETTINGS: Se considera peligroso porque una aplicación puede modificar los datos de configuración del sistema esto puede significar corromper la configuración de su sistema.

me.everything.badger.permission.BADGE_COUNT_READ: Se considera peligroso debido a que es un permiso desconocido de referencia de Android

me.everything.badger.permission.BADGE_COUNT_WRITE Se considera peligroso debido a que es un permiso desconocido de referencia de Android

5.5.2 Auditoria M2, almacenamiento de datos inapropiado

A continuación, se puede observar en la tabla 2 el análisis y calificación de los procedimientos detectados en las aplicaciones que pueden ser utilizados para obtener información que puede servir para futuros ataques. Esta auditoria está calificada de la siguiente manera.

Naranja: Medio, Amarillo: bajo

Tabla 2. Análisis M2 detectado

Análisis	Caracol Play	Pou	Udemy	Duolingo	Viki	
Base de datos SQLITE tipo texto sin cifrar	X		X	X	X	MEDIO
Datos codificados	X	X	X	X	X	BAJO
Exposición de datos potencialmente sensibles	X				X	BAJO
Almacenamiento de datos externos		X	X		X	MEDIO
Copia de seguridad habilitada de la aplicación			X		X	BAJO
Exposición a la información			X	X	X	BAJO

Fuente: este estudio

5.5.3 Explicación de cada uno de los factores de la tabla M2

Base de datos SQLITE tipo texto sin cifrar: La aplicación móvil utiliza una base de datos SQLite sin cifrar. Esto abre el acceso a un atacante que tenga intenciones de leer información de la aplicación y si el dispositivo este ruteado es peor ya que hay acceso físico. No es recomendable almacenar información en texto legible.

Datos codificados: La aplicación expone información técnica y depuraciones que sirven como base si alguien quiere realizar un ataque.

Exposición de datos potencialmente sensibles: La aplicación tiene proceso que exponen información sensible durante la ejecución.

Almacenamiento de datos externos: Cuando una aplicación puede acceder al almacenamiento externo como una tarjeta SD, cualquier aplicación maliciosa puede leer los datos de almacenamiento y puede llevar al mal funcionamiento de esta si el atacante corrompe el archivo por medio de manipulación de datos.

Copia de seguridad habilitada de la aplicación*:* Aunque es una opción que viene por defecto pues es un mecanismo de seguridad de Android para aplicaciones, esta puede almacenar información sensible y si se conoce los datos de acceso de la cuenta de Gmail por ejemplo se podría conocer la información almacenada.

Exposición a la información*:* la aplicación expone información de tipo url para acceder a host, servidores de producción. Esto sirve de recolección para un posible ataque.

A continuación, se puede observar en la tabla 3 el análisis y calificación del uso de comunicaciones con protocolo http lo que significa lectura de información cuando la comunicación es escuchada por un tercero. Esta auditoria está calificada de la siguiente manera.

Naranja: Medio.

5.5.4 Auditoria M3, comunicación insegura

Tabla 3. Análisis M3 detectado protocolo http

	Caracol Play	Pou	Udemy	Duolingo	Viki	
uso de protocolo http sin cifrar	X	X	X	X	X	MEDIO

Fuente: este estudio

5.5.5 Explicación de cada uno de los factores de la tabla M3

Uso de protocolo http sin cifrar*:* se detecta que las aplicaciones usan protocolo HTTP para enviar o recibir datos. Dicho protocolo no proporciona cifrado de ningún tipo por lo que si hay interceptación la información puede ser leída.

6 CONCLUSIONES

La comprensión de la metodología Owasp Mobile en el análisis de las aplicaciones móviles demuestra en su análisis estático y dinámico la necesidad de proteger las aplicaciones móviles de ataques de cibernéticos. La evidencia muestra falencias en el desarrollo que pueden sanearse con la implementación de metodologías maduras.

La arquitectura del sistema operativo Android muestra una robustez en su forma de aplicar la seguridad a las aplicaciones de modo que sea muy difícil el ataque por agujeros de seguridad. Se evidencia que los ciberdelincuentes usan técnicas como la ingeniería social para minar el sistema operativo.

Se ha tenido como referencia el marco de trabajo de Owasp Mobile 2016 haciendo énfasis sobre los riesgos M2 y M3 puntos débiles en la cadena de seguridad de los dispositivos. Android tiene como política la confianza entre su tecnología y los desarrolladores, permitiendo ver los permisos que solicita las aplicaciones antes de instalarse, quizá muchas veces los usuarios no tienen en cuenta estas advertencias e instalan aplicaciones falsas.

Se evidencia en el laboratorio ejecutado que hay riesgos que pueden calificarse como altos o bajos en almacenamiento y comunicación insegura según la metodología Owasp. La documentación que presenta Owasp ayuda a tener una visión estadística sobre la seguridad de cualquier aplicación. En el laboratorio se evidencia de forma recurrente el envío de información sin cifrar lo que constituye un grave problema de seguridad cuando las aplicaciones son usadas en redes abiertas donde puede existir interceptación de información.

El informe generado implementando la metodología Owasp y las herramientas de libre uso en el sistema operativo Linux ha mostrado aspectos claves que sirve para comprender las falencias que una aplicación móvil puede presentar respecto al riesgo de almacenamiento y comunicación insegura.

7 RECOMENDACIONES

La finalización de este estudio recomienda la implementación de pruebas automatizadas antes de salir al campo productivo. En el mercado existe muchas herramientas que son de pago, aunque también algunas gratis como la utilizada en este estudio llamado MobSF en Linux. Este tipo de software requiere de tiempo para su correcta configuración y poder observar los resultados para su posterior análisis.

Se recomienda la implementación de políticas de seguridad para tener y protocolos a seguir en la implementación de técnicas de cifrado. Toda política de seguridad debe ser de estricto cumplimiento y los directores deben hacer cumplirlas para evitar riesgos cibernéticos.

Tener en cuenta las opiniones de expertos en arquitectura de datos y servicios de comunicación que brinde la seguridad para la protección de la información.

BIBLIOGRAFÍA

A diario se registran 542.465 ataques informáticos en Colombia, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://www.eltiempo.com/tecnosfera/novedades-tecnologia/informe-sobre-ataques-informaticos-en-colombia-y-al-sector-financiero-135370

ABC tecnología. Android: el riesgo de los «Smartphone» inseguros. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: http://www.abc.es/tecnologia/moviles/telefonia/abci-android-riesgo-smartphones-inseguros-201511070317_noticia.html

Ali FEIZOLLAH A, Nor BADRUL ANUAR A, Rosli SALLEH A, Guillermo SUAREZ-TANGI, Steven FURNELL (2016). “AndroDialysis: análisis de la intención de Android, Eficacia en la detección de malware, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: http://www0.cs.ucl.ac.uk/staff/G.SuarezdeTangil/papers/2017cosec-androdialysis.pdf

Android fue el sistema operativo más atacado durante 2017. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://elcomercio.pe/tecnologia/empresas/youtube-android-sistema-operativo-atacado-2017-video-noticia-486089

Android Inc. [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://es.wikipedia.org/wiki/Android_Inc.

Arquitectura de la plataforma, [en línea] [citado el 8 de enero, 2019]. Disponible en internet: https://developer.android.com/guide/platform/?hl=es-419

Así fue la primera llamada por celular 45 años atrás, [en línea] [citado el 3 de abril, 2018]. Disponible en internet: https://mundo.sputniknews.com/tecnologia/201804031077570710-celular-primera-llamada/

Autocontrol y Ciberdelincuencia. Club Ciencias Forenses. [En línea] [Citado el 3 de marzo, 2019]. Disponible en internet: https://www.clubforenses.com/autocontrol-y-ciberdelincuencia-club-ciencias-forenses/

Centro de políticas de desarrolladores. [En línea] [Citado el 3 de marzo, 2019]. Disponible en internet: https://play.google.com/about/developer-content-policy/?hl=ES#!?modal_active=none

Colombia: el sexto país con más ataques cibernéticos en a. latina, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: http://www.enter.co/chips-bits/seguridad/colombia-sexto-pais-ataques-ciberneticos/

Cómo solucionar los problemas más comunes de Android, [en línea] [citado el 15 mayo, 2018]. Disponible en internet: https://elandroidelibre.elespanol.com/2015/02/como-solucionar-los-problemas-mas-comunes-de-android.html

Descubrimiento de las ondas de Radio: la confirmación de la Teoría Electromagnética, [en línea] [citado el 28 de abril, 2009]. Disponible en internet: https://www.investigacionyciencia.es/blogs/fisica-y-quimica/10/posts/descubrimiento-de-las-ondas-de-radio-la-confirmacin-de-la-teora-electromagntica-10186

DUNHAM, Ken; Abu NIMEH, SAEED; BECHER, MICHAEL (2008). Ataque y defensa maliciosos móviles. Syngress Media. ISBN 978-1-59749-298-0, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://books.google.com.co/books?id=Nd1RcGWMKnEC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

El Ransomware móvil se triplicó en el primer trimestre de 2017. El número de archivos detectados llegó a 218.625, según informe de Kaspersky Lab. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: http://www.eltiempo.com/tecnosfera/novedades-tecnologia/estadisticas-del-ransomware-movil-en-primer-trimestre-de-2017-91760

El teléfono celular. Historia y evolución de los celulares, [en línea] [citado el 26 de febrero, 2019]. Disponible en internet: https://tecnologia-informatica.com/telefono-celular-historia-evolucion-celulares/

Huawei: por qué Estados Unidos considera al gigante tecnológico chino una amenaza a la seguridad nacional, Redacción, BBC News Mundo. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://www.bbc.com/mundo/noticias-46475391

Ingeniería inversa de malware protegido [en línea] [citado el 15 mayo, 2018]. Disponible en internet: http://s3lab.deusto.es/ingenieria-inversa-malware-protegido/

KASPERSKY LAB. Kaspersky Security Bulletin 2013. Overall statistics for 2013. December de 2013. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://www.securelist.com/en/analysis/204792318/Kaspersky_Security_Bulletin_2013_Overall_statistics_for_2013

Kaspersky lab. Virus y Malware en Móviles Android. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: https://www.kaspersky.es/resource-center/threats/mobile

La historia del teléfono móvil: Origen, pasado y presente, [en línea] [citado el 28 de febrero, 2019]. Disponible en internet: http://culturacion.com/la-historia-del-telefono-movil-origen-pasado-y-presente/

La tecnología móvil y el internet de las cosas, [en línea] [citado el 7 de febrero, 2018]. Disponible en internet: http://noticias.universia.es/ciencia-tecnologia/noticia/2018/02/07/1157844/tecnologia-movil-internet-cosas.html

Las amenazas móviles. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: https://www.symantec.com/connect/blogs/las-amenazas-moviles

Las 'apps' más descargadas en Android e iOS durante el 2018, [en línea] [citado el 3 de marzo, 2019]. Disponible en internet: https://www.eltiempo.com/tecnosfera/novedades-tecnologia/las-aplicaciones-mas-descargadas-en-el-primer-trimestre-de-2018-213556

Los ataques cibernéticos se incrementaron este año, [en línea] [citado el 2 de marzo, 2019]. Disponible en internet: https://www.dinero.com/internacional/articulo/incremento-de-ataques-ciberneticos-en-el-2018/264180

M2-Almacenamiento de datos inseguros, [en línea] [citado el 1 marzo, 2019]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-M2-Insecure_Data_Storage

Más allá de Google Play y App Store. Los usuarios recurren a alternativas a las tiendas oficiales para instalar aplicaciones que aún no han sido lanzadas en su país o que ofrecen promociones. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://elpais.com/tecnologia/2017/05/31/actualidad/1496242186_229624.html

Más allá del PC. Este es el futuro de la computación. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: http://www.elmundo.es/economia/innovadores/2017/06/19/5947920f468aebac028b4612.html

Master en desarrollo de aplicaciones Android. Componentes de una aplicación. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/149-componentes-de-una-aplicacion

MIERES, Jorge. Ataques informáticos. Debilidades de Seguridad comúnmente explotadas. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: https://www.evilfingers.com/publications/white_AR/01_Ataques_informaticos.pdf

MOBILE MALWARE EVOLUTION 2016, [en línea] [citado el 8 de mayo, 2018]. Disponible en internet: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180734/Mobile_report_2016.pdf

Mobile Top 10 2016-Top 10, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

Nueva vulnerabilidad crítica en Android: si descargas de la Play Store, tranquilo, [en línea] [citado el 4 de julio, 2013]. Disponible en internet: https://www.xatakamovil.com/seguridad/nueva-vulnerabilidad-critica-en-android-si-descargas-de-la-play-store-tranquilo

OWASP Open Web Application Security Project. Category: Vulnerability. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://www.owasp.org/index.php/Category:Vulnerability

Paneles de control. Versiones de la plataforma. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: https://developer.android.com/about/dashboards/

Portafolio Juan José CÁNOVAS BUSTAMANTE. Elementos de una app de Android. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://juanjosecanbus.wordpress.com/2014/09/28/practica-1-elementos-de-una-app-de-android-pmm/

Portaltic Europa press. Detectadas ocho aplicaciones con 'malware' en Google Play Store capaces de saltarse sus sistemas de seguridad. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: http://www.europapress.es/portaltic/ciberseguridad/noticia-detectadas-ocho-aplicaciones-malware-google-play-store-capaces-saltarse-sistemas-seguridad-20171116144529.html

Proyecto de seguridad móvil OWASP, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Home

Ransomware en Android subió más del 50% durante 2016. [En línea] [Citado el 5 mayo, 2018]. Disponible en internet: https://www.dinero.com/empresas/confidencias-on-line/articulo/ransomware-en-android-subio-mas-del-50-durante-2016/242231

Razones que explican las menores ventas de computadores, [en línea] [citado el 2 de marzo, 2019]. Disponible en internet: https://www.dinero.com/economia/articulo/razones-para-caida-venta-computadores-nivel-mundial/208282

Seguridad en el desarrollo de aplicaciones móviles: los 5 mayores riesgos, de Alejandro CABALLERO, marzo 8, 2018, Noticias mundo móvil, [en línea] [citado el 15 mayo, 2018]. Disponible en internet: https://kingofapp.es/blog/seguridad-desarrollo-aplicaciones-moviles-mayores-riesgos/

Seguridad para Android: cinco consejos fundamentales. Seguridad en Internet. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://latam.kaspersky.com/resource-center/preemptive-safety/android-security-tips

Seguridad para aplicaciones móviles. ¿Cómo funciona el Sandbox Linux en Android? [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: http://seguridadparaaplicaciones.com/como-funciona-el-sandbox-linux-en-android/

The rise of mobile banker Asacub, By Tatyana SHISHKOVA on August 28, 2018. 10:00 am. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: https://securelist.com/the-rise-of-mobile-banker-asacub/87591/

Un reciente análisis de Avast sobre 160 millones de dispositivos móviles demuestra que el cibercrimen móvil está en aumento. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://blog.avast.com/es/nueva-investigacion-revela-el-incremento-de-amenazas-moviles

Universidad Nacional de Lujan. Departamento de Seguridad Informática. Análisis a la seguridad de la información. [En línea] [Citado el 8 de mayo, 2018]. Disponible en internet: http://www.seguridadinformatica.unlu.edu.ar/?q=node/12

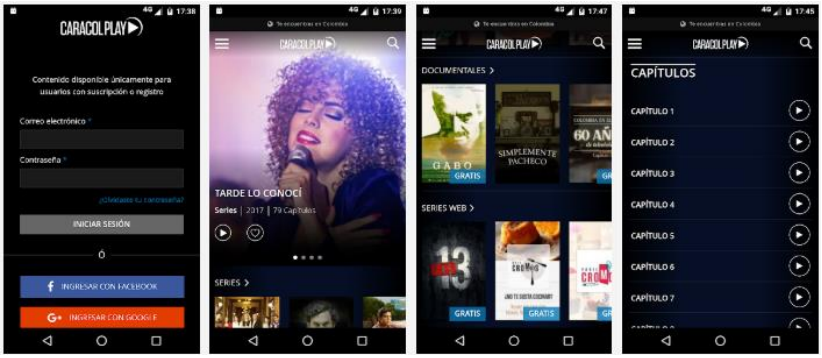
Versiones de Android [en línea] [citado el 5 de mayo, 2018]. Disponible en internet: https://actualizar-android.com/versiones

ANEXO A RESULTADOS EJECUCION PRUEBAS

I. CARACOL PLAY

Funcionalidad: Caracol Play es una experiencia única para sus usuarios con suscripción o registro. Con una amplia oferta de contenidos, los usuarios podrán disfrutar de series, telenovelas, documentales, realities y formatos web propios de Caracol Televisión desde cualquier dispositivo con acceso a internet de manera fácil y rápida[63].

Figura 34. Imagen funcionalidad



Fuente: https://play.google.com/store/apps/details?id=net.icck.CaracolPlay

1. Permisos solicitados por la aplicación a nivel de instalación

A continuación, se muestra los permisos que solicita el App al ser instalado

Figura 35. Permisos solicitados por la aplicación Caracol Play

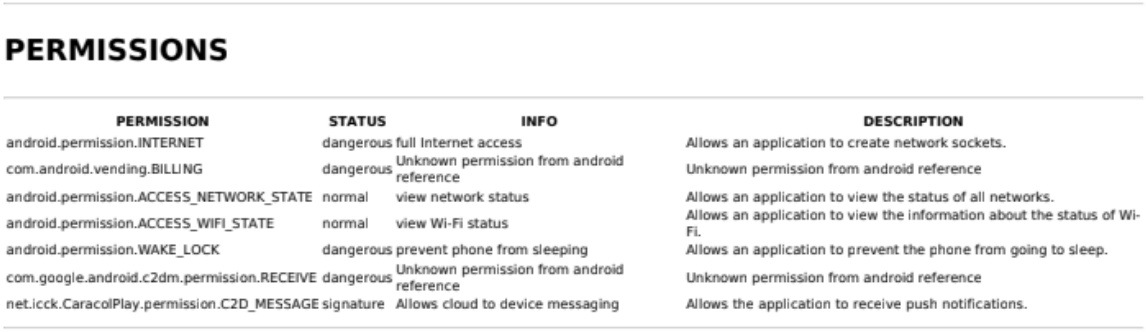
Fuente: este estudio

[63] Caracol Play, https://sensortower.com/android/CO/caracol-television-s-a/App/caracol-play/net.icck.CaracolPlay/overview

2. Permisos archivo Android Manifest.xml

A continuación, se observa los permisos que la aplicación pide en el archivo manifest.xml. Esta figura es arrojada por el proyecto de análisis estático MobSF

Figura 36. Permisos detectados por MobSF Caracol Play



PERMISSIONS

PERMISSION	STATUS	INFO	DESCRIPTION
android.permission.INTERNET	dangerous	full Internet access	Allows an application to create network sockets.
com.android.vending.BILLING	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.
android.permission.ACCESS_WIFI_STATE	normal	view Wi-Fi status	Allows an application to view the information about the status of Wi-Fi.
android.permission.WAKE_LOCK	dangerous	prevent phone from sleeping	Allows an application to prevent the phone from going to sleep.
com.google.android.c2dm.permission.RECEIVE	dangerous	Unknown permission from android reference	Unknown permission from android reference
net.icck.CaracolPlay.permission.C2D_MESSAGE	signature	Allows cloud to device messaging	Allows the application to receive push notifications.

Fuente: Esta investigación.

En la siguiente figura se observa los permisos configurados en el archivo físico de la aplicación manifest.xml

Figura 37. Visualización archivo Manifest.xml. Caracol Play

```
     <?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="19" android:versionName="2.1.0" package="net.icck.CaracolPlay" platformBuildVersionCode="25" platf
ormBuildVersionName="7.1.1"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="16" android:targetSdkVersion="25" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <permission android:name="net.icck.CaracolPlay.permission.C2D_MESSAGE" android:protectionLevel="signature" />
    <uses-permission android:name="net.icck.CaracolPlay.permission.C2D_MESSAGE" />
    <application android:theme="@style/Theme.AppCompat.NoActionBar" android:label="@string/app_name" android:icon="@mipmap/ic_lau
ncher" android:name="net.icck.CaracolPlay.Application" android:allowBackup="true" android:hardwareAccelerated="true">
        <activity android:label="@string/app_name" android:name="net.icck.CaracolPlay.MainActivity" android:configChanges="keyboa
rdHidden|orientation|screenSize">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

Fuente: esta investigación

3. Análisis estático M2

Base de datos SQLITE tipo texto sin cifrar

Se encontró por parte de la aplicación las sentencias de creación de base de datos exponiendo así su integridad.

Por parte de la aplicación.

Figura 38. Tablas usadas por la aplicación

Fuente: esta investigación

A continuación, la siguiente figura muestra la sentencia de creación de base de datos que genera la aplicación, aunque es común para muchas aplicaciones que usan SqlLiter.

Figura 39. Query de ejecución de la aplicación

