ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ESTUDIO DE LA AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS QUE UTILIZAN APLICACIONES ANDROID DESARROLLADAS PARA INSTITUCIONES PÚBLICAS DEL ECUADOR

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ARGUDO MURILLO JULIO ALEJANDRO
julio.argudo@epn.edu.ec

DIRECTOR: ING. GABRIEL ROBERTO LÓPEZ FONSECA MSc.
gabriel.lopez@epn.edu.ec

CODIRECTOR: ING. FRANKLIN LEONEL SÁNCHEZ CATOTA MSc.
franklin.sanchez@epn.edu.ec

Quito, Abril 2019

DECLARACIÓN

Yo, Julio Alejandro Argudo Murillo, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Julio Alejandro Argudo Murillo

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Julio Alejandro Argudo Murillo, bajo mi supervisión.

Ing. Gabriel Roberto López Fonseca MSc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

Ing. Franklin Leonel Sánchez Catota MSc.
CODIRECTOR DEL TRABAJO DE TITULACIÓN

AGRADECIMIENTOS

Agradezco a Dios, por regalarme la bendición de vivir, de colocar retos y dificultades diarias, por fortalecer mi corazón, iluminar mi mente y brindarme la oportunidad de esforzarme por lograr el objetivo de ser una mejor persona en primer lugar, y, en segundo lugar, un mejor profesional.

A mis queridas instituciones Instituto Tecnológico Superior Central Técnico y Escuela Politécnica Nacional en cuyas aulas me forjé, mis profesores, sus colaboradores y autoridades, porque hicieron de mi persona una entidad que no se rinde ante las dificultades de las circunstancias.

DEDICATORIA

El siguiente trabajo está dedicado a toda mi querida familia, quienes han estado a mi lado en todo tiempo y adversidad, soportando largas jornadas de estudio y trabajo, su comprensión y motivación ha sido fundamental en estas instancias de mi vida puesto que sin ustedes estuviera perdido.

A todas aquellas personas que han perdido la esperanza de un futuro mejor, invitándoles a Soñar, Esforzarse, Estudiar, Vivir y ser Valientes; puesto que Dios con su infinita misericordia conoce el tiempo exacto en el que vendrá una gran bendición, pues Él es el dueño de todo, y tiene su plan trazado, y aunque algunas cosas nos parezcan injustas o malas, solo DIOS sabe su perfecto plan.

Eclesiastés 3: 1-8. Todo tiene su tiempo.

Albert Einstein: "Locura es hacer la misma cosa una y otra vez esperando obtener diferentes resultados"

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Este documento está conformado por 4 capítulos.

En el primer capítulo se realiza un estudio al sistema operativo Android para conocer sus características, historia, arquitectura, principales API (*Application Programming Interface*) y librerías, y su modelo de seguridad. Además, con referencia al desarrollo de aplicaciones se revisa la estructura de los proyectos Android y los componentes de las aplicaciones.

Posteriormente se revisan los aspectos de análisis de seguridad para dispositivos móviles descritas en los lineamientos de las metodologías OWASP (*Open Web Application Security Project*) y la guía de análisis de seguridad OASAM(*Open Android Security Assessment Methodolog*).

También se revisa la metodología para la gestión del riesgo de privacidad CNIL(Comisión Nacional de Informática y de las Libertades), y la evaluación de los riesgos de privacidad en Android, con la finalidad de tener conocimiento de las áreas en las que se pueden tener vulnerabilidades en el OS (*Operating System*) Android y en los módulos y componentes de las aplicaciones. Además, se revisan distintas herramientas computacionales que permiten analizar vulnerabilidades orientadas al sistema operativo, vulnerabilidades generadas por la programación de las aplicaciones y uso de las API, la transferencia de información entre terminales e interacción entre aplicativos instalados en el OS Android.

En el segundo capítulo, se genera una guía de procedimientos que permitirá el análisis de vulnerabilidades existentes en el dispositivo móvil y en las aplicaciones instaladas, con la finalidad de elaborar un modelo estructurado de procedimientos para el análisis de vulnerabilidades que comprometan la privacidad de los usuarios.

En el tercer capítulo, se presenta la implementación de la guía de procedimientos propuesta en el capítulo anterior, para evaluar cuatro aplicaciones de instituciones públicas del Ecuador instaladas en ambientes físicos con OS Android 7, con la

finalidad de obtener un reporte que muestre todas las vulnerabilidades que podrían filtrar información privada de los usuarios de cada aplicativo, y realizar de esa forma la evaluación de la criticidad de cada aplicación, mediante el análisis de impacto sobre los activos de información del usuario que están presentes en Android y las vulnerabilidades encontradas.

En el cuarto capítulo, se presentan las conclusiones y recomendaciones obtenidas en el desarrollo del proyecto.

En la sección de Anexos, se presenta un índice de contenidos de los recursos que se encuentran almacenados en el medio digital de anexos, adjunto en este documento.

PRESENTACIÓN

El presente trabajo de Titulación tiene por objetivo realizar un análisis de la afectación a la privacidad que podría existir debido al uso de aplicaciones Android publicadas por instituciones públicas del Ecuador, que incluye un estudio del sistema operativo Android y su arquitectura, además de una revisión a los aspectos de análisis de seguridad para dispositivos móviles descritos en los lineamientos de las metodologías OWASP (*Open Web Application Security Project*), OASAM (*Open Android Security Assessment Methodolog*), gestión del riesgo de privacidad CNIL (Comisión Nacional de Informática y de las Libertades), y la evaluación de los riesgos de privacidad en Android.

Este trabajo tiene como finalidad desarrollar una guía de procedimientos que mediante su implementación y el uso de distintas herramientas computacionales, permitan conocer vulnerabilidades generadas por el sistema operativo, la forma del desarrollo, módulos o componentes de las aplicaciones y el uso de API (*Application Programming Interface*), la transferencia de información entre terminales y la interacción entre aplicativos instalados en el OS (*Operating System*) Android, que en el caso de ser explotadas permitirían la fuga de información, desde el dispositivo en uso hacia destinos conocidos o desconocidos y de esa forma afectar la privacidad de los usuarios.

Finalmente, con toda la información recolectada por la implementación de la guía de procedimientos, se realiza una evaluación de la criticidad de las aplicaciones de entidades públicas mediante un análisis de impacto sobre los activos de información del usuario que están presentes en las aplicaciones Android publicadas por entidades públicas del Ecuador.

CAPÍTULO I
MARCO TEÓRICO

En este capítulo se describe el fundamento teórico en el cual se basa el presente trabajo de Titulación. Se estudia principalmente al sistema operativo Android, su arquitectura, componentes y modelo de seguridad; a continuación, se analizan las metodologías de análisis de vulnerabilidades y los métodos para el estudio de afectación de riesgo relacionado al uso de aplicaciones Android, además del análisis de herramientas computacionales que permitan analizar vulnerabilidades en el sistema operativo y las aplicaciones instaladas.

1.1 INTRODUCCIÓN

Los dispositivos móviles están cambiando de forma significativa la sociedad actual tanto como Internet, dispositivos nuevos cada vez ofrecen capacidades similares a un computador personal, por lo cual se los puede usar para todo tipo de trabajo, de tal manera que se han convertido en dispositivos imprescindibles para la vida de muchas personas, pero, a diferencia de un ordenador, estos pueden ser portables y caber en la palma de la mano.

Estos dispositivos móviles son controlados por sistemas operativos (OS[1] – *Operating System*) que actualmente los proveen varios fabricantes, además de soportar la instalación de aplicativos de distintos tipos y orígenes, por lo cual se convierten en un blanco directo para los atacantes, debido a que personas con conocimientos de programación están en la capacidad de desarrollar una aplicación e inyectar código malicioso dentro de ellas, que al ser ejecutadas pueden dar como resultado pérdida o acceso a información privada y confidencial de los usuarios.

Uno de los sistemas operativos disponibles actualmente, es el OS Android, de código abierto y desarrollado por la Open Handset Alliance (OHA) [1], fundada el 5 de noviembre de 2007, la cual es una alianza comercial de 84 compañías que se dedica a desarrollar estándares abiertos para dispositivos móviles, entre ellos

[1] OS (Sistema Operativo): software principal o conjunto de programas de un sistema informático que gestiona los recursos de hardware y provee servicios a los programas de aplicación de software.

Samsung, HTC, Dell, Intel, Motorola, Qualcomm, Texas Instruments, LG, T-Mobile, Wind River Systems y Nvidia [2], entre otros y liderada por Google junto a otros 34 miembros entre los que se incluye fabricantes de dispositivos móviles, desarrolladores de aplicaciones, algunos operadores de comunicaciones y fabricantes de chips.

En la actualidad Android es el OS móvil más usado en todo el mundo, tal como se puede observar en la Figura 1.1, gracias a que puede instalarse prácticamente en todo tipo de dispositivos; por ello , ha causado un gran interés y tiene una importante aceptación de los usuarios, así como por parte de la industria. Por este motivo, al ser el sistema operativo más utilizado se incrementan los riesgos[2] que podrían perjudicar la seguridad de la información de los usuarios de los aplicativos.

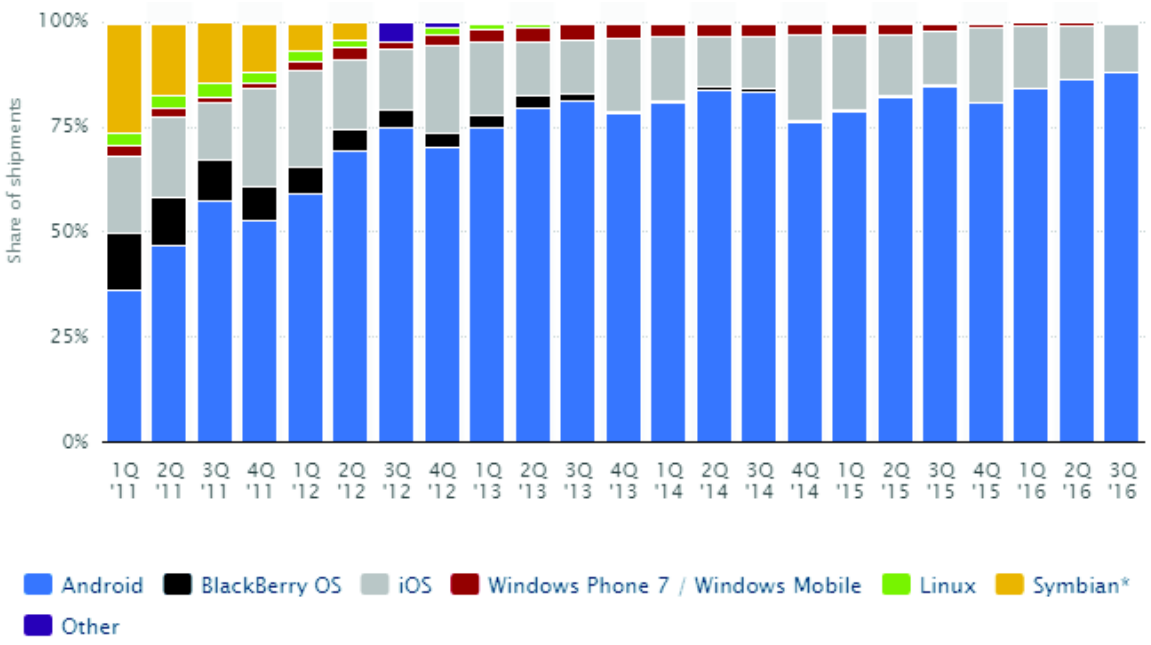

Figura 1.1. Mercado de sistemas operativos móviles de smartphones *alrededor del mundo desde el primer trimestre de 2011 al tercer trimestre de 2016 [3]*

En la actualidad existe una gran interacción entre las personas y los dispositivos móviles. Esta interacción ha logrado que los usuarios puedan intercambiar mensajes (chat) a través del Internet, hacer llamadas telefónicas, tomar fotografías,

[2] Riesgo: probabilidad de que una amenaza se materialice, utilizando la vulnerabilidad existente de un activo o grupos de activos de información, generándoles pérdidas o daños; Riesgo = Probabilidad de amenaza (Amenaza * Vulnerabilidad) * Impacto.

grabar videos, utilizar geolocalización GPS[3], compartir su ubicación al desplazarse hacia algún destino determinado, etc. por lo cual muchos de los datos personales como: domicilio, número de seguro social, números telefónicos, búsquedas realizadas, sitios web visitados, artículos leídos, detalles de compra e incluso patrones de comportamiento podrían estar disponibles en Internet.

El OS Android toma la seguridad como un factor prioritario por lo cual, en principio, es complicado vulnerarlo. Sin embargo, errores en la implementación de las aplicaciones, pueden generar comportamientos no deseados que provocan vulnerabilidades explotables [4], en sí los ataques más destacados en Android son de tipo *malware*[4], *spyware*[5] o virus[6] que están embebidos en las aplicaciones instaladas en el dispositivo móvil [5]. Por lo cual es muy importante conocer y asegurar el funcionamiento correcto de las aplicaciones y sus respectivos procesos.

De acuerdo a recientes estudios realizados por NowSecure [6] se conoce que al menos un cuarto de todas las aplicaciones móviles tienen al menos una falla de seguridad de alto riesgo generalmente, el 35 % de las comunicaciones enviadas por dispositivos móviles no están cifradas y el dispositivo móvil promedio se conecta a 160 servidores únicos cada día. Android al ser una plataforma abierta, se torna vulnerable por la alta incidencia de software no deseado que se descarga junto a las aplicaciones [7].

Por esta razón, el presente trabajo se concentra en analizar la posible afectación a la privacidad de los usuarios, mediante el análisis y la investigación de procedimientos que permitan conocer las vulnerabilidades que los dispositivos móviles pueden tener, y que servirían para filtrar información sensible que comprometan la privacidad de los usuarios que utilizan el OS Android y sus aplicaciones instaladas.

---

[3] GPS (*Global Positioning System*): sistema de Posicionamiento Global, es un sistema que permite determinar en toda la Tierra la posición de un objeto.
4 *Malware*: programa, código, software malicioso dañino o malintencionado, es un tipo de software que tiene como objetivo infiltrarse o dañar una computadora o sistema de información.
5 *Spyware*: *malware* que recopila información de un ordenador y después transmite esta información a una entidad externa sin el conocimiento o el consentimiento del propietario del ordenador.
6 Virus: software que tiene por objetivo alterar el funcionamiento normal del ordenador, sin el permiso o el conocimiento del usuario.

1.2 SISTEMA OPERATIVO ANDROID

1.2.1 ¿QUÉ ES ANDROID? [8] - [9]

Android es un OS inicialmente pensado para teléfonos móviles, al igual que Symbian, Blackberry OS e iOS; está basado en Linux, y permite desarrollar aplicaciones en Java que se ejecutarán en la máquina virtual Dalvik[7] o ART[8] optimizada para dispositivos móviles, dependiendo de sus versiones; proporcionando las interfaces necesarias para que las aplicaciones puedan acceder a las funciones del dispositivo móvil (llamadas, GPS, directorio telefónico, cámara, etc.) de forma sencilla, Android tiene como una de sus mejores características que es completamente libre, lo que permite que cualquier persona pueda descargar el código fuente, analizarlo, compilarlo e inclusive modificarlo.

1.2.2 CARACTERÍSTICAS [10]

Entre las principales características que posee Android se pueden citar las siguientes

- Núcleo del OS de código abierto, basado en el Kernel de Linux.
- Adaptable a muchas pantallas y resoluciones de distintos dispositivos.
- Ofrece diferentes formas de mensajería.
- Utiliza SQLite[9] para el almacenamiento de datos.
- Navegador web del OS basado en WebKit[10].
- Soporte de HTML, HTML5, Java y muchos formatos multimedia.
- Posee un catálogo de aplicaciones que pueden ser descargadas e instaladas desde Google Play.
- Incluye un emulador de dispositivos, herramientas para depuración de memoria y análisis del rendimiento del software.
- Aplicaciones propietarias para video llamadas y mensajería sobre Internet.
- Multitarea real de aplicaciones.

---

[7] Dalvik: máquina virtual que utiliza la plataforma para dispositivos móviles Android, que permite ejecutar aplicaciones programadas en Java.
[8] ART (Android *Runtime*): entorno de ejecución de aplicaciones utilizado por el sistema operativo Android en reemplazo a Dalvik, el cual transforma la aplicación en instrucciones de máquina, que posteriormente son ejecutadas por el entorno de ejecución nativo del dispositivo.
[9] SQLite: sistema de gestión de bases de datos relacional, contenida en una pequeña biblioteca escrita en C.
[10] WebKit: motor de navegación web de código libre, además un *framework* de Mac OS X, que se usó para construir aplicaciones como el mencionado Safari, Dashboard, Mail y otras

1.2.3 HISTORIA, VERSIONES DE ANDROID Y NIVELES DE API [11]

En el año 2003 se daba comienzo a la compañía Android Inc, creada por un grupo de visionarios, encabezado por Andy Rubín, cuyo objetivo fue la producción de software para terminales móviles. En el año 2005 Google adquiere Android Inc, y se empieza a trabajar en el desarrollo de la máquina virtual Dalvik, una máquina virtual optimizada para terminales móviles basada en Java.

En el año 2007 se crea el consorcio Handset Alliance y en el mes de noviembre se lanza una primera versión de Android SDK[11], con lo cual en 2008 aparece el primer móvil con Android (T-Mobile G1), desde entonces, hasta la actualidad Android ha desarrollado una serie de versiones compatibles siempre con sus versiones anteriores, de acuerdo con la siguiente lista:

- Android 1.0 - API 1 (septiembre 2008)
- Android 1.1 - API 2 (febrero 2009)
- *Cupcake*, Android 1.5 - API 3 (abril 2009)
- *Donut* Android 1.6 - API 4 (septiembre 2009)
- Éclair Android 2.0 - API 5 (octubre 2009)
- Éclair Android 2.1 - API 7 (enero 2010)
- Froyo, Android 2.2 - API 8 (mayo 2010)
- *Gingerbread* Android 2.3 - API 9 (diciembre 2010)
- *Honeycomb*, Android 3.0 - API 11 (febrero 2011)
- Android 3.1 - API 12 (mayo 2011)
- Android 3.2 - API 13 (julio 2011)
- *Ice Cream Sandwich*, Android 4.0 - API 14 (octubre 2011)
- Android 4.0.3 - API 15 (diciembre 2011)
- *Jelly Bea*n, Android 4.1 - API 16 (julio 2012)
- Android 4.2 - API 17 (noviembre 2012)
- Android 4.3 - API 18 (julio 2013)
- KitKat, Android 4.4 - API 19 (octubre 2013)
- *Lollipop*, Android 5.0 - API 21 (noviembre 2014)

[11] Android SDK (*Software Development Kit*): kit de desarrollo de software, con el que se puede desarrollar aplicaciones en lenguaje Java y ejecutar un emulador del sistema Android de cualquier versión.

- Android 5.1 - API 22 (marzo 2015)
- *Mashmallow* Android 6.0 - API 23 (octubre 2015)
- *Nougat* Android 7.0 - API 24 (julio 2016)
- *Nougat* Android 7.1 - API 25 (diciembre 2016)

1.2.4 VERSIONES MÁS UTILIZADAS [12]

De acuerdo con la información revisada, las distribuciones de las diferentes versiones de Android muestran que la versión 6 *Marshmallow* es la más utilizada, tal como se puede apreciar en la Tabla 1.1 y en la Figura 1.2.

Tabla 1.1. Dispositivos Android con porcentajes superiores al 0,1%, según la plataforma instalada [12]

Versión	Nombre	API	Porcentaje de Instalación
2.3.3 - 2.3.7	*Gingerbread*	10	0.4%
4.0.3 - 4.0.4	*Ice Cream Sandwich*	15	0.5%
4.1.x	*Jelly Bean*	16	2.0%
4.2.x		17	3.0%
4.3		18	0.9%
4.4	*KitKat*	19	13.4%
5	*Lollipop*	21	6.1%
5.1		22	20.2%
6	*Marshmallow*	23	29.7%
7	*Nougat*	24	19.3%
7.1		25	4.0%
8	*Oreo*	26	0.5%

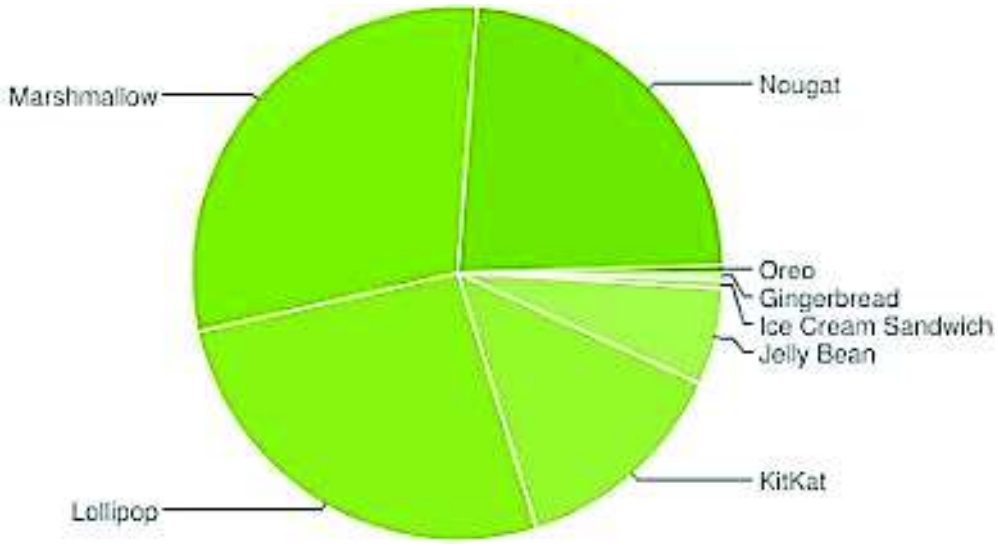

Figura 1.2. Dispositivos Android con porcentajes superiores al 0,1%, según la plataforma instalada [12]

1.2.5 ARQUITECTURA [13] - [14]

Android es una plataforma que contiene una pila de software donde se incluye un sistema operativo base, un *middleware*[12] y distintas aplicaciones básicas para el usuario tal como muestra la Figura 1.3.

Figura 1.3. Arquitectura de Android [14]

1.2.5.1 Núcleo de Linux

El kernel de Linux es la base de la plataforma Android, el cual permite que Android aproveche funciones de seguridad de Linux, la generación de subprocesos, la administración de memoria de bajo nivel gracias al *runtime* de Android (ART), el manejo de protocolos y el soporte de los *drivers* a los dispositivos.

[12] *Middleware*: software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos.

1.2.5.2 Capa de abstracción de hardware (HAL)

La capa de abstracción de hardware (HAL) brinda interfaces estándar que exponen las capacidades de hardware del dispositivo a la API del *framework* Java de nivel más alto, y consta de varios módulos de biblioteca que implementan una interfaz para un tipo específico de componente de hardware, como el módulo de la cámara o de *bluetooth*. Cuando el *framework* de una API realiza una llamada para acceder al hardware del dispositivo, el sistema Android carga el módulo de biblioteca para el componente de hardware en cuestión.

1.2.5.3 Runtime de Android (ART)

Es un entorno de ejecución de aplicaciones utilizado por Android, sucesor de la máquina virtual Dalvik, el cual lleva a cabo la transformación de la aplicación en instrucciones de máquina, que luego se ejecutan por entorno de ejecución del dispositivo, eliminando la necesidad de una máquina virtual o de interpretar el código en tiempo de ejecución generando mejoras en el desempeño de la aplicación y del dispositivo. Las funciones principales del ART son:

- Compilación *ahead-of-time* (AOT), que crea un archivo de compilación posterior a la instalación, el cual será utilizado al abrir la aplicación, evitando que la aplicación se compile continuamente, cada vez que ésta es ejecutada *y just-in-time* (JIT) para compilar el código cada vez que se inicia o se utilice otra parte de la aplicación.
- Recolección de elementos no usados (GC) optimizada.
- Mejor compatibilidad con la depuración, como un generador de perfiles de muestras dedicado, excepciones de diagnóstico detalladas e informes de fallos, y establecimiento de puntos de control de campos específicos.

Antes de Android 5.0, Dalvik era el *runtime* de ejecución del sistema operativo. Si la aplicación se ejecuta bien en ART, también debe funcionar en Dalvik, pero es posible que no suceda lo inverso. También se incluye un conjunto central de bibliotecas de tiempo de ejecución que proporcionan la mayor parte de la funcionalidad del lenguaje de programación Java y se incluyen algunas funciones del lenguaje Java 8, que la API del *framework* de Java usa.

1.2.5.4 Bibliotecas C/C++ nativas

Muchos de los componentes y servicios centrales del sistema Android, como el ART y la HAL, están basados en código nativo que requieren bibliotecas escritas en C y C++. Android proporciona la API del *framework* de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas a las aplicaciones. Por ejemplo, se puede acceder a OpenGL ES[13] a través del API Java OpenGL del *framework,* para agregar a la aplicación compatibilidad con dibujos y manipulación de gráficos 2D y 3D. Muchas de las librerías C/C++ utilizadas en varios componentes, utilizan proyectos de código abierto.

1.2.5.4.1 System C library

Derivación de la librería BSD de C estándar (libc), adaptada para dispositivos embebidos basados en Linux.

1.2.5.4.2 Media Framework

Librería derivada de OpenCORE[14] de PacketVideo que soporta *codecs* de reproducción y grabación de multitud de formatos de audio y vídeo e imágenes.

1.2.5.4.3 Surface Manager

Administra el acceso al subsistema de representación gráfica en 2D y 3D.

1.2.5.4.4 WebKit/Chromium

Soporta el navegador web utilizado Android y en la vista WebView. Desde la versión 4.4, WebKit ha sido reemplazada por Chromium/Blink, que es la base del navegador Chrome de Google.

1.2.5.4.5 SGL

Motor de gráficos 2D.

1.2.5.4.6 FreeType

Fuentes en bitmaps y renderizado vectorial.

---

[13] OpenGL ES (OpenGL *for Embedded Systems*): variante simplificada de la API gráfica OpenGL diseñada para dispositivos integrados, como teléfonos móviles, consolas de videojuegos, tabletas, PDA.

[14] OpenCORE: plataforma multimedia de código abierto Java.

1.2.5.4.7 Librerías 3D

Implementación derivada de OpenGL ES 1.0 API. Las librerías utilizan el acelerador de hardware 3D si está disponible, o el software altamente optimizado de proyección 3D.

1.2.5.4.8 SQLite

Motor de bases de datos relacionales potente y ligero, disponible para todas las aplicaciones.

1.2.5.4.9 SSL

Proporciona servicios de cifrado usando *Secure Socket Layer*.

1.2.5.5 API del *framework* de JAVA

El conjunto de funciones de Android está disponible mediante API escritas en el lenguaje Java, y son los cimientos necesarios para la creación de aplicaciones de Android y simplifican la reutilización de componentes del sistema, servicios centrales y modulares, tal como se citan a continuación.

- **Sistema de vista enriquecido y extensible (V*iew*):** Utilizado para compilar la interfaz del usuario (IU) de una aplicación, incluyen listas, cuadros de texto, cuadrículas, botones y un navegador web integrable.
- **Administrador de recursos (*Resource Manager*):** Brinda acceso a recursos sin código, como *strings*, gráficos y archivos de diseño almacenados.
- **Administrador de notificaciones (*Notification Manager*):** Permite que todas las aplicaciones muestren alertas personalizadas en la barra de estado.
- **Administrador de actividad (*Activity Manager*):** Administra el ciclo de vida de las aplicaciones y proporciona una pila de retroceso de navegación común.
- **Proveedores de contenido (*Content Providers*):** Permite que las aplicaciones accedan a datos desde otras aplicaciones o compartan sus propios datos.

1.2.5.6 Aplicaciones del sistema

Android incluye en la plataforma un conjunto de aplicaciones por defecto, tales como: correo electrónico, mensajería SMS, navegación en Internet, calendarios,

contactos, etc., que no tienen una característica especial en relación con aquellas que el usuario elije instalar; por este motivo, una aplicación externa puede convertirse en el navegador web, el sistema de mensajería SMS o, incluso, el teclado predeterminado del usuario, si así lo desea.

Las aplicaciones del sistema funcionan como aplicaciones para los usuarios y brindan capacidades claves a las cuales los desarrolladores pueden acceder desde sus propias aplicaciones. Por ejemplo, si en una aplicación se intenta entregar un mensaje SMS, no es necesario compilar esa funcionalidad, como alternativa, se puede invocar la aplicación de SMS que ya está instalada para entregar un mensaje al receptor.

1.2.6 MODELO DE SEGURIDAD [15]

La seguridad es un aspecto fundamental de todo sistema operativo, si se descarga una aplicación maliciosa de Internet y se instala en algún terminal móvil, esta podría tener la posibilidad de acceder a la información personal, tal como leer la lista de contactos, conocer la posición GPS, y enviar esta información vía Internet o enviarla mediante mensajería SMS. Para proteger a los usuarios, Android propone un esquema de seguridad, que no impone un sistema centralizado y controlado por una única empresa, fundamentándose en los siguientes pilares:

- Se puede aprovechar la seguridad que incorpora el sistema operativo Linux, de esta forma Android puede impedir que ciertas aplicaciones tengan acceso directo al hardware o interfieran recursos de otras aplicaciones.
- Las aplicaciones deben ser firmadas con un certificado digital que identificará a su autor, y que garantice que los archivos del fichero de la aplicación no hayan sido modificados. Si se requiere modificar la aplicación, esta tendrá que ser firmada nuevamente por el propietario de la clave privada. En Android no es necesario que un certificado digital sea firmado por una autoridad de certificación.
- Si se necesita que una aplicación tenga acceso a módulos del OS Android, comprometiendo su seguridad, se debe utilizar un modelo de permisos, de forma que el usuario conozca los riesgos del aplicativo antes de realizar la instalación de la aplicación.

- Todo aplicativo móvil a ser publicado en Google Play, está sujeto a una revisión basada en las políticas del programa para desarrolladores [16], antes de estar disponible para los usuarios.

1.2.6.1 Usuario Linux y acceso a ficheros

El OS Android crea una nueva cuenta de usuario Linux (*user ID*) por cada paquete APK[15] instalado y permanece constante durante toda su vida en el dispositivo, el cual sirve para proteger el acceso a recursos utilizados por otras aplicaciones.

Cualquier información almacenada por la aplicación será asignada al usuario Linux creado, por lo cual otras aplicaciones no tendrán acceso. Sin embargo, cuando se crea un fichero se pueden usar los modos **MODE_WORLD_WRITEABLE** y **MODE_WORLD_READABLE,** para permitir que otras aplicaciones puedan escribir o leer el fichero creado.

Aunque otras aplicaciones puedan escribir en el fichero, el propietario siempre será el usuario asignado a la aplicación que lo creó. Debido a que las restricciones de seguridad se garantizan a nivel de proceso, el código de dos aplicaciones, en circunstancias normales no puede ejecutarse en el mismo proceso. Para que dos aplicaciones sean ejecutadas en el mismo proceso sería necesario asignar un mismo usuario Linux a dos aplicaciones, para lo cual se podría utilizar el atributo *SharedUserId* en el archivo AndroidManifest.xml y ambas deben estar firmadas con el mismo certificado digital, consiguiendo que ambas aplicaciones sean tratadas como una sola para efectos de seguridad.

1.2.6.2 El esquema de permisos en Android [18 - 19]

Android define un esquema de permisos para proteger ciertos recursos y características especiales, por lo cual toda aplicación que acceda a estos recursos está obligada a declarar su intención de utilizarlos. Si una aplicación intenta acceder a un recurso del que no ha solicitado permiso, se generará una excepción de permisos y la aplicación será inmediatamente interrumpida.

---

15 APK (Android *application package*): es el formato de archivo del paquete utilizado por el sistema operativo Android para la distribución e instalación de aplicaciones móviles y middleware.

El usuario podrá examinar la lista de permisos que solicita la aplicación en el momento de instalación, lo cual le servirá para decidir si considera oportuno instalarla. A continuación, se muestra una lista con algunos de los permisos más utilizados en Android [17], los mismos que están agrupados por nivel de riesgo, como se puede observar en las **Error! Reference source not found.** y Tabla 1.3.

Tabla 1.2. Permisos peligrosos en Android [18] – 1 de 2

Tipo	Permiso	Descripción
Almacenamiento Externo	**WRITE_EXTERNAL_ STORAGE**	Modificar/eliminar almacenamiento USB (API 4). Permite el borrado y la modificación de archivos en la memoria externa.
	READ_EXTERNAL_ STORAGE	Leer almacenamiento USB (API 16). Permite leer archivos en la memoria externa.
Ubicación	**ACCESS_COARSE_ LOCATION**	Localización no detallada, basada en red (Cell-ID) y Wi-Fi.
	ACCESS_FINE_ LOCATION	Localización GPS detallada, basada en satélites GPS. También permite la localización ACCESS_COARSE_LOCATION.
Teléfono	**CALL_PHONE**	Permite realizar llamadas a números de teléfono directamente con cargos de paga y sin la intervención del usuario.
	READ_PHONE_ STATE	Permite consultar la identidad y el estado del teléfono. También para poner en pausa una aplicación cuando se recibe una llamada. Permite el acceso al número de teléfono, IMEI[16], IMSI[17] y al identificador único de 64 bits Google.
	READ_CALL_LOG WRITE_CALL_LOG	Permite leer y modificar el registro de llamadas telefónicas.
	ADD_VOICEMAIL	Permite crear nuevos mensajes de voz en el sistema.
	USE_SIP	Permite a tu aplicación usar el protocolo SIP[18]. (API 9).

[16] IMEI: identificador de teléfono GSM.
[17] IMSI: identificador de tarjeta SIM.
[18] SIP (*User Session Initial Protocol*): protocolo que sirve para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como video, voz y mensajería instantánea.

Tabla 1.2. Permisos peligrosos en Android [18] – 2 de 2

Tipo	Permiso	Descripción
Teléfono	**PROCESS_ OUTGOING_CALLS**	Permite a la aplicación controlar, modificar o abortar las llamadas salientes.
Mensajes de texto (SMS)	**SEND_SMS**	Permite a la aplicación enviar mensajes de texto SMS con cargo de paga.
	RECEIVE_SMS	Permite a la aplicación recibir y procesar SMS, modificando o borrando los mensajes recibidos.
	READ_SMS	Permite a la aplicación leer los mensajes SMS entrantes.
	RECEIVE_MMS	Permite monitorizar los mensajes multimedia entrantes.
	RECEIVE_WAP_ PUSH	Permite monitorizar los mensajes WAP PUSH entrantes.
Contactos	**READ_CONTACTS**	Permite leer información sobre los contactos almacenados).
	WRITE_CONTACTS	Permite modificar información sobre los contactos almacenados.
	GET_ACCOUNTS	Permiten acceder a la lista de cuentas almacenadas en el servicio de cuentas del dispositivo.
	READ_CALENDAR	Permite leer información del calendario del usuario.
	WRITE_CONTACTS	Permite escribir en el calendario del usuario, pero no leerlo.
Calendario	**READ_CALENDAR**	Permite leer información del calendario del usuario.
	WRITE_CONTACTS	Permite escribir en el calendario del usuario, pero no leerlo.
Cámara	**CAMARA**	Permite acceso al control de la cámara y a la toma de imágenes y vídeos, puede ser sin conocimiento del usuario.
Micrófono	**RECORD_AUDIO**	Permite acceso para grabar sonido desde el micrófono.
Sensores corporales	**BODY SENSORS**	Permite brindar acceso a los datos de sensores que monitorean al usuario.

Tabla 1.3. Permisos normales en Android [19] – 1 de 3

Tipo	Permiso	Descripción
Comunicaciones	**INTERNET**	Acceso a Internet sin límites, permitiendo establecer conexiones a través de Internet.
	ACCESS_NETWORK_ STATE	Permite conocer información sobre todas las redes.
	CHANGE_NETWORK_ STATE	Permite cambiar el estado de conectividad de redes.
	NFC	Control remoto de electrodomésticos, mediante un trasmisor infrarrojo NFC[19] (API 19).
Conexión WIFI	**ACCESS_WIFI_STATE**	Permite conocer el estado de las redes Wi-Fi disponibles.
	CHANGE_WIFI_STATE	Permite cambiar el estado de conectividad Wi-Fi.
	CHANGE_WIFI_ MULTICAST_STATE	Permite cambiar al modo Wi-Fi *Multicast* (API 4).
Bluetooth	**ACCESS_WIFI_STATE**	Permite conocer las redes Wi-Fi disponibles.
	BLUETOOTH	Permite a una aplicación conectarse con otro dispositivo Bluetooth, previamente emparejados.
	BLUETOOTH_ADMIN	Permite descubrir y emparejar dispositivos Bluetooth.
Consumo de batería	**WAKE_LOCK**.	Impide que el dispositivo entre en modo de suspensión.
	FLASHLIGHT	Permite encender la luz flash de la cámara.
	VIBRATE	Permite controlar de la vibración del dispositivo.
Aplicaciones	**RECEIVE_BOOT_ COMPLETED**	Ejecución automática al encender el teléfono. Permite a una aplicación recibir el anuncio *broadcast*.
	ACTION_BOOT_ COMPLETED	Ejecución cuando el sistema finaliza un inicio. La aplicación podrá ponerse en ejecución al arrancar el teléfono.

[19] NFC (*Near Field Communication*): tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos.

Tabla 1.3. Permisos normales en Android [19] – 2 de 3

Tipo	Permiso	Descripción
Aplicaciones	**BROADCAST_STICKY**	Enviar anuncios *broadcast* permanentes. Un *broadcast* permanente llegará a los *broadcast receivers* que actualmente estén escuchando, y a los que se instancien en un futuro.
	KILL_BACKGROUND_ PROCESSES	Permite invocar a *killBackgroundProcesses*(*String*). Al hacer esta llamada el sistema mata de inmediato a todos los procesos *background* asociados con el paquete indicado (API 9).
	REORDER_TASKS	Permite a una aplicación cambiar el orden de la lista de tareas.
	INSTALL_SHORTCUT UNINSTALL_SHORTCUT	Permite a una aplicación añadir o eliminar un acceso directo de la aplicación en el escritorio (API 19).
	GET_PACKAGE_SIZE	Permite a una aplicación conocer el tamaño de cualquier paquete.
	EXPAND_STATUS_ BAR	Permite a una aplicación expandir o contraer la barra de estado.
Configuraciones del sistema	**CHANGE_ CONFIGURATION**	Permite modificar la configuración del sistema.
	SET_WALLPAPER	Permite establecer fondo de pantalla en el escritorio.
	SET_WALLPAPER_ HITS	Permite a las aplicaciones establecer sugerencias de fondo de pantalla.
	SET_ALARM	Permite a la aplicación enviar una intención para poner una alarma o temporizador en la aplicación Reloj.
	SET_TIME_ZONE	Permite cambiar la zona horaria del sistema.
	ACCESS_ NOTIFICATION_ POLICY	Permite conocer y accede a la política de notificaciones del sistema (API 23).
Audio	**MODYFY_AUDIO_ SETTINGS**	Permite cambiar ajustes globales de audio.

Tabla 1.3. Permisos normales en Android [19] – 3 de 3

TIPO	PERMISO	DESCRIPCIÓN
Sincronización	**READ_SYNC_ SETTINGS**	Permite conocer si se tiene sincronización en segundo plano con alguna aplicación (como con un cliente de Gmail o Twitter).
	WRITE_SYNC_ SETTINGS	Permite registrar una aplicación como adaptador de sincronización (SyncAdapter).
	READ_SYNC_STATS	Permite leer estadísticas de sincronización.
Ubicación	**ACCESS_LOCATION_ EXTRA_COMMANDS**	Permite a una aplicación acceder a comandos adicionales de proveedores de localización.
Seguridad	**USE_FINGERPRINT**	Permite usar el hardware de reconocimiento de huella digital (API 23).
	DISABLE_KEYGUARD	Permite a las aplicaciones desactivar el bloqueo del teclado.

Hasta la versión Android 6.0, la declaración de permisos de la aplicación debía hacerse previamente en el archivo AndroidManifest.xml, antes de su instalación y ejecución en el dispositivo, concediendo o denegando el permiso a un grupo entero, sin importar si alguno de ellos está dentro de los permisos peligrosos. Por el contrario, a partir de la versión indicada, es posible declarar permisos de una aplicación en tiempo de ejecución, haciendo que el usuario se encargue de habilitar estos permisos para aquellos que son “peligrosos”.

1.2.7 MANIFIESTO DE LA APLICACIÓN

El archivo Android Manifest.xml [18] proporciona información esencial y necesaria de la aplicación al OS Android, para poder ejecutar el código de la aplicación. Realiza las siguientes acciones:

- Nombra el paquete de Java para la aplicación como un identificador único.
- Determina los procesos que alojan a cada componente de la aplicación.
- Declara los permisos que debe tener la aplicación para acceder a las secciones protegidas de una API e interactuar con otras aplicaciones.

- Declara los permisos que otras aplicaciones deben tener para interactuar con los componentes de aplicación.
- Describe los componentes de la aplicación, además nombra las clases que implementa cada componente de aplicación y publica sus capacidades, tal como mensajes *Intent*[20] con los que puede funcionar.
- Enumera las clases *Instrumentation* que proporcionan un perfil y otro tipo de información mientras la aplicación se ejecuta.
- Declara el nivel mínimo de la API Android que requiere la aplicación para ejecutarla correctamente.
- Enumera las bibliotecas con las que debe estar vinculada la aplicación.

El archivo AndroidManifest.xml contiene todos los elementos que se muestran en [18].

1.2.8 COMPONENTES DE LAS APLICACIONES [19]

El *framework* de aplicaciones de Android permite crear aplicaciones interesantes e innovadoras usando un conjunto de componentes reutilizables tales como se describen a continuación y se pueden observar en la Figura 1.4.

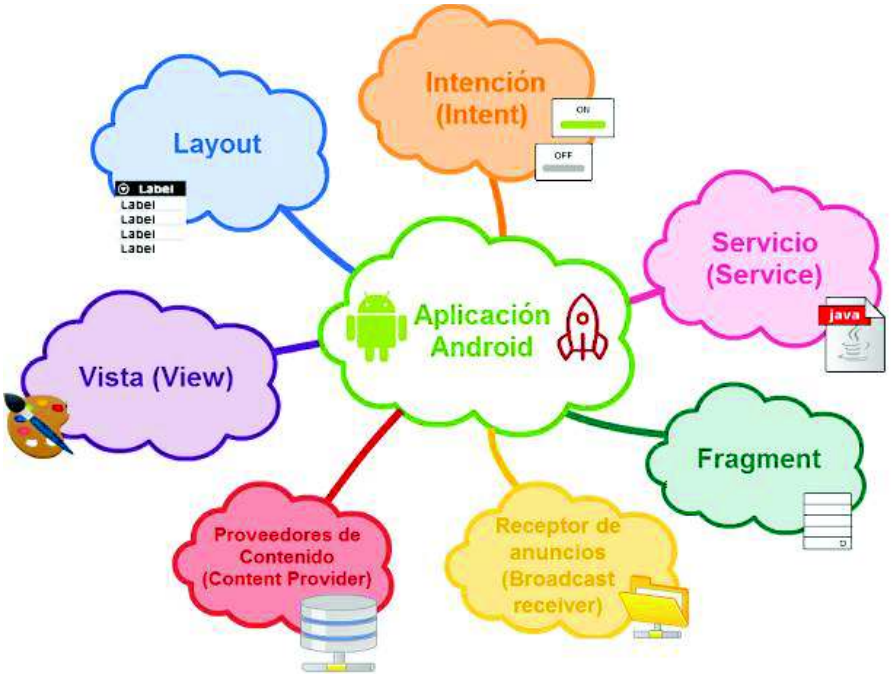

Figura 1.4. Componentes de una aplicación Android

[20] *Intent*: objeto que proporciona enlace de tiempo de ejecución entre componentes Android.

1.2.8.1 *Intents* y filtros de *Intents*

Una *Intent* es un objeto de acción que se puede utilizar para solicitar una acción de otro componente de aplicación, estos pueden ser internos o externos, tal como visualizar una página web, realizar una llamada telefónica, etc. y se utiliza cuando se requiera ejecutar un *Activity*, un *Service,* enviar un anuncio de *Broadcast* o*,* comunicarse con un *Service*. También se utilizan los *Intents* para el intercambio de información entre los componentes de aplicación.

1.2.8.1.1 Intents explícitos

Un *Intent* explícito se usa para iniciar un componente específico de la aplicación, como una actividad o un servicio particular en la aplicación mediante su nombre. Usualmente el programador usa un *Intent* explícito para iniciar un componente en su propia aplicación, porque conoce el nombre de clase de la actividad o el servicio que desea iniciar.

1.2.8.1.2 Intents implícitos

Un *Intent* implícito especifica una acción que puede invocar cualquier aplicación en el dispositivo que pueda realizar la acción. No se nombra el componente específicamente, pero se declara una acción general para realizar, lo que permite que un componente de otra aplicación la maneje, por ejemplo, si se desea mostrar al usuario una ubicación en un mapa, se puede usar un *Intent* implícito para solicitar que otra aplicación capaz muestre una ubicación específica en un mapa.

Un filtro de *Intents* es una expresión en el archivo Android Manifest.xml, que especifica el tipo de *Intents* que el componente podría recibir, por ejemplo, declarando un filtro de *Intents* para una actividad, permite que otras aplicaciones inicien directamente la actividad con cierto tipo de *Intent*. Así mismo, si no declara ningún filtro de *Intent* para una actividad, solo se la puede iniciar con un *Intent* explícito.

1.2.8.2 Activity

Un *Activity* es un componente de aplicación que contiene una pantalla con la que el usuario puede interactuar para realizar alguna acción, es decir la capa de

presentación de la aplicación. Las *activities* hacen uso de componentes tipo *View* para mostrar elementos de la interfaz gráfica al usuario.

1.2.8.3 Service

Un *Service* es un componente de aplicación que puede realizar operaciones de larga ejecución en segundo plano, sin proporcionar una interfaz de usuario. Por ejemplo, un componente de aplicación puede iniciar un servicio y siempre lo ejecutará en segundo plano, aunque el usuario cambie a otra aplicación.

1.2.8.4 Content Provider

Los *Content Provider* administran el acceso a un conjunto de datos estructurados, encapsulándolos, y proporcionando mecanismos para definir la seguridad de la información, siendo una interfaz estandarizada que conecta información en un proceso con código que se ejecuta en otro proceso.

1.2.8.5 Widgets de aplicaciones

Los *Widgets* de aplicaciones son vistas de aplicaciones en miniatura que se pueden incrustar en otras aplicaciones y recibir actualizaciones periódicas.

1.2.8.6 View

Un *View* es un elemento de la interfaz de usuario de una aplicación, descendiente de la clase Android.*View*. Por ejemplo, cuadros de texto, botones, imágenes, etc.

1.2.8.7 Layout

Un *Layout* es un contenedor de objetos *View* agrupados de una forma determinada, que controla su posición y comportamiento, también descendiente de la clase Android.*View*. Un *Layout* puede contener a otro *Layout*.

1.2.8.8 Fragment

Un *Fragment* es la unión de varios objetos *View* para crear un bloque funcional de la interfaz de usuario. Estos componentes aparecieron desde la versión 3.0 de Android debido a la aparición de las tabletas en las cuales los aplicativos debían soportar pantallas más grandes.

1.2.8.9 Broadcast receiver

Es el componente de aplicación destinado a detectar y reaccionar a determinados mensajes o eventos globales (*Broadcast Intents*) generados por el sistema, por ejemplo: alerta SMS, llamada entrante, batería baja, o por otras aplicaciones.
Los *Broadcast Receiver* no tiene interfaz de usuario, aunque si pueden iniciar un *Activity*.

1.2.9 PROCESOS Y SUBPROCESOS [20]

Por defecto, todos los componentes de una aplicación Android (<*activity*>, <*service*>, <*receiver*> y <*provider*>) se ejecutan en un mismo proceso, sin embargo, mediante el archivo AndroidManifest.xml se puede controlar un proceso de un determinado componente de ser necesario. Cada componente admite un atributo android:process, que puede determinar un proceso en el cual el componente debe ejecutarse, de forma única o compartida con otros procesos.

Android, trata de mantener el proceso de una aplicación el mayor tiempo posible, pero eventualmente se requiere eliminar procesos antiguos, con la finalidad de liberar memoria que necesitan procesos nuevos u otros más importantes. Para determinar que procesos mantener o finalizar, Android coloca cada proceso dentro de una “jerarquía de importancia”, según los componentes que se ejecutan en el proceso y en el estado de éstos, tal como se puede observar en la Figura 1.5.

Figura 1.5. Procesos de las aplicaciones Android.

Aquellos procesos con importancia baja son aquellos que se eliminan, luego los siguientes de menor importancia, y así sucesivamente, hasta recuperar los recursos del sistema

1.2.9.1 Proceso activo o en primer plano

Proceso requerido por la acción que está realizando el usuario en ese momento. Se considera en este estado si cualquiera de las siguientes condiciones es verdadero:

- Aloja una *Activity* con la que el usuario está interactuando.
- Aloja un *Service* que está vinculado con la *Activity* con la que el usuario está interactuando.
- Aloja un *Service* que se está ejecutando en primer plano.
- Aloja un *Service* que está ejecutando una *callback*[21] del ciclo de vida (onCreate(), onStart() o onDestroy()).
- Aloja un *BroadcastReceiver* que ejecuta su método onReceive().

Usualmente existen pocos procesos en primer plano en un momento determinado; estos se cancelan únicamente como último recurso, cuando el dispositivo ha alcanzado un estado de paginación de memoria, por lo que finalizar estos procesos es necesario para mantener la capacidad de respuesta de la interfaz del usuario.

1.2.9.2 Procesos visibles (*Visible process*)

Proceso que no tiene ningún componente en primer plano, pero puede afectar lo que el usuario observa en la pantalla. Se considera en este estado si es verdadera alguna de estas condiciones:

- Aloja una *Activity* que no esté en primer plano y sea visible para el usuario, por ejemplo, si la *Activity* del primer plano inicia un cuadro de diálogo que permite la visualización de la *Activity* principal detrás del cuadro.
- Aloja un *Service* que está vinculado con una actividad visible o en primer plano.

[21] *Callback*: métodos y especificaciones que permitirán implementar un paradigma de programación de tipo IoC (*Inversion of Control*), en el que será el objeto que está realizando la tarea el que se pondrá en contacto con el objeto que lo ha invocado, a través del *callback* cuando finalice de realizar la funcionalidad especificada en el método.

1.2.9.3 Procesos de servicio (*Started service process*)

Proceso que ejecuta un servicio que se ha iniciado con el método startService() y no es ninguna de las dos categorías anteriores. Los procesos de servicio no están relacionados directamente con lo que el usuario observa, y generalmente realizan procesos como copiar elementos del dispositivo a otro, reproducir música en segundo plano o descargar información, por este motivo el sistema mantiene estos procesos en ejecución a menos que no haya suficiente memoria en el dispositivo.

1.2.9.4 Procesos inactivos o en segundo plano (*Background process*)

Proceso que tiene un *Activity* que actualmente no es visible para el usuario, que puede ser finalizado por el sistema en cualquier momento para recuperar memoria que será utilizada por un proceso en primer plano, visible o de servicio. Estos procesos se mantienen en una lista LRU (menos usado recientemente) con la finalidad que la *Activity* recientemente usada sea la última en finalizar.

1.2.9.5 Procesos vacíos (*Empty process*)

Es un proceso que no aloja ningún componente activo de la aplicación, y se genera con la finalidad de tener memoria caché disponible para una próxima activación y de esa forma mejorar su tiempo de respuesta.

1.2.9.6 Subprocesos

Android crea un subproceso (hilo) de ejecución denominado “principal” cuando una aplicación se inicia, este subproceso está a cargo de distribuir eventos a los widgets correspondientes de la interfaz de usuario, incluidos los eventos de dibujo, e interactúa entre la aplicación y los componentes del kit de herramientas de la IU de Android (componentes de los paquetes android.widget y android.view).

El sistema no crea un subproceso independiente para cada componente de la aplicación, al contrario, todos sus componentes se ejecutan dentro de un mismo proceso en el hilo principal, por lo cual, los métodos que responden a eventos del sistema siempre se ejecutan en este hilo principal o en la interfaz del usuario.

1.2.9.7 Comunicación entre procesos

Android ofrece un mecanismo para comunicación entre procesos (IPC) mediante llamadas de procedimiento remoto (RPC), en el que una *Activity* u otro componente

invoca a un método, pero este se ejecuta de manera remota (en otro proceso), y los resultados son devueltos al emisor. Para realizar una IPC, la aplicación debe vincularse con un servicio mediante bindService().

1.2.10 DESARROLLO Y ESTRUCTURA DE PROYECTOS ANDROID [21]

Android por ser un sistema operativo *open source*, permite el acceso a su código fuente, es muy flexible y personalizable, haciendo que sus aplicaciones sean desarrolladas generalmente en el lenguaje de programación Java, mediante Android *Software Development Kit* (Android SDK) y Android Studio que es un IDE[22] que tiene incluido el SDK de Android.