```
CREATE TABLE meta(key LONGVARCHAR NOT NULL UNIQUE PRIMARY KEY, value LONGVARCHAR);CREATE TABLE autofill (name VARCHAR,
value VARCHAR, value_lower VARCHAR, date_created INTEGER DEFAULT 0, date_last_used INTEGER DEFAULT 0, count INTEGER
DEFAULT 1, PRIMARY KEY (name, value));CREATE INDEX autofill_name ON autofill (name);CREATE INDEX
autofill_name_value_lower ON autofill (name, value_lower);CREATE TABLE credit_cards ( guid VARCHAR PRIMARY KEY,
name_on_card VARCHAR, expiration_month INTEGER, expiration_year INTEGER, card_number_encrypted BLOB, date_modified
INTEGER NOT NULL DEFAULT 0, origin VARCHAR DEFAULT '', use_count INTEGER NOT NULL DEFAULT 0, use_date INTEGER NOT NULL
DEFAULT 0, billing_address_id VARCHAR);CREATE TABLE autofill_profiles ( guid VARCHAR PRIMARY KEY, company_name VARCHAR,
street_address VARCHAR, dependent_locality VARCHAR, city VARCHAR, state VARCHAR, zipcode VARCHAR, sorting_code VARCHAR,
country_code VARCHAR, date_modified INTEGER NOT NULL DEFAULT 0, origin VARCHAR DEFAULT '', language_code VARCHAR,
use_count INTEGER NOT NULL DEFAULT 0, use_date INTEGER NOT NULL DEFAULT 0);CREATE TABLE autofill_profile_names ( guid
VARCHAR, first_name VARCHAR, middle_name VARCHAR, last_name VARCHAR, full_name VARCHAR);CREATE TABLE
autofill_profile_emails ( guid VARCHAR, email VARCHAR);CREATE TABLE autofill_profile_phones ( guid VARCHAR, number
VARCHAR);CREATE TABLE autofill_profiles_trash ( guid VARCHAR);CREATE TABLE masked_credit_cards (id VARCHAR,status
VARCHAR,name_on_card VARCHAR,network VARCHAR,last_four VARCHAR,exp_month INTEGER DEFAULT 0,exp_year INTEGER DEFAULT 0,
bank_name VARCHAR, type INTEGER DEFAULT 0);CREATE TABLE unmasked_credit_cards (id VARCHAR,card_number_encrypted VARCHAR,
use_count INTEGER NOT NULL DEFAULT 0, use_date INTEGER NOT NULL DEFAULT 0, unmask_date INTEGER NOT NULL DEFAULT
0);CREATE TABLE server_card_metadata (id VARCHAR NOT NULL,use_count INTEGER NOT NULL DEFAULT 0, use_date INTEGER NOT
NULL DEFAULT 0, billing_address_id VARCHAR);CREATE TABLE server_addresses (id VARCHAR,company_name
VARCHAR,street_address VARCHAR,address_1 VARCHAR,address_2 VARCHAR,address_3 VARCHAR,address_4 VARCHAR,postal_code
VARCHAR,sorting_code VARCHAR,country_code VARCHAR,language_code VARCHAR, recipient_name VARCHAR, phone_number
VARCHAR);CREATE TABLE server_address_metadata (id VARCHAR NOT NULL,use_count INTEGER NOT NULL DEFAULT 0, use_date
INTEGER NOT NULL DEFAULT 0, has_converted BOOL NOT NULL DEFAULT FALSE);CREATE TABLE autofill_sync_metadata (storage_key
VARCHAR PRIMARY KEY NOT NULL,value BLOB);CREATE TABLE autofill_model_type_state (id INTEGER PRIMARY KEY, value BLOB);
```

Fuente: esta investigación

El siguiente cuadro muestra la exposición de urls en el código.

Cuadro 4. Líneas de código que exponen datos

Datos visibles en código
Archivo - net/icck/CaracolPlay/MainActivity.java
line 30: private static final **String** url = "https://playapp.caracoltv.com/user/login?destination=home";
Archivo - com/comscore/android/vce/c.java
line 48: static final **String** w = "https://sb.voicefive.com/rs/sdk/b.html";
Archivo - com/comscore/android/vce/f.java
line 19: this.d.deleteCookie("https://sb.voicefive.com/rs/sdk/b.html", str);
line 23: this.d.setCookie("https://sb.voicefive.com/rs/sdk/b.html", str, str2);
line 66: **String** cookie = this.d.getCookie("https://sb.voicefive.com/rs/sdk/b.html");
line 94: **String** cookie = this.d.getCookie("https://sb.voicefive.com/rs/sdk/b.html");
line 100: this.d.deleteCookie("https://sb.voicefive.com/rs/sdk/b.html", split2[0]);
line 107: **String**[] split = this.d.getCookie("https://sb.voicefive.com/rs/sdk/b.html").split(";");
Archivo - com/comscore/android/vce/c.java
line 49: static final **String** x = "https://sb.voicefive.com/rs/sdk/gg.js";
Archivo - com/comscore/android/vce/a.java
line 240: this.k = this.g.a(p.c ? "http://b.voicefive.com/rs/sdk/gg.js" : "https://sb.voicefive.com/rs/sdk/gg.js");

Fuente: Este estudio

Cuadro 5. (Continuación) Líneas de código que exponen datos

Archivo - com/comscore/android/vce/c.java
line 50: static final **String** y = "https://sb.scorecardresearch.com/rs/mobile/ntv/vce_st.js";
Archivo - com/comscore/android/vce/a.java
line 265: this.l = this.g.a(p.c ? "http://b.scorecardresearch.com/rs/mobile/ntv/vce_st.js" : "https://sb.scorecardresearch.com/rs/mobile/ntv/vce_st.js");
Archivo - com/comscore/android/vce/c.java
line 4: static final **String** A = "http://b.scorecardresearch.com/rs/mobile/ntv/vce_st.js";
Archivo - com/comscore/android/vce/a.java
line 265: this.l = this.g.a(p.c ? "http://b.scorecardresearch.com/rs/mobile/ntv/vce_st.js" : "https://sb.scorecardresearch.com/rs/mobile/ntv/vce_st.js");
Archivo - com/comscore/android/vce/c.java
line 51: static final **String** z = "http://b.voicefive.com/rs/sdk/gg.js";
Archivo - com/comscore/android/vce/a.java
line 240: this.k = this.g.a(p.c ? "http://b.voicefive.com/rs/sdk/gg.js" : "https://sb.voicefive.com/rs/sdk/gg.js");
Archivo - com/comscore/android/vce/aa.java
line 335: aa.this.K.loadDataWithBaseURL("http://localhost/", "<!DOCTYPE html><html><head></head><body><script type=\"text/javascript\">" + aa.this.g.c() + "</script></script></body></html>", "text/html", "UTF-8", null);
Archivo - com/comscore/android/vce/aa.java
line 24: private static final **String** I = "cvce://vce_m=";

Fuente: Este estudio

El siguiente cuadro muestra configuraciones que pueden suponer exposición de datos sensibles

Cuadro 6. Configuraciones de información sensible

Copia de seguridad habilitada de la aplicación
line 10: <application android:allowBackup="true"
android:hardwareAccelerated="true" android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:name="net.icck.CaracolPlay.Application"
android:theme="@style/Theme.AppCompat.NoActionBar">

Fuente: Este estudio

El siguiente cuadro muestra que la aplicación puede exponer información sensible mientras se ejecuta.

Cuadro 7. Solicitud de inicio de sesión en la cuenta de Google

Exposición de datos potencialmente sensibles
Nombre de usuario puede ser encontrado:
CheckinNowTaskTag Razón: 1 Fuerza: falsa Id. De usuario: 0 Significa que la aplicación solicita el inicio de sesión con la cuenta de Google,
Registro exitoso: CheckinNowTaskTag, https://android.clients.google.com/checkin (fragmento # 1): Significa la validación exitosa del inicio de sesión en Google
Comparte información con Urls cruzadas: Si existe una interceptación la información puede ser leída o incluso cambiada.
https://android.clients.google.com/checkin
https://www.googletagservices.com/tag/js/gpt.js
https://playapp.caracoltv.com/sites/default/files/js/js_uZwPoDiqXO6c7Pb0E3QANAXxCzFUvPIoFrth0dMviLs.js
https://tag.navdmp.com/tm35578.js

Fuente: Este estudio

4. Análisis estático M3

El siguiente cuadro muestra las configuraciones que permitirían conexiones tipo http

Cuadro 8. Configuraciones que muestran las conexiones con protocolo sin cifrar

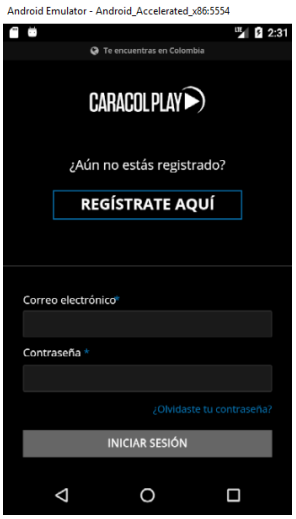
Uso de protocolo HTTP
Archivo - com/comscore/util/jni/JniComScoreHelper.java
line 116: try { line 117: httpURLConnection = (**HttpURLConnection**) new URL(str).openConnection(); line 118: if (httpURLConnection == null) {
Archivo - 'com/comscore/android/vce/r.java
line 61: public **HttpURLConnection** a(URL url) { line 62: **HttpURLConnection** = (**HttpURLConnection**) url.openConnection(); line 63: httpURLConnection.setConnectTimeout(60000);

Fuente: Este estudio

5. Análisis dinámico M2

Para este tipo de análisis se revisará las bases de datos que genera cada aplicación y si es legible ante herramientas de lectura de bases de datos tipo SQLite. Primero se instala la aplicación como se detalla a continuación.

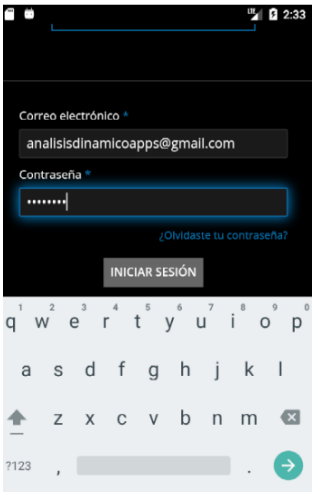
Figura 40. App en emulador Android estudio



Fuente: este estudio

A continuación, se inicia sesión con una cuenta de prueba en la App instalada en el emulador.

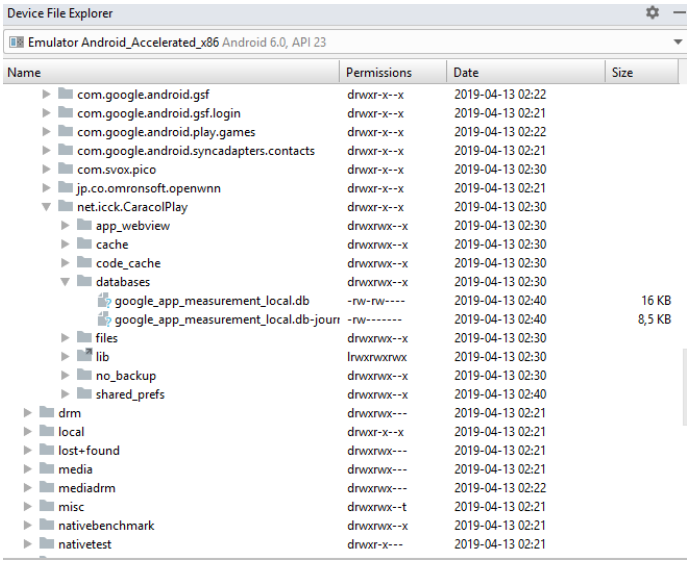
Figura 41. Inicio de sesión



Fuente: este estudio

Luego de instalar la aplicación y ejecutarla iniciando sesión se detectó la creación de una base de datos.

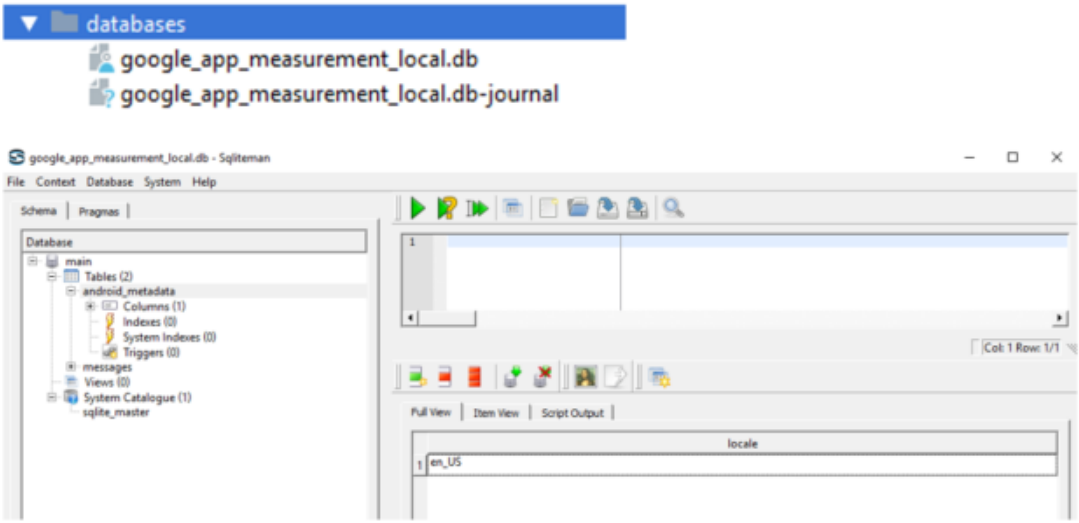
Figura 42. Bases de datos detectadas en la ejecución



Fuente: este estudio

Se procede a revisar la información de las bases de datos con la herramienta Sqliteman y no se observa datos sensibles, solo guarda la configuración de idioma y mensajes de error de ejecución, pero no se visualizó ningún dato sensible.

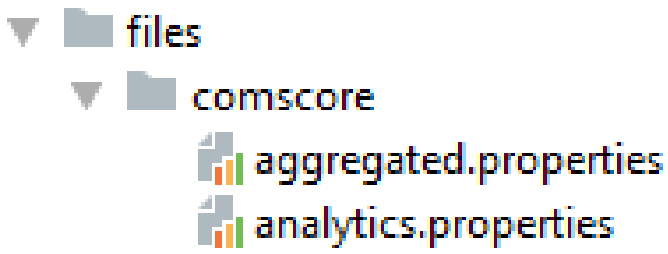
Figura 43. Revisión de bases de datos detectadas



Fuente: este estudio

Se revisa de igual forma la carpeta Files en busca de documentos que puedan guardar información sensible pero no se halló información sensible.

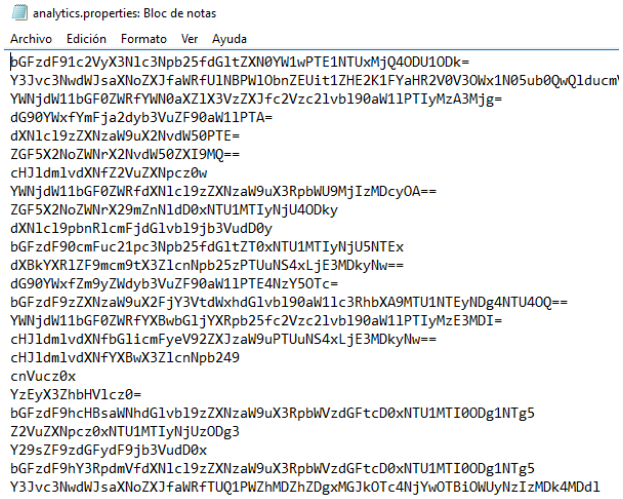
Figura 44. Carpeta Files



Fuente: este estudio

Aquí se observa como los datos provistos por Google si están

Figura 45. Archivo con datos encriptados de google analytics

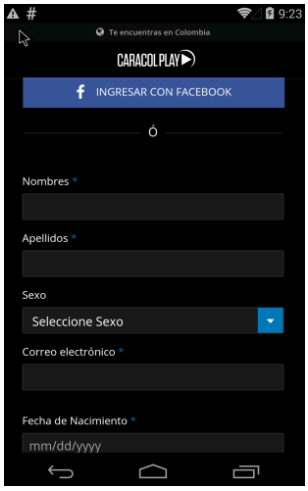


Fuente: este estudio

6. Análisis dinámico M3

Para el análisis dinámico se procede a instalar la app en cualquier emulador en este caso se usó el emulador de MobSF con la idea de analizar el comportamiento. A continuación, formulario inicial.

Figura 46. Formulario de registro de la App



Fuente: este estudio

Se observa con la herramienta Burp Suite como en cada validación el dato queda expuesto a cualquier atacante.

Figura 47. Datos expuestos en la validación de cada campo del formulario

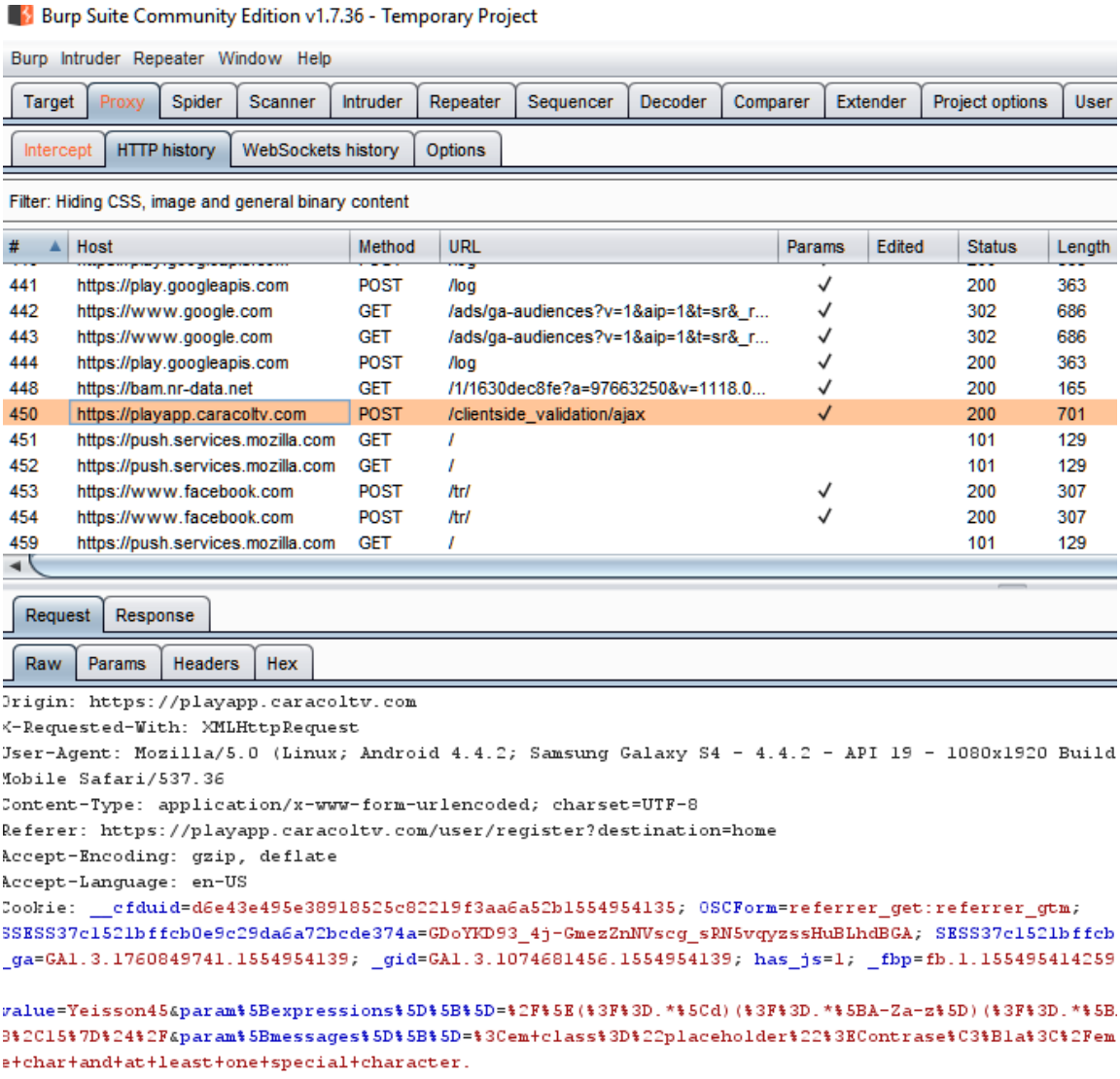
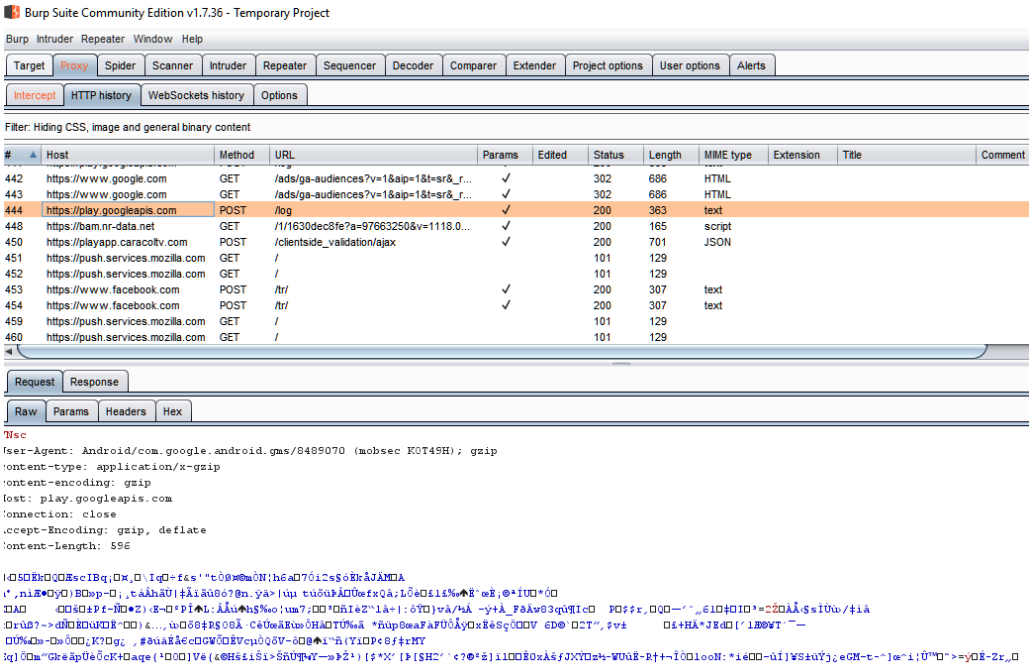


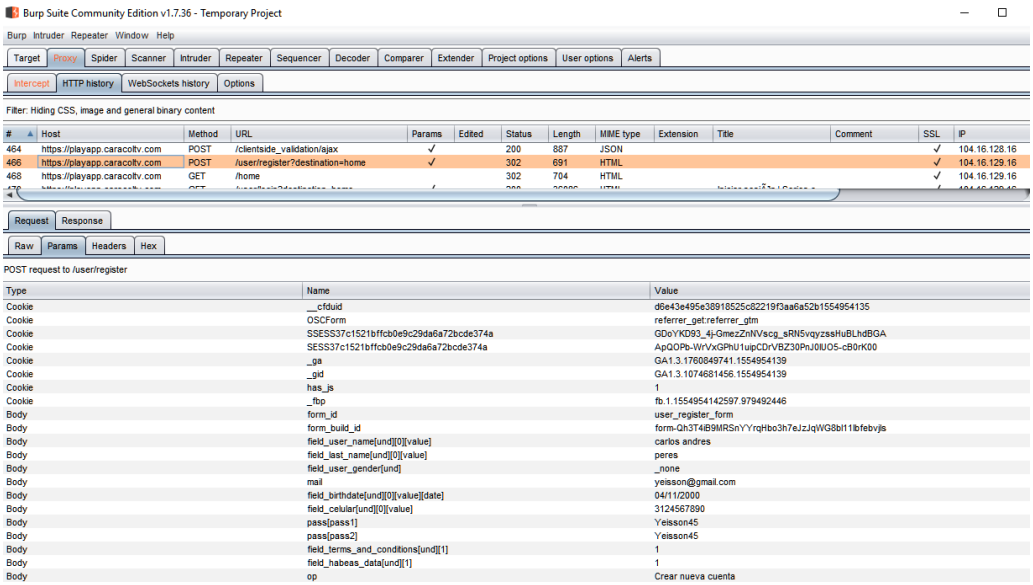
Figura 48. Ejemplo de petición a un servicio de google con datos encriptados



Fuente: este studio

También se evidencia él envió de datos personales de manera no segura.

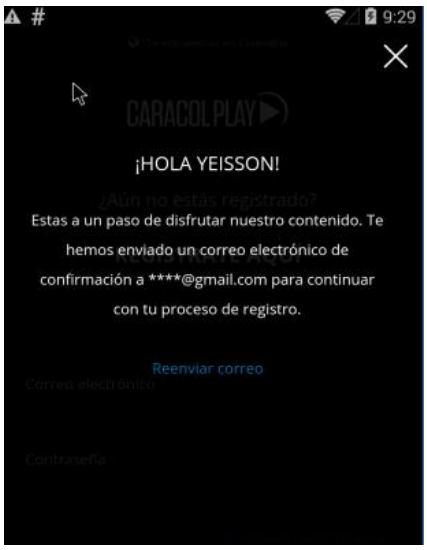
Figura 49. Datos expuestos en él envió de generación de usuario



Fuente: este estudio.

Se hace el registro y se evidencia él envió de información al correo electrónico.

Figura 50. Petición de creación de usuario



Fuente: este estudio

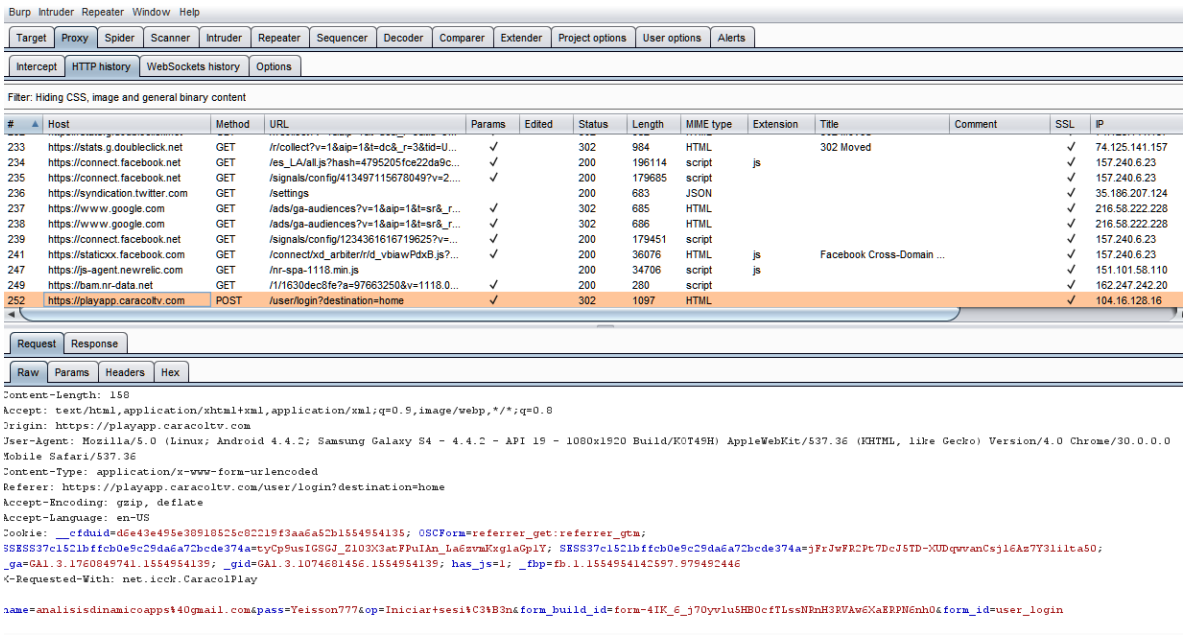
Luego se inicia sesión en la app para poder ingresar a los contenidos que ofrece.

Figura 51. Inicio de sesión

Fuente: este estudio

Cuando se envía la petición de inicio de sesión se observa datos sin encriptar

Figura 52. Petición de inicio de sesión exponiendo datos sensibles.



Fuente: este estudio

II. POU

Funcionalidad: Este App es de categoría game y su propósito es tener una mascota a la cual hay que cuidar. Entre sus cuidados esta alimentarlo, bañarlo, llevarlo a divertirse. Se escogió esta app porque usa información de pagos. A continuación, imágenes representativas.

Figura 53. Imágenes de la funcionalidad Pou en la App store

Fuente: este estudio.

1. Permisos solicitados por la aplicación a nivel de instalación

A continuación, se muestra los permisos que solicita el App al ser instalado.

Figura 54. Permisos solicitados por la aplicación Slither.

Fuente: este estudio.

2. PERMISOS ARCHIVO ANDROID MANIFEST.XML

A continuación, se muestra los permisos que se establecen en el archivo de configuración.

Figura 55. Permisos detectados por MobSF App Pou

PERMISSIONS

PERMISSION	STATUS	INFO	DESCRIPTION
android.permission.INTERNET	dangerous	full Internet access	Allows an application to create network sockets.
android.permission.WAKE_LOCK	dangerous	prevent phone from sleeping	Allows an application to prevent the phone from going to sleep.
android.permission.WRITE_EXTERNAL_STORAGE	dangerous	read/modify/delete SD card contents	Allows an application to write to the SD card.

PERMISSION	STATUS	INFO	DESCRIPTION
com.android.vending.BILLING	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.RECORD_AUDIO	dangerous	record audio	Allows application to access the audio record path.
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.
android.permission.ACCESS_WIFI_STATE	normal	view Wi-Fi status	Allows an application to view the information about the status of Wi-Fi.
android.permission.BLUETOOTH	dangerous	create Bluetooth connections	Allows an application to view configuration of the local Bluetooth phone and to make and accept connections with paired devices.
android.permission.BLUETOOTH_ADMIN	dangerous	bluetooth administration	Allows an application to configure the local Bluetooth phone and to discover and pair with remote devices.
android.permission.READ_PHONE_STATE	dangerous	read phone state and identity	Allows the application to access the phone features of the device. An application with this permission can determine the phone number and serial number of this phone, whether a call is active, the number that call is connected to and so on.
android.permission.READ_EXTERNAL_STORAGE	dangerous	read SD card contents	Allows an application to read from SD Card.
android.permission.ACCESS_COARSE_LOCATION	dangerous	coarse (network-based) location	Access coarse location sources, such as the mobile network database, to determine an approximate phone location, where available. Malicious applications can use this to determine approximately where you are.
android.permission.SYSTEM_ALERT_WINDOW	dangerous	display system-level alerts	Allows an application to show system-alert windows. Malicious applications can take over the entire screen of the phone.
com.google.android.gms.permission.ACTIVITY_RECOGNITION	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.ACCESS_FINE_LOCATION	dangerous	fine (GPS) location	Access fine location sources, such as the Global Positioning System on the phone, where available. Malicious applications can use this to determine where you are and may consume additional battery power.
com.google.android.providers.gsf.permission.READ_GSERVICES	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.GET_TASKS	dangerous	retrieve running applications	Allows application to retrieve information about currently and recently running tasks. May allow malicious applications to discover private information about other applications.

Fuente: este estudio.

Figura 56. Visualización archivo parcial Manifest.xml Pou