Todas las aplicaciones se encuentran comprimidas en formato APK y pueden instalarse en la mayoría de los dispositivos móviles desde cualquier explorador de archivos. En Android, cada módulo[23] está formado por un descriptor de aplicación (*manifest*), el código fuente en lenguaje Java (java), ficheros de recursos (res) y para la construcción del módulo (*Gradle Scripts*).

A continuación, se procede a revisar la estructura de un proyecto desarrollado en Android Studio. La estructura será muy similar para cualquier aplicación, independiente de su complejidad o tamaño, y se muestra en la Figura 1.6.

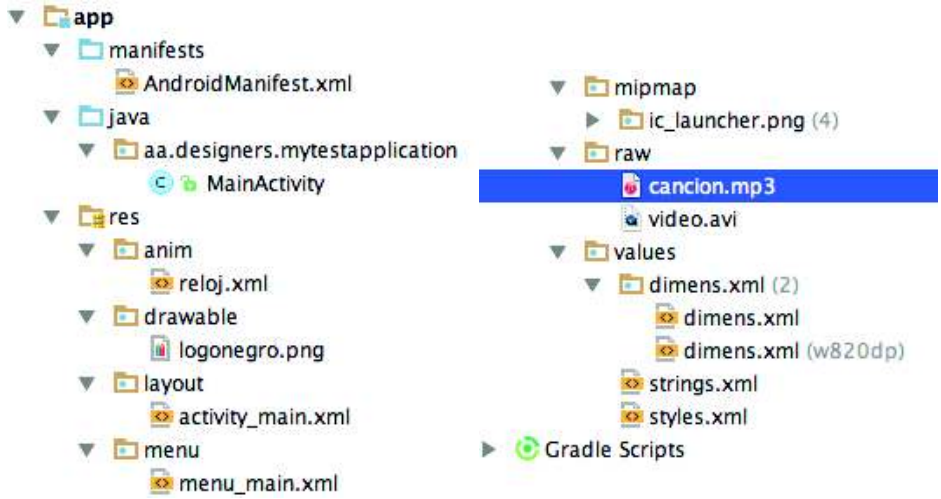

Figura 1.6. Estructura de un proyecto en Android Studio [21]

[22] IDE (*Integrated Development Environment*): aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.
[23] Módulo Android: librería que encapsula funcionalidades o unidad de funcionalidad para test unitario.

La carpeta raíz representa la aplicación Android, llamada generalmente "app", y está conformada por las secciones que se describen a continuación.

1.2.10.1 App/manifests

Contiene la definición de los aspectos principales de la aplicación, por ejemplo, componentes, identificación, o permisos necesarios para su ejecución.

En el archivo AndroidManifest.xml se pueden encontrar las siguientes secciones:

- El *package, es el* identificador de la aplicación y se define en el atributo *package* de la etiqueta *manifest*.
- La etiqueta *application,* especifica el inicio de la descripción de la aplicación y sus componentes.
- El atributo *allowBackup* indica si los datos de la aplicación pueden resguardarse y restaurarse, por ejemplo, con los comandos *adb backup* y *adb restore*.
- El icono y el nombre de la aplicación se introducen con los atributos *icon* y *label* de la etiqueta *application*.
- La opción *supportRtl,* indica si la aplicación soporta los idiomas que se escriben de derecha a izquierda, por ejemplo, el árabe, japonés.
- La opción *theme* indica el tema utilizado por la aplicación.
- La descripción *Activity* de la aplicación incluye:
    - El nombre de la clase que implementa *Activity*.
    - Los filtros de la *Activity*:
        - **MAIN**: indica que se trata de la actividad principal de la aplicación.
        - **LAUNCHER**: indica que esta actividad está presente en el arranque de la aplicación.

1.2.10.2 App/java

Contiene dos sub-paquetes, el primero representa el código fuente de la aplicación (*Activitys, Services*, código de negocio, clases auxiliares, etc.), y el segundo el código de pruebas.

1.2.10.3 App/res

Esta carpeta sirve para almacenar todos los recursos que se utilicen en la aplicación, se puede tener recursos diferentes en función de la resolución, de la orientación o del idioma del dispositivo. Los recursos son pre-compilados y añadidos al archivo R.java, el cual contiene referencias a todos los elementos presentes en la carpeta res, la cual está compuesta por las siguientes secciones.

- **Anim/Animator:** Contiene archivos XML que definen las animaciones.
- **Drawable:** Contiene imágenes y otros elementos gráficos. Se puede definir distintos recursos dependiendo de la resolución y densidad de la pantalla, pudiendo dividirse en otras subcarpetas.
- **Layout:** contiene todas las vistas de la aplicación (modos horizontal, vertical y tableta).
- **Menu:** Contiene archivos XML que definen las plantillas de los menús del proyecto.
- **Mipmap:** contiene los iconos de la aplicación para las distintas densidades de pantalla existentes**.**
- **Raw:** contiene los archivos multimedia descomprimidos o recursos adicionales, generalmente en formato distinto a XML.
- **Values**: contiene todos los archivos que contienen datos o valores que se usan en la aplicación (*strings*, *arrays*, *colors, dimensions, styles*, etc.).
    - **Dimens.xml:** contiene los tipos de resoluciones que podrá tomar la aplicación compilada.
    - **Strings.xml**: contiene las cadenas de texto de nuestra aplicación.
    - **Styles.xml:** contiene la definición de los estilos de nuestro proyecto.
    - **Color.xml:** contiene la definición de los colores que usaremos en el diseño de la aplicación.

1.2.10.4 Gradle Scripts

Contiene la información necesaria para la compilación del proyecto, tal como, la versión mínima de Android que soporta la aplicación, versión del SDK de Android, referencias a las librerías externas utilizadas, etc.

1.3 CAPAS DE SEGURIDAD EN DISPOSITIVOS MÓVILES [22]

Para poder analizar la seguridad de manera eficiente, se ha establecido seis capas de análisis como se muestran en la Figura 1.7, esto debido a que actualmente se han aumentado fallas de seguridad de información, gracias a la evolución de los móviles y sus sistemas operativos, y a personas mal intencionadas que aumentan su interés en obtener información confidencial de los usuarios.

Figura 1.7. Modelo de Seguridad por Niveles [22]

1.3.1 HARDWARE

En esta capa se evalúan los elementos que componen los terminales móviles utilizados, realizando de forma detallada un inventario de las características de memoria, procesador, pantalla, antenas, etc., con la finalidad de definir los riesgos y vulnerabilidades y establecer posibles puntos de ataque para minimizar las posibilidades de penetración a los equipos a través del hardware.

1.3.2 SISTEMA OPERATIVO

El sistema operativo se encuentra entre el hardware y las aplicaciones de software. Las principales amenazas que existen en esta capa son causadas por errores en el aislamiento de los recursos o el uso de versiones no oficiales de los sistemas operativos móviles, denominadas ROM. Los privilegios de usuarios, el aislamiento de procesos[24] y las actualizaciones, son los mecanismos de seguridad más importantes que se deben tomar en cuenta en esta capa.

[24] Aislamiento de procesos: aislar la ejecución de cada aplicación para no interferir en el funcionamiento de otras, incluso mediante un sistema de permisos.

1.3.3 ALMACENAMIENTO

Se debe prestar atención a la información que se almacena en las memorias internas y externas del terminal móvil, incluyendo bases de datos, archivos del usuario, binarios de los sistemas operativos, etc., por lo cual, definir procesos de validación, autenticación, autorización de accesos, cifrado de datos almacenados en memoria RAM y de archivos de internet, permisos de lectura/escritura son los mecanismos de seguridad más importantes que se deben tomar en cuenta en esta capa.

1.3.4 COMUNICACIONES

El uso de las comunicaciones especialmente las inalámbricas presentan una serie de ventajas, por ejemplo, la escalabilidad y la movilidad, sin embargo, en temas de seguridad presentan un vacío de seguridad, debido a que la información que viaja a través de este medio es más difícil de proteger, por lo cual es susceptible de vulnerabilidades y ataques, como se muestran a continuación.

- **Denegación de servicio:** proceso en la que el atacante consigue una interrupción o un retraso en el funcionamiento de servicios.
- ***Masquerading***: proceso por el cual el atacante suplanta la identidad de algún ente del sistema, con la finalidad de tener acceso al sistema, incidiendo directamente en los procesos de autenticación.
- ***Eavesdropping***: proceso por el cual el atacante obtiene información de una comunicación en la que no es ni emisor ni receptor. Un ataque de *eavesdropping* puede dar lugar a un posterior ataque de *masquerading*.
- **Confidencialidad de posicionamiento:** proceso por el cual el atacante obtiene la posición física de un dispositivo móvil usando diferentes técnicas.

Las técnicas de prevención más habituales que se aplican al medio físico son las de difusión del espectro[25], que también sirven para atenuar los ataques de *jamming*[26], la autenticación por medio del modelo reto-respuesta, con mensajes aleatorios e incorporación de métodos autenticación de dos etapas, y uso de criptografía.

---

[25] Difusión del espectro: fraccionamiento de la señal de radio para transmitirla de manera imperceptible por diferentes frecuencias.

[26] *Jamming*: obstrucción o perturbación generada intencionalmente en una comunicación con el objetivo de modificar y dificultar el intercambio de comunicaciones.

1.3.5 APLICACIONES

Las aplicaciones son uno de los componentes más importantes de los dispositivos móviles, por lo cual, es indispensable revisar las medidas de seguridad desarrolladas para que no puedan desestabilizar el sistema operativo o ser susceptible de vulnerabilidades que puedan afectar a la privacidad de los usuarios. Dentro del software hay varias formas de crear falencias de seguridad o vulnerabilidades, dentro de las cuales se tiene al *malware*, el cual es una aplicación de software o fracción de código con objetivo malicioso, que se ejecuta sin consentimiento o conocimiento del propietario para obtener información personal.

A continuación se lista los principales tipos de *malware* [23] .

- **Virus**: Programas informáticos cuyo objetivo es alterar el funcionamiento correcto de dispositivos. Generalmente, infectan archivos (normalmente ejecutables) del sistema con la finalidad de modificarlos o dañarlos, introduciendo código malicioso.
- **Gusano informático**: Programa informático tipo virus cuya característica principal es su capacidad para replicarse a sí mismo. A diferencia de los virus, tienen la capacidad de infectar sin la intervención del usuario.
- **Troyanos**: Software malicioso que se presenta aparentemente como un programa inofensivo y legítimo, pero al ser ejecutado se brinda al atacante acceso remoto al equipo infectado, con el objetivo principal de robo de datos personales y poder administrar remotamente de manera no autorizada recursos ajenos. Un virus suele ser destructivo, mientras un troyano accede y controla el dispositivo anfitrión, pero tratará de pasar inadvertido.
- **Spyware**: Es un software que permite recolectar información de forma no autorizada, para después transmitirla sin el conocimiento del propietario de la información a una entidad externa.
- **Backdoors**: Es un programa malicioso, usado para proporcionar al atacante acceso remoto no autorizado al dispositivo, explotando las vulnerabilidades del sistema operativo, trabajando en segundo plano y tiene la característica de poder ejecutar cualquier posible acción en el dispositivo comprometido.

- **Rootkits**: Es una complicación de programas utilizados por un atacante, para evitar ser detectado mientras se busca obtener acceso a un dispositivo de forma no autorizada, mediante la instalación de un módulo de *kernel* o el reemplazo de archivos o bibliotecas del sistema.
- **Keyloggers**: Es un dispositivo hardware o aplicaciones que graban o registran la pulsación de teclas y clics del ratón, con la finalidad de que la información recolectada sea utilizada posteriormente por el atacante.
- **Stealers**: Son programas o aplicaciones maliciosas del tipo troyano, introducidas a través de Internet, que obtienen información confidencial de los usuarios de forma fraudulenta, pero únicamente la que se encuentra guardada en el equipo y posteriormente la envían al atacante.

Para lograr prevenir este tipo de vulnerabilidades, es importante el uso de aplicaciones que están dentro del mercado oficial de aplicaciones; tener actualizado el navegador web que se utiliza en el dispositivo, tener cuidado de las páginas en las que se navega, y usar aplicaciones de seguridad que nos permitan aplicar métodos de autenticación más restrictivos, cifrado, aplicaciones antivirus, etc.

1.3.6 USUARIOS ATACANTES

En esta capa se toma en cuenta las intrusiones físicas que se puede realizar sobre el dispositivo, es decir cuando alguna persona ha tenido acceso a un dispositivo de manera física o tangible, con la finalidad de acceder a información sensible directamente del dispositivo. La instalación de algún programa que permita un acceso remoto a la información, o el adueñarse del equipo sin conocimiento o consentimiento de los usuarios.

Para evitar este tipo de ataques se puede establecer controles al acceso físico del dispositivo o de la información, que incluyan el establecer mecanismos de autenticación más robustas, evitar el uso de contraseñas débiles o fáciles de deducir, cifrar la información dentro del dispositivo, eliminar información de manera remota en casos de emergencias e instalar aplicaciones que permitan su rastreo.

1.4 ANÁLISIS DE SEGURIDAD DE APLICACIONES ANDROID

Una de las principales prioridades de los diferentes análisis de seguridades es ayudar a estandarizar y difundir metodologías de pruebas de aplicaciones móviles, para asegurarse que al momento del desarrollo de la aplicación no se introduzcan fallas de seguridad, y evaluar adecuadamente la superficie de ataque[27] de la aplicación móvil. La evaluación ideal combina análisis dinámico, análisis estático y análisis forense para garantizar que la mayoría de la superficie de ataque de la aplicación móvil sea cubierta. A continuación, se presenta un estudio de las metodologías de análisis de vulnerabilidades seleccionadas para este estudio.

1.4.1 OPEN WEB APPLICATION SECURITY PROJECT - OWASP [26]

El proyecto OWASP *Mobile Security* es un recurso destinado a desarrolladores y equipos de seguridad, para construir y mantener seguras las aplicaciones móviles, cuyo objetivo es clasificar riesgos de seguridad y proporcionar controles en el desarrollo para reducir la probabilidad de explotación de las aplicaciones por parte de terceros.

El análisis de la guía está principalmente enfocado en la capa aplicación y en las secciones de código fuente de la aplicación en las que el desarrollador está inmerso, a pesar de que también se toma en cuenta la plataforma móvil y los riesgos asociados a los controles y sus amenazas, así como también en la infraestructura de comunicación de las aplicaciones y el servidor.

Actualmente OWASP *Mobile Security* es un trabajo en progreso, el cual toma como procedimiento para el análisis de riesgos de seguridad el top 10 de controles móviles y principios de diseño, considerando en su metodología tres secciones.

- **Recopilación de información:** En esta sección se llevará a cabo un reconocimiento de la aplicación para identificar la magnitud y alcance de la aplicación, para lo cual, se necesita la recopilación de información que ayudará a identificar la infraestructura que soporta la aplicación y sus posibles vectores de ataque.

[27] Superficie de Ataque[24]: puntos de entrada o de salida que pueden ser utilizados por un usuario no autorizado para intentar introducir, modificar o extraer información de nuestros sistemas informáticos.

- **Análisis estático:** En esta sección se llevará a cabo un análisis del código fuente de la aplicación móvil en bruto, código descompilado o desensamblado mediante el uso de ingeniería inversa.
- **Análisis dinámico**: En esta sección se ejecuta una aplicación en el dispositivo o en un emulador para evaluar la comunicación existente entre procesos, realizar un análisis forense del sistema de archivos, evaluar las dependencias de servicios remotos y con los que la aplicación se comunica.

1.4.1.1 Top 10 de controles móviles y principios de diseño [27]

A continuación, se muestran los lineamientos a los cuales hacen referencia los controles que OWASP recomienda para la clasificación de riesgos de seguridad. Si se necesita conocer a detalle las recomendaciones o principios de diseño de cada control se puede revisar en el Anexo 17.

- **Identificar y proteger datos confidenciales en el dispositivo móvil – C1**

Riesgos: Almacenamiento inseguro de datos sensibles, se pueden generar ataques a teléfonos desmantelados debido a que los dispositivos móviles tienen un mayor riesgo de pérdida o robo.

- **Manejo de credenciales "*password*" seguras en el dispositivo – C2**

Riesgos: *spyware*, vigilancia, *malware* financiero. Las credenciales de un usuario, en caso de robo, no solamente proporcionan el acceso no autorizado al servicio de *backend*[28] móvil, sino que también pueden comprometer potencialmente muchos otros servicios y cuentas utilizadas por el usuario.

- **Garantizar que los datos sensibles están protegidos durante el transporte – C3**

Riesgos: Ataques de suplantación de red, vigilancia. Los datos sensibles que pasan a través de canales no seguros podrían ser interceptados.

- **Implementar correctamente autenticación, autorización y gestión de sesiones – C4**

Riesgos: Individuos no autorizados podrían obtener acceso a los datos o sistemas

[28] *Backend* (Lado del servidor): es la labor de ingeniería que compone el acceso a bases de datos y generación de plantillas/lógica del lado del servidor.

sensibles al eludir los sistemas de autenticación o mediante la reutilización de *tokens*[29] o *cookies*[30] válidas.

- **Mantener las API *backend* (servicio) y plataforma (servidor) seguros – C5**

Riesgos: Ataques contra los sistemas de *back-end* y la pérdida de datos a través de almacenamiento en la nube. La mayoría de las aplicaciones móviles interactúan con las API del servidor utilizando servicios web o protocolos propietarios.

- **Integración de datos seguros con servicios y aplicaciónes de terceros – C6**

Riesgos: Fuga de datos. Los usuarios pueden instalar aplicaciones que pueden ser maliciosas y pueden transmitir datos personales u otros con fines maliciosos.

- **Prestar especial atención en la recolección, almacenamiento y uso consentido de los datos del usuario – C7**

Riesgos: Divulgación no intencional de información personal o privada, procesamiento de datos ilegal.

- **Implementar controles para evitar el acceso no autorizado a recursos pagados (billetera, SMS, llamadas telefónicas, etc.) – C8**

Riesgos: Las aplicaciones brindan acceso automático mediante programación a llamadas telefónicas de tarificación adicional, SMS, datos *roaming*[31], pagos NFC[32], etc. Las aplicaciones con acceso privilegiado a estas API deben tener especial cuidado para evitar su abuso financiero.

- **Garantizar la distribución/aprovisionamiento seguro de aplicaciones móviles – C9**

Riesgos: El uso de prácticas seguras de distribución es importante para mitigar todos los riesgos descritos en el Top 10 de riesgos móviles OWASP y en el Top 10 de riesgos ENISA.

---

[29] *Token*: es una cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.
[30] *Cookie*: contenedor de datos que los sitios webs y el navegador utilizan con el propósito de almacenar información de los usuarios.
[31] *Roaming* (Itinerancia): función relacionada con la capacidad de un dispositivo para moverse de una zona de cobertura a otra.
[32] NFC - *Near field communication*: tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos, principalmente para hacer compras de bienes y servicios.

- **Revisar cuidadosamente cualquier interpretación de código de errores en tiempo de ejecución – C10**

Riesgos: La interpretación en tiempo de ejecución del código puede dar una oportunidad para que las partes no confiables proporcionen información no verificada. Por ejemplo, niveles adicionales en un juego, *scripts*, cabeceras de SMS interpretadas. Esto ofrece la oportunidad de que el *malware* evite los controles proporcionados por las tiendas de aplicaciones, y puede recaer en ataques de inyección que conducen a la fuga de datos, vigilancia, al *spyware*, y *diallerware*.

1.4.1.2 Top 10 de los riesgos de seguridad móvil – OWASP [27].

A continuación, se presenta el top 10 de riesgos de seguridad que OWASP menciona que se pueden presentar durante el desarrollo de aplicaciones móviles.

- **Uso inadecuado de la plataforma – M1:** Las plataformas móviles brindan una variedad de servicios, desde la autenticación hasta la comunicación segura de datos, y el almacenamiento de información confidencial en el dispositivo, haciendo uso de componentes clave del sistema operativo como el TouchId, keychain o cualquier otro control de seguridad que se implemente para la ejecución de una aplicación; esta categoría incluye las vulnerabilidades que podrían surgir si la seguridad de la plataforma no se usa correctamente.

  La falta de uso de esas características podría permitir que un atacante adquiera datos confidenciales presentes en el dispositivo o mientras la aplicación interactúa con el servidor *back-end*. Esto inclusive podría permitir que un atacante suplante al usuario y realice operaciones maliciosas que podría permitirle hacerse cargo de la aplicación o el dispositivo.

- **Combinación de almacenamiento inseguro de información – M2:** Esta categoría incluye vulnerabilidades relacionadas con información confidencial almacenados en el dispositivo en el entorno de la aplicación o en la tarjeta SD, o cualquier información que se filtre por algún canal sin el conocimiento del desarrollador, mediante datos del sistema operativo o cachés de teclado, logs con información detallada, almacenamiento en caché del portapapeles o caché de URL simple.

Si los datos almacenados de forma insegura son robados, podría conducir a una violación de la privacidad, robo de identidad o afectar la reputación del usuario.

- **Comunicación insegura – M3:** Esta categoría cubre las vulnerabilidades que pueden permitir que un atacante robe o modifique la comunicación entre dispositivos móviles, dispositivos y servidores o cualquier otro dispositivo de hardware, utilizando canales de comunicación como TCP / IP, GSM, 3G, 4G, Bluetooth, NFC, Wi-Fi, etc. Si un atacante realiza un ataque Man in the Middle, información confidencial puede ser robada o modificada durante la comunicación sin que el usuario lo sepa.
- **Autenticación no segura – M4:** Esta categoría cubre las vulnerabilidades relacionadas con la autenticación del usuario o con una administración de sesión débil. La aplicación no debe tener ningún método de autenticación más débil que una aplicación web normal, los identificadores del dispositivo, como dirección Mac, UDID, IMEI u otro identificador de hardware, no deben usarse para identificar la identidad o sesión del usuario. Una gestión de sesión segura incluye tokens de sesión no predecibles, vida útil de sesión limitada y desconectar al usuario en caso de un posible ataque. Si la sesión se ve comprometida, un atacante puede obtener acceso a información sensible del usuario o realizar operaciones que solo estarían disponibles para usuarios restringidos.
- **Criptografía insuficiente – M5:** Esta categoría cubre todas las vulnerabilidades que surgen cuando la aplicación intenta encriptar o proteger información usando algún tipo de sistema de encriptación, pero existe alguna debilidad que podría permitir que el sistema sea comprometido y que se filtre la información protegida. Esto incluye vulnerabilidades relacionadas con las contraseñas almacenadas, implementación de llaves encriptadas, tal como la fortaleza de su cifrado. Si hay una implementación criptográfica débil, un atacante puede obtener acceso a información confidencial que se supone están protegidos por la aplicación.
- **Autorización insegura – M6:** Esta categoría cubre vulnerabilidades relacionadas con el módulo de autenticación de la aplicación, cuando un

usuario no puede acceder a la aplicación con sus respectivas credenciales, sin embargo, existe debilidades en el sistema que permite a un atacante ver los datos que pertenecen a otro usuario. La autorización insegura se genera por una confianza excesiva del servidor en las solicitudes del cliente, si el servidor supone que todas las solicitudes enviadas por la aplicación son correctas y proporciona datos sin comprobaciones de autorización adicionales, un atacante puede realizar operaciones que de lo contrario no tendrían privilegios. Esto podría generar una violación grave a la información confidencial de los usuarios.

- **Problemas de calidad del código de cliente – M7:** Esta categoría maneja todos los problemas a nivel de código que puede existir en una aplicación móvil. Estos problemas incluyen código que puede generar desbordamientos de búfer, problemas de acceso a la memoria del dispositivo o ataques de cadenas que son específicos del lenguaje de programación en uso.

  Aunque este tipo de vulnerabilidades son difíciles de explotar, si un atacante puede hacerlo, se podría ejecutar código remoto sin restricciones en el dispositivo del usuario y de esa forma obtener acceso a cualquier información que se encuentre en el dispositivo. La explotación también podría permitirle a un atacante acceder a información confidencial como claves de cifrado, contraseñas e información API sensible desde la memoria del dispositivo.

- **Manipulación del código – M8:** Esta categoría cubre las vulnerabilidades que podrían surgir si un atacante pudiera modificar el recurso binario o local de la aplicación descargada desde la tienda de aplicaciones. Esto podría incluir una variedad de escenarios como: el parcheo de aplicaciones, manipulaciones de clases y métodos, etc. Si un atacante puede realizar un *bypass* de código mediante parcheo de la aplicación, puede obtener acceso a funciones premium o modificar la aplicación para que contenga malware.

- **Ingeniería inversa – M9:** Es uno de los métodos más comunes utilizados para descifrar aplicaciones y obtener acceso al código fuente, esto podría permitir accesos privilegiados en la aplicación, las cuales son extremadamente peligrosas. Esto podría incluir el acceso a métodos ocultos, algoritmos de

cifrado, secretos codificado, etc. La ingeniería inversa permite a un atacante obtener acceso a la propiedad intelectual del aplicativo, el mismo que incluye código fuente, constantes criptográficas, claves API, algoritmos de cifrado, e inclusive características de versiones futuras de la aplicación que se encuentran ocultas.

- **Funcionalidad Extraña – M10:** Esta vulnerabilidad es el resultado de la generación de pruebas deficientes durante la fase de desarrollo de la aplicación, debido a que los desarrolladores suelen agregar puertas traseras para obtener aplicaciones más rápidas y el acceso a datos gestionados por el aplicativo. Si estas funciones ocultas y puertas traseras son encontradas por un atacante, este puede aprovecharse para comprometer la aplicación completa y obtener acceso a las secciones de la aplicación que deberían haber sido restringidas.

1.4.2 OPEN ANDROID SECURITY ASSESSMENT METHODOLOGY – OASAM [28]

OASAM es una metodología para el análisis de seguridades, cuyo objetivo es ser una estructura referencial, de taxonomía completa y consistente, para el análisis de vulnerabilidades de aplicaciones Android, y sirve de apoyo a desarrolladores y encargados de buscar vulnerabilidades en aplicaciones, de acuerdo con los controles de seguridad presentes en las secciones descritas a continuación. Si se necesita conocer a detalle los lineamientos para el análisis de vulnerabilidades definido por esta metodología, se puede revisar en el Anexo 18.

- **OASAM-INFO - *Information Gathering* (Obtención de información y definición de superficie de ataque) – C1:** En esta sección se define la superficie de ataque, con lo que muchas de las vulnerabilidades que serán definidas en los controles serán probadas de acuerdo con la información obtenida en la esta sección.