```
<?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="2224" android:versionName="1.4.77" android:installLocation="auto" package="me.pou.app" platformBuildVersionCode="23" platformBuildVersionName="6.0-2438415"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="14" android:targetSdkVersion="27" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-feature android:name="android.hardware.microphone" android:required="false" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-feature android:name="android.hardware.bluetooth" android:required="false" />
    <supports-screens android:anyDensity="true" android:smallScreens="true" android:normalScreens="true" android:largeScreens="true" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <application android:label="@string/app_name" android:icon="@drawable/icon" android:name="dkqrw.DDHPW" android:hardwareAccelerated="true">
        </receiver>
        <service android:name="com.paypal.android.sdk.payments.PayPalService" android:exported="false" />
        <activity android:name="com.paypal.android.sdk.payments.PaymentActivity" />
        <activity android:name="com.paypal.android.sdk.payments.LoginActivity" />
        <activity android:name="com.paypal.android.sdk.payments.PaymentMethodActivity" />
        <activity android:name="com.paypal.android.sdk.payments.PaymentConfirmActivity" />
        <activity android:name="com.paypal.android.sdk.payments.PaymentCompletedActivity" />
        <activity android:name="io.card.payment.CardIOActivity" android:configChanges="keyboardHidden|orientation" />
        <activity android:name="io.card.payment.DataEntryActivity" />
        <activity android:name="com.adcolony.sdk.AdColonyInterstitialActivity" android:configChanges="keyboardHidden|orientation|screenSize" android:hardwareAccelerated="true" />
        <activity android:name="com.adcolony.sdk.AdColonyAdViewActivity" android:configChanges="keyboardHidden|orientation|screenSize" android:hardwareAccelerated="true" />
        <activity android:theme="@android:style/Theme.NoTitleBar.Fullscreen" android:name="com.vungle.publisher.FullScreenAdActivity" android:configChanges="keyboardHidden|orientation|screenSize" />
        <service android:name="com.vungle.publisher.VungleService" android:exported="false" />
        <receiver android:name="com.google.android.gms.analytics.AnalyticsReceiver" android:enabled="true" android:exported="false" />
        <service android:name="com.google.android.gms.analytics.AnalyticsService" android:enabled="true" android:exported="false" />
        <service android:name="com.google.android.gms.analytics.AnalyticsJobService" android:permission="android.permission.BIND_JOB_SERVICE" android:enabled="true" android:exported="false" />
        <meta-data android:name="android.support.VERSION" android:value="26.1.0" />
        <activity android:theme="@android:style/Theme.Translucent.NoTitleBar" android:name="com.google.android.gms.common.api.GoogleApiActivity" android:exported="false" />
        <meta-data android:name="android.arch.lifecycle.VERSION" android:value="27.0.0-SNAPSHOT" />
        <activity android:theme="@android:style/Theme.Translucent.NoTitleBar" android:name="me.pou.app.OutDebugExternalActivity" android:taskAffinity="me.pou.app.ValueAf" android:finishOnTaskLaunch="true" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" android:hardwareAccelerated="true" />
        <activity android:theme="@android:style/Theme.Translucent.NoTitleBar" android:name="me.pou.app.JarProxySetActivity_" android:taskAffinity="me.pou.app.ValueAf" android:finishOnTaskLaunch="true" android:excludeFromRecents="true" android:noHistory="true" />
        <activity android:name="com.mopub.mobileads.MoPubActivity" android:taskAffinity="me.pou.app.ValueAf" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" />
        <activity android:name="com.mopub.mobileads.MraidActivity" android:taskAffinity="me.pou.app.ValueAf" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" />
        <activity android:name="com.mopub.common.MoPubBrowser" android:taskAffinity="me.pou.app.ValueAf" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" />
        <activity android:name="com.mopub.mobileads.MraidVideoPlayerActivity" android:taskAffinity="me.pou.app.ValueAf" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" />
        <activity android:name="com.mopub.mod.mobileads.MoPubActivity" android:taskAffinity="me.pou.app.ValueAf" android:excludeFromRecents="true" android:configChanges="keyboardHidden|orientation|screenSize" android:noHistory="true" />
```

Fuente: este estudio.

3. Análisis estático M2

El siguiente cuadro muestra como hay urls expuestas que pueden servir para futuros ataques.

Cuadro 9. Líneas de código que exponen datos para Pou

Datos visibles en código
me/pou/app/App.java:
line 398: webView.loadUrl("http://help.pou.me/");
me/pou/app/i/a/a/j.java:
line 10: this.i.e("http://help.pou.me/privacy-policy.php");
me/pou/app/i/a/a/g.java:
line 11: this.i.e("http://help.pou.me");
me/pou/app/outside/a/a.java:
line 118: r9 = "http://help.pou.me";
me/pou/app/e/b.java:
line 32: private **String** f = "http://app.pou.me/";
me/pou/app/e/b.java:
line 33: private **String** g = "http://s3-ap-southeast-1.amazonaws.com/pou-list-thumbs/";
me/pou/app/e/b.java:
line 34: private **String** h = "http://s3-ap-southeast-1.amazonaws.com/pou-store-thumbs/";
me/pou/app/e/b.java:
line 1251: stringBuilder.append("http://www.zazzle.com/api/create/at-238804284116464489?rf=238804284116464489&ax=DesignBlast&sr=250733517293108788&cg=196314461686901523&ed=true&continueUrl=http%3A%2F%2Fwww.zazzle.com%2Fpoustore&rut=Go%20back%20to%20Pou%20Store&fwd=ProductPage&tc=&ic=&t_icon_iid=&icon=");
com/vungle/publisher/inject/EndpointModule.java:
line 5: **String** a = "http://api.vungle.com/api/v4/";
com/samsungapps/plasma/d.java:
line 38: private static final **String** Y = "http://hub-odc.samsungapps.com/ods.as";
com/samsungapps/plasma/k.java:
line 25: static final **Uri** j = **Uri**.parse("http://web.teledit.com/Danal/Notice/help/samsung/yak.html");
com/samsungapps/plasma/MobileMicroPurchasePaymentMethod.java:
line 40: this.f = "http://img.samsungapps.com/marketing/common/images/logo_on.gif";
me/pou/app/g/c/a/b.java:
line 23: this.i.e("https://twitter.com/poualien");

Fuente: este estudio.

Cuadro 10. (Continuación) Líneas de código que exponen datos para Pou

me/pou/app/g/c/a/a.java:
line 23: this.i.e("https://facebook.com/mypou");
com/vungle/publisher/FullScreenAdActivity.java:
line 185: **Intent** a = **IntentFactory**.a("android.intent.action.VIEW", **Uri**.parse("https://www.vungle.com/privacy/"));
com/vungle/publisher/inject/EndpointModule.java:
line 6: **String** b = "https://ingest.vungle.com/";
com/adcolony/sdk/c.java:
line 115: aw.c = bj.c(this.d, "use_staging_launch_server") ? "https://adc3-launch-server-staging.herokuapp.com/v4/launch" : "https://adc3-launch.adcolony.com/v4/launch";
com/adcolony/sdk/aw.java:
line 21: static **String** c = "https://adc3-launch.adcolony.com/v4/launch";
com/adcolony/sdk/q.java:
line 67: e = new y(new bk(new URL("https://wd.adcolony.com/logs")), **Executors**.newSingleThreadScheduledExecutor(), hashMap);
com/samsungapps/plasma/MobileMicroPurchasePaymentMethod.java:
line 247: this.h = "https://mobile.inicis.com/smart/mobile/";
line 408: stringBuilder.append("https://mobile.inicis.com/smart/mobile/?");
com/b/a/a/a/c.java:
line 29: c.this.c.executeOnExecutor(**AsyncTask**.THREAD_POOL_EXECUTOR, new **String**[]{"https://mobile-static.adsafeprotected.com/avid-v2.js"});
line 32: c.this.c.execute(new **String**[]{"https://mobile-static.adsafeprotected.com/avid-v2.js"});

Fuente: este estudio.

El siguiente cuadro muestra como el móvil por medio de la aplicación puede acceder a almacenamiento externo tanto para lectura y escritura. Esto puede derivarse en corrupción de datos almacenados.

Cuadro 11. Código que expone almacenamiento de datos en elementos externos

Almacenamiento de datos externos
me/pou/app/App.java:
line 436: if (bitmap.compress(**CompressFormat**.PNG, 100, byteArrayOutputStream)) {
line 437: **File** externalStorageDirectory = **Environment**.getExternalStorageDirectory();
line 438: **StringBuilder** stringBuilder = new **StringBuilder**();
line 545: try {
line 546: **File** externalStorageDirectory = **Environment**.getExternalStorageDirectory();

Fuente: este estudio.

Cuadro 12. (Continuación). Código que expone almacenamiento de datos en elementos externos

line 547: StringBuilder stringBuilder = new StringBuilder();
line 851: if (bitmap.compress(CompressFormat.PNG, 100, byteArrayOutputStream)) {
line 852: File file = new File(Environment.getExternalStorageDirectory(), "Pou");
line 853: file.mkdirs();
line 1173: try {
line 1174: File file = new File(Environment.getExternalStorageDirectory(), "Android/data/me.pou.app/cache");
line 1175: if (file.exists()) {
line 1253: try {
line 1254: File file = new File(Environment.getExternalStorageDirectory(), "Pou");
line 1255: file.mkdirs();
com/adcolony/sdk/ad.java:
line 211: r2.<init>(); Catch:{ NoClassDefFoundError -> 0x010e }
line 212: r3 = android.os.Environment.getExternalStorageDirectory(); Catch:{ NoClassDefFoundError -> 0x010e }
line 213: r3 = r3.toString(); Catch:{ NoClassDefFoundError -> 0x010e }
line 233: r6.<init>(); Catch:{ Exception -> 0x00ab }
line 234: r7 = android.os.Environment.getExternalStorageDirectory(); Catch:{ Exception -> 0x00ab }
line 235: r7 = r7.getPath(); Catch:{ Exception -> 0x00ab }
line 243: r7.<init>(); Catch:{ Exception -> 0x00ab }
line 244: r8 = android.os.Environment.getExternalStorageDirectory(); Catch:{ Exception -> 0x00ab }
line 245: r8 = r8.getPath(); Catch:{ Exception -> 0x00ab }
com/vungle/publisher/inject/CoreModule_ProvideOldAdTempDirectoryFactory.java:
line 32: public final String get() {
line 33: if (((Context) this.c.get()).getExternalCacheDir() == null) {
line 34: throw new fa();
line 35: }
line 36: return (String) e.a(fc.a(((Context) this.c.get()).getExternalCacheDir(), ".VungleCacheDir"), "Cannot return null from a non-@Nullable @Provides method");
line 37: }
com/a/a/g.java:
line 34: b = 0;
line 35: File externalCacheDir = ((p() || VERSION.SDK_INT >= 19) && a.getExternalCacheDir() != null) ? a.getExternalCacheDir() : a.getCacheDir() != null ? a.getCacheDir() : a.getFilesDir();

Fuente: este estudio

Cuadro 13. (Continuación). Código que expone almacenamiento de datos en elementos externos

com/vungle/publisher/inject/CoreModule_ProvideAdTempDirectoryFactory.java:
line 32: public final **String** get() {
line 33: if (((**Context**) this.c.get()).getExternalFilesDir(null) == null) {
line 34: throw new fa();
line 35: }
line 36: return (**String**) e.a(fc.a(((**Context**) this.c.get()).getExternalFilesDir(null).getAbsolutePath(), ".vungle"), "Cannot return null from a non-@Nullable @Provides method");
line 37: }
com/vungle/publisher/env/AndroidDevice.java:
line 297: public final boolean o() {
line 298: boolean equals = "mounted".equals(**Environment**.getExternalStorageState());
line 299: boolean a = ji.a(this.i, this);

Fuente: este estudio

4. Análisis estático M3

El siguiente cuadro muestra las configuraciones que permitirían conexiones tipo http

Cuadro 14. Configuraciones que muestran las conexiones con protocolo inseguro

Uso de protocolo HTTP
com/vungle/publisher/net/http/HttpURLConnectionFactory.java
line 14: public static **HttpURLConnection** a(**String** str) {
line 15: return (**HttpURLConnection**) new URL(str).openConnection();
line 16: }
com/adcolony/sdk/ba.java
line 142: } else {
line 143: this.f = (**HttpURLConnection**) new URL(this.a).openConnection();
line 144: this.f.setInstanceFollowRedirects(c ^ 1);
com/a/a/c.java
line 96: private **HttpURLConnection** a(**String** str, byte[] bArr, boolean z) {
line 97: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) new URL(str).openConnection();
line 98: httpURLConnection.setDoOutput(true);

Fuente: este estudio.

5. ANÁLISIS DINÁMICO M2

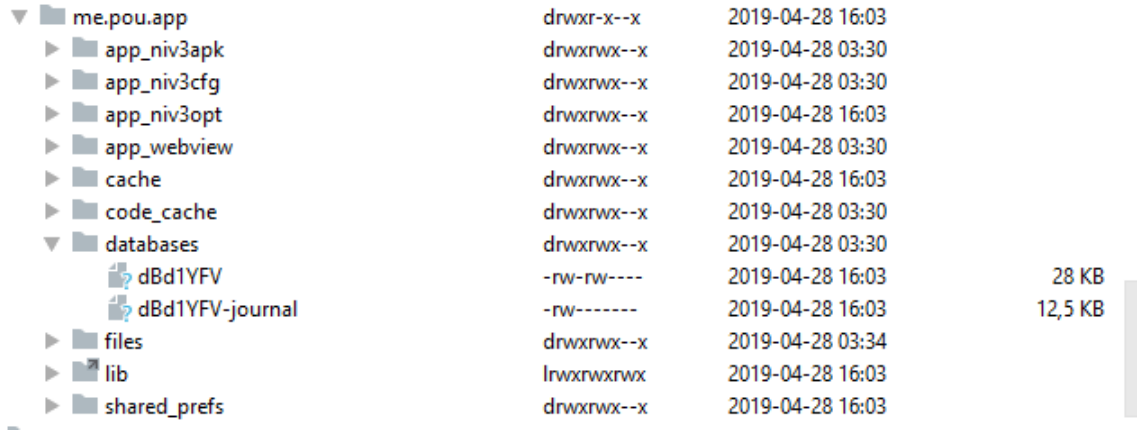
Para este tipo de análisis se revisará las bases de datos que genera cada aplicación y si es legible ante herramientas de lectura de bases de datos tipo SQLite. Primero se instala la aplicación como se detalla a continuación.

Figura 57. Capturas de instalación de la App

Fuente: este estudio.

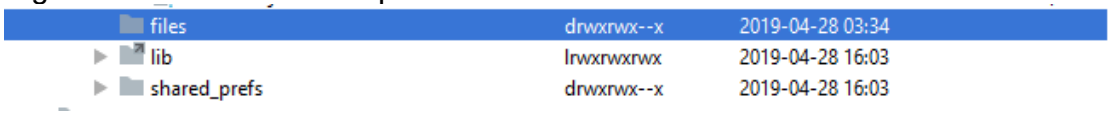
Luego de instalar la aplicación y ejecutarla iniciando sesión se detectó la creación de varios archivos de tipo SQLliter.

Figura 58. Archivos en la carpeta database.



Fuente: este estudio

Figura 59. Se revisa la carpeta files la cual esta vacía.



Fuente: este estudio

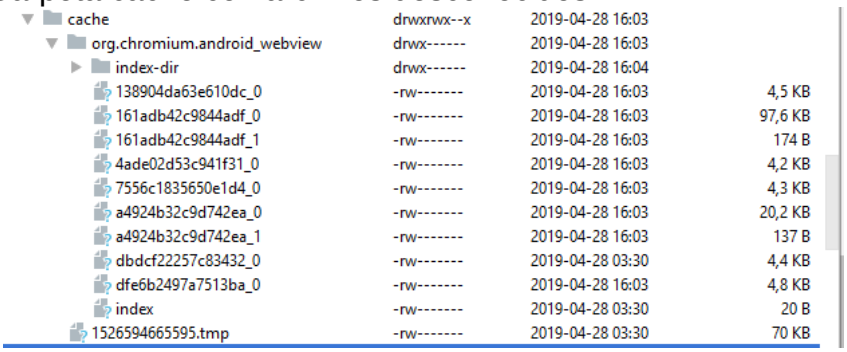
Se analizan otras carpetas en busca de información que pueda ser sensible

Figura 60. Archivos tipo xml en carpeta shared

Fuente: este estudio

Se analizan archivos en una carpeta llamada cache en busca de información.

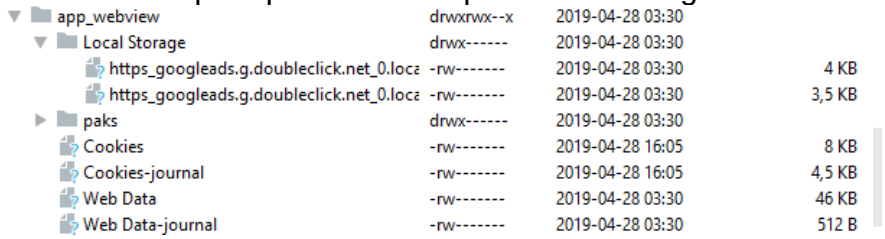
Figura 61. Carpeta cache con archivos desconocidos.



Fuente: este estudio

Revisión de otras carpetas. Al parecer la aplicación divide sus archivos de forma distinta.

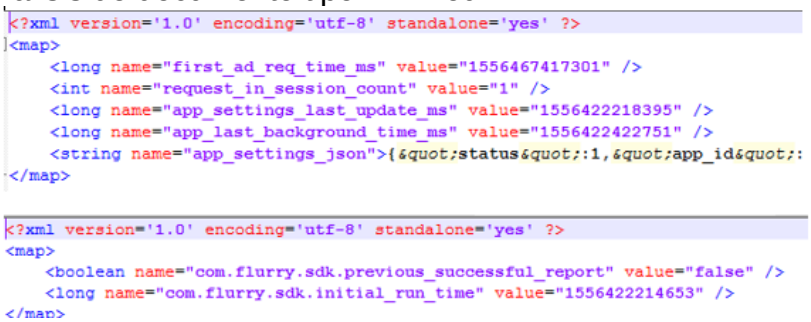
Figura 62. Archivos de tipo sqlliter en la carpeta local storage



Fuente: este estudio

Análisis de archivos xml donde se observan parámetros que pueden servir para realizar cambios en la aplicación.

Figura 63. Análisis de documento tipo xml Pou



Fuente: este estudio

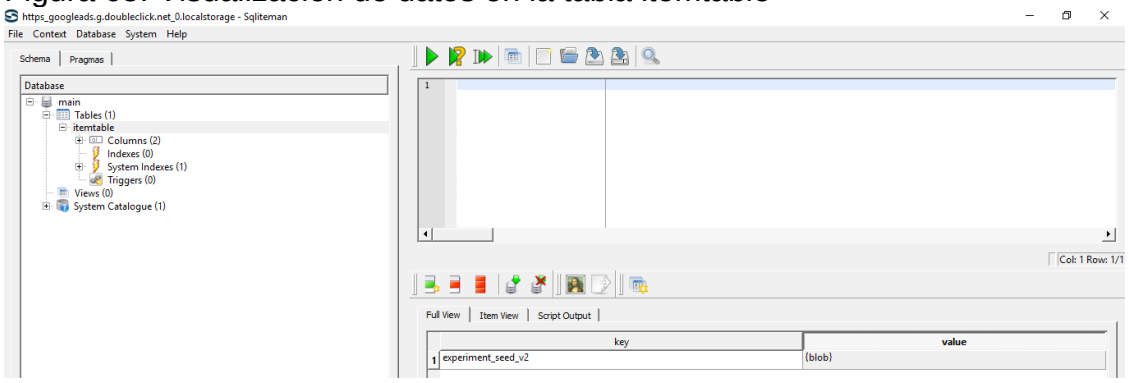
Figura 64. Parámetros encontrados en archivo de configuración que pueden ser modificados para uso de la aplicación.

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="session">qTNXNkCq10rFS8UZp1Vp176447468</string>
    <boolean name="muteMusic" value="false" />
    <string name="nickname">pou_XYBC09</string>
    <boolean name="muteSound" value="false" />
    <int name="lPS" value="1" />
    <int name="revision" value="224" />
    <string name="state">{&amp;quot;time&amp;quot;:1.556422546663E9,&amp;quot;rC&amp;quot;:2
    <string name="pL"></string>
    <string name="pLC"></string>
    <long name="lastStateUploadTime" value="1556422547330" />
    <boolean name="disableMicro" value="false" />
    <boolean name="decidedOnPersonalizedAds" value="false" />
    <int name="lGS" value="3" />
    <boolean name="disablePersonalizedAds" value="true" />
    <boolean name="dA" value="false" />
    <string name="check">801760d75a839578cdc18701b930110f</string>
    <boolean name="bluetoothPrefered" value="false" />
    <int name="version" value="4" />
    <int name="gameColors" value="1" />
    <boolean name="disableNotifications" value="false" />
    <boolean name="hSFLD" value="false" />
</map>
```

Fuente: este estudio

Visualización de base de datos *https_googleads_g_doubleclick.net_0.localstrorage*

Figura 65. Visualización de datos en la tabla itemtable

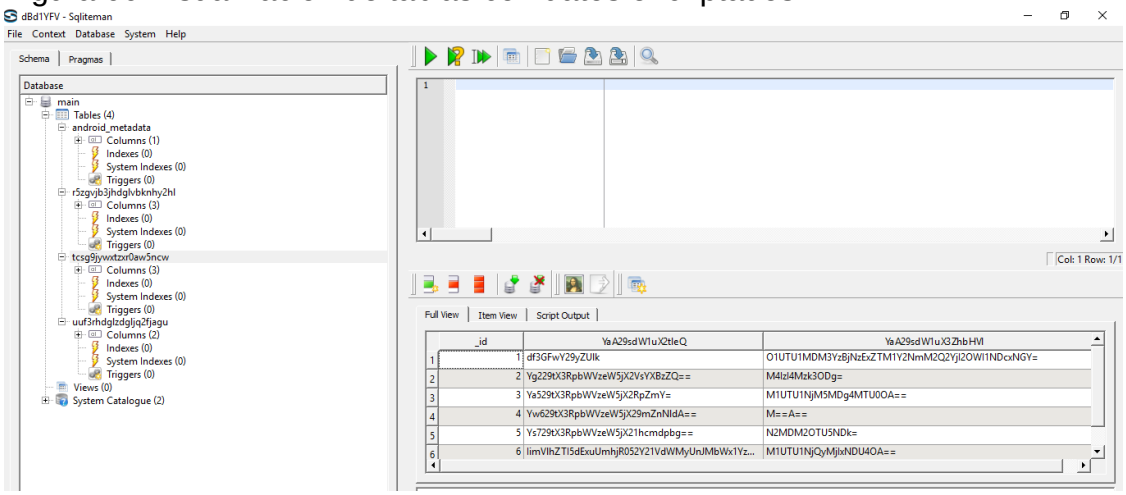


Fuente: este estudio

Visualización de base de datos *https_googleads_g_doubleclick.net_0.localstrorage*

Visualización de base de datos *dBd1YFV*

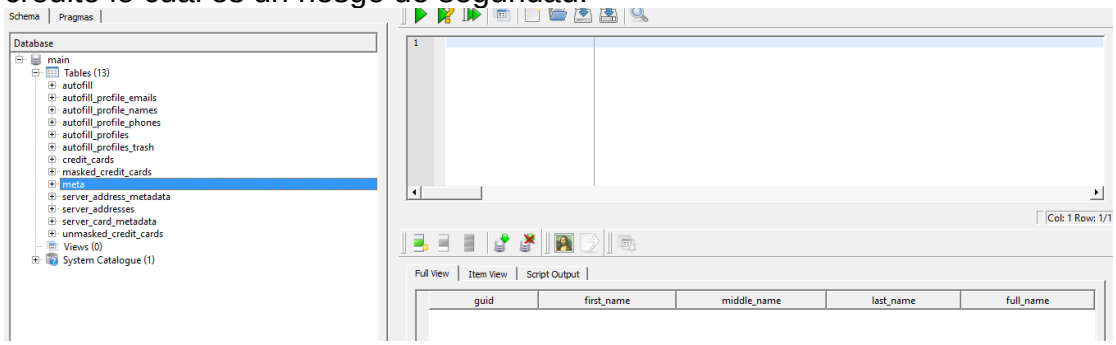
Figura 66.Visualizacion de tablas con datos encriptados



Fuente: este estudio

Visualización de base de datos *Web data*

Figura 67. Visualización de tablas que guardan datos encriptados de tarjetas de crédito lo cual es un riesgo de seguridad.

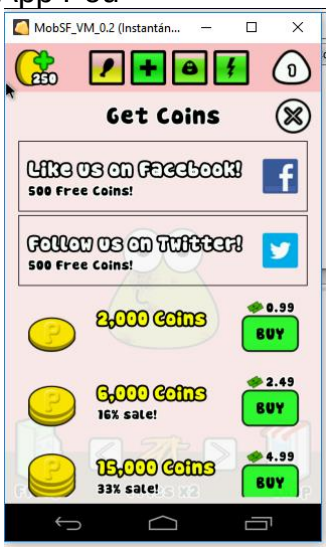


Fuente: este estudio

6. ANÁLISIS DINÁMICO M3

Para el análisis dinámico se procede a instalar la App en cualquier emulador en este caso se usó el emulador de MobSF con la idea de analizar el comportamiento. A continuación, formulario inicial de la App

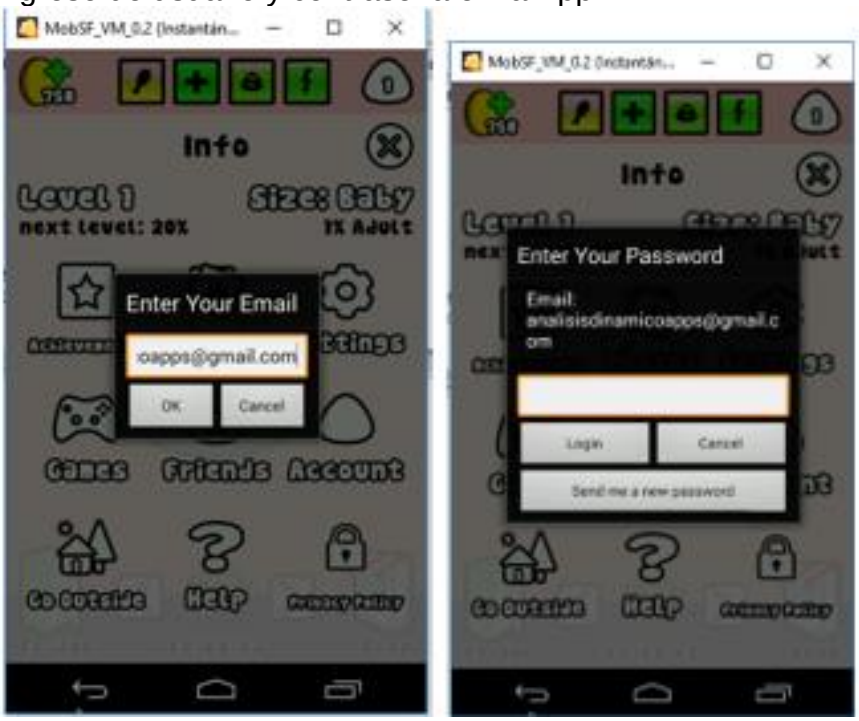
Figura 68. Inicialización de la App Pou



Fuente: este estudio

Se ingresa el usuario para iniciar sesión registrado

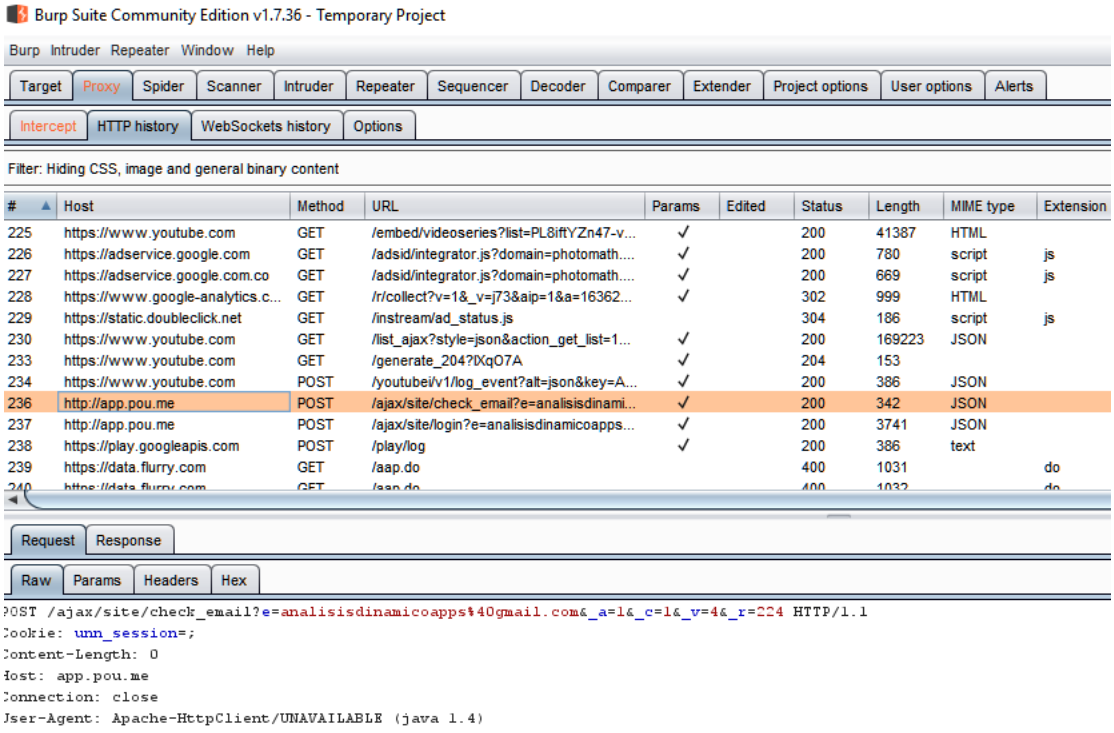
Figura 69. Ingreso de usuario y contraseña en la App



Fuente: este estudio

A continuación, se observa él envió del correo electrónico sin encriptar.

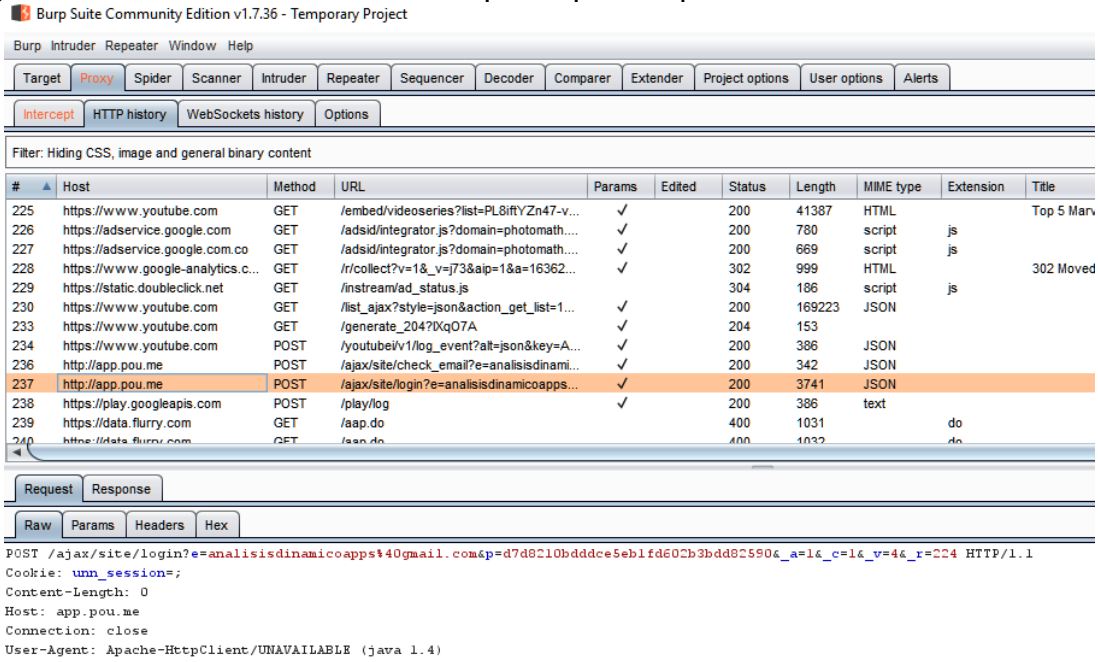
Figura 70. Envió de correo electrónico por la aplicación



Fuente: este estudio

A continuación, se observa como en el envío de contraseña se aplica un cifrado que permite la protección del dato

Figura 71. Envió de contraseña encriptada por la aplicación



Fuente: este estudio

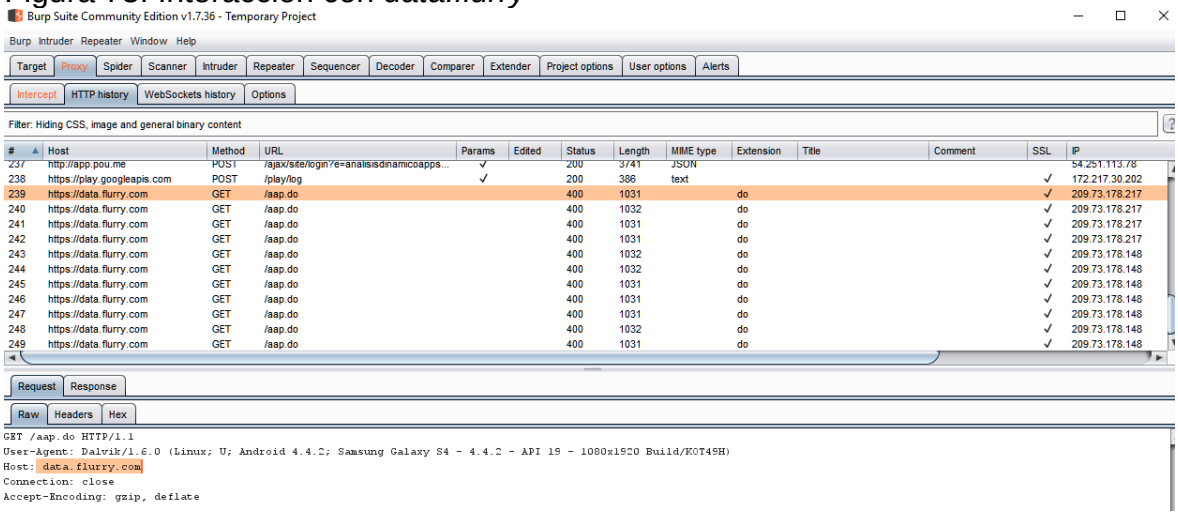
Se observa que la aplicación permite la recolección de datos de los usuarios por medio de páginas como flurry lo cual puede interpretarse como violación de privacidad. A continuación, el funcionamiento normal de la App.

Figura 72. Funcionamiento normal de la App Pou

Fuente: este estudio

Se observa cómo hay interacción con la página *data.flurry* y realización de descargas de paquetes .zip sin autorización del usuario

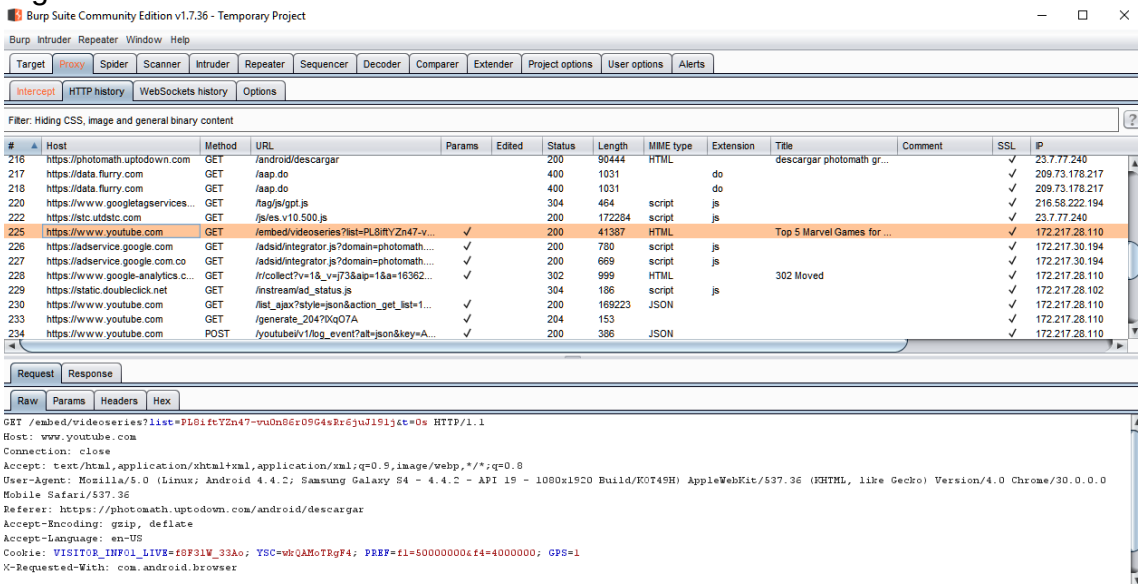
Figura 73. Interacción con *data.flurry*



Fuente: este estudio

A continuación, se observa la interacción constante de la aplicación con youtube son preaviso al usuario. Esto se traduce en violaciones de seguridad.

Figura 74. Interacción con Youtube sin aviso al usuario



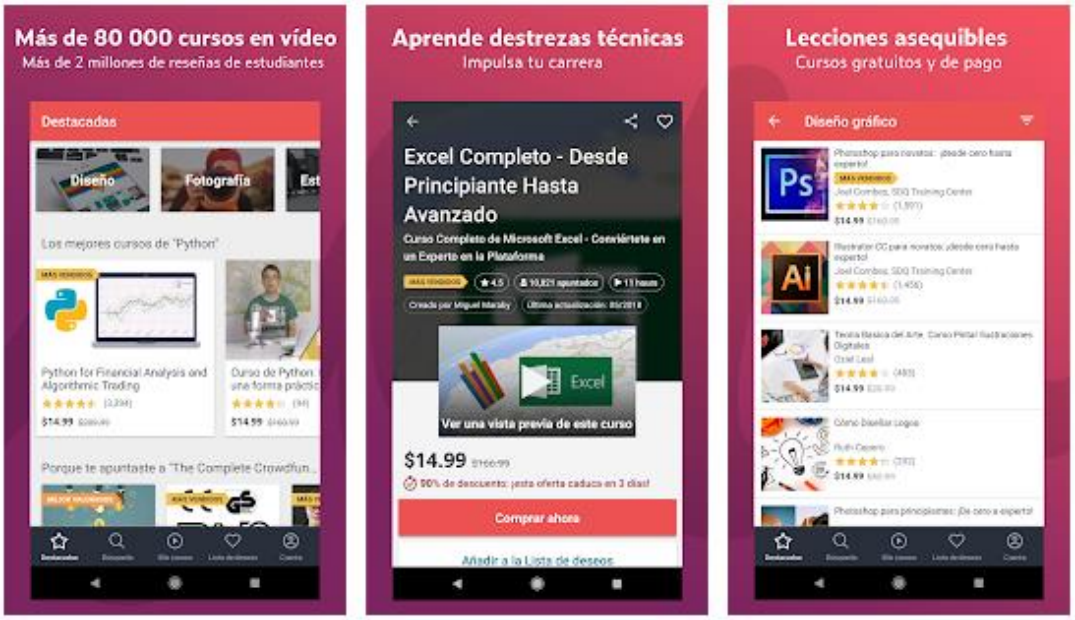
Fuente: este estudio

Se concluye que la aplicación maneja información sensible encriptada, pero hace uso constante de páginas ajenas como data.flurry y youtube.

III.Udemy

Funcionalidad: Udemy es una plataforma de aprendizaje online con más de 65 000 cursos en vídeo impartidos por instructores expertos. Toma cursos sobre todo tipo de temas, desde lenguajes de programación como Python y Java, hasta clases de desarrollo personal como diseño, dibujo, escritura y yoga. Únete a los más de 20 millones de estudiantes que ya están dominando nuevas capacidades, avanzando en sus carreras y explorando nuevas aficiones en Udemy[64].

Figura 75. Imágenes de la funcionalidad Udemy en la App store



Fuente: este estudio.

[64] https://play.google.com/store/Apps/details?id=com.udemy.android

1. Permisos solicitados por la aplicación a nivel de instalación

A continuación, se muestra los permisos que solicita la App al ser instalada.

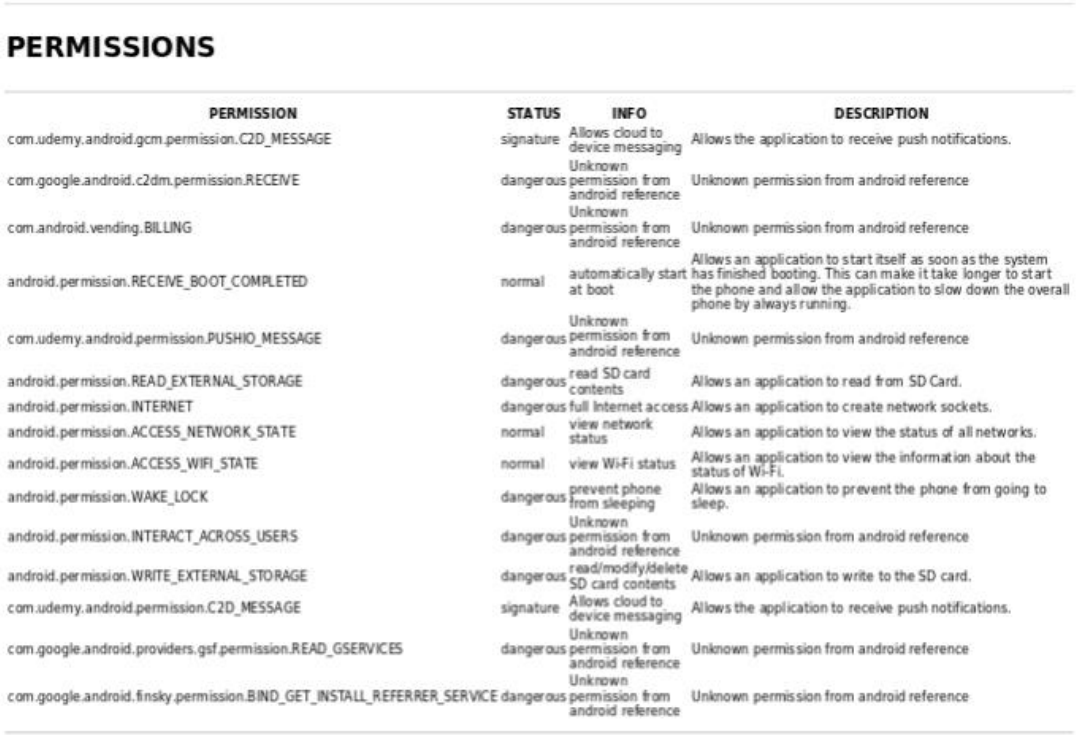
Figura 76. Permisos solicitados por la aplicación Udemy

Fuente: este estudio.

2. Permisos archivo Android Manifest.xml

A continuación, se muestra los permisos que se establecen en el archivo de configuración.

Figura 77. Permisos solicitados por la aplicación Udemy



PERMISSIONS

PERMISSION	STATUS	INFO	DESCRIPTION
com.udemy.android.gcm.permission.C2D_MESSAGE	signature	Allows cloud to device messaging	Allows the application to receive push notifications.
com.google.android.c2dm.permission.RECEIVE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.android.vending.BILLING	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.RECEIVE_BOOT_COMPLETED	normal	automatically start at boot	Allows an application to start itself as soon as the system has finished booting. This can make it take longer to start the phone and allow the application to slow down the overall phone by always running.
com.udemy.android.permission.PUSHIO_MESSAGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.READ_EXTERNAL_STORAGE	dangerous	read SD card contents	Allows an application to read from SD Card.
android.permission.INTERNET	dangerous	full Internet access	Allows an application to create network sockets.
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.
android.permission.ACCESS_WIFI_STATE	normal	view Wi-Fi status	Allows an application to view the information about the status of Wi-Fi.
android.permission.WAKE_LOCK	dangerous	prevent phone from sleeping	Allows an application to prevent the phone from going to sleep.
android.permission.INTERACT_ACROSS_USERS	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.WRITE_EXTERNAL_STORAGE	dangerous	read/modify/delete SD card contents	Allows an application to write to the SD card.
com.udemy.android.permission.C2D_MESSAGE	signature	Allows cloud to device messaging	Allows the application to receive push notifications.
com.google.android.providers.gsf.permission.READ_GSERVICES	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	dangerous	Unknown permission from android reference	Unknown permission from android reference

Fuente: este estudio.

Figura 78. Visualización archivo Manifest.xml parcial Udemy

```
<?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="229" android:versionName="5.5.2" package="com.udemy.android" platformBuildVersionCode="229" platformBuildVersionName="5.5.2"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="27" />
    <permission android:name="com.udemy.android.gcm.permission.C2D_MESSAGE" android:protectionLevel="signature" />
    <uses-permission android:name="com.udemy.android.gcm.permission.C2D_MESSAGE" />
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <permission android:name="com.udemy.android.permission.PUSHIO_MESSAGE" android:protectionLevel="signature" />
    <uses-permission android:name="com.udemy.android.permission.PUSHIO_MESSAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.INTERACT_ACROSS_USERS" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" android:maxSdkVersion="23" />
    <permission android:name="android.permission.MEDIA_CONTENT_CONTROL" />
    <uses-permission android:name="com.udemy.android.permission.C2D_MESSAGE" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
    <uses-permission android:name="com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE" />
    <permission android:name="com.udemy.android.permission.C2D_MESSAGE" android:protectionLevel="signature" />
    <uses-feature android:glEsVersion="0x00020000" android:required="true" />
    <application android:theme="@style/UdemyTheme" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" android:name="com.udemy.android.CombinedApplication" android:allowBackup="true" android:supportsRtl="true" android:networkSecurityConfig="@xml/network_security_config">
        <meta-data android:name="firebase_performance_logcat_enabled" android:value="false" />
        <activity android:theme="@style/Instructor.Theme.NoActionBar" android:name="com.udemy.android.CombinedDeepLinkActivity">
```

Fuente: este estudio.

3. Análisis estático M2

El siguiente cuadro muestra como el móvil por medio de la aplicación puede acceder a almacenamiento externo tanto para lectura y escritura. Esto puede derivarse en corrupción de datos almacenados.

Cuadro 15. Código que expone almacenamiento de datos

Almacenamiento de datos externos
Archivo: com/udemy/android/service/DownloadManager.java:
line 1281: if (**DownloadDirType**.externalFilesDirectory.equals(downloadDirType)) {
line 1282: return context.getExternalFilesDir(str);
line 1283: }
line 1292: if (!j()) {
line 1293: **File** externalFilesDir = context.getExternalFilesDir(str);
line 1294: **String** str2 = null;
Archivo: com/udemy/android/service/DownloadManager.java:
line 1298: } else if (VERSION.SDK_INT >= 23) {
line 1299: for (**File** file2 : context.getExternalCacheDirs()) {
line 1300: if (file2 != null && **Environment**.isExternalStorageRemovable(file2)) {
Archivo: com/udemy/android/helper/ExternalStorage.java:
line 13: try {
line 14: **String** externalStorageState = **Environment**.getExternalStorageState();
line 15: if ("mounted".equals(externalStorageState) || "mounted_ro".equals(externalStorageState)) {
line 23: public static boolean b() {
line 24: return "mounted".equals(**Environment**.getExternalStorageState());
line 25: }
Archivo: com/udemy/android/helper/ExternalStorage.java:
line 122: if (arrayMap.isEmpty()) {
line 123: arrayMap.put("sdCard", **Environment**.getExternalStorageDirectory());
line 124: }

Fuente: este estudio.

El siguiente cuadro muestra como la aplicación utiliza bases de datos SQLite sin cifrado. Esto puede dar información al atacante si hay un dispositivo rooteado y no hay datos cifrados.

Cuadro 16. Tablas y sentencias que expone la App Udemy

Base de datos SQLITE tipo texto sin cifrar
Archivo: __leanplum.db:
TABLES: android_metadata event RAW DUMP: CREATE TABLE android_metadata (locale TEXT);CREATE TABLE event(data TEXT);
Archivo: PushIOManager.db
TABLES: android_metadata events sqlite_autoindex_events_1 batches RAW DUMP: CREATE TABLE android_metadata (locale TEXT);CREATE TABLE IF NOT EXISTS 'events' (eventID TEXT PRIMARY KEY, eventName TEXT, extra BLOB, sessionID TEXT, timestamp INTEGER DEFAULT CURRENT_TIMESTAMP);CREATE TABLE IF NOT EXISTS 'batches' (batchID INTEGER PRIMARY KEY , retryCount INTEGER, sendTimestamp INTEGER DEFAULT CURRENT_TIMESTAMP, startEventID TEXT, endEventID TEXT);
Archivo: db_udemy-jobs:
TABLES: android_metadata job_holder sqlite_autoindex_job_holder_1 job_holder_tags TAG_NAME_INDEX RAW DUMP: CREATE TABLE android_metadata (locale TEXT);CREATE TABLE job_holder (insertionOrder integer primary key , `_id` text UNIQUE, `priority` integer, `group_id` text, `run_count` integer, `created_ns` long, `delay_until_ns` long, `running_session_id` long, `network_type` integer, `deadline` integer, `cancel_on_deadline` integer, `cancelled` intege);CREATE TABLE job_holder_tags (_id integer primary key , `job_id` text, `tag_name` text, FOREIGN KEY(`job_id`) REFERENCES job_holder(`_id`) ON DELETE CASCADE);CREATE INDEX TAG_NAME_INDEX ON job_holder_tags(tag_name);

Fuente: este estudio.

El siguiente cuadro muestra como hay datos quemados en el código que pueden servir para futuros ataques

Cuadro 17. Líneas de código que exponen datos para Udemy

Datos codificados
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 100: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/ignore-bogons", a(j));
line 145: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/ignore-bogons")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 101: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/bogons-empty", a(k));
line 148: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/bogons-empty")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 102: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/root-bogons", a(l));
line 151: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/root-bogons")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 103: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/default-attributes", a(m));
line 154: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/default-attributes")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 104: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/translate-colons", a(n));
line 157: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/translate-colons")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 105: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/restart-elements", a(o));
line 160: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/restart-elements")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 106: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/ignorable-whitespace", a(p));
line 163: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/ignorable-whitespace")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 107: this.A.put("http://www.ccil.org/~cowan/tagsoup/features/cdata-elements", a(q));
line 166: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/features/cdata-elements")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 181: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/properties/scanner")) {
line 205: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/properties/scanner")) {
Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 184: if (str.equals("http://www.ccil.org/~cowan/tagsoup/properties/schema")) {
line 211: } else if (str.equals("http://www.ccil.org/~cowan/tagsoup/properties/schema")) {
Archivo: com/pixplicity/htmlcompat/HtmlCompat.java:
line 53: parser.setProperty("http://www.ccil.org/~cowan/tagsoup/properties/schema", **HtmlParser**.a);

Fuente: este estudio

Cuadro 18. (Continuación) Líneas de código que exponen datos para Udemy

Archivo: org/ccil/cowan/tagsoup/Parser.java:
line 187: if (str.equals("http://www.ccil.org/~cowan/tagsoup/properties/auto-detector")) {
line 217: } else if (!str.equals("http://www.ccil.org/~cowan/tagsoup/properties/auto-detector")) {
Archivo: retrofit2/Response.java:
line 13: return success((**Object**) t, new **Builder**().code(200).message("OK").protocol(**Protocol**.HTTP_1_1).request(new **Request**.**Builder**().url("http://localhost/").build()).build());
line 17: return success((**Object**) t, new **Builder**().code(200).message("OK").protocol(**Protocol**.HTTP_1_1).headers(headers).request(new **Request**.**Builder**().url("http://localhost/").build()).build());
line 28: return error(responseBody, new **Builder**().code(i).message("Response.error()").protocol(**Protocol**.HTTP_1_1).request(new **Request**.**Builder**().url("http://localhost/").build()).build());
Archivo: com/udemy/android/receivers/RedirectURLReceiver.java:
line 10: **String** string = intent.getExtras().getString("redirect_url", "http://www.udemy.com/");
Archivo: com/leanplum/messagetemplates/b.java:
line 340: return new **ActionArgs**().with("Close URL", "http://leanplum:close").with("Open URL", "http://leanplum:loadFinished").with("Action URL", "http://leanplum:runAction").with("Track Action URL", "http://leanplum:runTrackedAction").with("Track URL", "http://leanplum:track").with("HTML Align", "Top").with("HTML Height", **Integer**.valueOf(0));
Archivo: com/leanplum/messagetemplates/WebInterstitialOptions.java:
line 32: return new **ActionArgs**().with("URL", "http://www.example.com").with("Close URL", "http://leanplum:close").with("Has dismiss button", **Boolean**.valueOf(true));
Archivo: com/leanplum/messagetemplates/HTMLTemplate.java:
line 48: **Leanplum**.defineAction(a, 3, new **ActionArgs**().with("Close URL", "http://leanplum:close").with("Open URL", "http://leanplum:loadFinished").with("Action URL", "http://leanplum:runAction").with("Track Action URL", "http://leanplum:runTrackedAction").with("Track URL", "http://leanplum:track").with("HTML Align", "Top").with("HTML Height", **Integer**.valueOf(0)), new **ActionCallback**() {
Archivo: com/leanplum/messagetemplates/e.java:
line 15: private static **String** m = "http://leanplum:close";
Archivo: com/leanplum/messagetemplates/a.java:
line 347: return new **ActionArgs**().with("Close URL", "http://leanplum:close").with("Open URL", "http://leanplum:loadFinished").with("Action URL", "http://leanplum:runAction").with("Track Action URL", "http://leanplum:runTrackedAction").with("Track URL", "http://leanplum:track").with("HTML Align", "Top").with("HTML Height", **Integer**.valueOf(0));
Archivo: com/leanplum/messagetemplates/e.java:
line 19: private static **String** q = "http://leanplum:loadFinished";
Archivo: com/leanplum/messagetemplates/e.java:
line 21: private static **String** s = "http://leanplum:runAction";

Fuente: este estudio

Cuadro 19. (Continuación) Líneas de código que exponen datos para Udemy

Archivo: com/leanplum/messagetemplates/e.java:
line 22: private static **String** t = "http://leanplum:runTrackedAction";
Archivo: com/leanplum/messagetemplates/e.java:
line 20: private static **String** r = "http://leanplum:track";
Archivo: com/leanplum/messagetemplates/e.java:
line 14: private static **String** l = "http://www.example.com";
Archivo: com/leanplum/messagetemplates/f.java:
line 9: **Leanplum**.defineAction(a, 2, new **ActionArgs**().with("URL", "http://www.example.com"), /* anonymous class already generated */);
Archivo: com/leanplum/messagetemplates/MessageTemplates.java:
line 36: **Leanplum**.defineAction("Open URL", 2, new **ActionArgs**().with("URL", "http://www.example.com"), new **ActionCallback**() {
Archivo: com/fasterxml/jackson/databind/ext/DOMDeserializer.java:
line 38: newInstance.setFeature("http://javax.xml.XMLConstants/feature/secure-processing", true);
Archivo: io/branch/referral/UniversalResourceAnalyser.java:
line 33: r2 = "https://cdn.branch.io/sdk/uriskiplist_v#.json";
Archivo: io/branch/referral/PrefHelper.java:
line 24: return "https://api.branch.io/";
Archivo: io/branch/referral/ServerRequestCreateUrl.java:
line 97: stringBuilder.append("https://bnc.lt/a/");
line 111: if (**Branch**.b().a() && !str.contains("https://bnc.lt/a/")) {
Archivo: zendesk/support/SupportSdkSettings.java:
line 66: return (this.mobileSettings == null || !**StringUtils**.hasLength(this.mobileSettings.getReferrerUrl())) ? "https://www.zendesk.com/embeddables" : this.mobileSettings.getReferrerUrl();
Archivo: com/udemy/android/CombinedAppPreferences.java:
line 46: return "https://bnc.lt/get-mobile-app";
Archivo: com/udemy/android/CombinedAppPreferences.java:
line 49: return "https://www.udemy.com";
Archivo: com/udemy/android/util/AppIndexHelper.java:
line 36: str2 = "https://www.udemy.com";
line 45: if (!(course == null || course.getUrl().contains("https://www.udemy.com"))) {
line 49: stringBuilder2.append("https://www.udemy.com");
line 54: stringBuilder2.append("https://www.udemy.com");
Archivo: com/udemy/android/CombinedAppNavigator.java:
line 94: context.startActivity(**Companion**.a(**WebViewActivity**.b, context2, "https://www.udemy.com/teaching/", context.getString(R.string.become_an_instructor), 0, 0, 24, null));

Fuente: este estudio

Cuadro 20. (Continuación) Líneas de código que exponen datos para Udemy

Archivo: com/udemy/android/activity/LoginBaseActivity.java:
line 52: protected final **List**<**String**> d = **Collections**.singletonList("https://www.googleapis.com/auth/userinfo.email");
Archivo: com/udemy/android/experiments/AndroidExperimentSet$instructorCommunityUrl$2.java:
line 13: return "https://www.udemy.com/api-2.0/lithium/community-url?next=/auth/udemysso";
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 42: longSparseArray.b(268, "https://udemy-images.udemy.com/course/480x270/1290678_79a8_2.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 43: longSparseArray.b(269, "https://udemy-images.udemy.com/course/480x270/381850_d819_10.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 44: longSparseArray.b(273, "https://udemy-images.udemy.com/course/480x270/394968_538b_5.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 45: longSparseArray.b(274, "https://udemy-images.udemy.com/course/480x270/356500_a78e_4.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 46: longSparseArray.b(276, "https://udemy-images.udemy.com/course/480x270/208726_966c_4.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 47: longSparseArray.b(277, "https://udemy-images.udemy.com/course/480x270/441920_0fd8_7.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 48: longSparseArray.b(278, "https://udemy-images.udemy.com/course/480x270/238934_4d81_4.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 49: longSparseArray.b(279, "https://udemy-images.udemy.com/course/480x270/1265618_008e_2.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 50: longSparseArray.b(288, "https://udemy-images.udemy.com/course/480x270/364426_2991_5.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 51: longSparseArray.b(290, "https://udemy-images.udemy.com/course/480x270/203556_5ff1_3.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 52: longSparseArray.b(292, "https://udemy-images.udemy.com/course/480x270/9287_d093_15.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 53: longSparseArray.b(294, "https://udemy-images.udemy.com/course/480x270/362328_91f3_10.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 54: longSparseArray.b(296, "https://udemy-images.udemy.com/course/480x270/59527_ac27_15.jpg");

Fuente: este estudio

Cuadro 21.(Continuación) Líneas de código que exponen datos para Udemy

Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 55: longSparseArray.b(298, "https://udemy-images.udemy.com/course/480x270/694190_ecb2_2.jpg");
Archivo: com/udemy/android/dao/model/CourseCategory.java:
line 56: longSparseArray.b(300, "https://udemy-images.udemy.com/course/480x270/116286_a3bd_10.jpg");
Archivo: com/udemy/android/helper/MainAppAccountConfiguration.java:
line 22: return "https://www.udemy.com/terms/?display_type=embed";
line 25: return "https://www.udemy.com/terms/privacy/?display_type=embed";
line 28: return "https://www.udemy.com/terms/copyright/?display_type=embed";
Archivo: com/udemy/android/login/LoginAndCreateAccountFragment.java:
line 109: **Intent** intent = new **Intent**("android.intent.action.VIEW", **Uri**.parse("https://www.udemy.com/terms/?display_type=embed"));
Archivo: com/udemy/android/helper/MainAppAccountConfiguration.java:
line 25: return "https://www.udemy.com/terms/privacy/?display_type=embed";
Archivo: com/udemy/android/helper/MainAppAccountConfiguration.java:
line 28: return "https://www.udemy.com/terms/copyright/?display_type=embed";
Archivo: com/udemy/android/helper/MainAppAccountConfiguration.java:
line 31: return "https://www.udemy.com/user/edit-notifications/?display_type=embed";
Archivo: com/udemy/android/instructor/account/AccountNavigator.java:
line 36: a("https://www.udemy.com/user/edit-notifications/?display_type=embed");
Archivo: com/udemy/android/helper/network/MainAppNetworkConfiguration.java:
line 16: return **Constants**.d ? **UdemyAPIConstants**.a : "https://www.udemy.com/api-2.0/";
Archivo: com/udemy/android/instructor/InstructorNavigator.java:
line 51: h().startActivity(**WebViewActivity**.b.a(h(), "https://www.udemy.com/statements/", h().getString(R.string.revenue_report), R.style.**Instructor_Theme_Toolbar**, R.color.instructor_primary_color));
Archivo: com/udemy/android/instructor/inbox/InstructorOverlayNavigator.java:
line 23: a("https://teach.udemy.com/course-creation/getting-started-as-a-udemy-instructor/");
Archivo: com/udemy/android/instructor/core/api/InstructorS3ApiClient.java:
line 26: **Object** create = **NetworkUtils**.a("https://udemy-images.s3.amazonaws.com/", builder, objectMapper).create(**InstructorS3ApiClient**.class);
Archivo: com/udemy/android/instructor/account/AccountNavigator.java:
line 30: a("https://about.udemy.com/");
Archivo: com/udemy/android/client/UdemyPageEventsAPI.java:
line 25: return (**UdemyPageEventsAPIClient**) **NetworkUtils**.a(**Constants**.d ? "https://page-events-ustats.dev.udemy.com/api-2.0/" : "https://page-events-ustats.udemy.com/api-2.0/", builder, objectMapper).create(**UdemyPageEventsAPIClient**.class);

Fuente: este estudio

Cuadro 22. (Continuación) Líneas de código que exponen datos para Udemy

Archivo: com/leanplum/LeanplumPushService.java:
line 228: ao.a("You are using adaptive icons without having a fallback icon for push notifications on Android Oreo. \nThis can cause a factory reset of the device on Android Version 26. Please add regular icon with name \"leanplum_default_push_icon.png\" to your \"drawable\" folder.\nGoogle issue: https://issuetracker.google.com/issues/68716460");
Archivo: com/appsflyer/AppsFlyerLib.java:
line 715: return "https://api.appsflyer.com/install_data/v3/";
Archivo: com/appsflyer/AppsFlyerLib.java:
line 740: stringBuilder.append("https://t.appsflyer.com/api/v");
Archivo: com/appsflyer/AppsFlyerLib.java:
line 747: stringBuilder.append("https://events.appsflyer.com/api/v");
Archivo: com/appsflyer/AppsFlyerLib.java:
line 754: stringBuilder.append("https://register.appsflyer.com/api/v");
Archivo: com/appsflyer/AppsFlyerLib.java:
line 996: backgroundHttpTask.execute(new **String**[]{"https://stats.appsflyer.com/stats"});
Archivo: com/udemy/android/instructor/core/deeplink/InstructorDeepLinks.java:
line 26: return **StringsKt__StringsJVMKt**.a(str, "udemy://instructor", false, 2, null);
Archivo: com/udemy/android/instructor/core/deeplink/AppDeepLink.java:
line 6: @DeepLinkSpec(prefix = {"udemy://instructor"})
Archivo: com/udemy/android/instructor/core/deeplink/InstructorDeepLinkModuleLoader.java:
line 9: public static final **List**<**DeepLinkEntry**> a = **Collections**.unmodifiableList(**Arrays**.asList(new **DeepLinkEntry**[]{new **DeepLinkEntry**("udemy://instructor/courseDiscussionDetails", **Type**.METHOD, **InstructorDeepLinks**.class, "qaMessage"), new **DeepLinkEntry**("udemy://instructor/courseRevenue", **Type**.METHOD, **InstructorDeepLinks**.class, "insightCourse"), new **DeepLinkEntry**("udemy://instructor/inboxAll", **Type**.METHOD, **InstructorDeepLinks**.class, "inbox"), new **DeepLinkEntry**("udemy://instructor/inboxCourseDiscussion", **Type**.METHOD, **InstructorDeepLinks**.class, "inbox"), new **DeepLinkEntry**("udemy://instructor/inboxMessages", **Type**.METHOD, **InstructorDeepLinks**.class, "inbox"), new **DeepLinkEntry**("udemy://instructor/messageDetails", **Type**.METHOD, **InstructorDeepLinks**.class, "directMessage"), new **DeepLinkEntry**("udemy://instructor/newEnrollments", **Type**.METHOD, **InstructorDeepLinks**.class, "insightOverview"), new **DeepLinkEntry**("udemy://instructor/revenue", **Type**.METHOD, **InstructorDeepLinks**.class, "insightOverview"), new **DeepLinkEntry**("udemy://instructor/reviewDetails", **Type**.METHOD, **InstructorDeepLinks**.class, "reviewDetails"), new **DeepLinkEntry**("udemy://instructor/reviews", **Type**.METHOD, **InstructorDeepLinks**.class, "reviews")}));

Fuente: este estudio

El siguiente cuadro muestra información que puede ser significar un riesgo de seguridad.

Cuadro 23. Exposición de datos sensibles.

Exposición a la información
Archivo: com/udemy/android/client/UdemyPageEventsAPI.java:
line 25: return (**UdemyPageEventsAPIClient**) **NetworkUtils**.a(**Constants**.d ? "https://page-events-ustats.dev.udemy.com/api-2.0/" : "https://page-events-ustats.udemy.com/api-2.0/", builder, objectMapper).create(**UdemyPageEventsAPIClient**.class);
hostnames with value dev.leanplum.com in following files:
Archivo: com/leanplum/a/h.java:
line 7: public static **String** d = "dev.leanplum.com";

Fuente: este estudio

EL siguiente cuadro muestra habilitado la copia de seguridad externa lo que puede significar divulgación de contenido sensible. Lo correcto sería estar en false.