- **OASAM-AUTH – *Authentication* (Análisis de la autenticación) – C2:** En este control se comprueban las funcionalidades relativas al uso de inicio de sesión través de la aplicación.

- **OASAM-CRYPT – *Cryptography* (Análisis del uso de criptografía) – C3:** En este control se comprueban las funcionalidades relativas al uso de la

criptografía en la aplicación que puede usarse al transmitir información o al almacenarla.

- **OASAM-CON - *Configuration and Deploy Management* (Análisis de la configuración e implantación) - C4:** En este control se analizan diferentes errores de configuración, o en sus componentes, y en las opciones de despliegue de las aplicaciones.
- **OASAM-LEAK - *Information Leak* (Análisis de fugas de información sensible) – C5:** En este control se comprueban las fugas de información sensible a diferentes medios, información sensible que puede ser relativa al usuario o al propio dispositivo.
- **OASAM-DV - *Data Validation* (Análisis de gestión de la entrada de usuario) – C6:** En esta sección se analizarán vulnerabilidades relacionadas con el manejo de la entrada recibida por parte del usuario en las aplicaciones. Uno de los principales vectores de ataque es la validación incorrecta de la entrada de información, ya que se puede inyectar código y afectar gravemente a la aplicación y a sus datos.
- **OASAM-IS - *Intent Spoofing* (Análisis de la gestión en la recepción de *Intents*) – C7:** Representan vulnerabilidades relacionadas con el envío de *Intents* arbitrarios a un componente que espera recibir otro tipo de *Intents*, debido a que los objetivos son públicos o exportables, en especial los *Broadcast Receivers* que pueden ser declarados en el archivo AndroidManifest o en tiempo de ejecución.
- **OASAM-UIR - *Unauthorized Intent Receipt* (Análisis de la resolución de *Intents) – C8:*** En esta sección se analiza la entrega de *Intents* implícitos, debido a que una aplicación maliciosa puede recoger dicho *Intent* mediante el registro de un *Intent Filter,* que es capaz de pasar la resolución (*action*[33], *data*[34] y *category*[35]), a menos que el *Intent* tenga declarado seguridades

[33] *Action*: objeto string que especifica la acción genérica a realizar, tal como ver o seleccionar.
[34] *Data*: objeto URI (Identificador unívoco de recursos de una red) que hace referencia a los datos sobre los que se actuará y/o el tipo MIME (Extensiones Multipropósito de Correspondencia de Internet) de esos datos.
35 *Category*: objeto *string* que contiene información adicional sobre el tipo de componente que debe manejar la *Intent.*

adicionales. Si una aplicación maliciosa pudiera interceptar los *Intents* implícitos, podría tener acceso a la ejecución del flujo de datos y realizar ataques de denegación de servicio o *phishing*. Esto puede manifestarse en componentes concretos como: *Broadcast Receiver, Activities, Services.*

- **OASAM-BL -*Business Logic*: Análisis de la lógica de negocio la aplicación – C9:** En esta sección se analizan vulnerabilidades relacionadas con el propio diseño y la codificación de los componentes, englobando la lógica de funcionamiento y la capacidad de la aplicación para funcionar de manera no esperada, alterando los flujos de trabajo de la misma. Estos ataques sobre la lógica de negocio son peligrosos, difíciles de detectar y específicos a la aplicación.

1.4.3 METODOLOGÍA PARA LA GESTIÓN DEL RIESGO DE PRIVACIDAD CNIL [29]

La metodología CNIL está basada en el tratamiento de información personal de los usuarios de dispositivos móviles, y su aplicación debe permitir:

- Tener una visión de los riesgos derivados del tratamiento de los datos personales.
- Saber cómo determinar las medidas de seguridad, necesarias para tomar todas las precauciones útiles con respecto a la naturaleza de los datos de los usuarios y los riesgos de su procesamiento, para preservar la seguridad de los datos, y en particular evitar su alteración, daño, o acceso por parte de terceros no autorizados.

1.4.3.1 Conceptos de gestión de riesgos

La metodología está enfocada en el contexto de la privacidad y es totalmente compatible con los estándares internacionales para la gestión de riesgos, por lo cual se utiliza en muchas áreas como seguridad de la información, seguridad, finanzas, seguros, etc.

1.4.3.1.1 Noción de riesgo para la privacidad

En el ámbito de la privacidad, los únicos riesgos a considerar son aquellos en los que el procesamiento de información personal afecta a la privacidad. Aquellos riesgos están compuestos por un evento temido (¿a qué se tiene miedo?), y todas

las amenazas que hacen que eso sea posible (¿cómo puede ocurrir esto?), para esto toma en consideración 3 aspectos.

- **Eventos temidos, lo que hay que evitar:** Para el tratamiento de información personal, los activos primarios a considerar son los flujos del proceso y la información personal, sobre los cuales se quiere evitar la no disponibilidad de procesos legales, cambios en los flujos de procesos sobre los datos, acceso ilegitimo a los datos, cambio no deseado y desaparición de información.
- **Amenazas, de lo que se tiene que proteger:** Para que ocurra un evento temido, tiene que haber una o más fuentes de riesgo causándolo, ya sea accidental o deliberadamente.

  - Personas pertenecientes a la organización: usuarios, especialistas informáticos, etc.
  - Personas ajenas a la organización: destinatario, proveedor, competidor, un tercero con autorizados, organización gubernamental, etc.
  - Fuentes no humanas: virus informáticos, desastres naturales, materiales inflamables, epidemias, roedores, etc.

  Las fuentes de riesgo actuarán accidental o deliberadamente, en los diversos componentes de los activos de información.

  - Hardware: Computadoras, enlaces de comunicaciones, discos, etc.
  - Software: Sistemas operativos, bases de datos, mensajería, etc.
  - Redes: Por cable, inalámbrica, fibra óptica, etc.
  - Personas: Usuarios, administradores, gerentes, etc.
  - Medios de papel: Impresión, fotocopias, etc.
  - Canales de transmisión de papel: Correo, flujo de trabajo, etc.

  La acción de las fuentes de riesgo sobre los activos de información, pueden suceder a través de diferentes amenazas.

  - Desviación de uso: Los activos se desvían de su uso previsto sin ser alterados o dañados.
  - Espionaje: Los activos se observan sin sufrir daños.
  - Daño: Los activos están parcial o totalmente dañados.
  - Cambios: Los activos se transforman.

- Superar los límites de operación: Los activos están sobrecargados, sobreexplotados o utilizados en condiciones que no les permite funcionar adecuadamente.
- Pérdidas de bienes: Los activos sufren pérdida, robo, son vendidos o regalados, por lo que ya no es posible ejercer los derechos de propiedad sobre los mismos.

- **Nivel de riesgos, ¿Cómo hacer la estimación?:** Un riesgo es un escenario que describe cómo las fuentes de riesgo pueden explotar las vulnerabilidades de los activos que conducen a provocar un incidente sobre los activos de información primarios y los impactos en la privacidad. Los niveles de riesgo se estiman en términos de severidad y probabilidad.
  - **Severidad:** Representa la magnitud de un riesgo y depende fundamentalmente del nivel de identificación de los datos personales y el nivel de consecuencias de los impactos potenciales.
  - **Probabilidad:** Representa la viabilidad de que ocurra un riesgo y depende fundamentalmente del nivel de vulnerabilidades de los activos de información frente al nivel de capacidad de que las fuentes de riesgo puedan explotarlos.

1.4.3.1.2 Enfoque de la gestión de riesgo de la privacidad.

Para evaluar los riesgos, en primer lugar, los eventos temidos tienen que ser identificados y estimados en términos de severidad. Después, se tiene que identificar aquellas amenazas cuya severidad es alta y estimar su probabilidad. Por lo tanto, los riesgos evaluados se pueden tratar mediante medidas proporcionadas y adecuadas. Este enfoque consiste en estudiar los aspectos que se muestran en la Figura 1.8 y se enlistan a continuación:

1. El contexto del procesamiento de información personal.
2. Los acontecimientos temidos en aquel contexto en particular.
3. Las posibles amenazas, de ser necesario.
4. Los riesgos implicados, de ser necesario.
5. Las medidas adecuadas para su tratamiento.

Al ser este un proceso de mejora continua, se requiere un seguimiento de cambios en el tiempo y actualizaciones cada vez que se produzca un cambio significativo

Figura 1.8. Etapas iterativas del enfoque de gestión de riesgo – CNIL [29]

En el proceso de identificación de las amenazas, la metodología menciona los riesgos que pueden afectar a la confidencialidad, integridad y disponibilidad, tal como se muestra en la Tabla 1.4.

Tabla 1.4. Amenazas genéricas del análisis de riesgo - CNIL [29] 1 de 2

Amenazas que pueden poner en peligro la confidencialidad	**Amenazas que pueden poner en peligro la disponibilidad**	**Amenazas que pueden poner en peligro la integridad**
C01. Uso anormal de hardware	A01. Uso de funciones de hardware ajenas al propósito para el que fueron configuradas	I01. Alteración de hardware
C02. Espionaje de hardware	A02. Sobrecarga de hardware	I02. Uso anormal de software
C03. Alteración de hardware	A03. Alteración de hardware	I03. Alteración de software
C04. Pérdida de hardware	A04. Daños en el hardware	I04. Ataques *Man-in-the-middle*[36] a través de canales informáticos
C05. Uso de funciones de software ajenas al propósito para el que fueron configuradas	A05. Pérdida de hardware	I05. Sobrecarga de trabajo
C06. Análisis de software	A06. Uso anormal de software	I06. Manipulación de personas

[36] *Man-in-the-Middle*: ataque que consiste en introducirse en la comunicación de dos equipos para que todo el tráfico pase por un tercero y así poder leer o modificar sus datos, contraseñas, etc.

Tabla 1.4. Amenazas genéricas del análisis de riesgo - CNIL [29] 2 de 2

Amenazas que pueden poner en peligro la confidencialidad	Amenazas que pueden poner en peligro la disponibilidad	Amenazas que pueden poner en peligro la integridad
C07. Alteración de software	A07. Sobrecarga de software	I07. Falsificación de documentos en papel
C08. Espionaje de canales de datos	A08. Alteración de software	I08. Manipulación de papel en los canales de transmisión
C09. Espionaje remoto de individuos	A09. Borrado total o parcial de un programa de software	
C10. Manipulación de individuos	A10. Pérdida de software	
C11. Adquisición de individuos	A11. Sobrecarga del canal de datos	
C12. Visualización de documentos en papel	A12. Daños del canal de datos	
C13. Robo de documentos en papel	A13. Desaparición del canal de datos	
C14. Espionaje de canales de transmisión de papel	A14. Sobrecarga de trabajo	
	A15. Daños corporales	
	A16. Salida de un colaborador	
	A17. Eliminación de documentos en papel	
	A18. Daño a los documentos en papel	
	D19. Desaparición de documentos en papel	
	A20. Sobrecarga de canales de transmisión de papel	
	A21. Daño a canal de transmisión de papel	
	A22. Alteración de los canales de transmisión de papel	
	A23. Desaparición de los canales de transmisión de papel	

1.4.4 EVALUACIÓN DE LOS RIESGOS DE PRIVACIDAD EN ANDROID [30]

En la actualidad, la evaluación de los riesgos de privacidad en las aplicaciones de Android no toma en cuenta la información del usuario, por lo que supone que el valor de cada activo se percibe de forma similar entre todos los usuarios. Por lo tanto, no es posible realizar una evaluación de riesgos de privacidad por usuario, en [30] se ha propuesto un proceso para evaluar el riesgo de privacidad de los usuarios de Android que incluye:

- Proporcionar una taxonomía de los datos de usuario que se encuentran en un *smartphone* y sus respectivos permisos, para discutir formas de evaluar el impacto de su divulgación para un usuario en particular.
- Identificar las amenazas de privacidad aplicables a los datos de los usuarios. Para cada amenaza, se asocian los permisos requeridos para que la amenaza ocurra, con lo cual se considera que cada vez que un usuario concede permisos a una aplicación, se introduce vulnerabilidades e incrementa la probabilidad de amenaza.
- Determinar estadísticas descriptivas de las combinaciones de permisos que pueden violar la privacidad del usuario, con base en su análisis de un conjunto de aplicaciones de Google Play.
- Evaluar el riesgo de privacidad por usuario, combinando la probabilidad de permisos con el aporte del usuario con respecto al impacto de la divulgación.

1.4.4.1 Evaluación del Impacto de la Privacidad [31 , 32]

La privacidad se refiere esencialmente a la protección de datos personales o Información Personal Identificable (PII, por sus siglas en inglés). La Evaluación del Impacto de la Privacidad (PIA, por sus siglas en inglés) se refiere a “un proceso sistemático para identificar y resolver los problemas de privacidad en un sistema de información que considera las consecuencias futuras para la privacidad de una acción actual o propuesta”.

Este proceso es una evaluación de riesgos, centrado en la privacidad, y está principalmente asociado con la recopilación, uso o divulgación de información personal, siendo una herramienta para el apoyo a la toma de decisiones. PIA se

utiliza actualmente en el Reino Unido, Canadá, EE. UU. y Australia en proyectos del sector público. Los principios de privacidad con respecto al mercado de Android son:

(i) Las solicitudes deben indicar claramente por qué se recopila la información personal, antes o durante el momento de la recopilación.

(ii) Los datos de usuario recopilados por las aplicaciones deben ser precisos, relevantes, actualizados y no excesivos en relación con el propósito de la recolección.

(iii) Las aplicaciones no deben retener los datos del usuario por períodos más largos que los necesarios para el propósito de la recolección.

(iv) Los datos del usuario deben ser recopilados por las aplicaciones con el conocimiento y consentimiento del individuo.

(v) Los datos del usuario no se deben comunicar a terceros, excepto bajo condiciones especificadas que pueden incluir el consentimiento del usuario.

(vi) Las aplicaciones deben asegurar que los datos del usuario se mantendrán en condiciones seguras.

(vii) Ser accesible al individuo para corrección o impugnación.

Estos principios indican que los usuarios deben poder revisar la política de privacidad de una aplicación antes de descargarla, esta política debe ser clara e indicar qué tipos de datos se recogen y con qué propósito. En Google Play, la inclusión de dicha política no es obligatoria para los desarrolladores, aunque Android requiere que los usuarios concedan permisos a las aplicaciones, los estudios han demostrado que los usuarios ignoran los permisos o no los entienden en absoluto, inclusive, los desarrolladores no indican por qué necesitan los datos, cómo los usarán, cómo serán almacenados o protegidos y si serán transferidos a un tercero.

1.4.4.1.1 Tipos de información personal

Al analizar el archivo manifiesto de Android, se identifican nueve tipos de datos que pueden ser objeto de una violación de privacidad, tales como:

a. Datos de comunicación como SMS[37], MMS[38], Voz, etc.
b. Datos de cámara o micrófono.
c. Datos de ubicación, GPS, ubicación de la red o redes sociales.
d. Datos en almacenamiento externo que incluyen datos de aplicaciones, documentos, correos electrónicos, etc.
e. Lista de contactos del dispositivo o de redes sociales.
f. Datos de historial/uso que indican las preferencias del usuario y pueden recogerse mediante marcadores.
g. Datos del calendario, que también podrían ser un indicador de los contactos o la ubicación del usuario.
h. Datos de identidad, que se refiere a todos los datos únicos que se pueden utilizar para identificar un usuario como: ID de dispositivo, direcciones de correo electrónico, ID de perfil.
i. Credenciales, por ejemplo: *tokens* de autenticación, contraseñas, PIN.

Para estos tipos de información, Android ha asignado permisos para permitir a las aplicaciones su uso o recolección, estos permisos se muestran en la Tabla 1.5.

1.4.4.1.2 Niveles de acceso a los activos del Smartphone.

De acuerdo con el análisis de riesgos, un PIA para *smartphones* se refiere típicamente a (a) un dispositivo móvil y (b) una o más aplicaciones. Para cada dispositivo, es necesario definir la información personal accedida o recogida, junto con los respectivos permisos de acceso de las aplicaciones o los controles. Para cada aplicación, necesitamos identificar las condiciones de recopilación y uso de las PII, los riesgos de privacidad derivados de combinaciones de permisos peligrosos, innecesarias o la falta de contramedidas apropiadas.

En la Tabla 1.5 se resume el nivel de acceso de permisos que protegen a un tipo de datos o un canal de comunicación en las versiones de Android Jelly Bean 4.1 (API 17, API 16), Ice Cream Sandwich 4.0.x (API 15) y Gingerbread 2.3.x (API 10).

[37] SMS (*Short Message Service*): servicio disponible en teléfonos móviles que permite el envío de mensajes cortos, conocidos como mensajes de texto
38 MMS: (*Multimedia Messaging Service*): servicio de mensajería que permite a teléfonos móviles enviar y recibir contenidos multimedia, incorporando sonido, video o fotos.

Tabla 1.5. Nivel de acceso de permisos [30]

	Permission Access levels per API			
	17	16	15	10
ACCESS_COARSE_LOCATION	d	d	d	d
ACCESS_FINE_LOCATION	d	d	d	d
ACCESS_NETWORK_STATE	n	n	n	n
ACCESS_WIFI_STATE	n	n	n	n
AUTHENTICATE_ACCOUNTS	d	d	d	d
BLUETOOTH	d	d	d	d
BLUETOOTH_ADMIN	d	d	d	d
CALL_PHONE	d	d	d	d
CAMERA	d	d	d	d
GET_ACCOUNTS	n	n	n	n
GET_TASKS	d	d	d	d
INTERNET	d	d	d	d
MOUNT_UNMOUNT_FILESYSTEMS	s	d	d	d
PROCESS_OUTGOING_CALLS	d	d	d	d
READ_CALENDAR	d	d	d	d
READ_CALL_LOG	d	d	N/A	N/A
READ_CONTACTS	d	d	d	d
READ_EXTERNAL_STORAGE	n	n	N/A	N/A
READ_HISTORY_BOOKMARKS	d	d	d	d
READ_LOGS	s/t	s/t	d	d
READ_PHONE_STATE	d	d	d	d
READ_PROFILE	d	d	d	N/A
READ_SMS	d	d	d	d
READ_SOCIAL_STREAM	d	d	d	N/A
READ_USER_DICTIONARY	d	d	d	d
RECEIVE_MMS	d	d	d	d
RECEIVE_SMS	d	d	d	d
RECEIVE_WAP_PUSH	d	d	d	d
RECORD_AUDIO	d	d	d	d
SEND_SMS	d	d	d	d
SUBSCRIBED_FEEDS_READ	n	n	n	n
USE_CREDENTIALS	d	d	d	d
USE_SIP	d	d	d	d
WRITE_EXTERNAL_STORAGE	d	d	d	d
d: dangerous s: signature or system t: development n: normal				

1.4.4.2 Evaluación del Riesgo de Privacidad en Smartphones

El documento describe un enfoque para la evaluación del riesgo a la privacidad, centrado en los usuarios de teléfonos inteligentes, para lo cual los usuarios (a) instalan aplicaciones en sus dispositivos, de forma regular (b) los instalan sólo en el mercado oficial de aplicaciones y (c) protegen su *smartphone* sólo con los controles de seguridad predeterminados de la plataforma, sin tomar en cuenta los dispositivos con sistema operativo modificado o instalando algún software de seguridad adicional. Finalmente, el teléfono inteligente debe tener conectividad a Internet desde el portador móvil.

1.4.4.2.1 Amenazas y permisos

Se considera a los permisos de aplicaciones de terceros como vulnerabilidades, debido a que en Android el acceso a recursos protegidos sólo ocurre si la aplicación solicitante recibe el permiso correspondiente para el recurso. Incluso si la aplicación solicitante es benigna, su privilegio de acceder a datos privados puede ser mal utilizado por otra aplicación que puede realizar un ataque secundario, o por alguna biblioteca maliciosa que se incluya con ella.

Los usuarios de Android deben aceptar permisos en el enfoque del todo o nada, porque no pueden autorizar sólo un subconjunto de los permisos solicitados. Entre las amenazas de los teléfonos inteligentes, cinco de ellas plantean un potencial impacto en la privacidad como son:

a) Seguimiento / Vigilancia (T1): Se refiere al seguimiento del contexto de los usuarios. Por ejemplo, el uso de los sensores del dispositivo.
b) Interceptación/escucha a escondidas (T2): Se refiere a la interceptación ilícita de comunicaciones y es aplicable a todos los tipos de datos de comunicación.
c) Perfil del usuario (T3): Se refiere al seguimiento de las actividades realizadas recurrentes del usuario.
d) *Phishing* (T4): Se refiere al adquirir credenciales de autenticación desde el terminal móvil de forma fraudulenta, mediante engaño a los usuarios.

e) *Divulgación* de Información Personal (T5): Se refiere a la divulgación de todos los otros tipos de información personal, que no caen en los otros cuatro tipos de amenazas como documentos, archivos multimedia, etc.

El análisis omite amenazas que podrían ser usadas como vector de ataque para una violación de privacidad. Por ejemplo, de un ataque exitoso de suplantación pueden desencadenar otros ataques de privacidad, tales como phishing, escuchas telefónicas, desclasificación de SSL, etc. Tampoco considera spyware, o las amenazas que explotan vulnerabilidades en el sistema operativo y que no están relacionadas con un permiso específico o relacionado con un dispositivo rooteado.

En la Tabla 1.6 se muestran datos personales de un usuario a los que se les asignan permisos que pueden permitir una violación de privacidad. Para que una aplicación revele cualquier tipo de información privada a un tercero, se requiere un canal de datos disponibles como: (a) conexión a Internet, (b) mensajes cortos y (c) Bluetooth. Cada uno de ellos atado a un permiso peligroso (a) INTERNET, (b) SEND_SMS, y (c) BLUETOOTH.

Dado que cada amenaza está habilitada por un conjunto de combinaciones de permisos, la probabilidad de amenaza por usuario es el valor promedio de las frecuencias en cada categoría especificada. Por último, el nivel de vulnerabilidad por usuario se evalúa en función de la frecuencia media de una combinación de permisos determinada en las categorías seleccionadas, basada en la siguiente escala cuantitativa: (a) 1. Despreciable <= 10 %, (b) 2. Limitado: >10 % y <= 40%, (c) 3. Significativo >40 % y <= 70 %, y (d) 4. Máximo > 70 %.

Tabla 1.6. Mapeo de activos de datos, permisos y amenazas [30] 1 de 2

Categoría	**Tipo de activo**	**Permiso Android**	**Amenazas de privacidad**				
			T1	**T2**	**T3**	**T4**	**T5**
Comunicacion.	SMS	RECEIVE_SMS		✔			
	MMS	READ_SMS		✔			
	Voz	RECEIVE_MMS		✔			
	Push Wap	PROCESS_OUTGOING_ CALLS		✔			
		RECEIVE_WAP_PUSH		✔			

Tabla 1.6. Mapeo de activos de datos, permisos y amenazas. [30] 2 de 2

Categoría	Tipo de activo	Permiso Android	Amenazas de privacidad				
			T1	T2	T3	T4	T5
Sensor/ Localización	Video	CAMERA	✔	✔			
	Audio	RECORD_AUDIO	✔ ✔	✔			
	Localización	ACCESS_COARSE_ LOCATION	✔		✔		
		ACCESS_FINE_LOCATION	✔ ✔		✔		
		BLUETOOTH_ADMIN	✔		✔		
		ACCESS_NETWORK_ STATE	✔ ✔		✔ ✔		
		ACCESS_WIFI_STATE	✔		✔		
		READ_SOCIAL_STREAM	✔ ✔		✔ ✔		
Almacenamiento externo		WRITE_EXTERNAL_ STORAGE		✔ ✔	✔ ✔		✔ ✔
		READ_EXTERNAL_ STORAGE		✔	✔		✔
Contactos		READ_CONTACTS		✔ ✔			✔ ✔
		READ_SOCIAL_STREAM		✔			✔
Historial / Uso		READ_CALL_LOG		✔	✔		
		READ_HISTORY_ BOOKMARKS			✔		
		GET_TASKS			✔		
		READ_LOGS			✔		
		READ_USER_ DICTIONARY			✔		✔
		READ_SOCIAL_STREAM			✔		
		SUBSCRIBED_FEEDS_ READ			✔		
Calendario		READ_CALENDAR	✔				✔
Identidad		READ_PROFILE			✔	✔	✔
		GET_ACCOUNTS			✔	✔	✔
		READ_PHONE_STATE			✔	✔	✔
Credenciales		READ/RECEIVE_SMS				✔	
		AUTHENTICATE_ ACCOUNTS				✔	
		USE_CREDENTIALS				✔	

1.4.4.2.2 Amenazas e impactos

Para evaluar el impacto de las brechas de privacidad, se genera un ajuste a la metodología para la Gestión del Riesgo de Privacidad CNIL. Inicialmente, los permisos de la Tabla 1.6, se asignan con un nivel de identificación, el cual hace referencia a la capacidad de identificar a un individuo únicamente evaluando los datos que el permiso protege. El **nivel de identificación** del permiso se evalúa mediante una escala de 4 ítems:

1. **Insignificante**: Identificar a un usuario que utiliza este permiso puede ser prácticamente imposible.
2. **Limitado**: Identificar a un usuario que use este permiso puede ser difícil, pero es posible en ciertos casos.
3. **Significativo**: Identificar relativamente fácil a un usuario que use este permiso.
4. **Máximo**: Identificar extremadamente fácil a un usuario que utiliza este permiso.

Después de la identificación del individuo, se hacen preguntas al usuario para evaluar su impacto individual de una violación a su privacidad. Cada pregunta describe el efecto de la divulgación o mal uso en varios datos personales.

Las respuestas del cuestionario también están predefinidas, describiendo el efecto de la amenaza de privacidad para el usuario, e indican el **nivel de severidad**, basados en la siguiente escala cualitativa:

1. **Insignificante**: El usuario no se ve afectado, o encontrar algunos inconvenientes que puede superar sin ningún problema.
2. **Limitado**: El usuario puede encontrar inconvenientes significativos, que puede ser capaz de superar a pesar de algunas dificultades.
3. **Significativo**: El usuario puede encontrar consecuencias significativas, que puede ser capaz de superar con serias dificultades.
4. **Máximo**: El usuario puede encontrar consecuencias significativas o incluso irreversibles, que no puede superar.

El nivel de impacto total (ImL) de un escenario de amenaza específico se evalúa como la suma del **nivel de identificación** y el **nivel de severidad**, basándose en

la siguiente escala: a) 1 - Despreciable: ImL < 5%, (b) 2 - Limitado: ImL = 5% (c) 3 - Significativo: ImL = 6% y (d) 4 Máximo: ImL > 6%.