Cuadro 24. Propiedad de Backup habilitado.

android/AndroidManifest.xml:
line 22: <application android:allowBackup="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:name="com.udemy.android.CombinedApplication" android:networkSecurityConfig="@xml/network_security_config" android:supportsRtl="true" android:theme="@style/UdemyTheme">

Fuente: este estudio

4. Análisis estático M3

El siguiente cuadro muestra las configuraciones que permitirían conexiones tipo http

Cuadro 25. Configuraciones que muestran posibles conexiones con protocolo sin cifrar

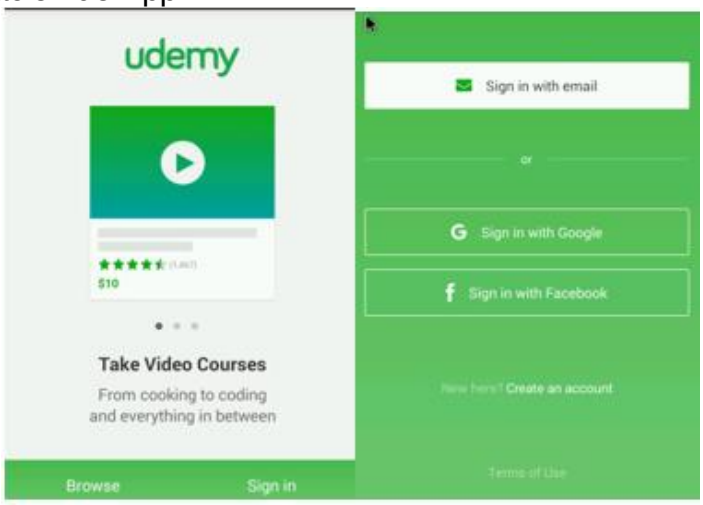
Uso de protocolo HTTP
io/branch/referral/BranchViewHandler.java:
line 111: try {
line 112: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(this.b.e).openConnection()));
line 113: httpURLConnection.setRequestMethod(**HttpGetHC4**.METHOD_NAME);
There is **'(HttpURLConnection)'** found in file 'com/leanplum/a/bo.java':
line 563: static **HttpURLConnection** a(**String** str, **String** str2, boolean z, int i) throws **IOException** {
line 564: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(str).openConnection()));
line 565: if (z) {
line 861: public static **JSONObject** a(**HttpURLConnection** httpURLConnection) throws **JSONException**, **IOException** {
line 862: **String** b = b(httpURLConnection);
line 863: if (h.o && h.l) {
com/pushio/manager/PIOAPIConnectorService.java:
line 53: try {
line 54: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(str).openConnection()));
line 55: httpURLConnection.setRequestMethod(str2);
line 68: }
line 69: a(httpURLConnection);
line 70: } catch (**IOException** e) {
com/pushio/manager/tasks/PushIOGetImageTask.java:
line 27: try {
line 28: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(strArr[0]).openConnection()));
line 29: httpURLConnection.setDoInput(true);
com/bumptech/glide/load/data/HttpUrlFetcher.java:
line 35: public **HttpURLConnection** a(URL url) throws **IOException** {
line 36: return (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(url.openConnection()));
line 37: }

Fuente: este estudio

5. Análisis dinámico M2

Para este tipo de análisis se revisará las bases de datos que genera cada aplicación y si es legible ante herramientas de lectura de bases de datos tipo SQLite. Primero se instala la aplicación como se detalla a continuación.

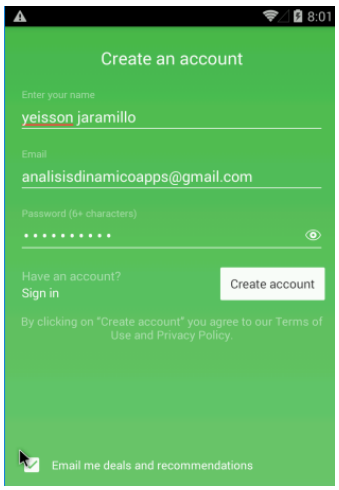
Figura 79. Instalación de App



Fuente: este estudio

A continuación, se genera un usuario para revisar él envió de datos

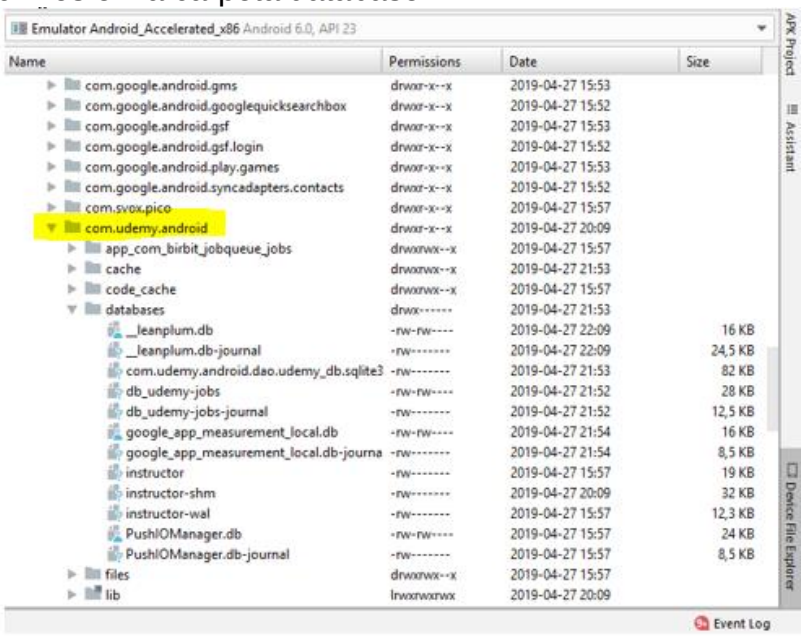
Figura 80. Generación de usuario



Fuente: este estudio

A continuación, se revisa las bases de datos sql que genera la aplicación

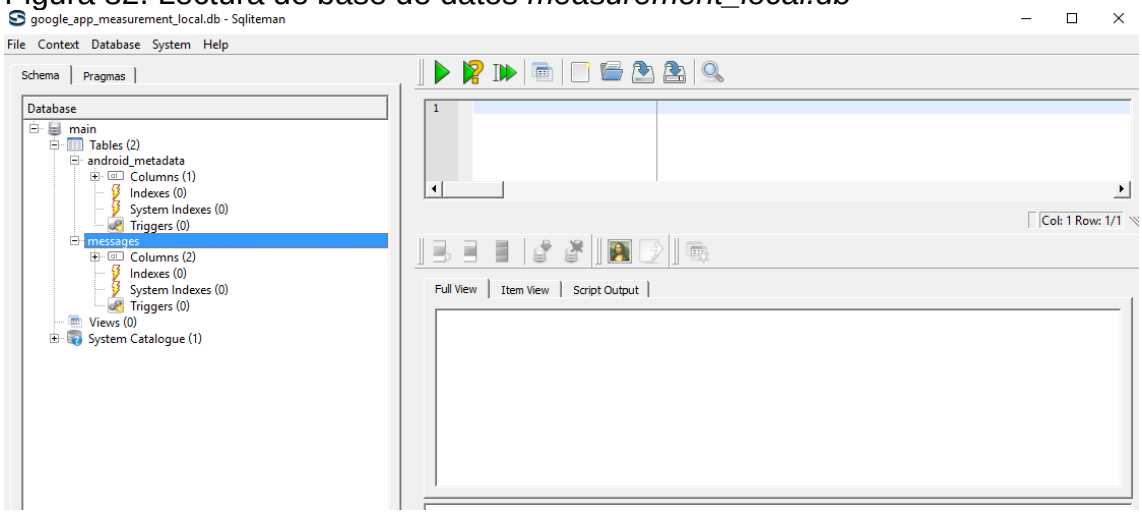
Figura 81. Archivos en la carpeta database.



Fuente: este estudio

Se realiza la revisión de los archivos con extensión *measurement_local*.db y no se visualizan datos sensibles.

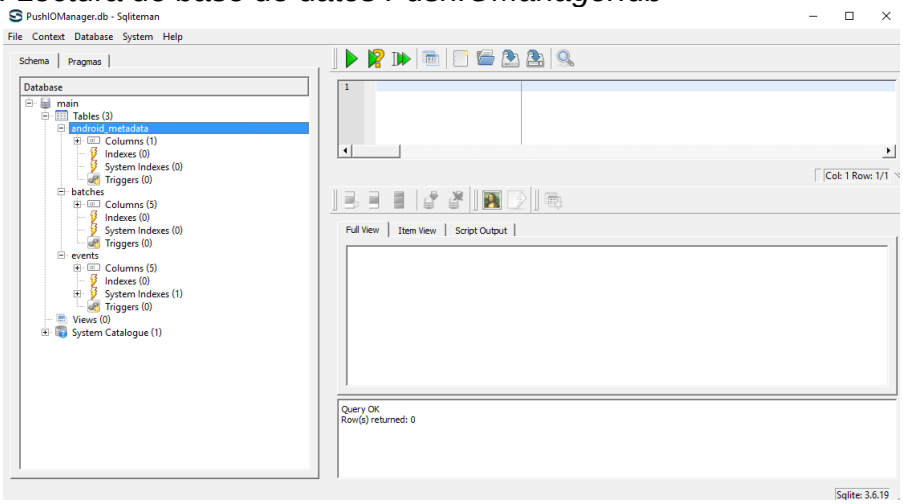
Figura 82. Lectura de base de datos *measurement_local.db*



Fuente: este estudio

Se realiza la revisión de los archivos con extensión *PushIOManager.db* y no se visualizan datos sensibles.

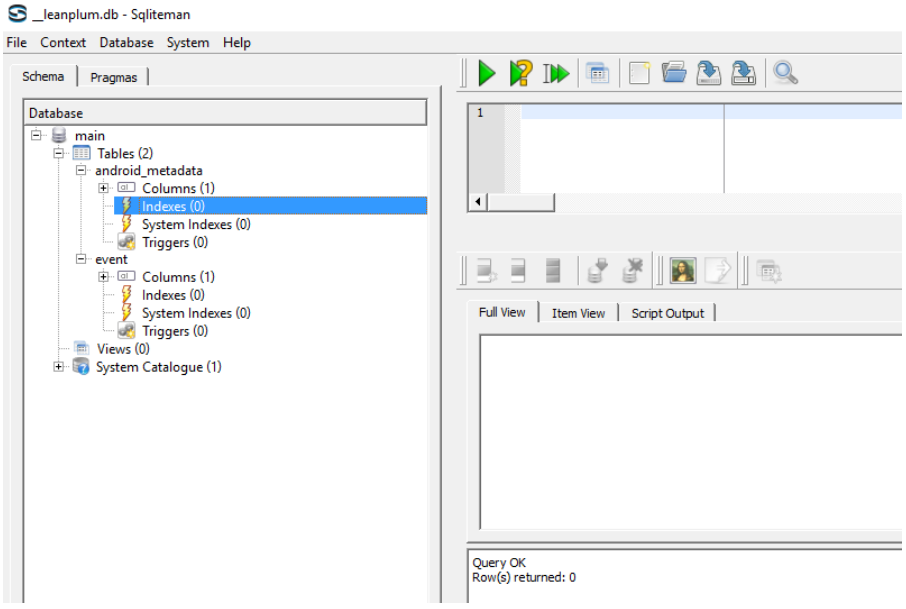
Figura 83. Lectura de base de datos *PushIOManager.db*



Fuente: este estudio

Se realiza la revisión de los archivos con extensión *leanplum.db* y no se visualizan datos sensibles.

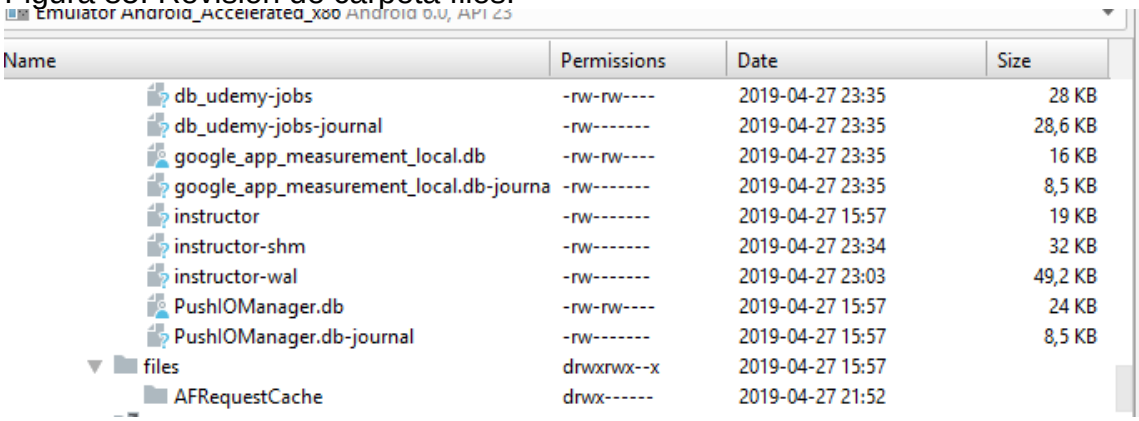
Figura 84. Lectura de base de datos *__leanplum.db*



Fuente: este estudio

Se revisa de igual forma la carpeta Files en busca de documentos que puedan guardar información sensible pero no se halló información sensible.

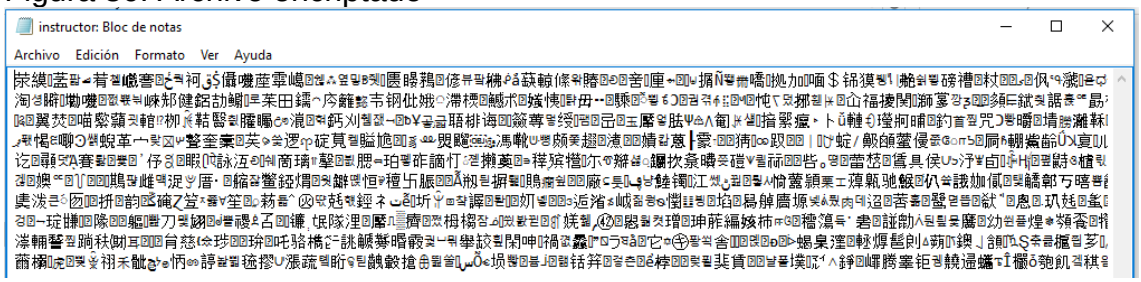
Figura 85. Revisión de carpeta files.



Fuente: este estudio

A continuación, se observa un archivo encriptado con información de los instructores cumpliendo con políticas de seguridad de cifrado

Figura 86. Archivo encriptado

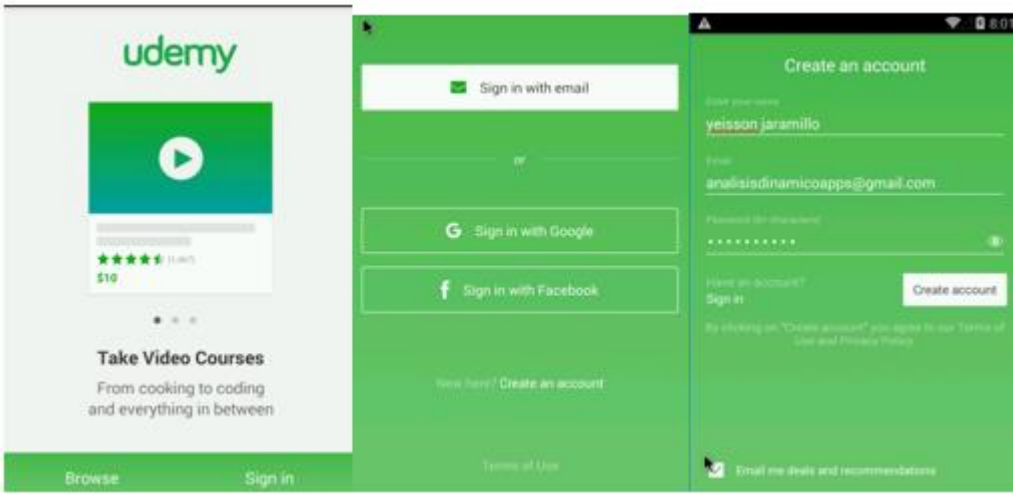


Fuente: este estudio

6. Análisis dinámico M3

Para el análisis dinámico se procede a instalar la App en cualquier emulador en este caso se usó el emulador de MobSF con la idea de analizar el comportamiento. A continuación, formulario inicial e inicio de sesion de la App

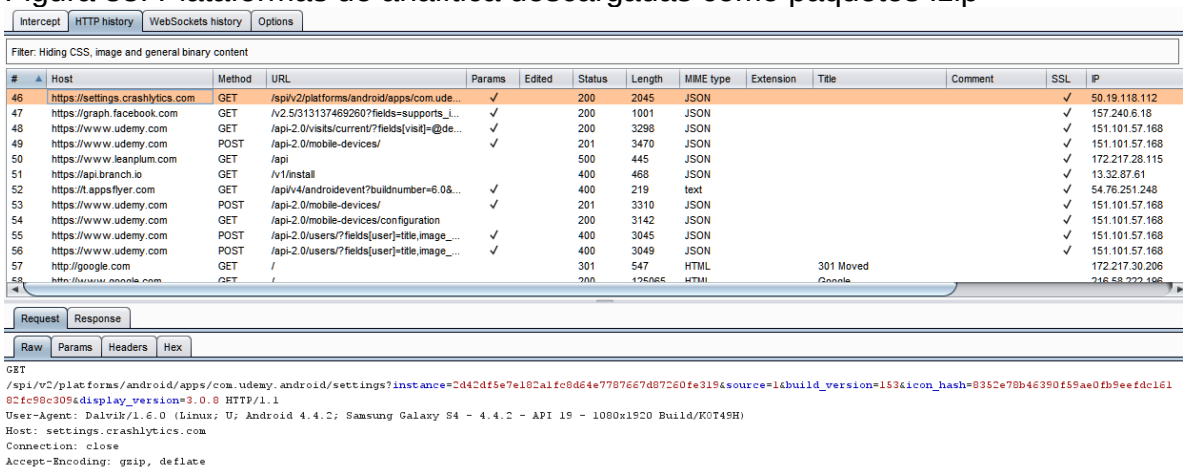
Figura 87. Inicialización de la App y creación de usuario.



Fuente: este estudio

Inicialmente se observa la ejecución de varias plataformas que son utilizadas para capturar datos y patrones de usuario. Esto puede significar violación de la privacidad

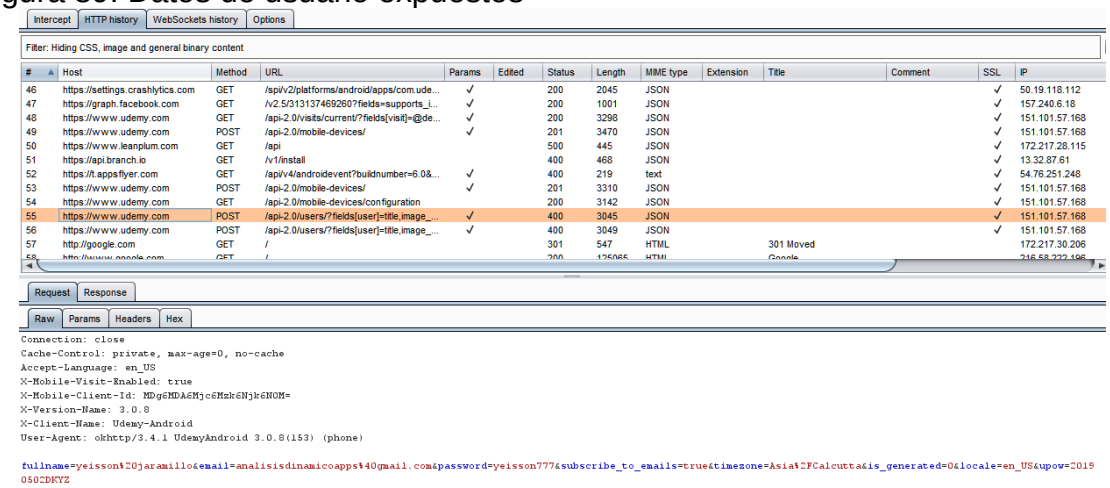
Figura 88. Plataformas de analítica descargadas como paquetes .zip



Fuente: este estudio

Luego cuando se inicia sesión se puede ver como la plataforma no encripta la información enviada a los servidores. A continuación, los datos de usuario visibles desde la herramienta Burn

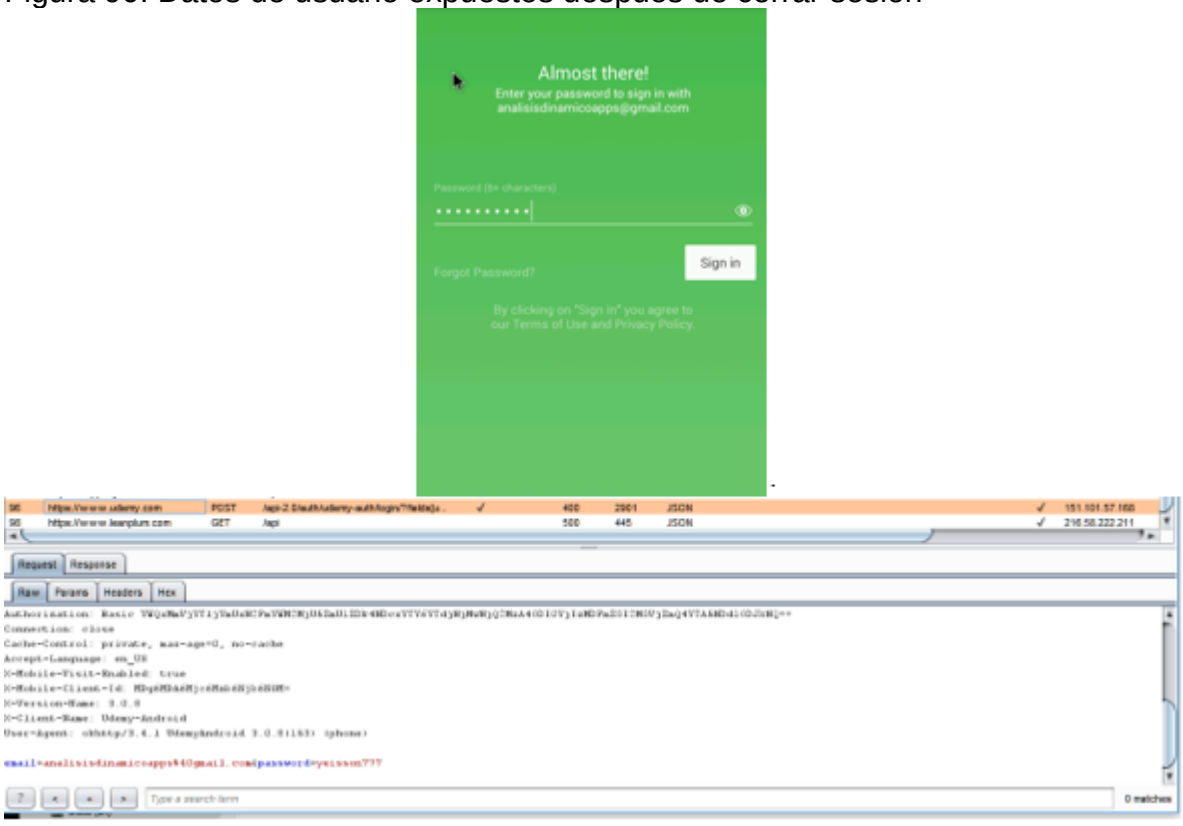
Figura 89. Datos de usuario expuestos



Fuente: este estudio

También se cierra sesión y al tratar de ingresar de nuevo hay exposición de datos.

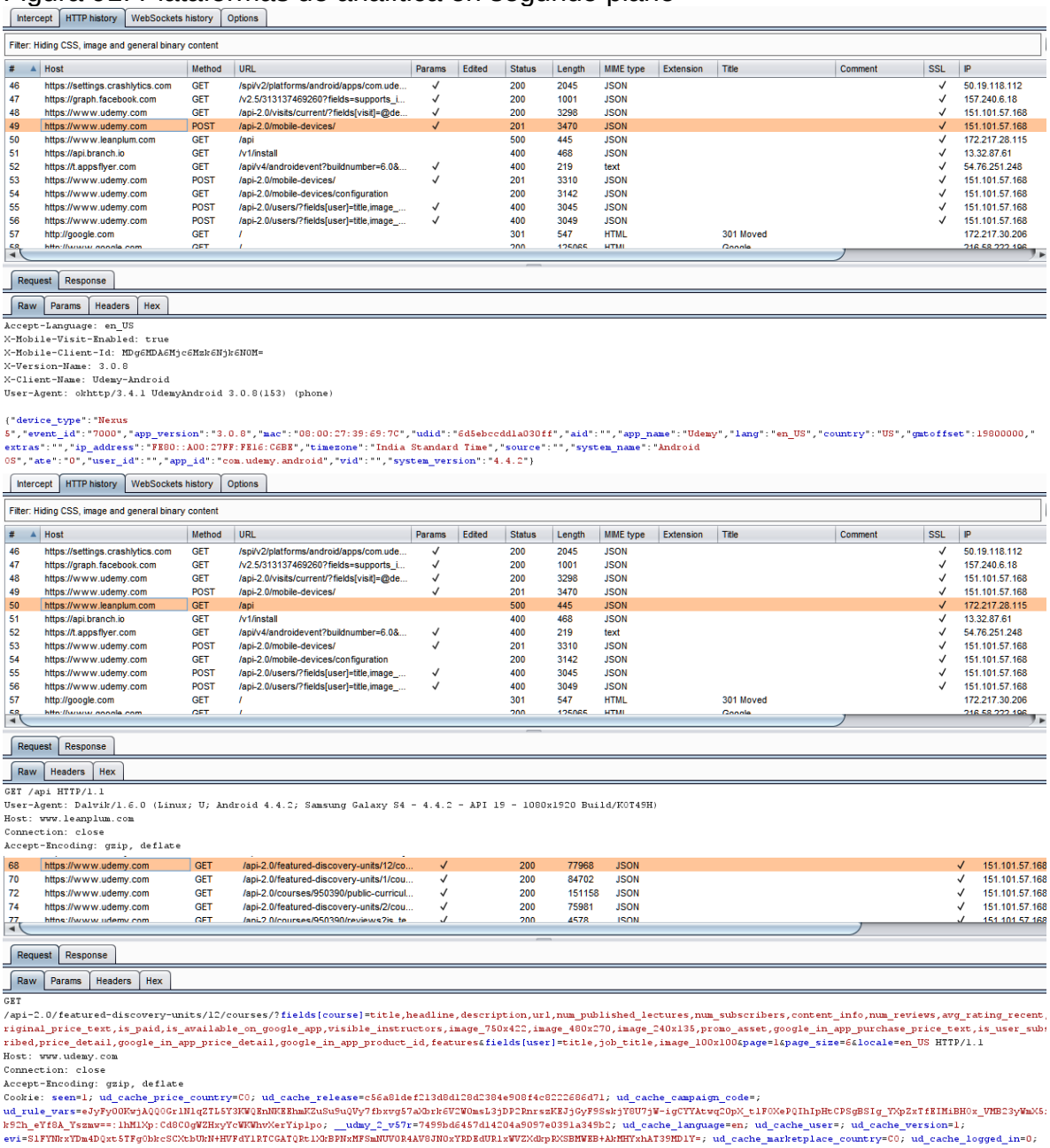
Figura 90. Datos de usuario expuestos después de cerrar sesión



Fuente: este estudio

Por último, se observa varias plataformas de análisis de patrones de usuario ejecutándose en segundo plano de forma continua.

Figura 91. Plataformas de analítica en segundo plano

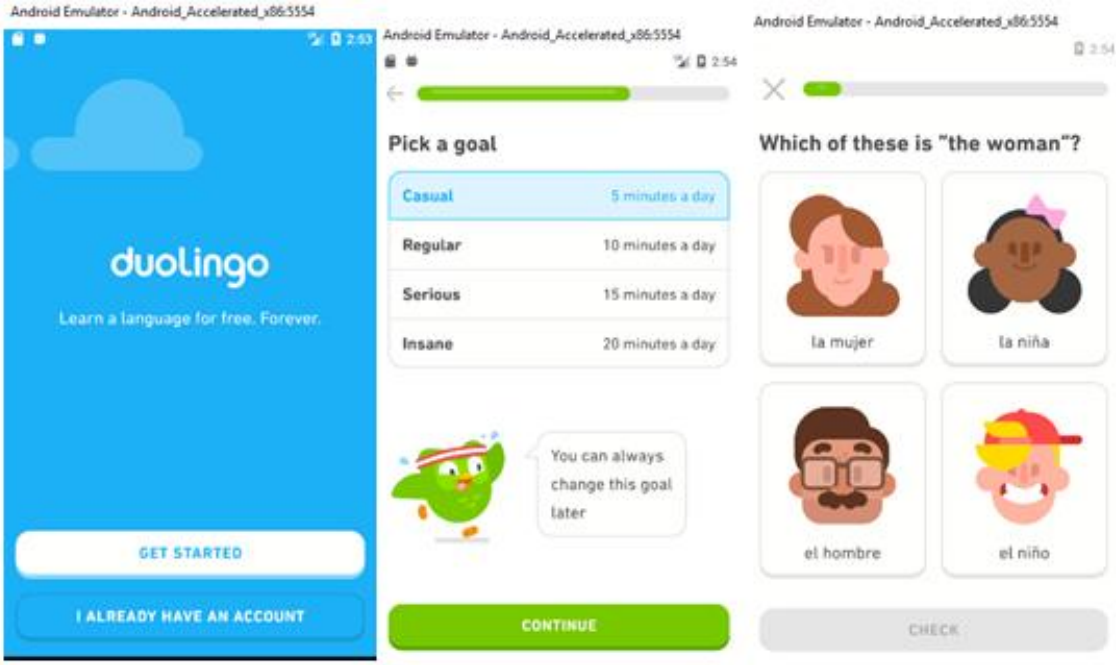


Fuente: este estudio

IV.Duolingo - idiomas gratis

Funcionalidad: Esta es una App muy famosa de aprendiza de idiomas. Su interfaz es amigable y de fácil uso además de dar lecciones gratis. Hay una sección de pagos, pero sus ganancias se basan en la publicidad.

Figura 92. Imagen de la funcionalidad Duolingo en la App store

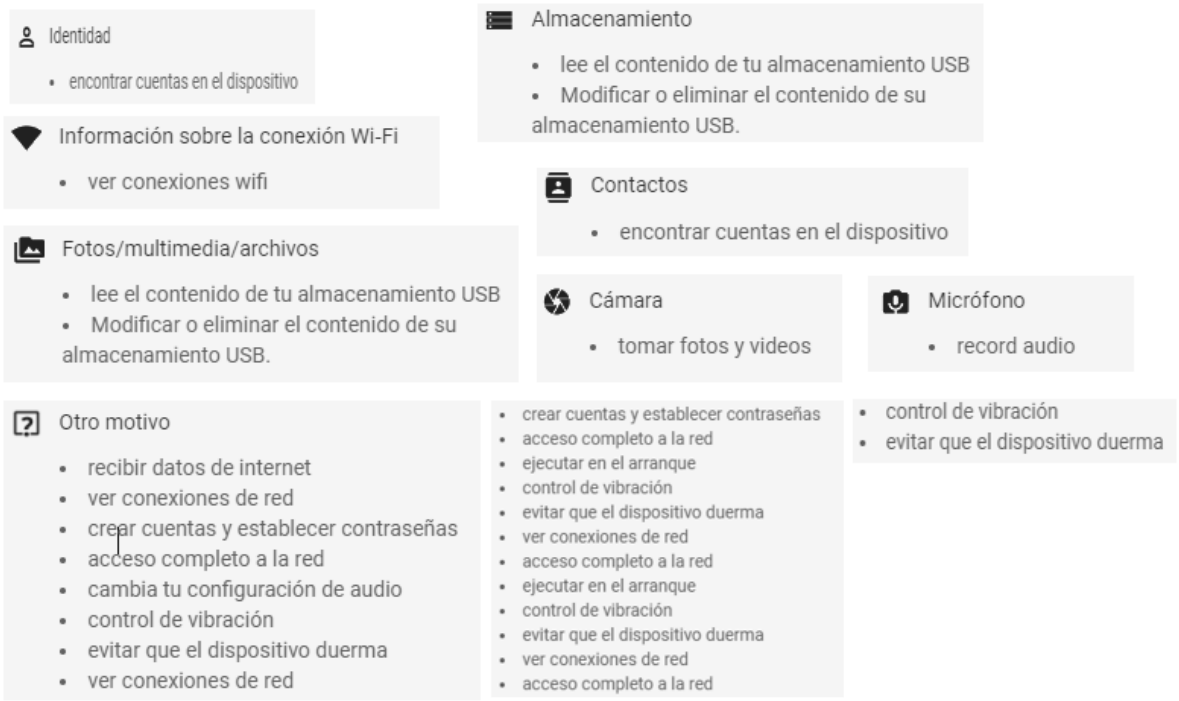


Fuente: este estudio

1. Permisos solicitados por la aplicación a nivel de instalación

A continuación, se muestra los permisos que solicita la App al ser instalada.

Figura 93. Permisos solicitados por la aplicación Duolingo



Fuente: este estudio

2. Permisos archivo Android Manifest.xml

A continuación, se muestra los permisos que se establecen en el archivo de configuración.

Figura 94. Archivo manifest.xml parcial

```
<?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="711" android:versionName="4.6.3" android:installLocation="auto" android:compileSdkVersion="28" android:compileSdk
VersionCodename="9" package="com.duolingo" platformBuildVersionCode="711" platformBuildVersionName="4.6.3"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="28" />
    <uses-feature android:name="android.hardware.touchscreen" android:required="false" />
    <uses-feature android:name="android.hardware.camera" android:required="false" />
    <uses-feature android:name="android.hardware.camera.autofocus" android:required="false" />
    <uses-feature android:name="android.hardware.microphone" android:required="false" />
    <uses-permission android:name="android.permission.AUTHENTICATE_ACCOUNTS" android:maxSdkVersion="22" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission-sdk-23 android:name="android.permission.CAMERA" />
    <permission android:name="com.duolingo.permission.C2D_MESSAGE" android:protectionLevel="signature" />
```

Fuente: este estudio

Figura 95. Permisos detectados por MobSF App Duolingo

PERMISSIONS

PERMISSION	STATUS	INFO	DESCRIPTION
android.permission.AUTHENTICATE_ACCOUNTS	dangerous	act as an account authenticator	Allows an application to use the account authenticator capabilities of the Account Manager, including creating accounts as well as obtaining and setting their passwords.
android.permission.INTERNET	dangerous	full Internet access	Allows an application to create network sockets.
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.
android.permission.FOREGROUND_SERVICE	normal		Allows a regular application to use
android.permission.VIBRATE	normal	control vibrator	Allows the application to control the vibrator.
android.permission.WAKE_LOCK	dangerous	prevent phone from sleeping	Allows an application to prevent the phone from going to sleep.
android.permission.GET_ACCOUNTS	normal	discover known accounts	Allows an application to access the list of accounts known by the phone.
android.permission.RECORD_AUDIO	dangerous	record audio	Allows application to access the audio record path.
android.permission.WRITE_EXTERNAL_STORAGE	dangerous	read/modify/delete SD card contents	Allows an application to write to the SD card.
com.duolingo.permission.C2D_MESSAGE	signature	Allows cloud to device messaging	Allows the application to receive push notifications.
com.google.android.c2dm.permission.RECEIVE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.android.vending.BILLING	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.READ_EXTERNAL_STORAGE	dangerous	read SD card contents	Allows an application to read from SD Card.
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	dangerous	Unknown permission from android reference	Unknown permission from android reference
android.permission.MODIFY_AUDIO_SETTINGS	dangerous	change your audio settings	Allows application to modify global audio settings, such as volume and routing.
android.permission.ACCESS_WIFI_STATE	normal	view Wi-Fi status	Allows an application to view the information about the status of Wi-Fi.
com.sec.android.provider.badge.permission.READ	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.sec.android.provider.badge.permission.WRITE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.htc.launcher.permission.READ_SETTINGS	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.htc.launcher.permission.UPDATE_SHORTCUT	dangerous	Unknown permission from android reference	Unknown permission from android reference

PERMISSION	STATUS	INFO	DESCRIPTION
com.sonyericsson.home.permission.BROADCAST_BADGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.sonymobile.home.permission.PROVIDER_INSERT_BADGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.anddoes.launcher.permission.UPDATE_COUNT	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.majeur.launcher.permission.UPDATE_BADGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.huawei.android.launcher.permission.CHANGE_BADGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.huawei.android.launcher.permission.READ_SETTINGS	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.huawei.android.launcher.permission.WRITE_SETTINGS	dangerous	modify global system settings	Allows an application to modify the system's settings data. Malicious applications can corrupt your system's configuration.
android.permission.READ_APP_BADGE	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.oppo.launcher.permission.READ_SETTINGS	dangerous	Unknown permission from android reference	Unknown permission from android reference
com.oppo.launcher.permission.WRITE_SETTINGS	dangerous	modify global system settings	Allows an application to modify the system's settings data. Malicious applications can corrupt your system's configuration.
me.everything.badger.permission.BADGE_COUNT_READ	dangerous	Unknown permission from android reference	Unknown permission from android reference
me.everything.badger.permission.BADGE_COUNT_WRITE	dangerous	Unknown permission from android reference	Unknown permission from android reference

Fuente: este estudio

3. Análisis estático M2

El siguiente cuadro muestra como hay datos quemados en el código que pueden servir para futuros ataques

Cuadro 26.Información de tablas y sentencias tipo SQL

Base de datos SQLITE tipo texto sin cifrar
Archivo: google_conversion_tracking.db:
TABLES: android_metadata conversiontracking sqlite_sequence RAW DUMP: CREATE TABLE android_metadata (locale TEXT);CREATE TABLE conversiontracking (conversion_ping_id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, string_url TEXT NOT NULL, preference_key TEXT, is_repeatable INTEGER, parameter_is_null INTEGER preference_name TEXT, record_time INTEGER, retry_count INTEGER,last_retry_time INTEGER);CREATE TABLE sqlite_sequence(name,seq);
Archivo: Web Data
TABLES: meta sqlite_autoindex_meta_1 autofill sqlite_autoindex_autofill_1 autofill_name autofill_name_value_lower credit_cards sqlite_autoindex_credit_cards_1 autofill_profiles sqlite_autoindex_autofill_profiles_1 autofill_profile_names autofill_profile_emails autofill_profile_phones autofill_profiles_trash masked_credit_cards unmasked_credit_cards server_card_metadata server_addresses server_address_metadata autofill_sync_metadata sqlite_autoindex_autofill_sync_metadata_1 autofill_model_type_state

Fuente: este estudio

Cuadro 27. Líneas de código que exponen datos para Duolingo

Exposición a la información
com/duolingo/DuoApp.java:
line 537: r1 = "https://social-test-bddb4.firebaseio.com";
com/duolingo/app/ai.java:
line 265: **Zendesk**.INSTANCE.init(**DuoApp**.a(), "https://duolingotest.zendesk.com", "db861434db5cae7a18adfd2936b0d4c58666797b123bc855", "mobile_sdk_client_e51e8c3d953d55ef0f5c");
com/duolingo/d/a/b.java:
line 189: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) new URL(this.h ? "https://excess-dev.duolingo.com/batch" : "https://excess.duolingo.com/batch").openConnection();

Fuente: este estudio

El siguiente cuadro muestra como hay datos quemados en el código que pueden servir para futuros ataques

Cuadro 28. Líneas de código que exponen datos para Duolingo

Datos codificados
zendesk/support/SupportSdkSettings.java:
line 66: return (this.mobileSettings == null || !d.a(this.mobileSettings.getReferrerUrl())) ? "https://www.zendesk.com/embeddables" : this.mobileSettings.getReferrerUrl();
com/duolingo/DuoApp.java:
line 537: r1 = "https://social-test-bddb4.firebaseio.com";
com/duolingo/ads/PodcastPromoActivity.java:
line 32: this.a.startActivity(new **Intent**("android.intent.action.VIEW", **Uri**.parse("https://podcast.duolingo.com/")));
com/duolingo/app/ai.java:
line 265: **Zendesk**.INSTANCE.init(**DuoApp**.a(), "https://duolingotest.zendesk.com", "db861434db5cae7a18adfd2936b0d4c58666797b123bc855", "mobile_sdk_client_e51e8c3d953d55ef0f5c");
com/duolingo/networking/ApiOrigin.java:
line 4: API("https://android-api.duolingo.com"),
com/duolingo/grade/network/a.java:
line 21: z.a(this.a, b.a("https://d3kwyfyztuo0xs.cloudfront.net/config/latest/0.9.3", **Method**.GET), false).a(new f() {
com/duolingo/grade/a/b.java:
line 147: return new **Config**(0, new **HashMap**(), new **UrlGeneration**(new **Test**[0], "https://d3kwyfyztuo0xs.cloudfront.net/{language_id}/{grading_data_version}/{id}"), new **HashMap**());

Fuente: este estudio

Cuadro 29. (Continuación) Líneas de código que exponen datos para Duolingo

com/duolingo/v2/a/b.java:
line 113: if (kotlin.b.b.j.a((**Object**) str, (**Object**) "https://social.duolingo.com")) {
line 233: return a.a(d, "https://social.duolingo.com");
com/duolingo/v2/request/c.java:
line 16: private final **String** f = "https://social.duolingo.com";
com/duolingo/v2/request/d.java:
line 30: this.a = "https://explanations.duolingo.com";
com/duolingo/v2/request/e.java:
line 13: private final **String** f = "https://duolingo-leaderboards-prod.duolingo.com";
com/duolingo/v2/request/j.java:
line 32: this.f = "https://athena.duolingo.com";
com/duolingo/d/a/b.java:
line 189: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) new URL(this.h ? "https://excess-dev.duolingo.com/batch" : "https://excess.duolingo.com/batch").openConnection();
com/duolingo/app/DebugActivity.java:
line 1856: **StringBuilder** stringBuilder = new **StringBuilder**("http://next-");
com/duolingo/networking/ApiOrigin.java:
line 5: CN("http://api.duolingo.cn"),
com/duolingo/networking/ApiOrigin.java:
line 6: AVD("http://10.0.2.2:8080");
com/duolingo/model/VersionInfo.java:
line 19: private static final **String** DEFAULT_DICT_BASE_URL = "http://d.duolingo.com/";
com/duolingo/model/VersionInfo.java:
line 25: private static final **String** DEFAULT_TTS_BASE_URL = "http://t.duolingo.com/";
com/duolingo/model/VersionInfo.java:
line 26: private static final **String** DEFAULT_TTS_CDN_URL = "http://static.duolingo.com/";
com/duolingo/app/DeepLinkHandler.java:
line 934: r1 = "duolingo://switch_course/?ui_language=";
com/duolingo/app/DeepLinkHandler.java:
line 965: r0 = "duolingo://clubs";
line 967: r1 = "Uri.parse(\"duolingo://clubs\")";
com/duolingo/app/DeepLinkHandler.java:
line 984: r0 = "duolingo://premium";
line 986: r1 = "Uri.parse(\"duolingo://premium\")";
com/duolingo/app/DeepLinkHandler.java:
line 1001: r1 = "duolingo://profile?user_id=";
com/duolingo/e/e.java:
line 168: **StringBuilder** stringBuilder = new **StringBuilder**("asset:///");

Fuente: este estudio

4. Análisis estático M3

El siguiente cuadro muestra las configuraciones que permitirían conexiones tipo http

Cuadro 30. Configuraciones que muestran posibles conexiones con protocolo sin cifrar

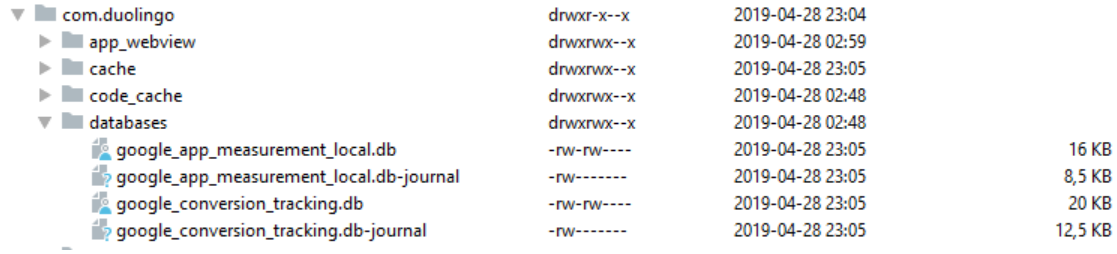
Uso de protocolo HTTP
com/vungle/publisher/net/http/HttpURLConnectionFactory.java
line 14: public static **HttpURLConnection** a(**String** str) {
line 15: return (**HttpURLConnection**) new URL(str).openConnection();
line 16: }

Fuente: este estudio

5. Análisis dinámico M2

Para este tipo de análisis se revisará las bases de datos que genera cada aplicación y si es legible ante herramientas de lectura de bases de datos tipo SQLite. Luego de instalar la aplicación y ejecutarla iniciando sesión se detectó la creación de varios archivos de tipo SQLliter. A continuación, los archivos de datos que genera la instalación de la App.

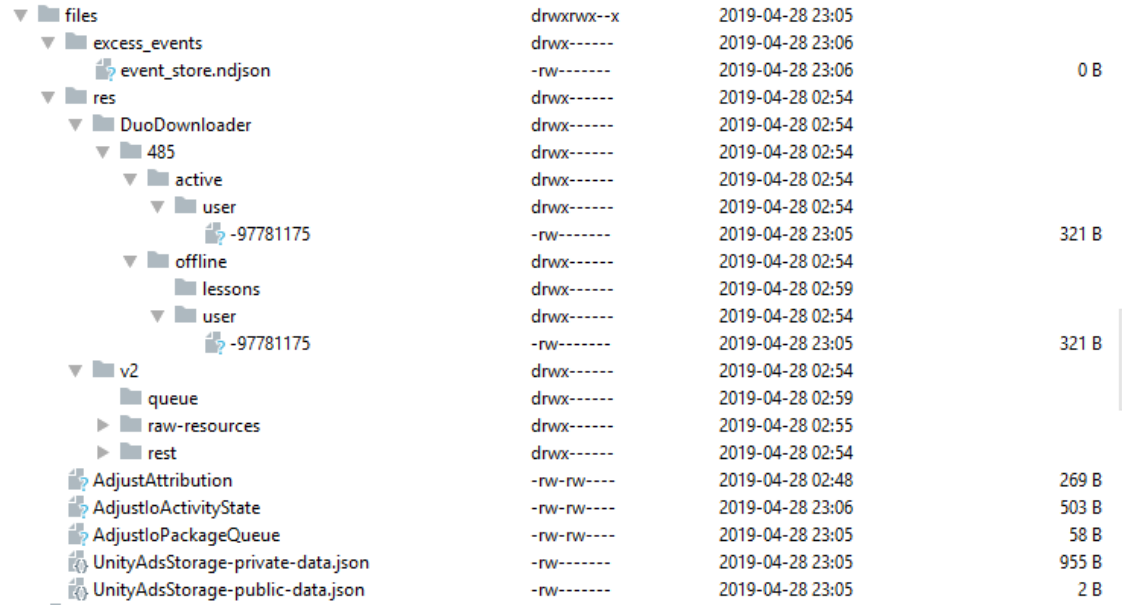
Figura 96. Archivos en la carpeta database.



Fuente: este estudio

A continuación, archivos de configuración que genera la aplicación

Figura 97. Archivos con datos encriptados



Fuente: este estudio

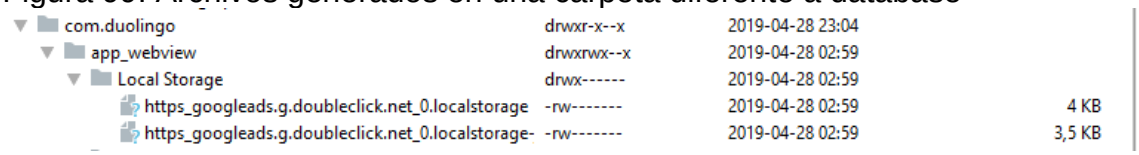
A continuación, se relaciona archivos, pero no se encuentra información sensible.

Figura 98. Archivos y visualización de información no sensible.

Fuente: este estudio

A continuación, se analizan los archivos de la carpeta *Local storage*

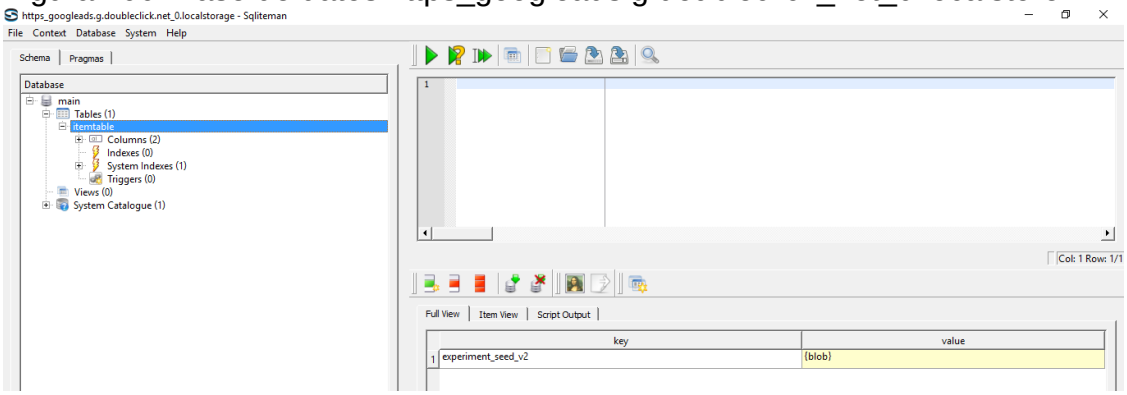
Figura 99. Archivos generados en una carpeta diferente a database



Fuente: este estudio

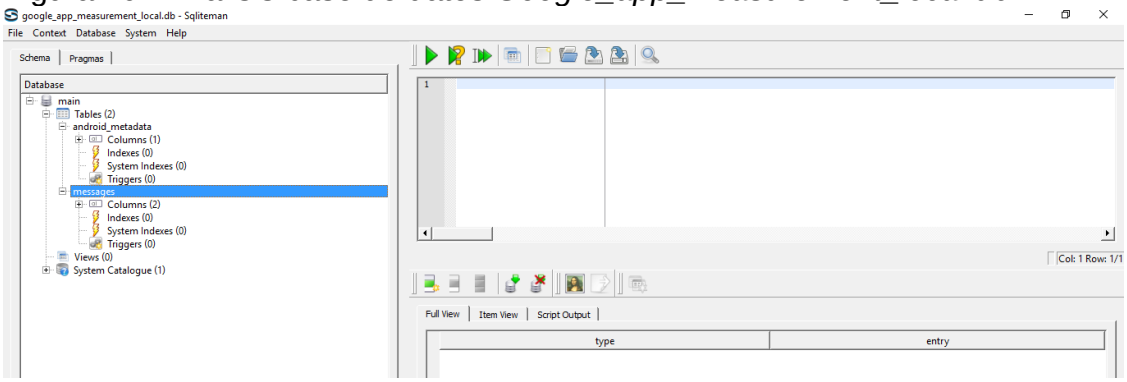
A continuación, se relacionan las bases de datos encontradas y visualizadas con la herramienta SQLiteman.

Figura 100. Base de datos https_googleads.g.doubleclick_net_0.localstore



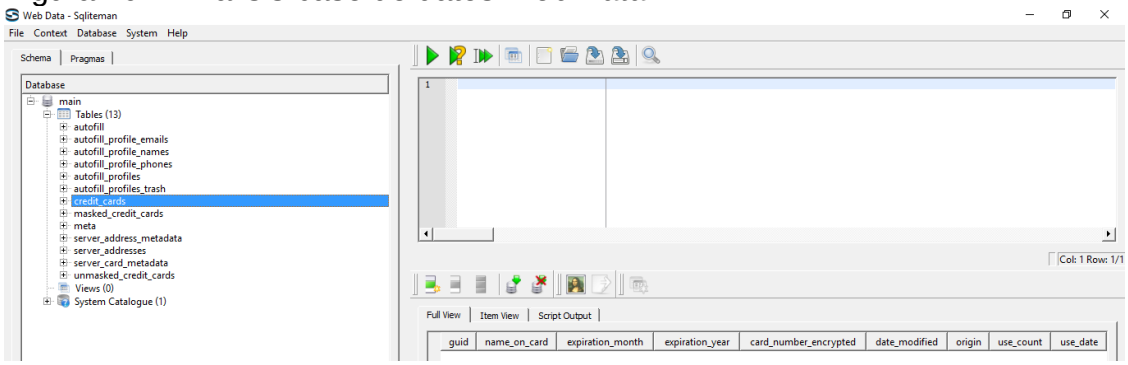
Fuente: este estudio

Figura 101 Análisis base de datos *Google_app_measurement_local.db*



Fuente: este estudio

Figura 102. Análisis base de datos *Web Data*



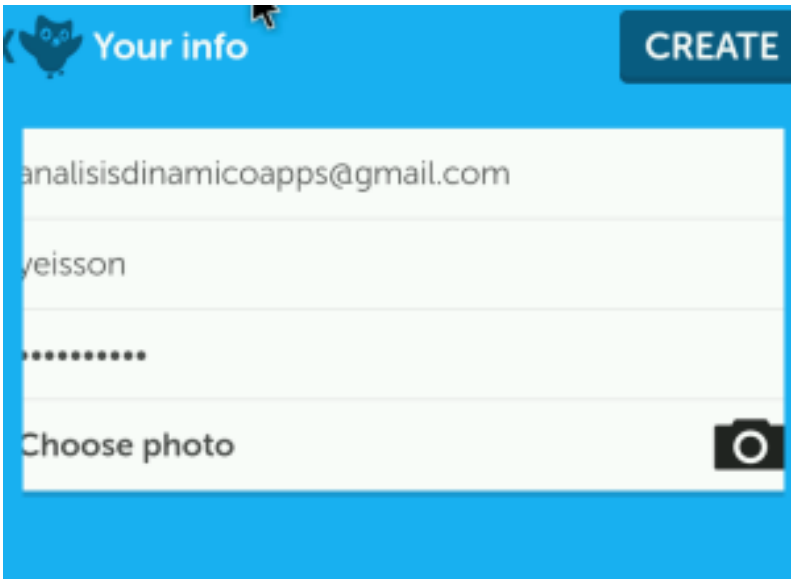
Fuente: este estudio

Se determina que no hay datos sensibles almacenados en el dispositivo para el análisis de riesgos tipo M2.

6. Análisis dinámico M3

Para el análisis dinámico se procede a instalar la app en cualquier emulador en este caso se usó el emulador de MobSF con la idea de analizar el comportamiento. A continuación, creación de cuenta en la plataforma.

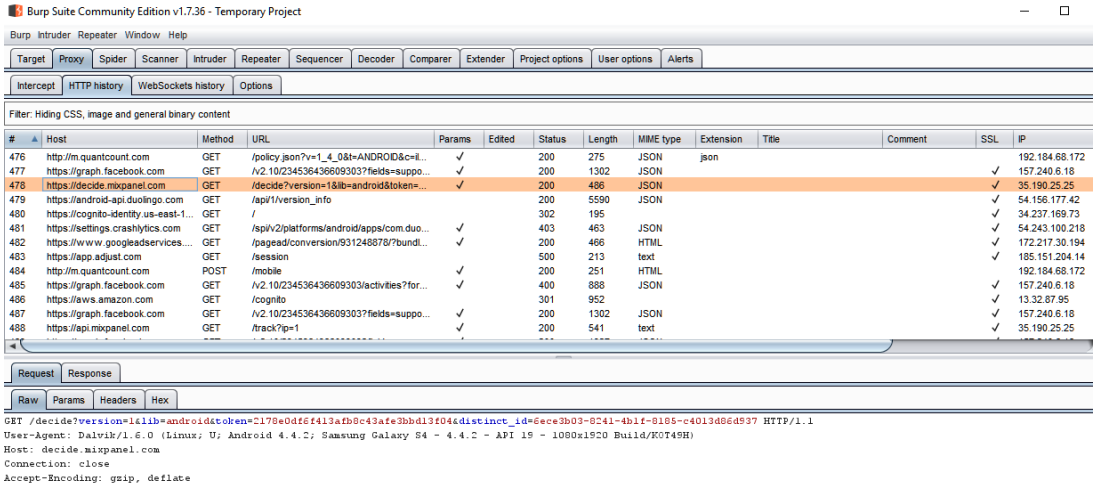
Figura 103. Creación de cuenta de usuario



Fuente: este estudio

Se observa en la inicialización de la App múltiples plataformas de captura de patrones de información de usuario. Acciones que pueden traducirse en violación de privacidad.

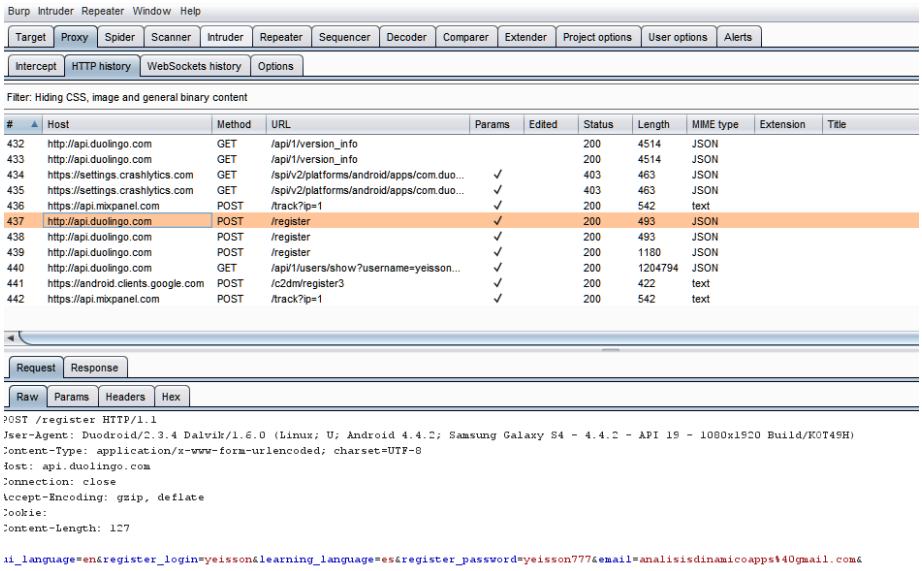
Figura 104. Múltiples plataformas descargan paquetes .zip



Fuente: este estudio

En el envío de información se observa los datos de usuario legibles, no se observa técnicas de cifrado.