1.4.4.2.3 Riesgo de amenaza de privacidad

Dado que cada amenaza de privacidad requiere la presencia de combinaciones específicas de permisos en el dispositivo del usuario, el nivel de riesgo por amenaza de privacidad se evalúa como una suma del nivel de Impacto y nivel de vulnerabilidad, en la siguiente escala: (a) 1 - Despreciable: 1 a 3, (b) 2 - Limitado: 4 a 5, (c) 3 - Significativo: 6 a 7, y (d) 4 - Máximo: 8. Este valor corresponde al perfil de riesgo específico del usuario (PIA personalizado), como (a) El impacto fue evaluado en base a la opinión del usuario y refleja sus propias suposiciones con respecto a los efectos potenciales de una violación de privacidad. Y (b) La probabilidad de amenaza (nivel de vulnerabilidad) se evalúa de acuerdo con el tipo de uso del dispositivo que el usuario describe, identificando las categorías comunes de sus aplicaciones.

1.5 HERRAMIENTAS COMPUTACIONALES PARA EL ANÁLISIS DE VULNERABILIDADES DE APLICACIONES ANDROID

A continuación, se muestran las diferentes herramientas computacionales que sirven para realizar análisis de vulnerabilidades presentes en los terminales móviles físicos o virtualizados, siendo estas vulnerabilidades generadas por el sistema operativo o por las aplicaciones instaladas en el dispositivo. Si se necesita obtener información detallada sobre las herramientas, su configuración y uso, se puede revisar el Anexo 19.

1.5.1 AMBIENTES VIRTUALIZADOS ANDROID

La emulación de un sistema operativo es un recurso ideal cuando se necesita probar alguna característica específica del terminal o de alguna aplicación, sin tener que disponer de un laboratorio completo con hardware y software especializado de alto costo.

1.5.1.1 Emulador oficial Android [32].

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android, el cual incluye el Kit de Desarrollo (SDK) que contiene un emulador completo de Android, el cual servirá para realizar la compilación rápida

y la ejecución de la aplicación en tiempo real, directamente desde el móvil virtualizado.

1.5.1.2 Genymotion [33]

Es uno de los emuladores de Android que puede integrarse con diversos entornos, como Android Studio, Eclipse, inclusive Visual Studio 2015. Su principal característica es que posee una gran variedad de máquinas virtuales configuradas de acuerdo con los equipos disponibles en el mercado permitiendo tener fácil acceso al hardware del dispositivo virtualizado.

1.5.1.3 Andy Android Emulator [34]

Andy es un emulador que permite ejecutar cualquier tipo de aplicaciones Android directamente en un PC o Mac, ofreciendo a los usuarios la capacidad de almacenamiento ilimitado. Su principal característica es que proporciona sincronización entre el escritorio y los dispositivos sin necesidad de conexión fija o inalámbrica mediante una interfaz amigable con el usuario, rápida e intuitiva.

1.5.2 HERRAMIENTAS COMPUTACIONALES PARA EL ANÁLISIS DE VULNERABILIDADES EN EL DISPOSITIVO MÓVIL Y SUS APLICACIONES

1.5.2.1 Nmap (*Network Mapper*) [35]

Nmap es una herramienta de código abierto que sirve para exploración de red y auditoría de seguridad y monitoreo, que utiliza paquetes en formas originales para determinar equipos disponibles en una red, servicios, sistemas operativos, tipos de filtros de paquetes o cortafuegos que se utilizan, etc. La salida de Nmap es una lista de objetivos, cuya información principal es la “tabla de puertos interesantes”. Esta tabla enumera el número de puerto, protocolo, nombre del servicio más común, y su estado que puede ser: *open* (abierto), *filtered* (filtrado) y *closed* (cerrado).

1.5.2.2 OpenVAS (*Open Vulnerability Assessment*) [36]

OpenVAS, es una plataforma de software libre que integra herramientas y servicios especializados en el escaneo y gestión de vulnerabilidades de seguridad de sistemas informáticos. El núcleo de esta arquitectura orientada a servicios seguros SSL, es el escáner OpenVAS. El escáner ejecuta eficientemente las **Pruebas de**

Vulnerabilidad de Red[39] (NVT), que son provistos por NVT OpenVAS, que tienen rutinas que comprueban la presencia de problemas potenciales de seguridad en los sistemas.

La distribución de Kali Linux cuenta con esta herramienta instalada de forma predefinida, y puede ser utilizada a través de dos métodos, línea de comandos (OpenVAS CLI) o interfaz web (*Greenbone Security Assistant*).

1.5.2.3 Wireshark [37]

Wireshark es un analizador de protocolos de software libre, utilizado para realizar análisis de información sobre paquetes de red de comunicaciones, que provee una funcionalidad similar a tcpdump[40], pero incorpora una interfaz gráfica con opciones filtrado y organización de información, de esa forma permite observar todo el tráfico generado a través de una red. Además, la herramienta permite examinar información de datos capturados en vivo o de un archivo de captura almacenado previamente, ya que incluye un completo lenguaje para filtrar información importante y poder mostrar el flujo de una sesión TCP reconstruida.

1.5.2.4 SandDroid [38]

SandDroid es un sistema automático de análisis de aplicaciones de Android que combina técnicas de análisis estático y dinámico, disponibles en la Internet [38], en el cual permite realizar la carga de aplicativos APK de Android para realizar el respectivo análisis de vulnerabilidades.

En el portal se tiene la facilidad de revisar reportes anteriores de análisis de aplicativos APK. Para buscar un reporte se puede utilizar la opción de filtrado mediante los valores del hash MD5 de la aplicación, el valor SHA-1 de la firma digital de la aplicación Android, el nombre del paquete de la aplicación de Android, o el nombre de la familia del malware encontrado en la aplicación de Android.

---

[39] Pruebas de vulnerabilidad de red (NVT): pruebas que sirven para verificar problemas de seguridad existentes en sistemas remotos, en OpenVAS, están escritas en el lenguaje de programación NASL (Nessus *Attack Scripting Language*).

[40] Tcpdump: herramienta para línea de comandos, el cual permite al usuario capturar, mostrar y analizar en tiempo real los paquetes transmitidos y recibidos por la red a la cual el ordenador está conectado.

1.5.2.5 Dexter [39]

Dexter es una plataforma de análisis de aplicaciones para Android, se encuentra en versión Beta y cuenta con una interfaz de usuario web basado en un *back-end y front-end* desacoplado. La plataforma realiza una extracción de la mayor cantidad de información posible de los archivos APK para mostrarlas en diferentes vistas, entre las cuales se puede realizar una revisión de los bytecodes[41] de forma gráfica en una vista básica de bloque.

1.5.2.6 Androbugs *Framework* [40]

AndroBugs, es un sistema de análisis de vulnerabilidades que sirve para encontrar posibles vulnerabilidades de seguridad en las aplicaciones Android, que funciona mediante línea de comandos generando un análisis eficiente.

La herramienta entrega un reporte en formato txt, donde se puede observar las características principales de la aplicación analizada, además de un listado de las vulnerabilidades donde se las pondera mediante los siguientes grados de criticidad: Critico, Advertencia, Observación e Informes.

1.5.2.7 IBM Appscan Mobile Analyzer [41]

IBM Security AppScan, es una familia de herramientas con suscripción, para prueba y supervisión de seguridad de aplicaciones de la división *Rational Software* de IBM, que tiene la intención de probar aplicaciones en busca de vulnerabilidades de seguridad, y generar reportes y recomendaciones de correcciones. Sus principales características son:

- Ayuda a proteger fácilmente la aplicación móvil.
- Mejora la postura de la seguridad de la aplicación al detectar una variedad de tipos de vulnerabilidades de alto nivel que suelen utilizar los programas maliciosos.
- Simplifica las pruebas sin necesidad de configuración adicional.
- Reduce la necesidad de trabajo manual con un escáner de seguridad completamente automático y móvil que no requiere interacción del usuario.

[41] *Bytecode*: código intermedio más abstracto que el código máquina, habitualmente tratado como archivo binario que contiene un programa ejecutable, que es un archivo binario producido por el compilador.

- Facilita el uso de AppScan *Mobile Analyzer*, eliminando la necesidad del análisis de código fuente.
- Proporciona informes que contienen los detalles necesarios para el desarrollo, eliminando la necesidad de involucrar a un experto en seguridad.
- Ayuda a acelerar el tiempo de comercialización al permitirle integrar el servicio con sus procesos de generación en curso.

Para poder acceder al análisis de aplicaciones es necesario crear una cuenta en IBM del tipo prueba, en el enlace https://myibm.ibm.com/products-services/products, en donde después de realizar la respectiva validación es posible acceder al módulo de análisis de aplicaciones móviles.

1.5.2.8 Zanti - Zimperium Mobile Security [42]

Zanti es un conjunto de herramientas desarrollado para Android, que sirve para realizar pruebas de penetración, el cual permite simular ataques maliciosos en una red con distintos tipos de operaciones tales como ataques *Man In The Middle*(MITM), clonación de direcciones MAC, exploración de red, auditoría de contraseñas, comprobaciones de seguridad, captura de sesiones HTTP, entre otras.

Para poder hacer uso de la herramienta es necesario instalar la aplicación, descargada del portal de Zimperium, en un dispositivo Android que tenga permisos de *root*. A continuación, la herramienta permite realizar un escaneo a los dispositivos de red a los que se puede hacer una prueba de penetración con su respectiva dirección IP y MAC, esto mediante un módulo ligero de Nmap.

1.5.2.9 Drozer (MWR Labs) [43]

Drozer es una de las mejores plataformas para la evaluación de seguridad de Android y sus aplicaciones, la cual permite asumir el rol de una aplicación que permite interactuar con otras aplicaciones mediante el mecanismo de Comunicación entre Procesos (IPC) de Android y el sistema operativo. Drozer es una herramienta interactiva, por lo que, para realizar una evaluación de seguridad se tiene que ejecutar los diferentes comandos en una consola propia de la herramienta, los cuales serán enviados al agente instalado en el dispositivo objetivo, para ejecutar la tarea correspondiente.

Drozer ejecuta código Java dinámico en un dispositivo, para evitar la necesidad de compilar e instalar pequeños scripts de prueba. Sus características principales son:

- Descubrir los paquetes instalados en el dispositivo objetivo.
- Enviar *Intents* a los IPC *Endpoints*[42].
- Acceder a bases de datos de otras aplicaciones.
- Interactuar con servicios de otras aplicaciones.
- Ejecución de comandos en un *shell* interactivo.
- Acceder a un dispositivo con *Exploits* remotos.
- Escalamiento de privilegios *root*.
- Es posible usar Drozer en dispositivos físicos y emulados.

1.5.2.10 Qark [44]

Es una herramienta de análisis de código estático, que sirve para reconocer vulnerabilidades potenciales de seguridad en las aplicaciones Android basadas en Java, proporcionando descripciones claras de los problemas y enlaces a fuentes de referencia autorizadas. Qark también intenta proporcionar comandos ADB (Android *Debug Bridge*) generados dinámicamente para ayudar en la validación de vulnerabilidades, incluso al finalizar el escaneo, crea una aplicación de prueba personalizada APK, diseñada específicamente para explotar las vulnerabilidades y demostrar los posibles problemas que descubre.

1.5.2.11 Santoku [45]

Santoku Linux es una distribución de sistema operativo gratuita y de código abierto basada en Linux, desarrollada especialmente para auditar dispositivos móviles en busca de vulnerabilidades, fallos o cualquier aspecto que comprometa la privacidad de los usuarios. Esta distribución incluye herramientas para realizar pruebas de penetración clasificadas en 3 apartados, las cuales son:

- **Análisis forense:** En este apartado se encuentran herramientas para adquirir y analizar datos de los dispositivos, herramientas para instalar o cambiar imágenes de sistemas operativos de varios fabricantes y herramientas para analizar memorias NAND, tarjetas multimedia y RAM.

[42] IPC (*Inter Process Communication*) *Endpoints*: mecanismo por el cual diferentes componentes de Android se comunican entre sí.

- **Análisis de *malware* móvil:** Dentro de este apartado se encuentran emuladores que permiten utilizar máquinas virtuales con la finalidad de poder analizar el funcionamiento de los diferentes sistemas operativos móviles y hallar la presencia de *malware*.
- **Análisis de seguridad:** Dentro de este apartado se puede acceder a herramientas y *scripts* que sirven para la detección de vulnerabilidades y fallas, también contiene *scripts* que facilitan la tarea de descifrado y de-compilación de herramientas y aplicaciones para un análisis a detalle del código de la aplicación.

1.5.2.12 Androguard [46]

Androguard es un *framework* que permite interactuar con un código malicioso con la finalidad de leer sus recursos, acceder al código fuente, etc., inclusive para comparar distintas amenazas realizando comparación de métodos, clases y recursos para encontrar similitudes o diferencias.

Por defecto tiene una serie de *scripts* y herramientas para facilitar el análisis, las cuales permiten interactuar con las amenazas de manera sencilla. Androlyze[47] es una de las opciones más prácticas que permite iniciar una shell interactiva para el análisis de vulnerabilidades de Android.

1.5.2.13 BurpSuite [48]

BurpSuite es una herramienta para realizar el análisis de seguridad de aplicaciones Web, que cuenta con la integración de varios componentes que permiten verificar desde la asociación inicial y análisis de la superficie de ataque, hasta la detección y explotación de vulnerabilidades de seguridad de una aplicación, combinando técnicas manuales avanzadas con la automatización de procedimientos.

Entre los principales componentes de la aplicación se tiene un *proxy*, que permite interceptar información con la finalidad de inspeccionar y modificar el tráfico entre la aplicación de destino y el navegador. La herramienta permite además realizar la escritura de *plugins* personalizados, para realizar tareas complejas y personalizadas dentro de BurpSuite.

1.5.2.14 DEX2JAR y JD-GUI [49]

La ingeniería inversa es una de las áreas de la seguridad informática que se utiliza para encontrar vulnerabilidades en una aplicación, analizando la presencia de código sospechoso dentro de la lógica de funcionamiento de una aplicación. Esta técnica es aplicada con la finalidad de realizar un análisis sobre los archivos APK de las aplicaciones usando los programas DEX2JAR y JD-GUI para navegar entre las clases de la aplicación para validar algún tipo de código malicioso o un error de programación dentro de la codificación que permitan la presencia de algún tipo de *malware* generando fuga de información sensible de los usuarios.

CAPÍTULO 2

DESARROLLO DE GUÍA DE BUENAS PRÁCTICAS Y ANÁLISIS DE VULNERABILIDADES

En este capítulo se desarrollará una guía de procedimientos para el análisis de vulnerabilidades que afectan a la privacidad de los usuarios, por lo cual se considerará el uso de las metodologías y buenas prácticas de análisis de seguridad de aplicaciones Android, además de las herramientas computacionales revisadas, que servirán para generar ambientes de virtualización Android y la generación de diferentes pruebas de penetración sobre los ambientes virtuales o físicos y sus aplicaciones instaladas.

Para el análisis de vulnerabilidades de las aplicaciones Android, se ha establecido un procedimiento que consta de seis fases, en donde se hace referencia a cuatro controles (C) de análisis de vulnerabilidades que inciden sobre la información recopilada del dispositivo, métodos utilizados de autenticación y cifrado, información transferida entre dispositivos, y análisis del código fuente y ejecución de los componentes de la aplicación. Estos controles servirán para conocer si el dispositivo y las aplicaciones pueden afectar la información privada de los usuarios.

Para la evaluación de la afectación a la privacidad de los usuarios, se ha establecido un procedimiento que consta de cuatro fases, en donde se hace relación a los factores: activos de información, amenazas, identificación del usuario, gravedad o severidad percibida por el usuario, nivel de Impacto, nivel de vulnerabilidad, que se mencionan en la metodología de evaluación de riesgo debido al uso de aplicaciones de dispositivos móviles.

2.1 GUÍA PARA EL ANÁLISIS DE VULNERABILIDADES DE APLICACIONES ANDROID Y SU AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS

2.1.1 LINEAMIENTOS PARA EL ANÁLISIS DE VULNERABILIDADES RELACIONADAS CON LA PRIVACIDAD DE LOS USUARIOS

En base a las metodologías de análisis de vulnerabilidades revisadas, existen diferentes lineamientos que analizan o controlan vulnerabilidades similares

relacionadas con la protección de la privacidad del usuario, estos lineamientos revisados de las diferentes metodologías: OWASP [50], OASAM [28], y la evaluación de riesgos de privacidad Android [49], se han agrupado en cuatro fases de estudio, que servirán para establecer la información necesaria para conocer si el dispositivo y las aplicaciones que se utilizan poseen vulnerabilidades que puedan afectar la información privada de los usuarios. Las fases a las que se hace referencia están alineadas a los controles que se enuncian a continuación.

2.1.1.1 Información recopilada del dispositivo – C1

En esta fase, se analiza toda la información disponible acerca de los dispositivos móviles, con lo cual se puede definir la superficie de ataque, correspondiente a la información general del sistema operativo, sus aplicaciones, permisos, y también aquellas vulnerabilidades relacionadas a los agujeros de seguridad presentes en el sistema operativo o a sus aplicaciones instaladas, que pueden generar vulnerabilidades que podrían permitir la fuga de información privada de los usuarios.

Para esta fase se han asociado los lineamientos de las metodologías revisadas que se muestran en la Tabla 2.1.

Tabla 2.1. Asociación de lineamientos referentes a recopilación de información de metodologías revisadas

OWASP *Mobile Security Project*	**OASAM - *Open* Android *Security Assessment Methodology***	**Evaluación de los riesgos de privacidad en Android**
C1. Identificar y proteger datos confidenciales en el dispositivo móvil.	C1. OASAM-INFO - Information Gathering: Obtención de información y definición de superficie de ataque.	Identificar los tipos de datos que pueden ser el objetivo de una violación de privacidad.

Para el análisis de esta fase se seleccionan las siguientes herramientas computacionales para su uso, considerando aquellas que brindan más información para el análisis, mayores prestaciones y facilidad en su uso, de acuerdo con el estudio realizado de las diferentes herramientas computacionales para análisis de vulnerabilidades que se encuentran en el Anexo 19, tal como se observa en la Tabla 2.2.

Tabla 2.2. Herramientas seleccionadas para la fase de recopilación de información

Herramienta	Información Colectada	Uso de la Información
NMAP	Información del sistema operativo, listados de puertos TCP y UDP abiertos.	Conocer posibles huecos de seguridad que pueden servir para el ingreso no autorizado y posible pérdida de información en el dispositivo.
OPENVAS	Listado de vulnerabilidades presentes en el dispositivo analizado.	Conocer vulnerabilidades presentes en el dispositivo que pueden ser explotadas para extraer información sensible de los dispositivos.
DROZER	Información de aplicaciones presentes en el dispositivo y sus respectivos permisos.	Conocer el listado de aplicaciones presentes en el sistema operativo para hallar posibles aplicaciones tipo *malware*, también hallar los permisos que solicitan las aplicaciones para el uso de recursos o información.

2.1.1.2 Métodos de autenticación y cifrado – C2

En esta fase, se validan los métodos relacionados con aquellas aplicaciones que dispongan de procedimientos de inicio de sesión y autorización, analizando aquellas vulnerabilidades generadas por el uso de métodos de programación que gestionan el acceso a las aplicaciones, nombres de usuarios y contraseñas que se pueden establecer de forma predeterminada en los dispositivos, o colocado en el código de la aplicación. En la Tabla 2.3 se muestra la asociación de los lineamientos de las metodologías revisadas en la sección 1.4.

Para el análisis de esta fase se seleccionan las herramientas computacionales que se muestran en la Tabla 2.4, considerando aquellas que ofrecen más información para el análisis, mayores prestaciones y facilidad en su uso, de acuerdo con el estudio realizado de las diferentes herramientas computacionales para análisis de vulnerabilidades que se encuentran en el Anexo 19.

Tabla 2.3. Mapeo de lineamientos referentes al análisis de inicios de sesión

OWASP *Mobile Security Project*	**OASAM - *Open* Android *Security Assessment Methodology***	**Evaluación de los riesgos de privacidad en Android**
C2. Manejo de credenciales "*password*" seguras en el dispositivo.	C2. OASAM-AUTH – *Authentication*: Análisis de la autenticación.	d.- Adquisición fraudulenta de credenciales de autenticación - Phishing.
C4. Implementar correctamente autenticación, autorización y gestión de sesiones.	C3. OASAM-CRYPT – *Cryptography*: Análisis del uso de criptografía.	

Tabla 2.4. Herramientas seleccionadas para la fase de análisis de inicios de sesión

Herramienta	**Información Colectada**	**Uso de la Información**
ZANTI	Información de usuarios y contraseñas capturadas que viajan desde la aplicación utilizando métodos de ataque MITM.	Conocer vulnerabilidades relacionadas a usuarios y contraseñas que son utilizados en las aplicaciones que viajan a través de la red, y pueden ser capturadas; con las cuales se pueda realizar accesos no autorizados a las aplicaciones; inclusive conociendo aquellas que no utilizan alguna metodología de cifrado.
BURPSUITE	Información de usuarios y contraseñas capturadas que viajan desde la aplicación utilizando métodos de vigilancia a través de *proxy*.	
WIRESHARK	Información de usuarios y contraseñas capturadas que viajan desde la aplicación utilizando capturas de paquetes en la red.	
DEX2JAR-JDGUI	Información de usuarios y contraseñas quemadas en el código fuente de la aplicación en los métodos de autenticación.	Conocer vulnerabilidades relacionadas a usuarios y contraseñas que son utilizados en las aplicaciones y se encuentran en el código fuente de la aplicación o registros generados por la aplicación, con las cuales se pueda realizar accesos no autorizados a la aplicación.
LOGCAT	Información de logs del sistema operativo donde se puede registrar variables de sesión o valores de autenticación.	

2.1.1.3 Información transferida entre dispositivos – C3

En esta fase, se realiza un análisis de la información transferida entre las aplicaciones y los servidores del aplicativo, analizando el contenido de los paquetes transferidos o interceptando información de las peticiones web y variables de sesión; inclusive verificar aquellos que no implementen ningún mecanismo de cifrado a nivel de datos o canal de transmisión.

Además, mediante este análisis se puede validar si existe alguna comunicación hacia algún *host* desconocido o si información confidencial se filtra sin el conocimiento de los usuarios, inclusive si esta es ajena a la lógica de la aplicación utilizada, pudiendo ser interceptadas por terceros.

Para esta fase se han asociado aquellos lineamientos de las metodologías revisadas con base en la Tabla 2.5.

Tabla 2.5. Mapeo de lineamientos referentes al análisis de información transferida entre dispositivos

OWASP *Mobile Security Project*	**OASAM - *Open* Android *Security Assessment Methodology***	**Evaluación de los riesgos de privacidad en Android**
C3. Garantizar que los datos sensibles están protegidos durante el transporte.	C5. OASAM-LEAK - *Information Leak*: Análisis de fugas de información sensible.	a.- Seguimiento / Vigilancia
C6. Integración de datos seguro con servicios y aplicaciones de terceros.		b.- Interceptación de información/Escucha a escondidas.

Para el análisis de esta fase se seleccionan las siguientes herramientas computacionales para su uso, considerando aquellas que ofrecen más información para el análisis, mayores prestaciones y facilidad de uso, de acuerdo con el estudio realizado de diferentes herramientas computacionales para análisis de vulnerabilidades que se encuentran en el Anexo 19, y se observan en la Tabla 2.6.

2.1.1.4 Análisis de aplicaciones Android – C4

En esta fase, se realiza el análisis de los componentes de las aplicaciones que tienen vulnerabilidades debido al uso inapropiado de permisos en el momento del

desarrollo de la aplicación, al uso de las API de terceros desconocidos o maliciosos, o incidentes en tiempos de ejecución, para lo cual se han asociado aquellos lineamientos de las metodologías revisadas, y que se muestran en la Tabla 2.7.

Tabla 2.6. Herramientas seleccionadas para la fase de análisis de información transferida entre dispositivos

Herramienta	Información Colectada	Uso de la Información
WIRESHARK	Información que viaja desde la aplicación hacia servidores utilizando capturas de paquetes en la red.	Conocer información sensible que viaja en diferentes protocolos de red utilizando o no cifrado hacia diferentes hosts servidores que podrían ser del tipo malicioso.
BURPSUITE	Información que viaja desde la aplicación hacia los servidores utilizando método de *proxy* en la red.	

Tabla 2.7. Mapeo de lineamientos referentes al análisis de aplicaciones Android

OWASP *Mobile Security Project*	**OASAM - *Open* Android *Security Assessment Methodology***	**Evaluación de los riesgos de privacidad en Android**
C5. Mantener las API *backend* (servicios) y plataforma (servidor) seguros.	C4. OASAM-CON - *Configuration and Deploy Management*: Análisis de la configuración e implantación.	(e) Divulgación de Información Personal, como documentos, archivos multimedia, etc., relacionado principalmente al almacenamiento de información.
C7. Prestar especial atención en la recolección, almacenamiento y uso consentido de los datos del usuario.	C7. OASAM-IS - *Intent Spoofing*: Análisis de la gestión en la recepción de *Intents*.	
C10. Revisar cuidadosamente cualquier interpretación de código de errores de tiempo de ejecución.	C8. OASAM-UIR - *Unauthorized Intent Receipt*: Análisis de la resolución de *intents*.	
	C9. OASAM-BL -*Business Logic*: Análisis de la lógica de negocio.	

Este análisis se puede realizar a través de dos procedimientos, debido a que la detección de vulnerabilidades se puede hacer mediante una validación estática

sobre el código fuente de la aplicación previo a su ejecución; y también mediante la validación de la interacción de la aplicación con otras, en el momento de su ejecución, tal como se muestra a continuación.

C4.1 Análisis del código fuente de la aplicación

Se define el análisis de vulnerabilidades que podrían generarse por el uso inapropiado de APIs de terceros, inyección de *malware*, o errores de código que tengan un impacto en permisos de: almacenamiento de datos o archivos, seguridad en los componentes y bases de datos. Para el análisis de esta fase se seleccionan las siguientes herramientas computacionales tal como se observa en la Tabla 2.8.

Tabla 2.8. Herramientas seleccionadas para la fase de análisis del código fuente de la aplicación

Herramienta	Información Colectada	Uso de la Información
APK EXTRACTOR	Archivos en formato APK de la aplicación a analizar.	Conocer el código fuente de la aplicación para generar sobre ellos pruebas de penetración.
ANDROBUGS	Información de vulnerabilidades de seguridad presentes en las aplicaciones.	Conocer fallos de seguridad en el código fuente, mejores prácticas de desarrollo no aplicadas y uso de comandos shell peligrosos.
SANDDROID	Información de vulnerabilidades presentes en APIs sensibles y librerías, incluido un análisis dinámico del código de la aplicación, score y resumen de riesgos.	Conocer los riesgos asociados al uso de librerías y APIs de terceros que representan un riesgo a la información que procesan los usuarios.