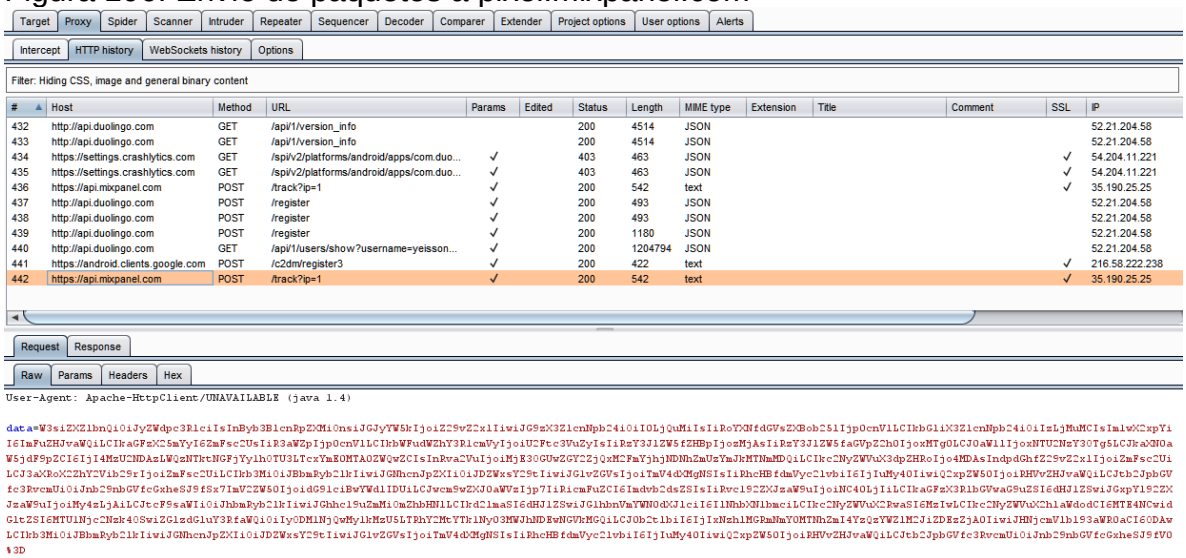
Figura 105. Se observa los datos de usuarios sin cifrado



Fuente: este estudio

Adicional se observa envío de datos en paquetes encriptados a páginas de analítica.

Figura 106. Envío de paquetes a pixel.mixpanel.com



Fuente: este estudio

V.Wiki

Funcionalidad: Esta App permite visualizar contenido multimedia tipo serias, dramas, o programas de espectáculos. También tiene la posibilidad de ver subtítulos lo que amplía la plataforma y rompe con la barrera del idioma. Su propósito es el entretenimiento. A continuación, una imagen de la App.

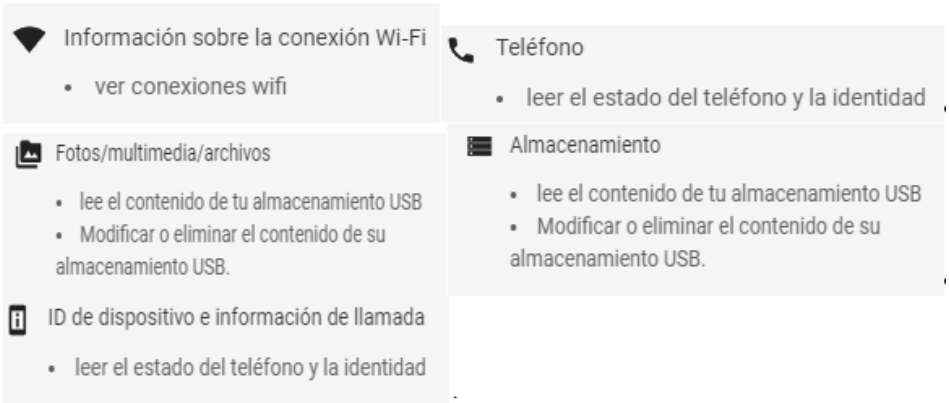
Figura 107. Imágenes de la funcionalidad Wiki en la App store

Fuente: este estudio

1. Permisos solicitados por la aplicación a nivel de instalación

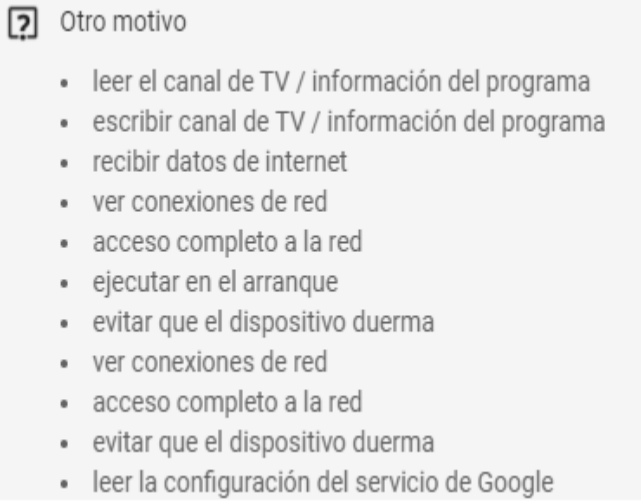
A continuación, se muestra los permisos que solicita el App al ser instalado.

Figura 108. Permisos solicitados por la aplicación Viki



Fuente: este estudio

Figura 109. (Continuación) Permisos solicitados por la aplicación Viki



Fuente: este estudio

2. Permisos archivo Android Manifest.xml

A continuación, se muestra los permisos configurados en el archivo físico manifest.xml

Figura 110. Muestra parcial del archivo mafiest de la aplicación.

```
      <?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="80401" android:versionName="2.3.1" android:installLocation="auto" package="com.viki.android"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="26" />
    <supports-screens android:anyDensity="true" android:smallScreens="true" android:normalScreens="true" android:largeScreens="true" andro
id:resizeable="true" android:xlargeScreens="true" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <uses-permission android:name="com.amazon.device.permission.COMRADE_CAPABILITIES" />
    <uses-permission android:name="com.android.providers.tv.permission.READ_EPG_DATA" />
    <uses-permission android:name="com.android.providers.tv.permission.WRITE_EPG_DATA" />
    <uses-feature android:name="android.hardware.telephony" android:required="false" />
    <uses-feature android:name="android.hardware.microphone" android:required="false" />
    <uses-feature android:name="android.hardware.touchscreen" android:required="false" />
    <uses-feature android:name="android.software.leanback" android:required="true" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE" />
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <application android:theme="@style/Theme.Leanback" android:label="@string/app_name" android:icon="@mipmap/new_viki_launcher" android:n
ame="com.viki.android.VikiApplication" android:allowBackup="true" android:hardwareAccelerated="true" android:largeHeap="true" android:supp
ortsRtl="true" android:banner="@drawable/app_tv_banner">
        <meta-data android:name="com.google.android.gms.version" android:value="@integer/google_play_services_version" />
        <activity android:theme="@android:style/Theme.Translucent" android:name="com.google.android.gms.ads.AdActivity" android:exported="
false" android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|screenSize|smallestScreenSize|uiMode" />
        <activity android:theme="@style/SplashTheme" android:name="com.viki.android.tv.activity.SplashActivity" android:launchMode="single
Top" android:screenOrientation="landscape" android:logo="@drawable/viki_logo">
            <intent-filter>
```

Fuente: este estudio

3. Análisis estático M2

El siguiente cuadro muestra como la aplicación utiliza bases de datos SQLite sin cifrado. Esto puede dar información al atacante si hay un dispositivo rooteado y no hay datos cifrados

Cuadro 31. Tablas y sentencias que expone la App Wiki

Base de datos SQLITE tipo texto sin cifrar

Archivo: vikidatabase.db:

```
TABLES:
android_metadata
languageTable
sqlite_sequence
countries
videoPositions
EntertainmentAgenciesTable
ReviewVoteTable
WatchMarkerTable

RAW DUMP:
CREATE TABLE android_metadata (locale TEXT);CREATE TABLE languageTable (_id integer primary key autoincrement, code text
not null, name text not null, native_name text not null, direction text not null);CREATE TABLE
sqlite_sequence(name,seq);CREATE TABLE `countries` (`code` VARCHAR , `native_name` VARCHAR , `names` VARCHAR , `_id`
INTEGER PRIMARY KEY AUTOINCREMENT );CREATE TABLE `videoPositions` (`channel_title` VARCHAR , `video_id` VARCHAR ,
`user_id` VARCHAR , `resource_type` VARCHAR , `created_at` BIGINT , `position` INTEGER , `media_count` INTEGER , `_id`
INTEGER PRIMARY KEY AUTOINCREMENT , `notified` INTEGER , `watched_percentage_col` INTEGER );CREATE TABLE
EntertainmentAgenciesTable (  _id INTEGER PRIMARY KEY AUTOINCREMENT,  id TEXT NOT NULL,  type TEXT NOT NULL,  titles
TEXT NOT NULL );CREATE TABLE ReviewVoteTable (  _id INTEGER PRIMARY KEY AUTOINCREMENT,  id TEXT NOT NULL,  userid TEXT
NOT NULL,  vote INTEGER NOT NULL DEFAULT 0,  flag INTEGER NOT NULL DEFAULT 0);CREATE TABLE WatchMarkerTable (_id INTEGER
PRIMARY KEY, type TEXT, timestamp TEXT, container_id TEXT, video_id TEXT, episode_number NUMERIC, duration NUMERIC,
watch_marker NUMERIC, credits_marker NUMERIC, updated_till NUMERIC, user_id TEXT);
```

Archivo: viki_vikilitics.db:

```
TABLES:
android_metadata
EventTable
sqlite_sequence

RAW DUMP:
CREATE TABLE android_metadata (locale TEXT);CREATE TABLE EventTable (  _id INTEGER PRIMARY KEY AUTOINCREMENT,  EventId
TEXT NOT NULL,  EventKey TEXT NOT NULL,  EventValue TEXT NOT NULL );CREATE TABLE sqlite_sequence(name,seq);
```

Archivo: kodb:

```
TABLES:
android_metadata
events
sqlite_sequence
updates

RAW DUMP:
CREATE TABLE android_metadata (locale TEXT);CREATE TABLE events (_id INTEGER PRIMARY KEY AUTOINCREMENT, data TEXT NOT
NULL);CREATE TABLE sqlite_sequence(name,seq);CREATE TABLE updates (_id INTEGER PRIMARY KEY AUTOINCREMENT, data TEXT NOT
NULL);
```

Fuente: este estudio

El siguiente cuadro muestra como hay datos quemados en el código que pueden servir para futuros ataques

Cuadro 32. Líneas de código que exponen datos para Wiki

Datos codificados
c/b/o.java:
line 117: b.a((**Object**) a, "The RxJavaPlugins.onSubscribe hook returned a null SingleObserver. Please check the handler provided to RxJavaPlugins.setOnSingleSubscribe for invalid null returns. Further reading: https://github.com/ReactiveX/RxJava/wiki/Plugins");
c/b/i.java:
line 265: c.b.e.b.b.a((**Object**) a, "The RxJavaPlugins.onSubscribe hook returned a null Observer. Please change the handler provided to RxJavaPlugins.setOnObservableSubscribe for invalid null returns. Further reading: https://github.com/ReactiveX/RxJava/wiki/Plugins");
c/b/f.java:
line 51: c.b.e.b.b.a((**Object**) a, "The RxJavaPlugins.onSubscribe hook returned a null MaybeObserver. Please check the handler provided to RxJavaPlugins.setOnMaybeSubscribe for invalid null returns. Further reading: https://github.com/ReactiveX/RxJava/wiki/Plugins");
c/b/a.java:
line 20: b.a(a, "The RxJavaPlugins.onSubscribe hook returned a null CompletableObserver. Please check the handler provided to RxJavaPlugins.setOnCompletableSubscribe for invalid null returns. Further reading: https://github.com/ReactiveX/RxJava/wiki/Plugins");
c/b/c/d.java:
line 11: stringBuilder.append("The exception was not handled due to missing onError handler in the subscribe() method call. Further reading: https://github.com/ReactiveX/RxJava/wiki/Error-Handling | ");
c/b/c/f.java:
line 5: stringBuilder.append("The exception could not be delivered to the consumer because it has already canceled/disposed the flow or the exception has nowhere to go to begin with. Further reading: https://github.com/ReactiveX/RxJava/wiki/What's-different-in-2.0#error-handling | ");
com/iterable/iterableapi/q.java:
line 69: r0 = "https://api.iterable.com/api/";

Fuente: este estudio

Cuadro 33. (Continuación). Líneas de código que exponen datos para Wiki

com/iterable/iterableapi/e.java:
line 280: f fVar = new f(this.e, "https://links.iterable.com/", "a/matchFp", com.iterable.iterableapi.a.a.a(this.d).a(), f.b, new c() {
com/viki/auth/b/d.java:
line 27: return "https://disqus.com/api/3.0/threads/create.json";
line 43: stringBuilder.append("https://disqus.com/api/3.0/threads/create.json?api_key=DEeLE2sHKKtTOVs7zdLK5smRlsjgGbfxegRGt1y7g7XANNdhQTkhAVXhZcYpxGGM&api_secret=Rb7LBRTOhHSREFdb76y2ANV2YSLJkBN7UQ9jxIadSCwNT5qzgcDi1Umyn4McmG6X&forum=vikiorg&title=");
com/viki/auth/b/d.java:
line 30: return "https://disqus.com/api/3.0/threads/set.json";
com/viki/auth/b/d.java:
line 33: return "https://disqus.com/api/3.0/posts/create.json?api_key=DEeLE2sHKKtTOVs7zdLK5smRlsjgGbfxegRGt1y7g7XANNdhQTkhAVXhZcYpxGGM&api_secret=Rb7LBRTOhHSREFdb76y2ANV2YSLJkBN7UQ9jxIadSCwNT5qzgcDi1Umyn4McmG6X";
line 83: stringBuilder.append("https://disqus.com/api/3.0/posts/create.json?api_key=DEeLE2sHKKtTOVs7zdLK5smRlsjgGbfxegRGt1y7g7XANNdhQTkhAVXhZcYpxGGM&api_secret=Rb7LBRTOhHSREFdb76y2ANV2YSLJkBN7UQ9jxIadSCwNT5qzgcDi1Umyn4McmG6X&message=");
com/viki/auth/b/d.java:
line 36: return "https://disqus.com/api/3.0/posts/list.json";
com/viki/a/d.java:
line 7: private static **String** a = "https://collector.viki.io/production";
com/viki/android/RakutenLoginActivity.java:
line 20: **String** b = "https://ap.accounts.global.rakuten.com/globalweb/pages/login.xhtml";
com/viki/android/g/g.java:
line 165: webView.postUrl("https://ap.accounts.global.rakuten.com/globalweb/pages/login.xhtml", stringBuilder.toString().replace("\n", "").getBytes("UTF-8"));
com/viki/android/SplashActivity.java:
line 406: com.google.android.gms.auth.api.a.g.a(this.l, new com.google.android.gms.auth.api.credentials.a.a().a("https://accounts.google.com").a(true).a()).a(new -$$Lambda$SplashActivity$Ih3eEOimBUZT2MZnjxp2XSGYgOk(this));

Fuente: este estudio

Cuadro 34. (Continuación). Líneas de código que exponen datos para Wiki

com/viki/android/utils/j.java:
line 315: **PendingIntent** a = com.google.android.gms.auth.api.a.g.a(c(), new com.google.android.gms.auth.api.credentials.**HintRequest**.a().a(new com.google.android.gms.auth.api.credentials.**CredentialPickerConfig**.a().a(true).b(true).a()).a(true).a("https://accounts.google.com").a());
com/viki/android/utils/a.java:
line 113: stringBuilder.append("https://pubads.g.doubleclick.net/gampad/ads?");
com/viki/android/f/c.java:
line 7: private final **String** b = "https://api.viki.io";
line 8: private final **String** c = "https://api.viki.io";
com/viki/android/customviews/o.java:
line 230: r2 = "https://www.viki.com/mobile_copyright";
com/kochava/base/i.java:
line 359: optString = "https://kvinit-prod.api.kochava.com/track/kvinit";
com/kochava/base/i.java:
line 365: optString = "https://control.kochava.com/track/json";
line 374: optString = "https://control.kochava.com/track/json";
line 380: optString = "https://control.kochava.com/track/json";
line 392: optString = "https://control.kochava.com/track/json";
line 408: return "https://control.kochava.com/track/json";
com/kochava/base/i.java:
line 386: optString = "https://control.kochava.com/track/kvquery";
com/kochava/base/i.java:
line 398: optString = "https://token.api.kochava.com/token/add";
com/kochava/base/i.java:
line 404: optString = "https://token.api.kochava.com/token/remove";
com/surveymonkey/surveymonkeyandroidsdk/b.java:
line 46: stringBuilder.append("https://www.surveymonkey.com/r/");
com/surveymonkey/surveymonkeyandroidsdk/a/a.java:
line 54: r3 = "https://api.surveymonkey.net/sdk/v1/respondents?api_key=";
com/d/b.java:
line 216: a(a, a.a ? "https://turing.viki.io/v4/settings.json" : "https://api-staging.viki.net/v4/settings.json", hashMap, g, new com.android.b.o.b<**String**>() {
com/viki/auth/b/c.java:
line 52: stringBuilder.append("android-app://com.viki.android/viki/");

Fuente: este estudio

Cuadro 35. (Continuación). Líneas de código que exponen datos para Wiki

com/viki/android/settings/fragment/MiscellaneousPreferenceFragment.java:
line 9: private static **String** b = "http://www.viki.com/mobile_terms_of_use";
com/viki/android/settings/fragment/MiscellaneousPreferenceFragment.java:
line 10: private static **String** c = "http://www.viki.com/mobile_copyright";
com/viki/android/video/j.java:
line 115: com.viki.android.utils.d.a("http://support.viki.com/hc/en-us/articles/200138684--Not-available-in-your-region-error-message", j.this.getActivity());
com/viki/android/customviews/o.java:
line 238: r2 = "http://support.viki.com/hc/en-us/articles/200138684--Not-available-in-your-region-error-message";
com/viki/android/g/e.java:
line 17: c.a(context, new j(1, "http://grp01.gidapi-pri.stg.jp.local/v1.2/auth/token/get", hashMap, new **HashMap**(), stringBuilder.toString(), new b<**String**>() {
com/viki/android/g/g.java:
line 34: return "http://rakutenlogin/success";
com/viki/android/g/g.java:
line 37: return "http://rakutenlogin/failure";
com/viki/android/utils/a.java:
line 155: stringBuilder3 = "http://pubads.g.doubleclick.net/gampad/ads?cust_params=test_key%3Dandroid_redirect&sz=640x360&iu=/50449293/Video_Test&impl=s&gdfp_req=1&env=vp&output=xml_vast3&unviewed_position_start=1&url=[referrer_url]&correlator=[timestamp]&cmsid=893&vid=1037996v&description_url=www.viki.com&max_ad_duration=30000&sdmax=120000";
com/viki/android/utils/a.java:
line 159: stringBuilder4.append("http://www.viki.com&vid=");
com/viki/android/chromecast/c/a.java:
line 550: return intent.getData() != null && intent.getData().equals(**Uri**.parse("viki://cast/join"));

Fuente: este estudio

A continuación, se muestra datos expuestos que podrían revelar sitios claves del sitio.

Cuadro 36. Exposición de datos sensibles

Exposición a la información
com/d/b.java:
line 216: a(a, a.a ? "https://turing.viki.io/v4/settings.json" : "https://api-staging.viki.net/v4/settings.json", hashMap, g, new com.android.b.o.b<**String**>() {

Fuente: este estudio

El siguiente cuadro muestra configuraciones que pueden suponer exposición de datos sensibles

Cuadro 37. Configuraciones de información sensible

Copia de seguridad habilitada de la aplicación
android/AndroidManifest.xml:
line 19: <application android:allowBackup="true" android:appComponentFactory="androidx.core.app.CoreComponentFactory" android:hardwareAccelerated="true" android:icon="@mipmap/new_viki_launcher" android:label="@string/viki_app_name" android:largeHeap="true" android:name="com.viki.android.VikiApplication" android:supportsRtl="true" android:testOnly="false" android:theme="@style/VikiTheme">

Fuente: este estudio

4. Análisis estático M3

El siguiente cuadro muestra las configuraciones que permitirían conexiones tipo http

Cuadro 38. Configuraciones que muestran posibles conexiones con protocolo sin cifrar

Uso de protocolo HTTP
c/a/a/a/a/e/d.java:
line 48: public **HttpURLConnection** a(URL url) {
line 49: return (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(url.openConnection()));
line 50: }
line 52: public **HttpURLConnection** a(URL url, **Proxy** proxy) {
line 53: return (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(url.openConnection(proxy)));
line 54: }

Fuente: este estudio

Cuadro 39. (Continuación). Configuraciones que muestran posibles conexiones con protocolo sin cifrar

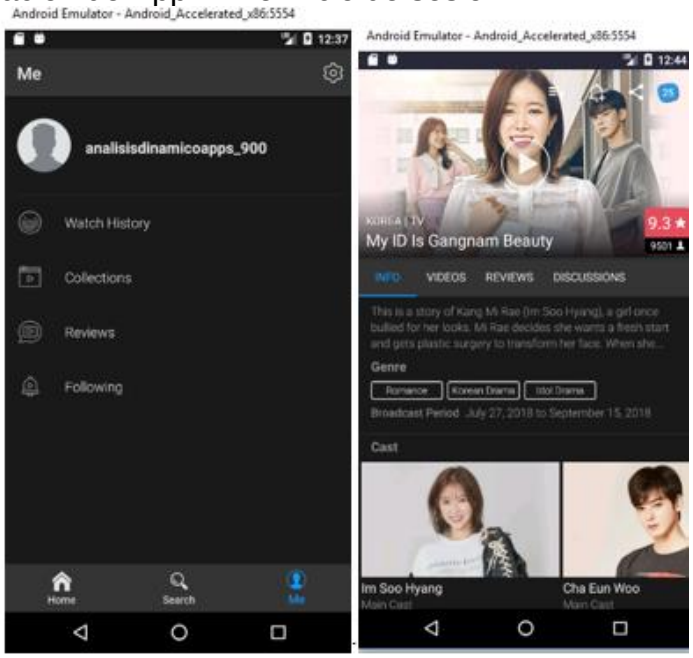
com/kochava/base/d.java:
line 120: **String** property;
line 121: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(str).openConnection()));
line 122: try {
com/bumptech/glide/d/a/f.java:
line 33: public **HttpURLConnection** a(URL url) {
line 34: return (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(url.openConnection()));
line 35: }
com/c/b/af.java:
line 41: public **HttpURLConnection** a(**Uri** uri) {
line 42: **HttpURLConnection** httpURLConnection = (**HttpURLConnection**) ((**URLConnection**) **FirebasePerfUrlConnection**.instrument(new URL(uri.toString()).openConnection()));
line 43: httpURLConnection.setConnectTimeout(15000);

Fuente: este estudio

5. Análisis dinámico M2

Para este tipo de análisis se revisará las bases de datos que genera cada aplicación y si es legible ante herramientas de lectura de bases de datos tipo SQLite. Primero se instala la aplicación como se detalla a continuación

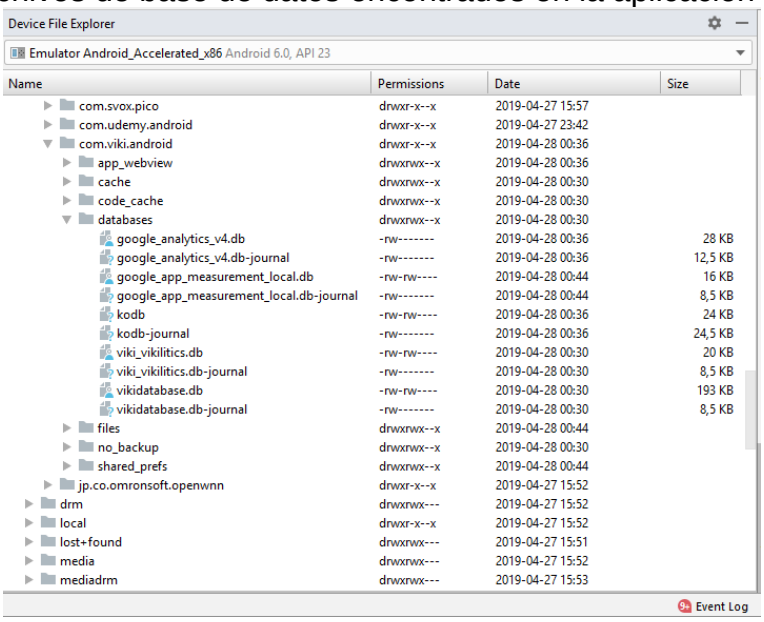
Figura 111. Instalación de App Viki e inicio de sesión



Fuente: este estudio

Luego de instalar la aplicación y ejecutarla iniciando sesión se detectó la creación de varios archivos de base de datos.

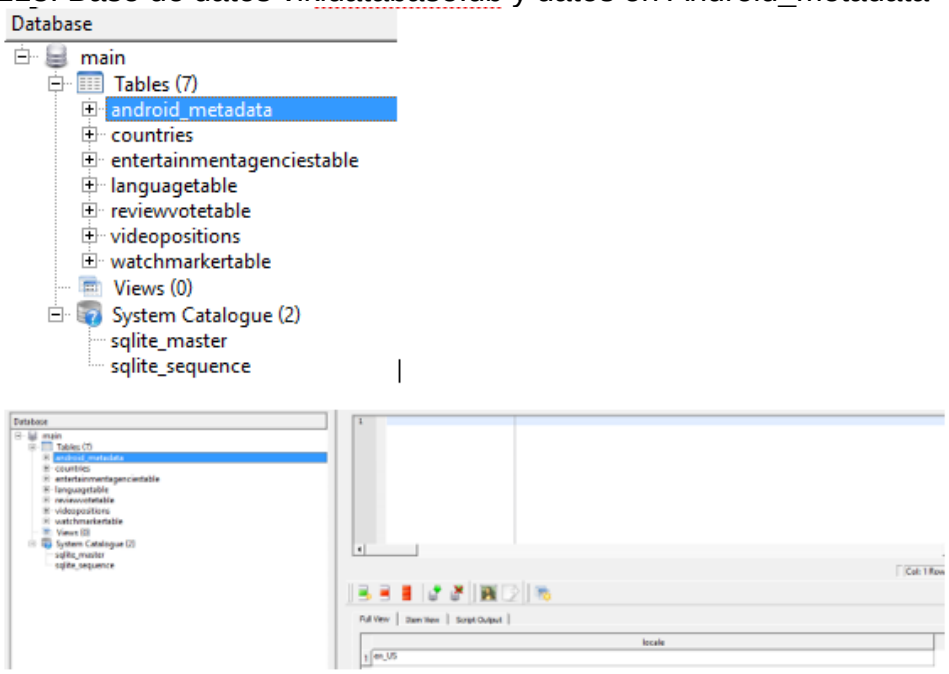
Figura 112. Archivos de base de datos encontrados en la aplicación



Fuente: este estudio

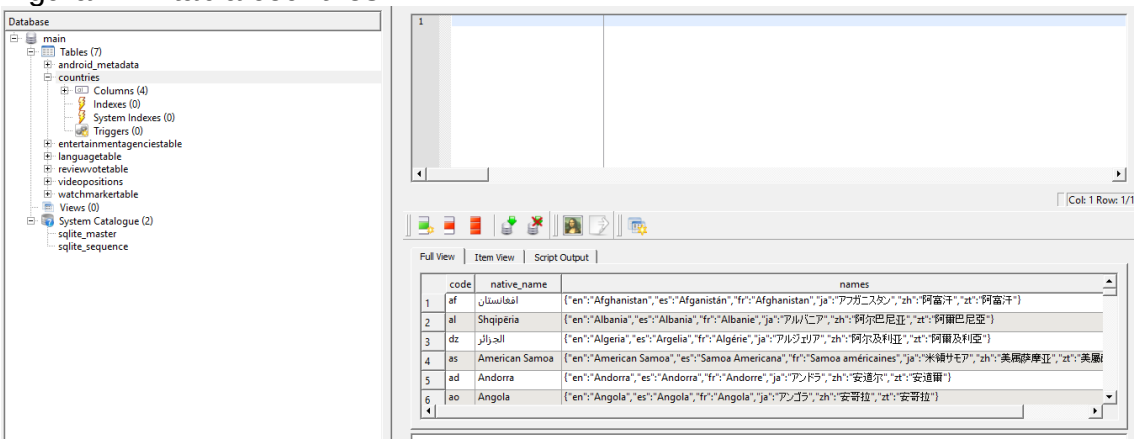
Se procede a revisar la información de las bases de datos con la herramienta Sqliteman y no se observa datos sensibles, solo guarda la configuración de idioma y mensajes de error de ejecución, pero no se visualizó ningún dato sensible. A continuación, las imágenes de cada una de las bases de datos creadas.

Figura 113. Base de datos vikidatabase.db y datos en Android_metadata



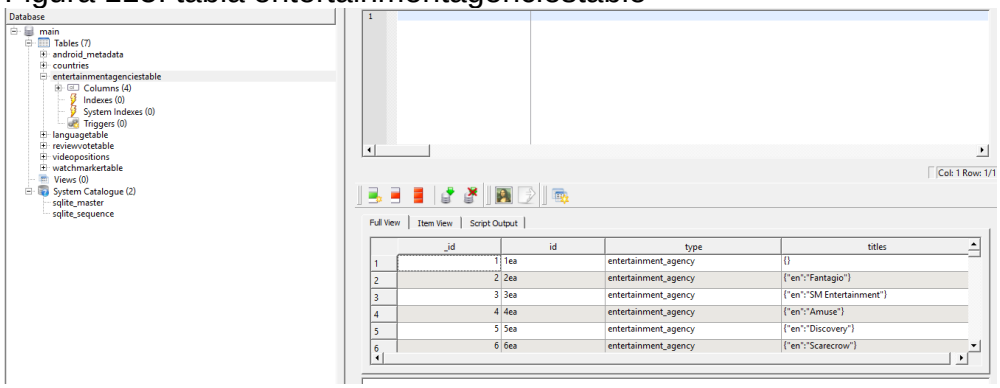
Fuente: este estudio

Figura 114. tabla countries



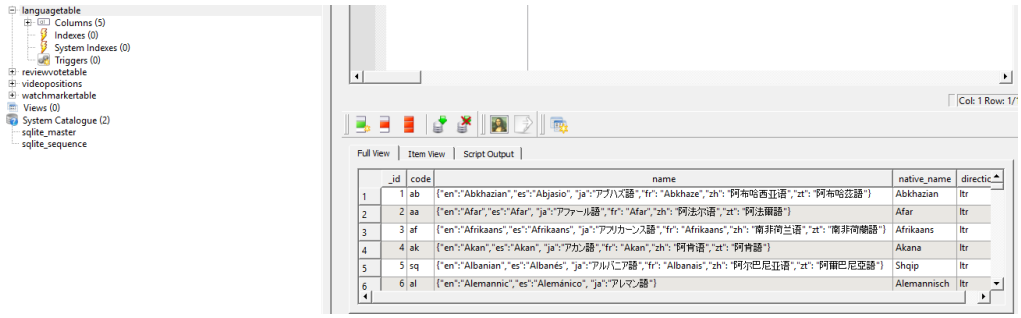
Fuente: este estudio

Figura 115. tabla entertainmentagenciestable



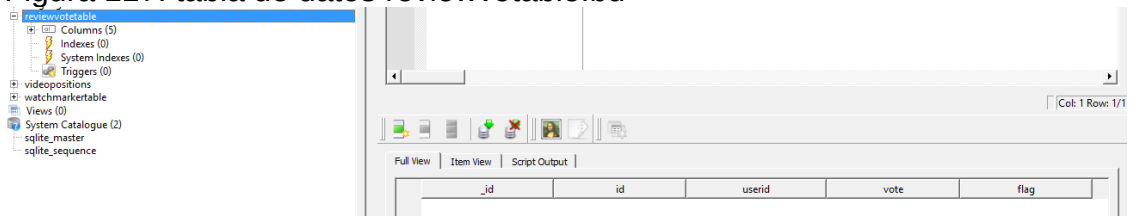
Fuente: este estudio

Figura 116. tabla languagetable



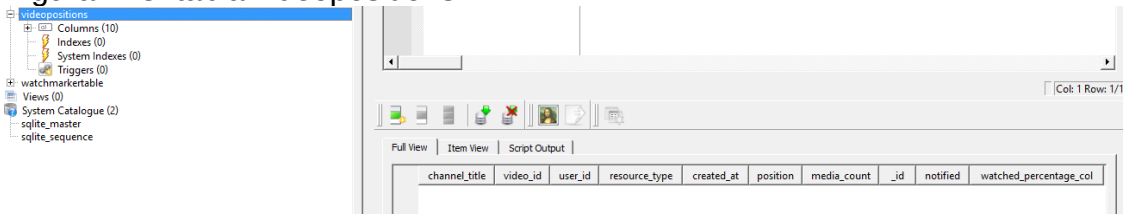
Fuente: este estudio

Figura 117. tabla de datos reviewvotetable.bd



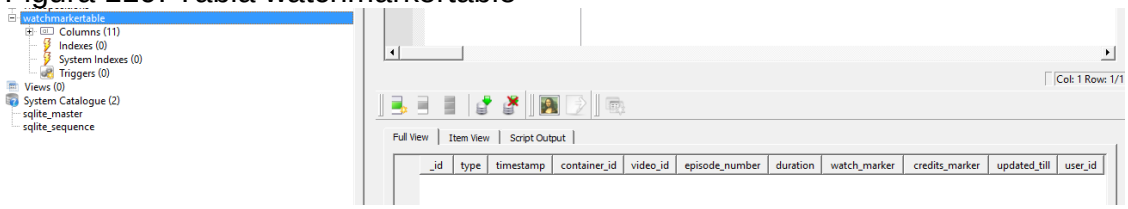
Fuente: este estudio

Figura 118. tabla videopositions



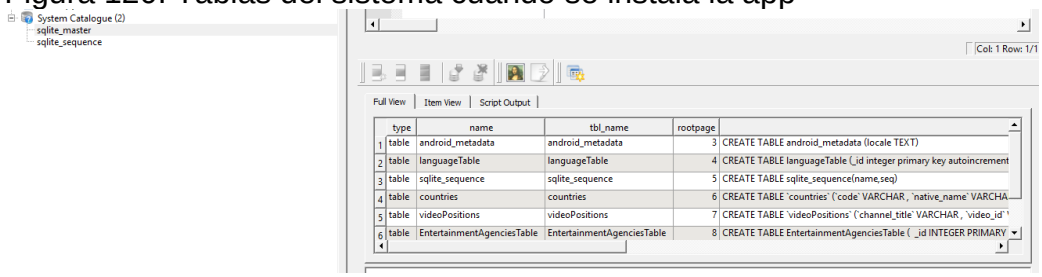
Fuente: este estudio

Figura 119. Tabla watchmarkertable



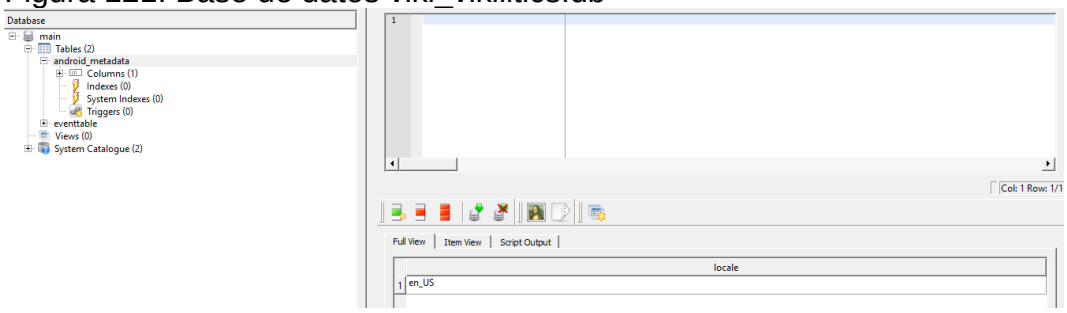
Fuente: este estudio

Figura 120. Tablas del sistema cuando se instala la app



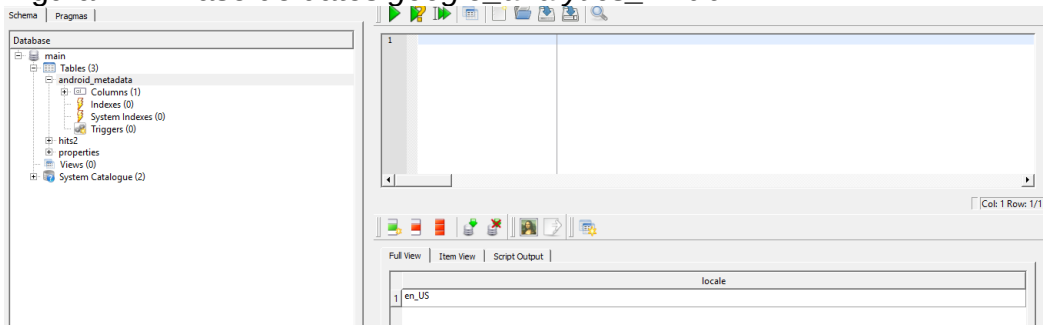
Fuente: este estudio

Figura 121. Base de datos viki_vikilitics.db



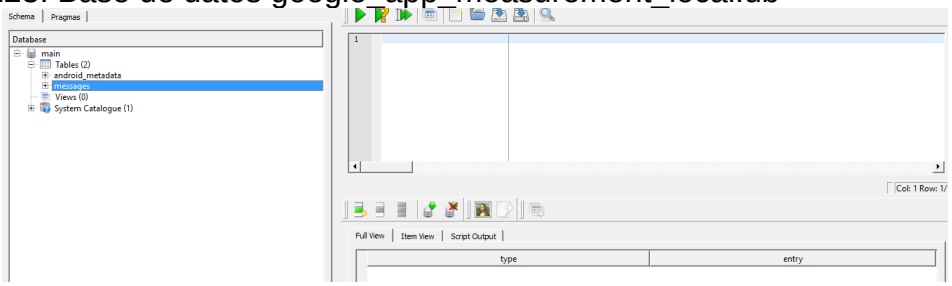
Fuente: este estudio

Figura 122. Base de datos *google_analytics_v4.db*



Fuente: este estudio

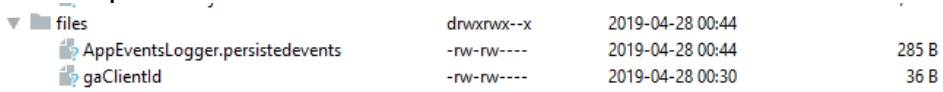
Figura 123. Base de datos google_app_measurement_local.db



Fuente: este estudio

Se revisa de igual forma la carpeta Files en busca de documentos que puedan guardar información sensible pero no se halló información.

Figura 124. Carpeta files.

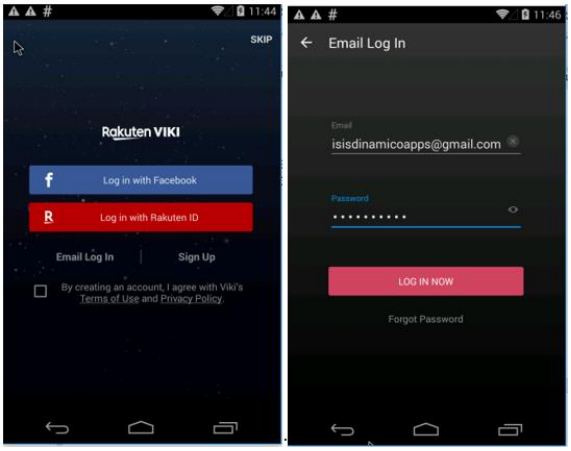


Fuente: este estudio

6. Análisis dinámico M3

Para el análisis dinámico se procede a instalar la App en cualquier emulador en este caso se usó el emulador de MobSF con la idea de analizar el comportamiento. A continuación, formulario inicial

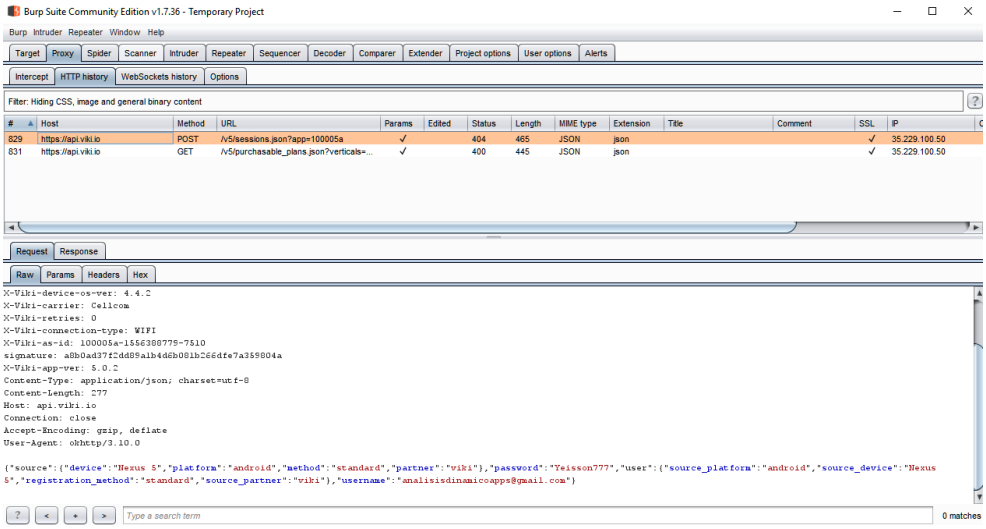
Figura 125. Inicio de sesión en la App



Fuente: este estudio

Se observa con la herramienta Burp Suite como en cada validación el dato queda expuesto a cualquier atacante

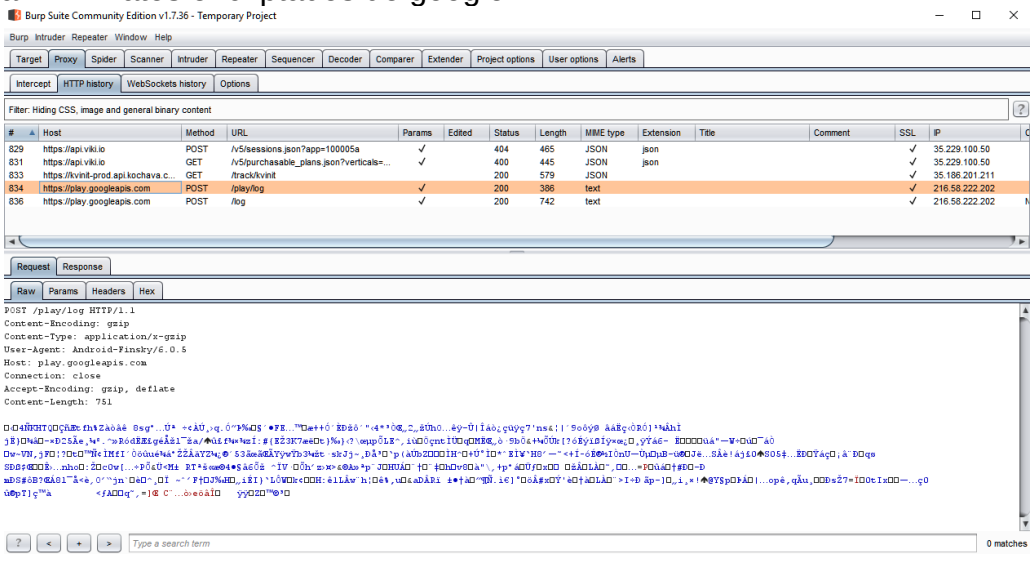
Figura 126. Visualización de usuario y contraseña



Fuente: este estudio

También se observa interacción con el api de google pero su comunicación esta encriptada

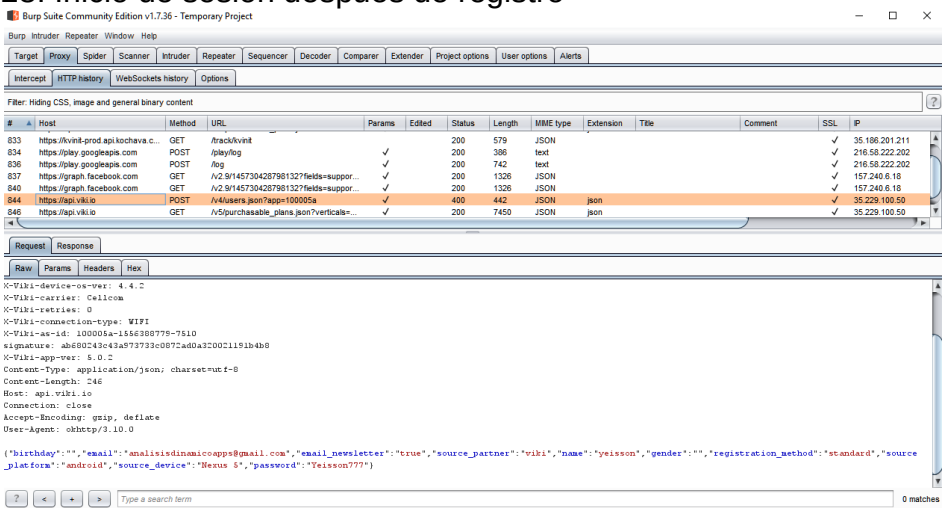
Figura 127. Datos encriptados de google



Fuente: este estudio

Al cerrar sesión y volver a ingresar se observa datos expuestos sin cifrar

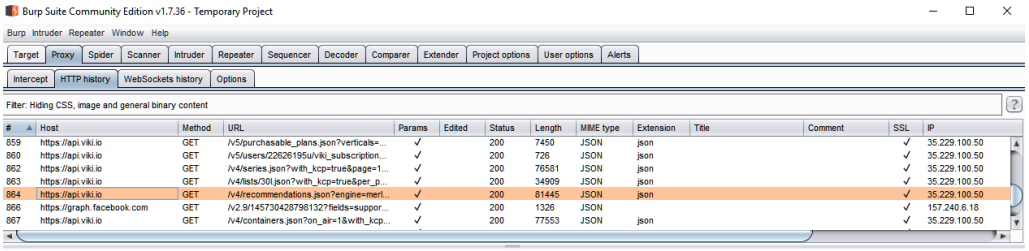
Figura 128. Inicio de sesión después de registro



Fuente: este estudio

En algunas peticiones se observa uso de protocolo sin cifrar http.

Figura 129. Uso de protocolo sin cifrar viki



Fuente: este estudio

Para esta aplicación se genera solo comunicación tipo post creación de usuarios, lo demás es de tipo streaming.

ANEXO B REPORTE EJECUTIVO

En este reporte se visualiza de modo grafico los riesgos M2 y M3 de tipo estático y dinámico y sirve para entender de forma visual el análisis realizado en este laboratorio académico.

Ver adjunto: “Reporte *Ejecutivo.pdf*”

RAE

1. Información General	
Tema	Realizar un laboratorio controlado que analice los riesgos de almacenamiento y comunicación insegura .de acuerdo con la metodología de seguridad móvil Owasp Mobile 2016
Titulo	Análisis dinámico de seguridad en aplicaciones Android con el proyecto de seguridad móvil Owasp.
Autor(es)	Yeisson Valentino Jaramillo Quirama
Director	José Hernando Pena Hidalgo
Fuente Bibliográfica	Se referencia 47 fuentes bibliográficas, Algunas de tipo principal como la página oficial de Owasp Mobile. M2-Almacenamiento de datos inseguros, [en línea] [citado el 1 marzo, 2019]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-M2-Insecure_Data_Storage Mobile Top 10 2016-Top 10, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10 OWASP Open Web Application Security Project. Category: Vulnerability. [En línea] [Citado el 5 de mayo, 2018]. Disponible en internet: https://www.owasp.org/index.php/Category:Vulnerability Proyecto de seguridad móvil OWASP, [en línea] [citado el 17 octubre, 2018]. Disponible en internet: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Home
Año	2019
Resumen	El propósito de este estudio es la comprensión de la seguridad móvil haciendo énfasis en el sistema operativo Android para lo cual se analiza la historia y su evolución, malware y formas de ataque, arquitectura y funcionamiento en pro de una mejor comprensión de la tecnología. Posteriormente se ejecutarán herramientas de análisis disponibles en el mercado que ayudaran en la evaluación de posibles fallos de seguridad y como la metodología OWASP Mobile hace recomendaciones basadas en los juicios de expertos a nivel mundial. Los resultados obtenidos de este estudio servirán como guía para futuros ambientes de pruebas mejorando la calidad del desarrollo y evitando la exposición de datos no cifrados y la comunicación insegura http
Palabras Claves	OWASP, Android, seguridad, análisis estático, análisis dinámico

Contenido	

	7 CONCLUSIONES 8 RECOMENDACIONES BIBLIOGRAFÍA ANEXO A. RESULTADOS EJECUCION PRUEBAS

2. Descripción del problema
ANTECEDENTES DEL PROBLEMA La App Store, tienda de aplicaciones presenta una herramienta de acceso a desarrollos de gran utilidad para las necesidades de cada uno, sin embargo, es cierto que los desarrollos allí dispuestos a veces no tienen el análisis necesario antes de ser liberada al público. Una de las críticas de muchos especialistas es la libertad de instalar Apps de terceros que no están en la plataforma de Google Play. Esto por supuesto es un riesgo innecesario que abre la puerta a todo tipo de vulnerabilidad. Un filtro establecido en la instalación de Apps es dar conocimiento al usuario de los permisos que requiere, acceso a datos y características de conexión. Pues bien, un usuario puede saber por ejemplo si la aplicación puede enviar mensajes de texto, pero, aunque sea claro estos detalles no ayudan mucho pues muchas veces hay acciones en segundo plano que son invisibles para el usuario. **PLANTEAMIENTO DEL PROBLEMA** ¿Cómo puede un equipo de desarrollo construir y mantener aplicaciones móviles seguras según la metodología OWASP MOBILE? El incremento del uso de aplicaciones móviles viene en aumento considerable, tanto así que la venta de equipos de cómputo a caído vertiginosamente. Son más las organizaciones como bancos, juegos, redes sociales y gobiernos que ponen sus funcionalidades en aplicaciones de tipo móvil haciendo de esta variedad un blanco potencial para los ciberdelincuentes con diferentes fines. Google indico que el aumento de aplicaciones maliciosas va en aumento ya que de las 8.5 millones un 77% son consideradas malware con ataques tipo Ransomware en aumento hasta en un 50%. Pero la movilidad es un servicio que da demasiadas ventajas a sus usuarios y por eso se debe mirar con mucho cuidado las nuevas formas de ataques y las formas de implementación para tener las aplicaciones blindadas. Son tantas las utilidades y funciones de los móviles y las aplicaciones que existe la posibilidad de manejar datos de todo tipo, videos, fotos, conversaciones, claves e información bancaria, todo esto hace que la vulnerabilidad crezca con ataques de malware, software espía entre muchos más pues el abanico es muy grande. También juega el mal uso de los usuarios respecto a las aplicaciones y el manejo que requieren ya que es un activo critico volviéndose en bancos de información personal teniendo así un riesgo muy alto. Ahora bien, ese es el tema de datos, pero hay factores contrarios a la voluntad del usuario que hace que sea crucial blindar la información como por ejemplo cuando se pierde el móvil o es robado sin saber el uso que le darán a la información allí almacenada. Android como sistema operativo predominante permiten que los desarrolladores piensen más en este tipo de aplicaciones y generalmente publican desarrollos solo con las mínimas medidas de seguridad y lo que es peor cuando se instalan aquellas Apps que ni siquiera están en la App Store hace que la vulnerabilidad alcance niveles críticos.

Los móviles se han vuelto en bancos de información personal siendo así un riesgo muy alto de seguridad.

FORMULACION DEL PROBLEMA

¿Qué nivel de seguridad tienen las aplicaciones nativas para el sistema operativo Android según los riesgos OWASP Mobile M2 y M3?

3. Objetivos

OBJETIVO GENERAL

Realizar un estudio del análisis dinámico de la seguridad en el desarrollo de las aplicaciones móviles a partir de la metodología OWASP Mobile

OBJETIVOS ESPECÍFICOS

- Realizar un estudio del esquema de la arquitectura Android y su seguridad.
- Identificar riesgos de almacenamiento y comunicación inseguros según OWASP
- Realizar un laboratorio controlado que permita evaluar la seguridad en aplicaciones móviles
- Realizar un informe de diagnóstico de las aplicaciones evaluadas, donde se evidencie aspectos claves de seguridad respecto a almacenamiento y transporte de datos inseguro.

4. Diseño metodológico

La vulnerabilidad de las aplicaciones se presenta cuando no hay un análisis de pruebas y por medio de herramientas especializadas se encuentran agujeros que ponen en riesgo la integridad, confidencialidad y disponibilidad del producto. Se realizará una clasificación de los riesgos según OWASP 2016 para sus ítems M2 y M3 con el propósito de describir múltiples vulnerabilidades que servirán para cualquier ataque de malware. Se tomará como base los riesgos M2 y M3 de la metodología testing OWASP Top Ten móvil Riesgos de 2016. Esta metodología permite ver los defectos comunes ante auditorias de seguridad y comenta los diferentes riesgos ante debilidades e impactos.

OWASP M2: Almacenamiento de datos inseguros, fugas de información.

OWASP M3: Comunicación insegura. Configuraciones incorrectas de SSL, no cifrado, negociación débil.

TÉCNICAS DE RECOLECCION DE DATOS

En pro de realizar una correcta investigación sobre los riesgos que serán analizados se han realizado las siguientes tareas.

Investigación de la metodología OWASP Mobile en su página oficial , investigación de la arquitectura Android, políticas de seguridad, investigación sobre desarrollo de aplicaciones y sus fallas de programación y configuración, Investigación sobre herramientas que puedan realizar testing en ambiente seguro para aplicaciones que funcionen en Android, análisis de técnicas de ingeniería para descompilar y revisar APKS ,anexo con resultados de las pruebas realizadas sobre las apps seleccionadas basadas en un hacking ético, análisis e imágenes de las pruebas realizadas en el emulador de Android para este propósito.

Fuentes de información primarias: Como técnicas de fuentes primarias se utilizarán las investigaciones sobre la arquitectura de Android y artículos investigativos de seguridad en las

aplicaciones, la forma en que la tecnología móvil ha cambiado la interacción humana. Todas las lecturas realizadas para entender el tema de las amenazas y riesgos de las aplicaciones móviles y como los hackers están pasando sus ataques de la web a la tecnología móvil, para este caso Android.

Tipos de fuentes: Monografías, informes de investigación, patentes y normas, fuentes no publicadas como lo son videos y blogs personales.

Fuentes de información secundarias: Como fuentes primarias se utilizarán trabajos e investigaciones que ya estén depurados según el propósito de este trabajo. Estas servirán como base para iniciar el trabajo que dé cumplimiento a los objetivos trazados.

Tipo de fuentes: Índices de contenido, fuentes bibliográficas, citación de fuentes según normas requeridas, enciclopedias, manuales, tablas demostrativas, traducciones

Fuentes de información y resultados de los datos obtenidos: Como fuentes de información se utiliza las herramientas que permitan realizar el análisis estático de entorno seguro sea en un móvil o un emulador. Se pretende con esto obtener un análisis demostrativo que permita identificar las falencias halladas.

5. Referentes teóricos

En pro de realizar una correcta investigación sobre los riesgos que serán analizados se han realizado las siguientes tareas.

- Investigación de la metodología OWASP Mobile en su página oficial.
- Investigación de la arquitectura Android, políticas de seguridad.
- Investigación sobre desarrollo de aplicaciones y sus fallas de programación y configuración.
- Investigación sobre herramientas que puedan realizar testing en ambiente seguro para aplicaciones de que funcionen en Android.
- Análisis de técnicas de ingeniería para descompilar y revisar APKS.
- Anexo con resultados de las pruebas realizadas sobre las apps seleccionadas basadas en un hacking ético.
- Análisis e imágenes de las pruebas realizadas en el emulador de Android para este propósito.

6. Conclusiones

Es imperativo comprender la necesidad de proteger las aplicaciones móviles, la sociedad se vuelta a estas tecnologías y con ellas el creciente mercado de ataques de delincuentes cibernéticos con miras a minar la seguridad de los sistemas operativos en este caso Android.

En este trabajo se comprenden mejor como los desarrollos no tienen en cuenta la protección del código exponiendo urls y datos de archivos que pueden ser usados para futuros ataques. Este estudio ha cumplido con el propósito de sus objetivos dando bases para una mejor comprensión de la seguridad móvil.

Se ha tenido como referencia el marco de trabajo de Owasp Mobile 2016 haciendo énfasis sobre los riesgos M2 y M3 los cuales se presentan como un punto débil en la cadena de seguridad de los dispositivos. Android tiene como política la confianza entre su tecnología y los

desarrolladores, por lo que permite ver los permisos que solicita las aplicaciones antes de instalarse, quizá muchas veces los usuarios no tienen en cuenta estas advertencias y caen en aplicaciones falsas que se instalan con el propósito de analizar patrones y conseguir datos personales

Es muy importante que los profesionales en desarrollo cumplan con las políticas de seguridad establecidas en sus organizaciones, la implementación de sistemas de calidad que ejecuten pruebas de todos tipos sobre las aplicaciones antes de salir al mercado para sanear aquellos riesgos que pueden convertirse en ataques a la seguridad y por último el desprestigio de la casa de software.

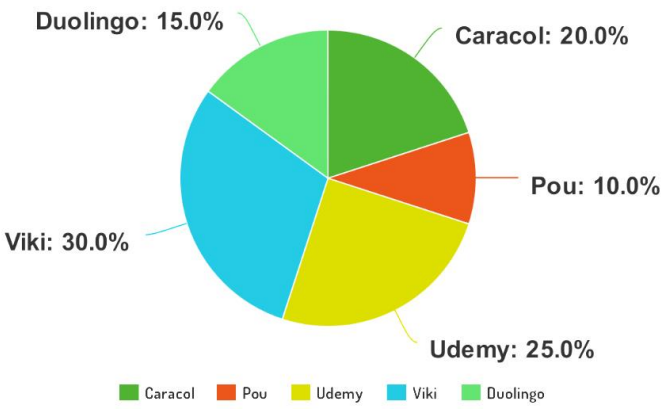
Se ha hecho referencia sobre los archivos Manifiest.xml y se analizan con herramientas especializadas que explican y califican este tipo de permisos en la aplicación y como pueden convertirse en riesgos de seguridad. Debe entenderse correctamente este tipo de archivos para tener un panorama de lo seguro que puede ser una aplicación.

REPORTE
EJECUTIVO

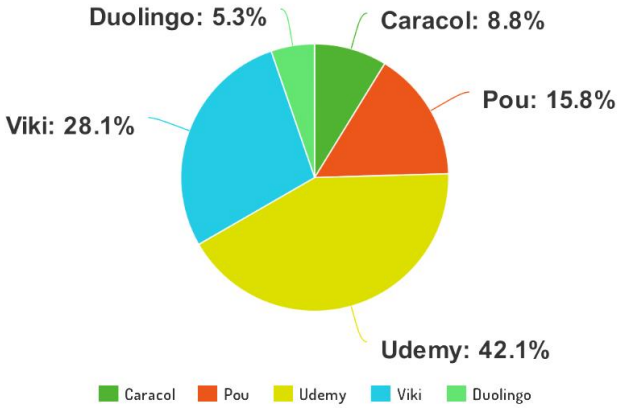
ANÁLISIS DINÁMICO DE SEGURIDAD EN APLICACIONES ANDROID CON EL PROYECTO DE SEGURIDAD MÓVIL OWASP M2 Y M3

APPS ANALIZADAS

RIESGO ESTÁTICO M2



RIESGO ESTÁTICO M3



SEGURO
MEDIO
INSEGURO

RIESGO DINAMICO M2

RIESGO DINÁMICO M3