C4.2 Análisis de las vulnerabilidades en los componentes de la aplicación

Se define el análisis de los componentes de la aplicación (*Activities, Services, Intents, Broadcast Receivers, Content Provider*) para validar sus niveles de seguridad y detectar vulnerabilidades asociadas a su interacción entre dos o más aplicaciones. Para el análisis de esta fase se seleccionan las siguientes herramientas computacionales tal como se observa en la Tabla 2.9.

Tabla 2.9. Herramientas seleccionadas para la fase de análisis de los componentes de la aplicación

Herramienta	Información Colectada	Uso de la Información
DROZER	Información de componentes de aplicación que contienen vulnerabilidades explotables.	Conocer fallos de seguridad presentes en los componentes de aplicaciones que aplicaciones terceras podrían explotar.

2.1.2 GUÍA DE PROCEDIMIENTOS PARA LA EVALUACIÓN DE AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DE APLICACIONES ANDROID

Para el análisis de afectación a la privacidad, se toma como metodología base el recurso analizado “Evaluación de los riesgos de privacidad en Android” [30], el cual toma como referencia para la ponderación del riesgo asociado a los dispositivos y aplicaciones móviles la metodología para la gestión del riesgo de privacidad CNIL [29]. De acuerdo con la revisión de esta metodología, para hallar el riesgo se consideran los aspectos que se describen a continuación.

2.1.2.1 Metodología de evaluación del riesgo que afecta la privacidad de los usuarios

Para la evaluación del riesgo presente en el uso de las aplicaciones de dispositivos móviles, es necesario tener en cuenta los siguientes factores.

2.1.2.1.1 Identificación de los Activos de Información

En esta fase se analizan los activos de información presentes en el dispositivo móvil, relacionados con los permisos solicitados al momento de la instalación o uso de las aplicaciones Android, tomando como referencia lo revisado en la Tabla 1.6, y que pueden ser objeto de una violación a la privacidad, tales como:

a. Datos de comunicación, SMS, MMS, Voz, etc.
b. Datos de sensores, cámara, micrófono, etc.
c. Datos de ubicación, GPS, ubicación de la red o redes sociales.
d. Datos de almacenamiento externos, datos de aplicaciones, documentos, correos electrónicos, etc.
e. Lista de contactos del dispositivo.
f. Datos de historial/uso, indicador de las preferencias del usuario
g. Datos del calendario, indicador de los contactos y/o la ubicación del usuario.

h. Datos de identidad, ID de dispositivo, perfiles, direcciones de correo electrónico.
i. Credenciales, tokens de autenticación, contraseñas, PIN.

2.1.2.1.2 Asociación de Amenazas (Th)

Con base en los activos de información que pueden ser objeto de violación a la privacidad, se toma como amenazas los siguientes aspectos:

T1. **Seguimiento / Vigilancia**: uso de la información de sensores del dispositivo.

T2. **Interceptación / escucha a escondidas**: interceptación ilícita de comunicaciones.

T3. **Perfil del usuario**: identificación de actividades que realiza el usuario.

T4. ***Phishing***: obtención de credenciales de usuarios vía engaño.

T5. **Divulgación de Información Personal**: divulgación de información personal, como documentos, archivos multimedia, etc.

La respectiva asociación de activos de información con sus amenazas se genera con base en la información revisada en la Tabla 1.6. Mapeo de activos de datos, permisos y amenazas

2.1.2.1.3 Nivel de Identificación del usuario (IdL)

Se analiza el nivel con el cual se pueden identificar a los usuarios de las aplicaciones Android, con base en los permisos que estos solicitan para su uso. Se los establece en la Tabla 2.10.

Si la lógica de la aplicación usada administra información con la cual se puede identificar fácilmente a un usuario o su información básica como identificación, nombres, etc., se pondera en la escala de Máximo aquel permiso que pueda filtrar la información por medio de un canal de transmisión INTERNET, SEND_SMS, BLUETOOTH.

2.1.2.1.4 Nivel de Severidad percibida por el usuario (SeL)

Se analiza el nivel de severidad que cada usuario percibe sobre los activos de información que están expuestos a las respectivas amenazas revisadas con la

siguiente escala de valores: 1–Insignificante, 2–Moderado, 3–Significativo, 4–Máximo.

Tabla 2.10. Nivel de identificación de usuarios de acuerdo con permisos de las aplicaciones

Nivel de Identificación		Permisos
1	Insignificante	ACCESS_COARSE_LOCATION
		ACCESS_FINE_LOCATION
		ACCESS_ NETWORK_STATE
		ACCESS_WIFI_STATE
		BLUETOOTH_ADMIN
		GET_TASKS
		READ_CALENDAR
		READ_HISTORY_BOOKMARKS
		READ_LOGS
		READ_USER_DICTIONARY
		RECEIVE_WAP_PUSH
		SUBSCRIBED_FEEDS_READ
2	Moderado	CAMERA
		PROCESS_OUTGOING_CALLS
		READ_CALL_LOG
		READ_CONTACTS
		READ_EXTERNAL_STORAGE
		READ_SMS
		READ_SOCIAL_ STREAM
		RECEIVE_MMS
		RECEIVE_SMS
		RECORD_AUDIO
		WRITE_EXTERNAL_STORAGE
3	Significativo	AUTHENTICATE_ACCOUNTS
		GET_ACCOUNTS
		USE_CREDENTIALS
4	Máximo	READPHONE_STATE
		READ_PROFILE

2.1.2.1.5 Nivel de Impacto (ImL)

Se analiza el impacto de cada amenaza sobre el activo de información, para lo cual es necesario conocer el valor del Impacto (Im) el cual corresponde a la suma del Nivel de Identificación (IdL) y el Nivel de severidad (SeL), con el cual se pondera su valor mediante la siguiente escala: 1–Insignificante: Im < 5, 2–Moderado: Im = 5, 3–Significativo: Im = 6, 4–Maximo: Im > 6. (Im = IdL + SeL)

2.1.2.1.6 Nivel de Vulnerabilidad (VL)

Se analiza el grado de factibilidad que una amenaza afecte a los activos de información mediante las vulnerabilidades encontradas, para lo cual se pondera el grado de Vulnerabilidad (VL), en base a la siguiente escala: 1–Insignificante, 2–Moderada, 3–Significativa, 4–Máximo.

La ponderación de la Vulnerabilidad(V) está basada en el análisis de los lineamientos revisados en la sección 2.1.1, con relación a los diferentes fallos de seguridad presentes en el dispositivo y aplicación. Además, del lineamiento CNIL que menciona que se debe validar hasta qué punto pueden explotarse las propiedades de los activos de información, para lo cual se realiza un promedio a las ponderaciones individuales para cada control que se detallan en las Tabla 2.11, Tabla 2.12, Tabla 2.13, Tabla 2.14 y Tabla 2.15.

En caso de que se encuentren vulnerabilidades relacionadas al sistema operativo, el cual permitan acceso con perfiles de súper usuario o el dispositivo puede ser accedido físicamente o sustraído. Se ponderan todos los lineamientos con valores críticos, debido a que, con el nivel de privilegios mencionado, es posible realizar cualquier modificación al sistema operativo o a los programas utilizados.

- **C1. Información recopilada del dispositivo**

Tabla 2.11. Ponderación de la vulnerabilidad relacionada con a los activos de información presentes en el terminal móvil

1–Insignificante	2–Moderada	3–Significativa	4–Critica
• Difícil acceso a las características del dispositivo.	• Se puede conocer características básicas de Android o los Aplicativos instalados desde otras aplicaciones o de forma remota.	• Existen vulnerabilidades que permiten fácil acceso a información del dispositivo, aplicaciones o usuarios, relacionados a Android o a los permisos de los aplicativos.	• Existen fallos de seguridad que permiten acceso total a información del dispositivo, aplicaciones o usuario mediante autorización root.

➢ C2. Métodos de autenticación y cifrado

Tabla 2.12. Ponderación de la vulnerabilidad relacionada con los métodos de autenticación y cifrado de las aplicaciones

1–Insignificante	2–Moderada	3–Significativa	4–Critica
• No es posible la captura de información relacionadas con autenticación.	• La aplicación permite ataques de fuerza bruta sobre variables de autenticación. • Se puede capturar nombres de usuarios para autenticación en la aplicación.	• Existen variables de autorización o sesiones almacenadas en el dispositivo o componentes de aplicaciones.	• Es posible capturar variables de autenticación por cualquier método. • Existen variables de autenticación quemadas en código con autorizaciones especiales.

➢ C3. Información transferida entre dispositivos

Tabla 2.13. Ponderación de la vulnerabilidad relacionada con la información transferida entre dispositivos

1–Insignificante	2–Moderada	3–Significativa	4–Critica
• Existe información transferida con encriptación que no puede ser leída bajo ningún concepto.	• Existe información transferida con algún tipo de encriptación, pero puede ser leída por cualquier método.	• Existe información transferida sin ningún tipo de encriptación de canal o de datos.	• Existe transferencia de información con host servidores no identificados.

➢ C4.1. Análisis del código fuente de la aplicación

Tabla 2.14. Ponderación de la vulnerabilidad relacionada con el análisis del código fuente de la aplicación

1–Insignificante	2–Moderada	3–Significativa	4–Critica
• El código fuente de la aplicación está ofuscado y no puede ser descompilado ni alterado.	• Uso de librerías de terceros que tengan un nivel de riesgo moderado, considerado por las herramientas computacionales de análisis.	• Existen *bugs* generados por el desarrollo del dispositivo, que permitan su mal funcionamiento. • El código fuente de la aplicación puede ser descompilado y alterado. • Existen permisos solicitados que no guardan relación con la lógica de la aplicación.	• Uso de librerías de terceros que tengan un nivel de riesgo alto, o *malware* que permitan la extracción de información sensible del dispositivo o usuario.

- **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación**

Tabla 2.15. Ponderación de la vulnerabilidad relacionada con los componentes de la aplicación

1–Insignificante	2–Moderada	3–Significativa	4–Critica
• No puede invocar, acceder, ser accedido, ser invocado desde o hacia ningún componente de la aplicación.	• El aplicativo puede acceder o invocar componentes de otras aplicaciones o recursos del dispositivo.	• Existen componentes expuestos que entregan información sensible de los usuarios implementando alguna restricción, pero pueden ser vulnerados de alguna forma.	• Existen componentes expuestos que entregan información sensible de los usuarios sin ningún tipo de restricción.

2.1.2.1.7 Nivel de Riesgo (RL)

Se analiza la posibilidad que una amenaza pueda afectar activos de información mediante las vulnerabilidades encontradas, para lo cual es necesario obtener el valor del Riesgo(R) el cual es la suma del nivel de Impacto (ImL) y el nivel de Vulnerabilidad (VL), y se lo pondera en base a la siguiente escala 1–Insignificante R: 1 a 3, 2–Moderada R: 4 a 5, 3–Significativa R: 6 a 7, 4–Máximo R:8. (R = ImL + VL).

En la Tabla 2.16 se muestra un ejemplo del cálculo del nivel del riesgo de una aplicación Android “A”, en donde, debido a los permisos que la aplicación solicita para su instalación/ejecución, se han asociado las Amenazas (T) y Activos de información (A) en base a la Tabla 1.6, obteniendo los valores de las columnas amenazas y activos, sobre los cuales se realizará la ponderación del nivel de identificación (IdL), nivel de severidad (SeL), impacto (Im), nivel de impacto (ImL), nivel de vulnerabilidad VL, riesgo (R), y nivel de riesgo (RL) de la aplicación, basados en los factores revisados de la sección 2.1.2.1.

2.1.2.2 Análisis de vulnerabilidades presentes en las aplicaciones Android

Para realizar la evaluación de la afectación a la privacidad de los usuarios, es necesario realizar el siguiente procedimiento con la finalidad de hallar el nivel de riesgo que representa el uso de una aplicación en particular.

Tabla 2.16. Ejemplo de cálculo de riesgo sobre el aplicativo Android (A)

AMENAZAS(T)	ACTIVO	IdL(A)	SeL(A)	Im	ImL(A)	VL(A)	R	RL(A)
T1	A1	1	4	5	2	3	5	2
T2	A2	2	3	5	2	4	6	3
T3	A3	3	1	4	1	3	4	2
T4	A4	4	2	6	3	3	6	3
T5	A5	4	4	8	4	4	8	4
							Riesgo	2.8

Fase 0: Configurar los ambientes donde se desarrollará el análisis de vulnerabilidades.

- Configurar la herramienta BurpSuite para utilizar las herramientas de Proxy y sus diferentes utilidades de intercepción de peticiones web.
- Configurar el respectivo direccionamiento IP para disponer de conectividad entre todos los dispositivos requeridos para el análisis.
- Instalar el agente del aplicativo Drozer en el dispositivo, utilizando el archivo APK disponible en el portal de la aplicación revisada, que también se encuentra disponible en [51] .
- Conectar el dispositivo móvil vía USB en algún computador o dispositivo habilitado para poder acceder a los registros LOGCAT.
- Habilitar el modo desarrollador en el dispositivo móvil, para Android 6 y 7 se realiza presionando de manera consecutiva la versión de compilación del dispositivo (ruta: Configuración>Acerca del dispositivo>Compilación).
- Dentro del menú de opciones de desarrollador habilitar la opción de Depuración USB.

Fase 1: Recopilar toda la información posible del dispositivo a analizar, tomando en cuenta las aplicaciones presentes en el mismo, es necesario que el sistema operativo Android sobre el cual se van a instalar los aplicativos a analizar, esté lo más puro posible, es decir sin aplicaciones de ningún tipo instaladas.

- Usando la herramienta ADB[43] se procede a recolectar la información del dispositivo mediante el comando C:\adb>adb logcat > log_[APK

[43] ADB: Acrónimo de "*Android Debugging Bridge*", Es una Herramienta que viene junto con el SDK de Android y nos permite acceder y controlar un dispositivo Android desde una PC.

ANALIZADA].txt, para un posterior análisis de la información recolectada durante la prueba de penetración.

- Utilizando la herramienta NMap se ejecutan los siguientes comandos para hallar información del dispositivos y puertos TCP y UDP abiertos.
  nmap -p 1-65535 -T4 -A -v [IP ADDR]
  nmap -sS -sU -T4 -A -v [IP ADDR]
- Encender el módulo servidor "Embedded Server" del agente Drozer instalado en el dispositivo.
- Utilizando la herramienta Drozer y el shell de comandos, se ejecutan los siguientes comandos:
  drozer console connect --server [IP ADDR]
  dz> run information.deviceinfo > [DIRECTORIO]\2.2.1[ARCHIVO].TXT
  dz> run app.package.list > [DIRECTORIO]\2.2.2[ARCHIVO].TXT
- Desactivar el programa Drozer en el dispositivo.

Fase 2: Realizar un análisis de vulnerabilidades al dispositivo móvil utilizando la herramienta OpenVas, para conocer fallos de seguridad que puede tener Android, almacenar sus resultados para el posterior análisis de información.

- Crear nuevo *target*: *Configuration> Targets > NewTarget.*
- Crear nuevo *task*: *Scan Management > Tasks > New Task*, seleccionar el *Target*, y seleccionar *Scan Config.*
- Ejecutar *task* creado para iniciar el análisis de vulnerabilidades.

Fase 3: Instalar el aplicativo seleccionado a analizar en el dispositivo móvil, y proceder a ejecutar nuevamente los comandos revisados en la fase 1 y fase 2, para validar si después de instalada la aplicación hay cambios sobre el sistema operativo Android, especialmente presencia de fallas de seguridad.

Fase 4: Se analiza el aplicativo instalado para obtener los permisos requeridos para su funcionamiento, y de esa forma descubrir los activos de información que pueden ser susceptibles a las amenazas definidas.

- Encender el módulo servidor "Embedded Server" del agente Drozer instalado en el dispositivo.

- Mediante el shell del programa Drozer, ejecutar los siguientes comandos para hallar la información de permisos sobre la aplicación a analizar:
  drozer console connect --server [IP ADDR]
  dz> run app.package.list
  dz> run app.package.list -f [PATRON DE BUSQUEDA]
  dz> run app.package.info -a [PAQUETE A ANALIZAR] > [DIRECTORIO]\5.2[ARCHIVO].TXT

Fase 5: Para la valoración de la Vulnerabilidad (V) se analizan cada uno de los lineamientos de la sección 2.1.2.1.6, en base al siguiente procedimiento.

- **C1. Recopilación de información**: Con la información recolectada en las fases 1, 2 y 4, se procede a analizar la ponderación que se colocará con base de las vulnerabilidades encontradas.
- **C2. Métodos de autenticación y cifrado:** se recolecta información que servirá para colocar la ponderación de vulnerabilidad a este lineamiento mediante el siguiente procedimiento:
  - Utilizando Zanti, se realiza un ataque tipo MITM al dispositivo móvil para validar si es posible capturar variables de sesión o algún tipo de información referente a los métodos de autenticación.
    Para la visualización del resultado de la ejecución de los comandos se debe seguir la siguiente ruta: Mapeo de Red > Selección del *host* > Ataques *Man in the Middle* > Pedidos registrados > Ver > Contraseñas.
  - Utilizando la herramienta BurpSuite, se configura los módulos y las interfaces para el uso de la utilidad *proxy* configurado en el puerto 8080, para lo cual es necesario configurar el dispositivo de la siguiente forma:
    - Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas > Proxy > Configuración Manual.
    - Para interceptar tráfico SSL, es necesario instalar el certificado digital de la herramienta BurpSuite (http://[BurpSuite_Host]:8080/cert) en el dispositivo. Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.

- Después de realizar la configuración se procede a revisar métodos POST en el historial de registros HTTP o interceptando peticiones, con palabras claves “LOGIN”, “SECURE”, “USER”, “PASSWORD”, “SESSION”, “AUTHORIZATION”, etc., con la finalidad de validar si es posible capturar variables de autenticación.

- Utilizando la herramienta Wireshark, se configura la captura de paquetes en la opción *Capture* > *Start*. Después se pueden utilizar los siguientes filtros para captura de contraseñas:

  pop.request.command == "USER" || pop.request.command == "PASS"
  smtp.req.command == "AUTH"
  imap.request contains "login"
  http.request.method == "POST"

  - Cuando se encuentren estos paquetes, se puede realizar el análisis usando: [selección del registro] > clic derecho > *Follow* > TCP/UDP *Stream*.

- Utilizar la herramienta DEX2JAR y JDGUI y proceder a obtener y observar el código fuente de la aplicación, para hallar patrones o analizar los métodos de autenticación que usa el aplicativo.
  - Se descomprimen los contenidos del archivo APK utilizando 7Zip[44].
  - Copiar las librerías del programa DEX2JAR en el directorio de los archivos desempaquetados del aplicativo a analizar.
  - En el shell ejecutar el comando:
    C:\d2j-dex2jar.bat classes.dex
  - Verificar el código fuente usando el programa JD-GUI y buscar patrones de código como “*user*”, “*psw*”, etc.
  - Analizamos los métodos de autenticación de la aplicación.

- Utilizando la herramienta Logcat, se capturan los registros de *logs* del dispositivo a analizar, mientras se interactúa con el aplicativo instalado.

[44] 7Zip: Es un potente compresor y descompresor de archivos que soporta un gran número de formatos, representando una excepcional alternativa gratuita. Por defecto utiliza el formato de archivo 7z.

Posteriormente se buscan patrones de búsqueda en la información colectada, buscando llaves como “*user*”, “*pass*”, nombres de usuario, etc.

- **C3. Información transferida entre dispositivos:** se recolecta información que servirá para colocar la ponderación de vulnerabilidad a este lineamiento mediante el siguiente procedimiento**:**

  - Utilizando la herramienta BurpSuite, se configuran los módulos y las interfaces para el uso del proxy configurado en el puerto 8080, para lo cual es necesario configurar el dispositivo de la siguiente forma:
    - Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas> Proxy > Configuración Manual.
    - Para interceptar tráfico SSL, es necesario instalar el certificado digital de la herramienta BurpSuite (http://[BurpSuite_Host]:8080/cert) en el dispositivo. Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.
    - Después de realizar la configuración se procede a revisar métodos GET y POST en el historial de registros HTTP o interceptando las peticiones web, con la finalidad de validar si es posible capturar información transferida entre dispositivos.
  - Utilizando la herramienta Wireshark, se configura la captura de paquetes en la opción *Capture* > *Start*. Después se pueden utilizar los siguientes filtros (de visualización) para validar actividad maliciosa o verificar la transferencia de información entre los dispositivos:
    - dns
    - http.request
    - http.request.method == "GET"
    - http.request.method == "POST"

  Cuando se encuentren estos paquetes, se puede realizar el análisis usando: [selección del registro] > clic derecho > *Follow* > TCP/UDP *Stream*.

- **C4.1. Análisis del código fuente de la aplicación:** se recolecta información que servirá para colocar la ponderación de vulnerabilidad a este lineamiento mediante el siguiente procedimiento:

- Utilizando la herramienta APK Extractor de preferencia en un dispositivo ajeno al analizado, se procede a extraer el archivo APK de la aplicación, seleccionando la aplicación a extraer y usando la opción compartir.
- Utilizando la herramienta AndroBugs, se procede a realizar un análisis sobre los códigos fuentes de la aplicación presentes en el archivo APK extraído usando el siguiente comando:
  - C:\AndroBugs_Framework>androbugs.exe -f [DIRECTORIO_APK]

  Posteriormente analizar la información presentada por el informe de la herramienta para realizar la ponderación de vulnerabilidad en base a los riesgos de seguridad encontrados.
- Utilizando la herramienta SandDroi, se procede a realizar un análisis sobre los códigos fuentes de la aplicación presentes en el archivo APK extraído, utilizando el siguiente procedimiento:
  - Acceder al portal de la herramienta y proceder a cargar el archivo apk para su análisis Aplicación > *Upload*.
  - Después de completar el análisis respectivo, procederemos a buscar el reporte en la sección *Overview*, para analizar especialmente las secciones: comportamiento de riesgo, malware detectado, clasificación, características del código, permisos, componentes de aplicación, URL usadas, API sensibles y usos de permisos.

➢ **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación:** se recolecta información que servirá para colocar la ponderación de vulnerabilidad a este lineamiento mediante el siguiente procedimiento:

- Encender el agente Drozer instalado en el dispositivo y utilizando la herramienta *shell* Drozer, ejecutar los siguientes comandos:

  drozer console connect --server [IP ADDR]
  - dz> run app.package.list -f [CLAVE DE BUSQUEDA PAQUETE]
  - dz> run app.package.manifest [PAQUETE] > [DIRECTORIO]\6.5.1[ARCHIVO].TXT
  - dz> run app.package.attacksurface [PAQUETE] > [DIRECTORIO]\6.5.1[ARCHIVO].TXT

- Después de generado el ataque de superficie, se procede a listar los componentes que presentan exposición a algún tipo de vulnerabilidad, para lo cual se usan los siguientes comandos:
  - *Activities*: dz> run app.activity.info -a [PAQUETE]
  - *Content Provider:* dz> run app.provider.info -a [PAQUETE]
    dz> run scanner.provider.finduris -a [PAQUETE]
  - *Services*: dz> run app.service.info -a [PAQUETE]
  - *Broadcast Receiver:* dz> run app.broadcast.info --package [PAQUETE]

- Con la información recolectada previa, se procede a realizar pruebas de seguridad y explotar los componentes según sea el caso, con la finalidad de encontrar vulnerabilidades sobre los componentes de la aplicación de acuerdo con los siguientes comandos:
  - Explotar *Activities*:
    dz> run app.activity.start –component [PAQUETE] [NOMBRE ACTIVITY]
  - Explotar *Content Provider* para obtener información*:*
    dz> run app.provider.query [URI CONTENT PROVIDER]
  - Comprobar rutas de acceso transversales[45] en *Content Provider*:
    dz> run scanner.provider.traversal -a [PAQUETE]
  - Comprobar vulnerabilidades de SQL *Injection*:
    dz> run scanner.provider.injection -a [PAQUETE]
  - Explotar *Services:*
    dz> run app.service.send [PAQUETE] [NOMBRE SERVICIO] --msg [PARAMETROS]
  - Explotar *Broadcast Receiver*:
    dz> run app.broadcast.send --action [ACCION del archivo android_manifest] --component [PAQUETE] [*BROADCAST RECIEVER*] --extra string [PARAMETROS DE ENTRADA(Validados en código)]

- Desactivar el agente Drozer en el dispositivo.

[45] Rutas de acceso transversal. Vulnerabilidad informática presente cuando no existe suficiente seguridad en la validación de un usuario, permitiéndole acceder a cualquier tipo de directorio superior sin ningún control.

2.1.2.3 Análisis de información recolectada y evaluación de la afectación a la privacidad de los usuarios

Con base en la información recolectada de la sección 2.1.2.2, se procede a evaluar la afectación a la privacidad de los usuarios mediante el siguiente procedimiento:

Fase 1: De acuerdo a la información obtenida en la cuarta fase de los procedimientos para el análisis de vulnerabilidades presentes en las aplicaciones Android, se deben asociar los permisos encontrados con la finalidad de hallar los Activos de Información, Amenazas(Th) y Nivel de Identificación de los Usuarios(IdL), con base en la sección 2.1.2.1 donde se menciona los lineamientos para el análisis de vulnerabilidades relacionadas con la privacidad de los usuarios, que posteriormente servirá para el análisis de nivel de riesgo.

Fase 2: Se establecen los valores de Nivel de Severidad (SeL) en base a la percepción que un usuario en particular tiene sobre qué tan grave es que se filtre su información, sea al momento de la instalación o del uso de los aplicativos Android.

Fase 3: Con la información colectada acerca de las vulnerabilidades encontradas en el aplicativo seleccionado, se procede a realizar el cálculo respectivo de las ponderaciones de vulnerabilidad en base a los lineamientos de la sección 2.1.1.

Fase 4: Posteriormente con toda la información colectada se obtiene los valores de Riesgo(R) y Nivel de Riesgo (RL) asociado al uso del aplicativo Android analizado con base en la sección 2.1.2.1.7, los cuales serán presentados como parte del reporte de la afectación a la privacidad de los usuarios de aplicaciones.

CAPÍTULO 3

ANÁLISIS DE AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DE APLICACIONES DE INSTITUCIONES PUBLICAS DEL ECUADOR

En este capítulo se procede a implementar la sección 2.1.2.2 referente al análisis de vulnerabilidades presentes en las aplicaciones Android, sobre las aplicaciones seleccionadas SRI Móvil, ECU911, Gestión de Riesgos Ecuador, CNT Móvil, con la finalidad de detectar fallos de seguridad que puedan incidir sobre los activos de información gestionados, afectando la privacidad de los usuarios.

3.1 SELECCIÓN DE APLICATIVOS ANDROID A ANALIZAR

Para la selección de aplicativos Android a analizar, se han elegido aquellas aplicaciones que tienen relevancia al momento de administrar, almacenar o transferir información confidencial de los usuarios, siendo estas aplicaciones parte de sectores gubernamentales críticos o estratégicos del Ecuador, gracias al tipo de información que gestionan. Las aplicaciones seleccionadas se muestran en la Tabla 3.1.

Tabla 3.1. Aplicaciones Android de instituciones públicas del Ecuador seleccionadas para análisis de afectación a la privacidad de los usuarios

APLICACIÓN	INFORMACIÓN GESTIONADA	FORMA DE ACCESO A INFORMACIÓN
SRI Móvil - SRI ECUADOR	información de impuestos fiscales de los usuarios	Portal con autenticación
CNT Móvil	Información de suscripción a planes de telefonía y datos	Portal con autenticación, posterior a suscripción
Ecuador Seguro	Información de riesgos a nivel de país, para emisión de información y alertas tempranas	No necesita autenticación, posterior a suscripción
ECU 911	Información de seguridad ciudadana y gestión de denuncias	No necesita autenticación, posterior a suscripción

3.2 CONFIGURACIÓN DE AMBIENTES PARA DESARROLLAR EL ANÁLISIS DE VULNERABILIDADES SOBRE LOS APLICATIVOS ANDROID SELECCIONADOS

Para el procedimiento del análisis de vulnerabilidades se realiza en primer lugar la configuración de los ambientes y herramientas revisadas en la sección 2.1.2.2 Fase 0, la cual es común para todas las aplicaciones seleccionadas, y su procedimiento se cita a continuación.

- Configuración de la herramienta BurpSuite en sus secciones de proxy y parámetros de intercepción de peticiones web tal como se muestra en la Figura 3.1.

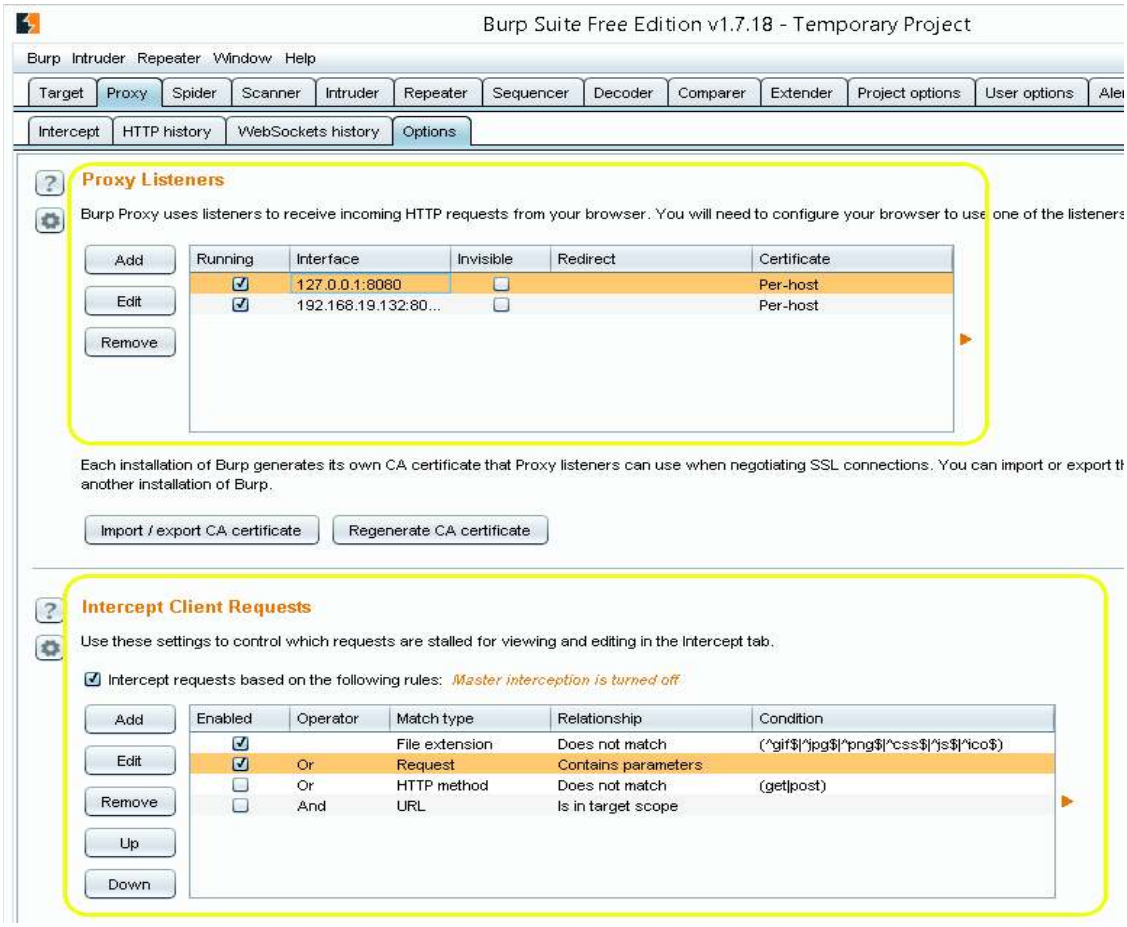

Figura 3.1. Configuración proxy BurpSuite

- Configuración de dirección IP en el dispositivo móvil (192.168.19.104), para lo cual se utiliza una dirección de una red Wifi previamente configurada en la infraestructura de red del ambiente de pruebas, tal como se muestra en la Figura 3.2.

Figura 3.2. Direccionamiento IP del dispositivo móvil

- Instalación de agente Drozer en el dispositivo móvil, para lo cual se utiliza el archivo .apk disponible en el portal de la aplicación revisada, disponible en [51] , cuya interfaz se muestra en la Figura 3.3.

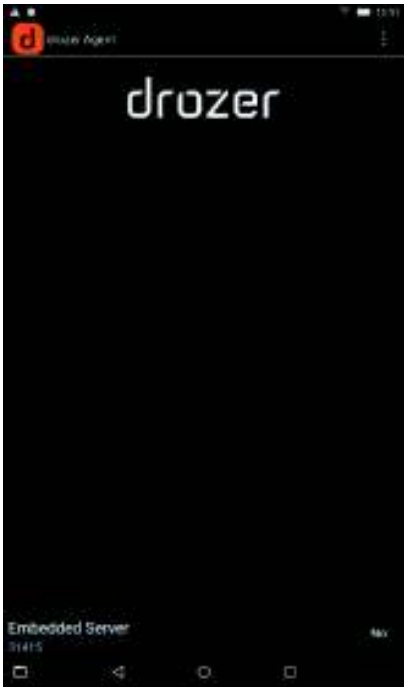

Figura 3.3. Agente Drozer instalado en el dispositivo móvil

- Establecimiento de la conexión USB entre el dispositivo móvil y el computador, para transferencia de información y ejecución de comandos, tal como se muestra en la Figura 3.4.

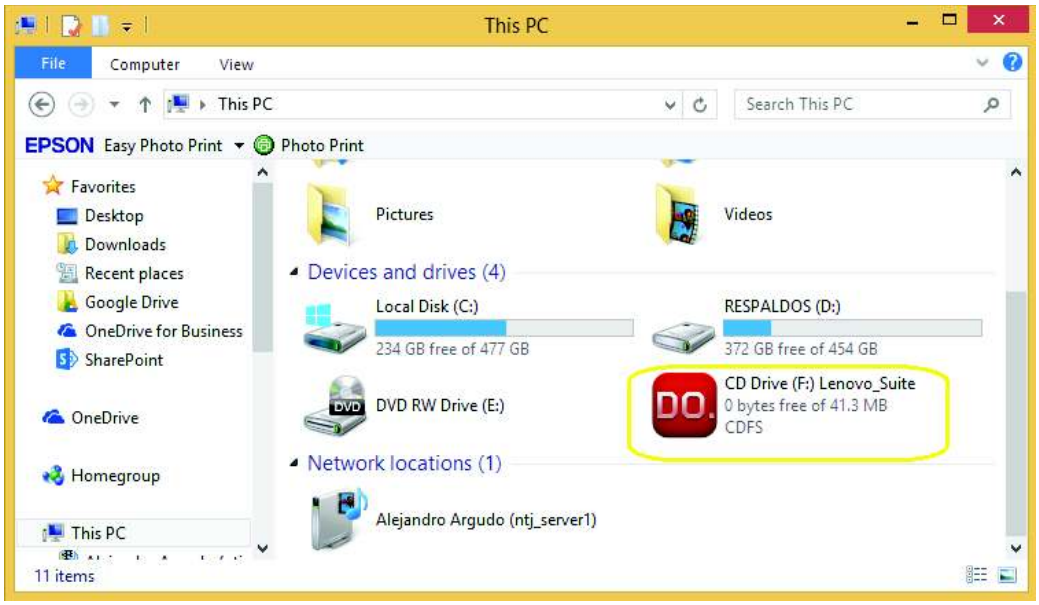

Figura 3.4. Conexión del dispositivo mediante interfaz USB

- Habilitación del modo desarrollador y depuración USB en el dispositivo móvil, tal como se observa en la Figura 3.5.

Figura 3.5. Modo desarrollador, Depuración por USB activo

- Recolección de información de logs del dispositivo usando la conexión USB, mediante la ejecución del comando adb logcat, tal como se observa en la Figura 3.6.

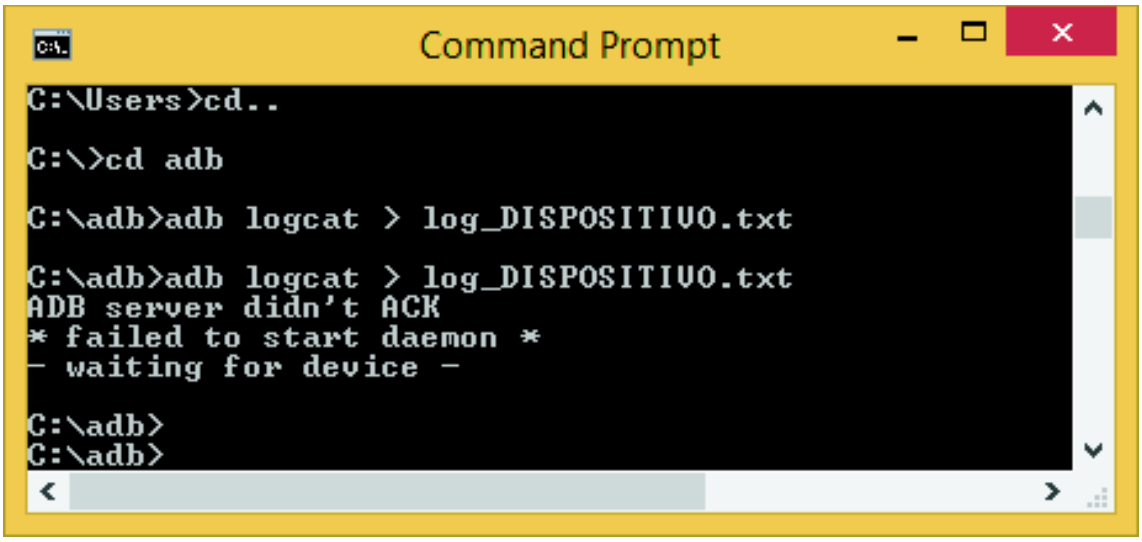

Figura 3.6. Extracción de logs del dispositivo móvil

3.3 IMPLEMENTACIÓN DEL ANÁLISIS DE VULNERABILIDADES SOBRE EL APLICATIVO SRI MÓVIL - SRI ECUADOR FINANZAS

La aplicación SRI Móvil es un canal de información del Servicio de Rentas Internas de la República del Ecuador, que permite brindar a la ciudadanía acceso a consultas tributarias públicas, noticias, información de agencias y redes sociales de la institución desde dispositivos móviles Android, de manera ágil y de fácil uso.

Los servicios de consulta que se ofrecen son: Estado Tributario, Valor de Matrícula Vehicular, Deudas con el SRI, Validez de Documentos Físicos, Impuesto a la Renta Causado, Seguimiento de Trámites, Validación de códigos QR para verificación en línea de declaraciones, certificados y documentos emitidos por la Administración del SRI, Solicitud de Turnos en línea, Calculadoras de impuestos a pagar y Denuncias.

De acuerdo con el procedimiento establecido en la sección 2.1.2.2 relacionado al análisis de vulnerabilidades presentes en las aplicaciones Android, se realiza el análisis de la aplicación SRI Móvil, de acuerdo con las siguientes fases:

Fase 0: Configurar los ambientes donde se desarrollará el análisis de vulnerabilidades, en base al procedimiento mostrado en la sección 3.2, con la finalidad de obtener un ambiente limpio para pruebas, que sirva para realizar el análisis de vulnerabilidades a la aplicación SRI móvil.

Fase 1: Recopilación de información del dispositivo móvil.

A continuación, se procede a recopilar toda la información posible del dispositivo a analizar, para lo cual se aplica el siguiente procedimiento:

- Ejecutar el siguiente comando: nmap -p 1-65535 -T4 -A -v 192.168.19.104 y nmap -sS -sU -T4 -A -v 192.168.19.104, tal como se muestra en la Figura 3.7, para rastrear puertos presentes en el dispositivo móvil.

Los resultados de este comando se encuentran en el Anexo 10.1.1. y en el Anexo 10.1.2.

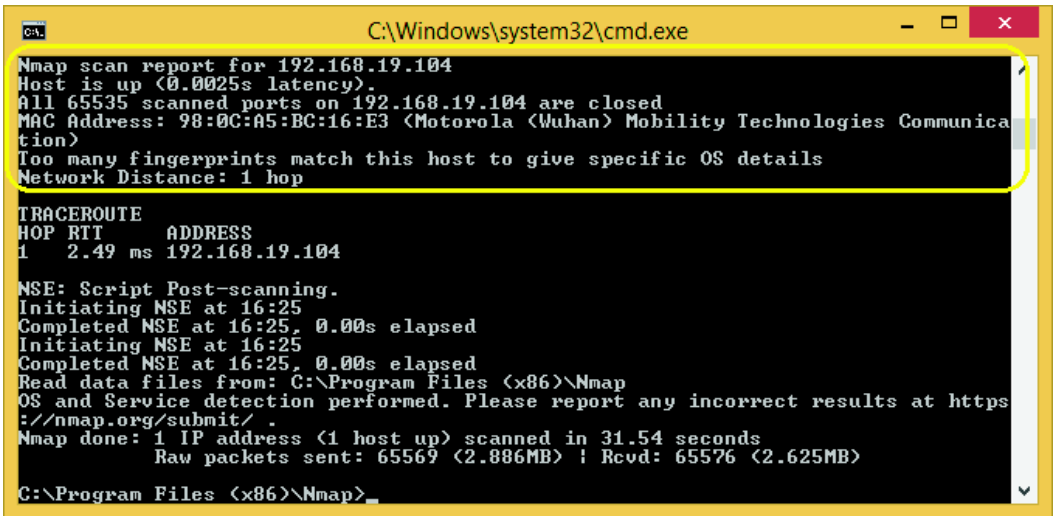

Figura 3.7. Captura de respuesta comando Nmap con información relevante

- Encender el módulo servidor “Embedded Server” del agente Drozer instalado en el dispositivo, tal como se muestra en la Figura 3.8

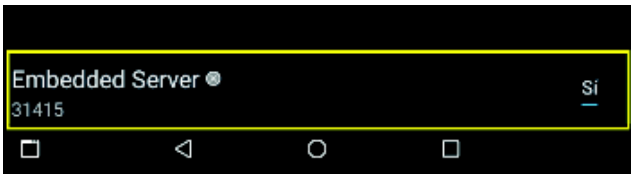

Figura 3.8. Activación del módulo Embedded Server en Drozer

- Realizar la conexión entre la aplicación Drozer y shell de comandos, tal como se muestra en la Figura 3.9, con la finalidad de ejecutar los comandos listados a continuación, que sirven para obtener la información básica del dispositivo móvil, y el listado de aplicaciones presentes en el mismo:

  dz> run information.deviceinfo > c:\DRZ_EXT\2.3.1.APK_SRI.txt

  dz> run app.package.list > c:\DRZ_EXT\2.3.2.APK_SRI.txt

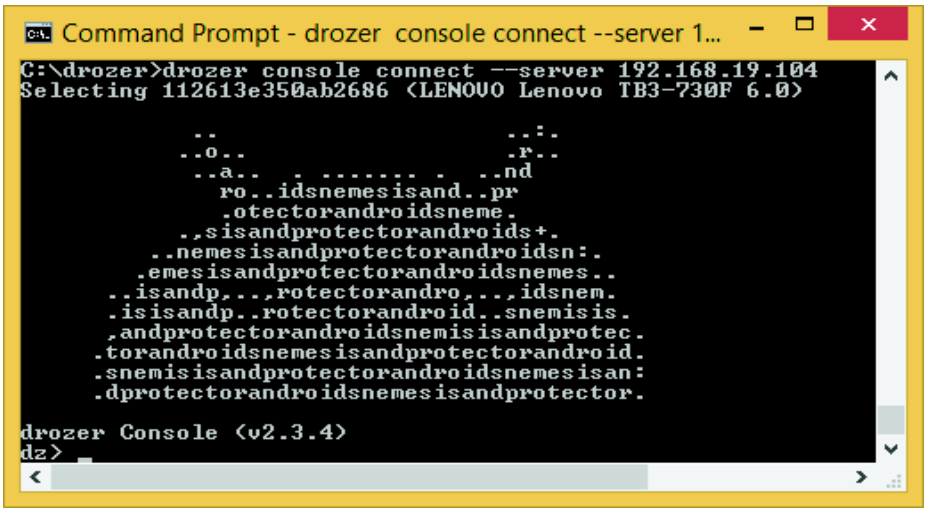

Figura 3.9. Shell De herramienta Drozer

Los resultados de este procedimiento se encuentran en el Anexo 3.1.

- Desactivación del programa Drozer en el dispositivo.

Fase 2: Análisis de vulnerabilidades al dispositivo móvil utilizando la herramienta OpenVAS.

A continuación, se procede a ejecutar un análisis de vulnerabilidades presente en el dispositivo móvil, con la finalidad de conocer fallos de seguridad que pueda tener Android, para lo cual se aplica el siguiente procedimiento:

- Creación de nuevo *Target* TB_LENOVO, tal como se muestra en la Figura 3.10.

Figura 3.10. Creación de Target en OpenVAS.

- Creación de nuevo *Task VULNERABILIDADES_TB_LENOVO,* tal como se muestra en la Figura 3.11

Figura 3.11. Creación de Task en OpenVAS.

- Ejecución de *Task y revisión del reporte, tal* como se muestra en la Figura 3.12.

Figura 3.12. Reporte de Task ejecutado en OpenVAS.

En esta etapa, el aplicativo OpenVAS no encontró ningún incidente relacionado con vulnerabilidades presentes en el dispositivo móvil.

Fase 3: Instalación del aplicativo SRI Móvil.

A continuación, se procede a realizar la instalación del aplicativo SRI móvil, tal como se muestra en la Figura 3.13, con la finalidad de realizar posteriormente el

respectivo análisis de vulnerabilidades sobre el aplicativo instalado y el dispositivo móvil.

Figura 3.13. Instalación de aplicación SRI Móvil

Después de instalado el aplicativo, se realiza una nueva ejecución de los procedimientos de la Fase 2, primer procedimiento, y Fase 3, para validar si después de instalada la aplicación existe cambios sobre Android, o algún fallo de seguridad. Los resultados de esta fase se encuentran en el Anexo 10.1.3, y 10.1.4.

Fase 4: Análisis de los permisos del aplicativo instalado SRI Móvil.

A continuación, se procede a obtener los permisos requeridos para el funcionamiento de la aplicación instalada con la finalidad de descubrir los activos de información que pueden ser susceptibles de amenazas, para lo cual se aplica el siguiente procedimiento:

- Encender el módulo servidor “Embedded Server” del agente Drozer.
- Ejecución de comandos Drozer, tal como se muestra en la Figura 3.14.
  - Conexión al Shell de comandos.
    drozer console connect --server 192.168.19.104
  - Enumerar todos los paquetes instalados en el dispositivo.
    dz> run app.package.list
  - Búsqueda de algún paquete instalado con la clave “sri”.
    dz> run app.package.list -f sri

- Obtener información básica sobre el paquete encontrado SRI.

  dz> run app.package.info -a ec.gob.sri.movil.app > c:\DRZ_EXT\5.2.INFO_SRI.txt

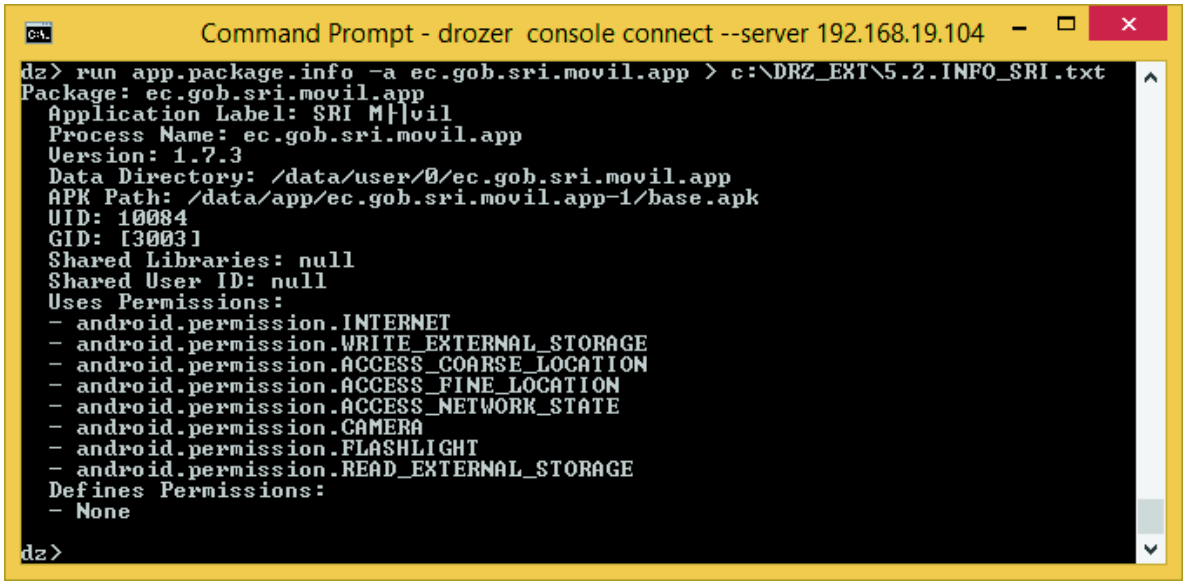

Figura 3.14. Captura de información del paquete SRI utilizando Drozer.

Los resultados relacionados con esta fase se encuentran en el Anexo 3.1.3.

Fase 5: Valoración de la Vulnerabilidad (V).

A continuación, se procede a analizar vulnerabilidades presentes en el dispositivo y en la aplicación SRI móvil, con la finalidad de obtener información que permita conocer la presencia de vulnerabilidades que podrían afectar la privacidad de los usuarios, para lo cual se aplica el siguiente procedimiento de acuerdo a cada uno de los controles establecidos en la sección 2.1.1:

- **C1. Recopilación de información**. La información fue recolectada en las fases 1, 2 y 4 del procedimiento, y servirá posteriormente para realizar la ponderación de nivel de vulnerabilidad para este control, en base a los lineamientos de la sección 2.1.2.1.6.
- **C2. Métodos de autenticación y cifrado**
  - Ataque tipo MITM utilizando Zanti

    A continuación, se procede a ejecutar el paquete Zanti para realizar la captura de contraseñas y variables de sesión, tal como se muestra en la Figura 3.15.

Para la visualización del resultado de la ejecución de la funcionalidad se debe seguir la siguiente ruta: Mapeo de Red > Selección del host > Ataques Man in the Middle > Pedidos registrados > Ver > Contraseñas.

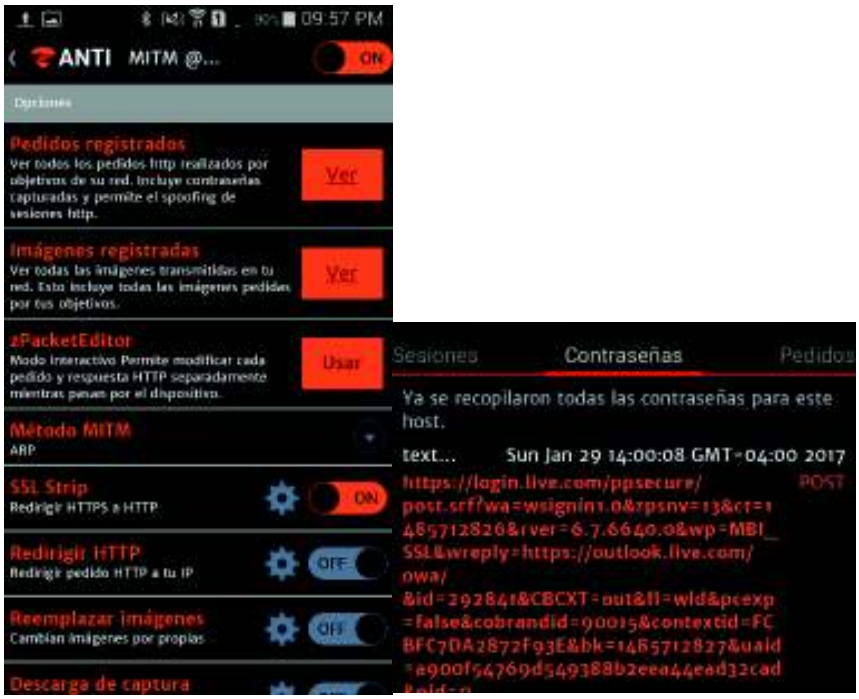

Figura 3.15. Ejemplo de captura de contraseñas en Zanti mediante ataques MiTM

En esta etapa no se obtuvo capturas de contraseñas ni variables de sesión utilizadas en el programa SRI móvil.

- Configuración de la conexión proxy en el dispositivo móvil.
  A continuación, se procede a configurar en el dispositivo móvil el uso del proxy BurpSuite configurado en la Fase 0, con la finalidad de interceptar historial de registros http, peticiones entre dispositivos y tráfico SSL relacionado con autenticación, para lo cual se aplica el siguiente procedimiento:
  - Configuración de las direcciones y parámetros del proxy en el dispositivo móvil, tal como se muestra en la Figura 3.16. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas> Proxy > Configuración Manual.

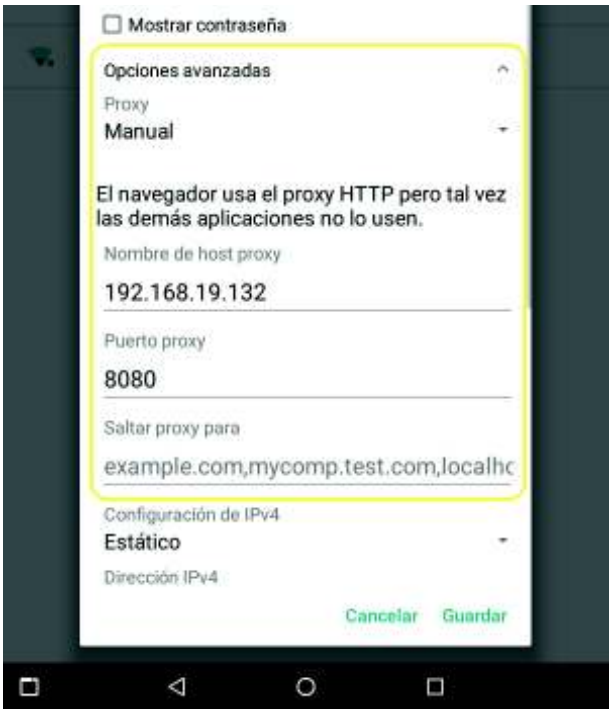

Figura 3.16. Configuración de proxy en el dispositivo móvil

- Instalación del certificado digital de la herramienta BurpSuite, tal como se muestra en la Figura 3.17. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.

Figura 3.17. Instalación del certificado digital BurpSuite en el dispositivo móvil

- Revisión de métodos POST con claves de palabras referentes a autenticación en el historial de registros usando BurpSuite, tal como se muestra en la Figura 3.18. Los resultados relacionados con esta fase se encuentran en el Anexo 2.1

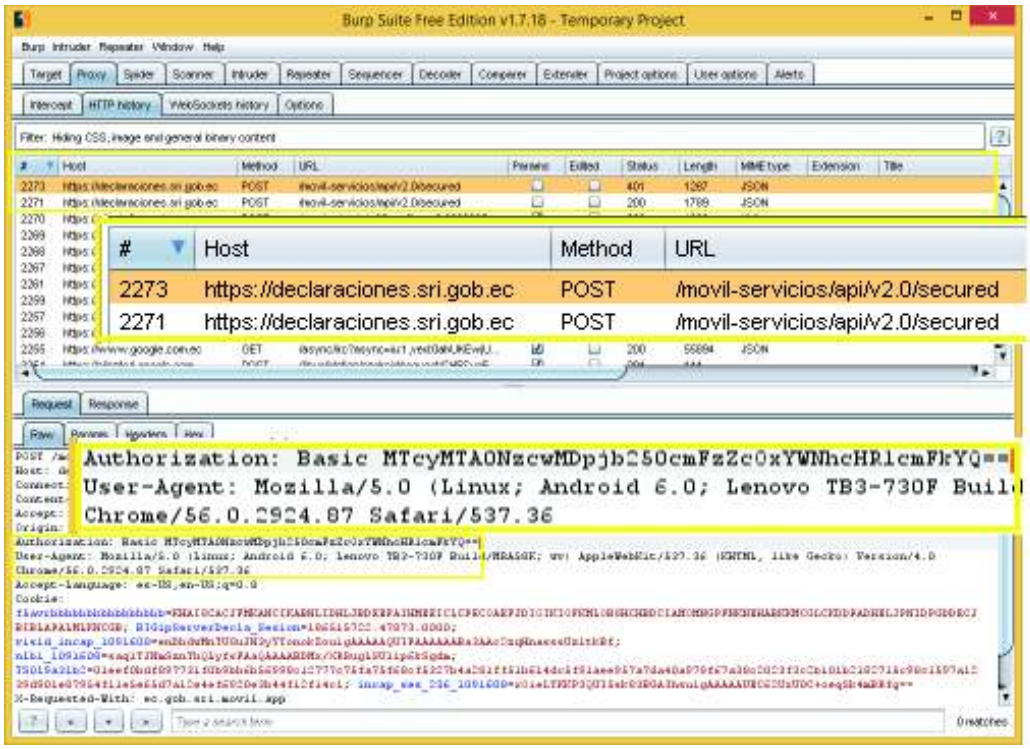

Figura 3.18. Captura de variables de autenticación en métodos POST

En esta etapa se obtuvieron capturas de variables tales como: claves de autorización que se transfieren entre cliente y servidor, versión del cliente web desde el que se invocan las peticiones, nombre de host, etc. que pueden ser visualizadas en texto plano mediante el uso de la herramienta BurpSuite.

- Revisión de paquetes de red transferidos entre el dispositivo móvil y servidor relacionado con autenticación, utilizando la herramienta Wireshark tal como se muestra en la Figura 3.19.

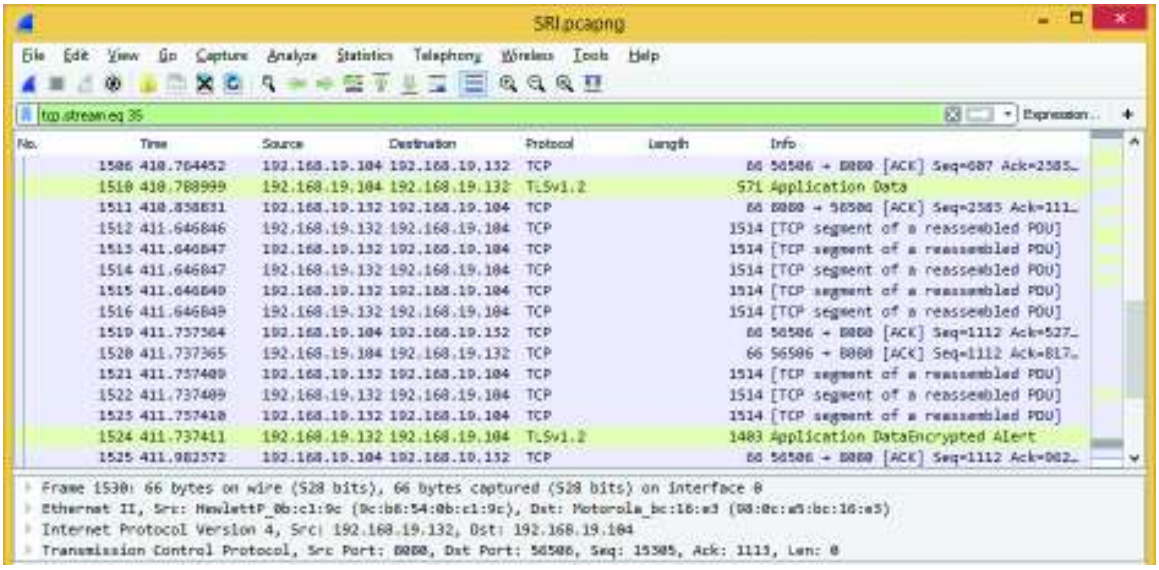

Figura 3.19. Captura de paquetes de red usando Wireshark

En esta etapa se obtuvieron capturas de información transferidas entre el dispositivo y el servidor, pero no se logró capturar variables relacionadas con autenticación. Los resultados relacionados con esta fase se encuentran en el Anexo 6.1.2.

- Revisión del código fuente de la aplicación usando DEX2JAR y JDGUI, donde se puede observar el código de las clases, especialmente aquellas relacionadas con autenticación y cifrado, como se muestra en la Figura 3.20.

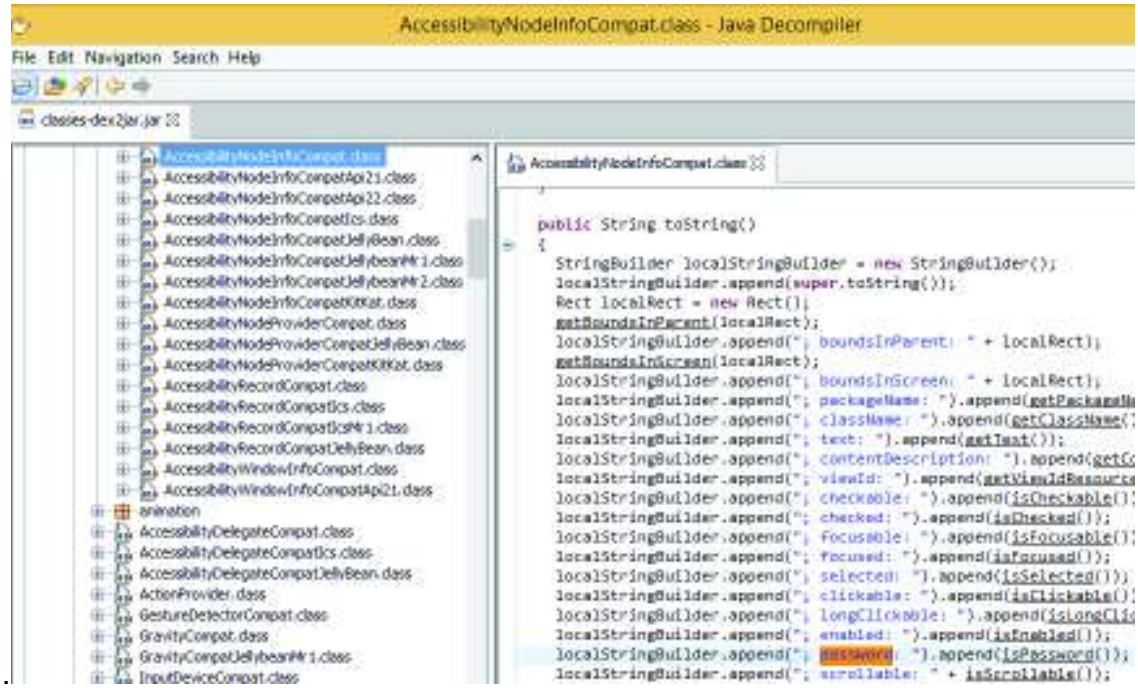

Figura 3.20. Revisión de código fuente de la aplicación SRI

En esta etapa no se obtuvieron errores detectados en código que podrían afectar a los métodos de autenticación, ni tampoco claves quemadas en el código fuente de la aplicación. Los resultados relacionados con esta fase se encuentran en el Anexo 16.1.

- Captura de registros log mediante Logcat

  A continuación, se procede a extraer los registros generados por la aplicación SRI móvil que se almacenan en los archivos log de Android, utilizando la herramienta Logcat.

  En esta etapa se detectó, que las variables de autenticación se estaban almacenando en texto plano en los logs de Android, tal como se muestra en

la Figura 3.21. Los resultados relacionados con esta fase se encuentran en el Anexo 4.1.

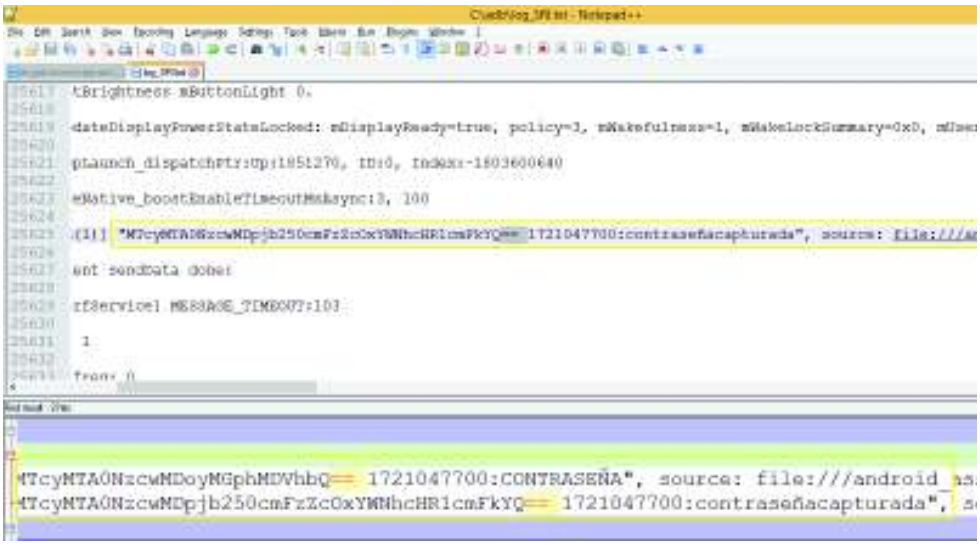

Figura 3.21. Captura de registros Log utilizando Logcat

- **C3. Información transferida entre dispositivos:**
  - Revisión de métodos GET y POST utilizando BurpSuite

    A continuación, se procede a revisar las peticiones y respuestas generados por los métodos GET y POST en el historial de registros de la herramienta, tal como se muestra en la Figura 3.22, donde se puede observar que la información sensible gestionada por el aplicativo SRI Móvil, se transfiere usando protocolo https, pero puede ser visualizada en texto plano mediante el uso del proxy BurpSuite, como se muestra en la opción *Response* de la herramienta.

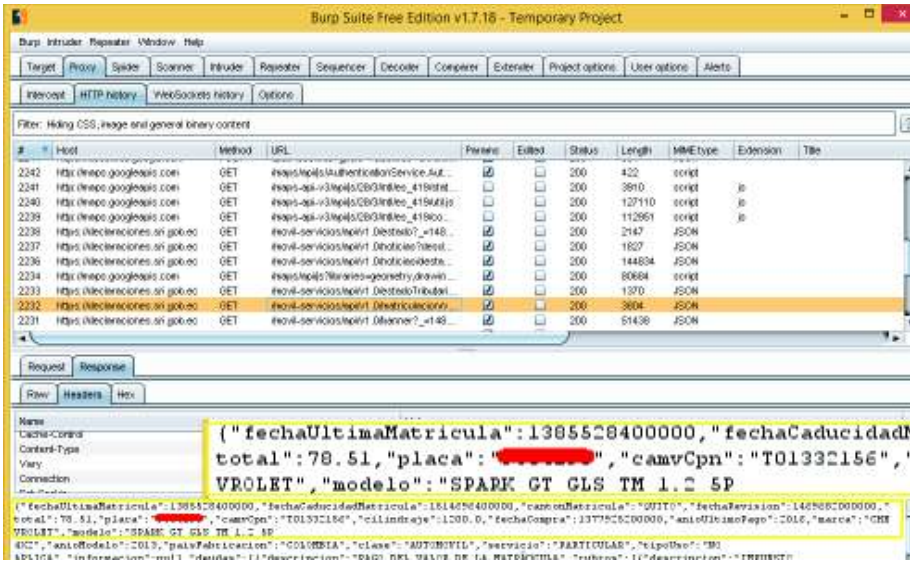

Figura 3.22. Método GET capturado usando BurpSuite

Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 2.1.

- Captura de paquetes Utilizando Wireshark

  A continuación, se procede a analizar los paquetes transferidos entre el aplicativo SRI Móvil y el servidor utilizando Wireshark, tal como se muestra en la Figura 3.23.

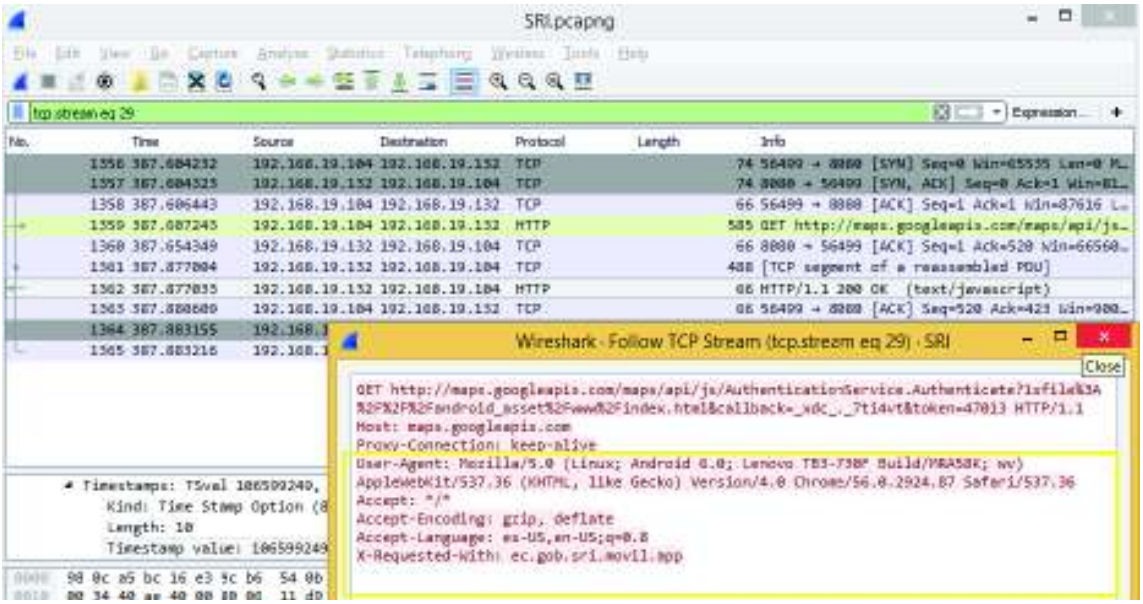

Figura 3.23. Captura de paquetes de métodos GET usando Wireshark

En esta etapa, se logró capturar información relacionada con el dispositivo móvil que contiene el aplicativo SRI Móvil tal como se puede observar en la sección *Follow* TCP *Stream* de la Figura 3.23, pero no se pudo observar información sensible gestionada por el aplicativo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 6.1.2.

➢ C4.1. Análisis del código fuente de la aplicación

- Extracción del archivo APK usando APK Extractor

  A continuación, se procede a extraer el archivo APK del aplicativo SRI móvil, para lo cual se utiliza el aplicativo APK Extractor instalado en el mismo dispositivo donde se va a realizar la extracción, tal como se muestra en la Figura 3.24.

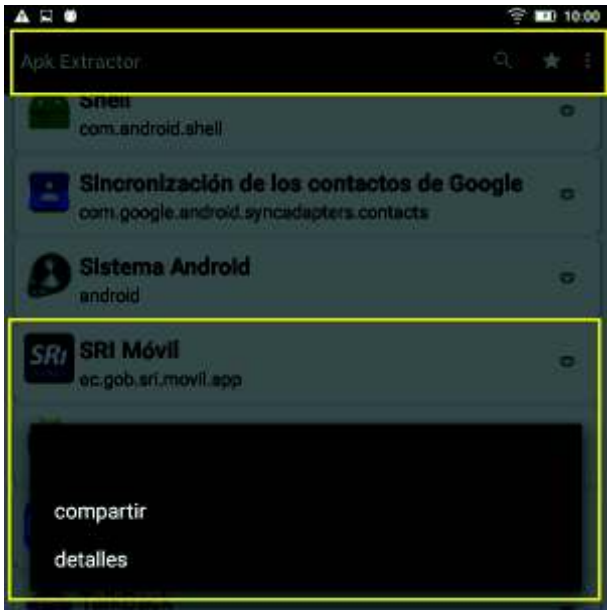

Figura 3.24. Extracción de archivos apk de aplicación SRI Móvil usando ApkExtractor

- Análisis estático de la aplicación utilizando AndroBugs

  A continuación, se procede a realizar un análisis de vulnerabilidades sobre el archivo apk del aplicativo SRI Móvil, utilizando la herramienta AndroBugs, teniendo como resultado un archivo de texto donde se pueden observar las incidencias encontradas durante el análisis, tal como se muestra en la Figura 3.25. El comando ejecutado en este procedimiento es: C:\AndroBugs_Framework>androbugs.exe -f sri.movil.app.apk.

ec.gob.sri.movil.app_733190b0d4afdc3fc6114dd64c4f3e9470f6c875c232650647eb0946e07fc66a3d643af2ea61dae4c845ffd3efe4fb214475e88b478472eee9a5df107e

```
1  ********************************************************************************
2  **   AndroBugs Framework - Android App Security Vulnerability Scanner  **
3  **                          version: 1.0.0                            **
4  **     author: Yu-Cheng Lin (@AndroBugs, http://www.AndroBugs.com)     **
5  **               contact: androbugs.framework@gmail.com                **
6  ********************************************************************************
7  Platform: Android
8  Package Name: ec.gob.sri.movil.app
9  Package Version Name: 1.7.3
10 Package Version Code: 298
11 Min Sdk: 14
12 Target Sdk: 23
13 MD5   : 9062f168ac2cdabd0ec28b5b1ebd07ae
14 SHA1  : 8cb6c8e3b1ec1683e60f42e03dbf5b849a4224b3
15 SHA256: 841711b50c173b779d8fdb24f21ec5023e32dc242ee9b8a46acea6d083bd14f1
16 SHA512: 75ac7f45ba004bfa3fdd05a263e0a0b0a7cb4741eafd0e7bd520a343014088440685655 3fb2f57ad563c52f1
17 Analyze Signature: 733190b0d4afdc3fc6114dd64c4f3e9470f6c875c232650647eb0946e07fc66a3d643af2ea61c
18 --------------------------------------------------------------------------------
19 [Critical] <#BID 64208, CVE-2013-6271#> Fragment Vulnerability Checking:
20            'Fragment' or 'Fragment for ActionbarSherlock' has a severe vulnerability prior to Ar
21            Please check:
22            (1)http://developer.android.com/reference/android/os/Build.VERSION_CODES.html#KITKAT
23            (2)http://developer.android.com/reference/android/preference/PreferenceActivity.html#
```

Figura 3.25. Informe de vulnerabilidades generado por AndroBugs

En esta etapa, se pudo validar en el resultado generado, la presencia de varias incidencias del tipo crítico, en donde una de las más críticas

mencionaba el uso de urls sin SSL para transferencia de información usada en el aplicativo.

Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 1.1

- Análisis de vulnerabilidades del archivo apk utilizando SandDroid.

A continuación, se procede a realizar la carga del archivo apk del aplicativo SRI Móvil en el portal de la herramienta SandDroid, donde se ejecuta un análisis de vulnerabilidades sobre la aplicación, generando un resultado dentro del portal [46], tal como se puede observar en la Figura 3.26.

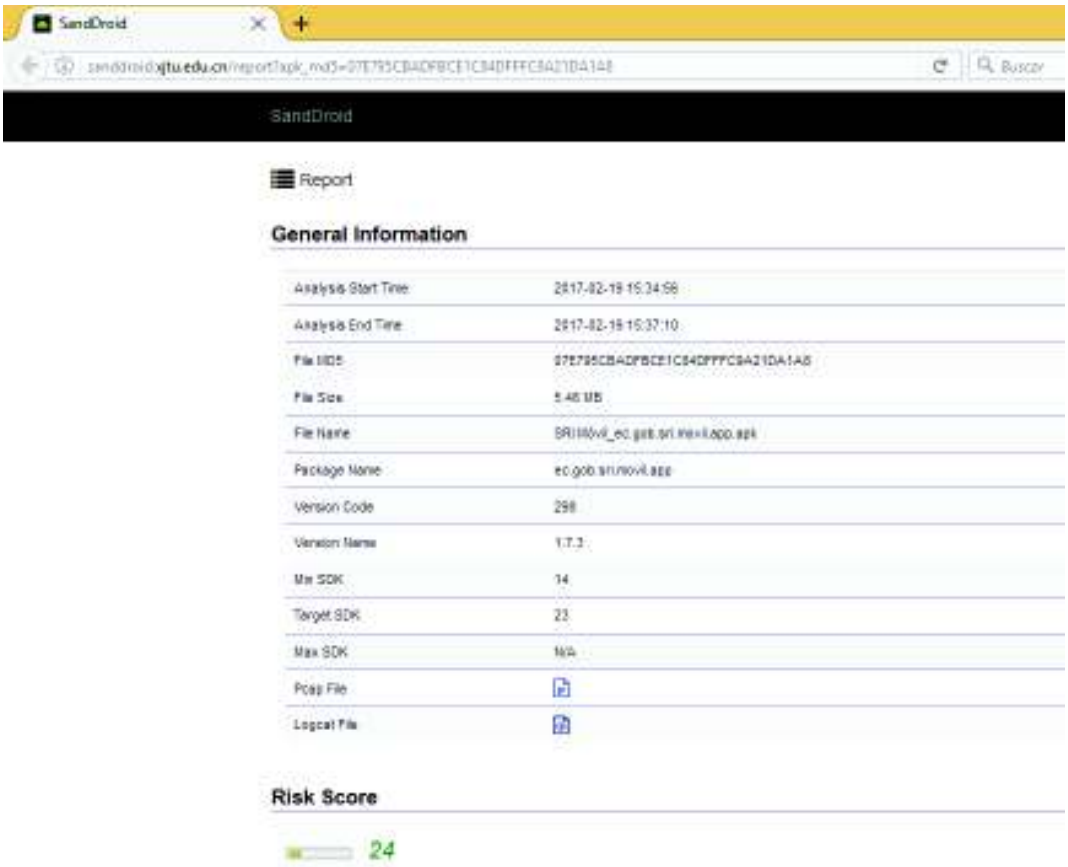

Figura 3.26. Reporte del archivo apk SRI Móvil, generado por SandDroid

En el reporte entregado por SandDroid se pudo validar la presencia de varias incidencias, entre las más notables se tiene que existe permisos brindados no utilizados por la aplicación, tener acceso a cámara, u obtener información de aplicaciones presentes en el dispositivo, entre otros. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 5.1.

[46] http://sanddroid.xjtu.edu.cn/report?apk_md5=07E795CBADFBCE1C84DFFFC9A21DA1A8

- **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación:**

  - Análisis de componentes de la aplicación Android utilizando Drozer.

    A continuación, se procede a realizar la ejecución del análisis de componentes presentes en el aplicativo SRI Móvil, con la finalidad de realizar una prueba de penetración sobre el aplicativo donde muestre incidentes de seguridad presentes en la aplicación, para lo cual utilizamos el Shell de comandos que se muestra en la Figura 3.27, de acuerdo al procedimiento:

    - Conexión al Shell de comandos.

      drozer console connect --server 192.168.19.104

    - Búsqueda de algún paquete instalado con la clave “sri”.

      dz> run app.package.list -f sri

    - Obtener información sobre el archivo manifest del aplicativo SRI móvil.

      dz> run app.package.manifest ec.gob.sri.movil.app > c:\DRZ_EXT\6.5.1.MANIFEST_SRI.txt

    - Realizar una prueba de penetración sobre el aplicativo SRI móvil para obtener un detalle de sus componentes.

      dz> run app.package.attacksurface ec.gob.sri.movil.app > c:\DRZ_EXT\6.5.1.ATTSURFACE_SRI.txt

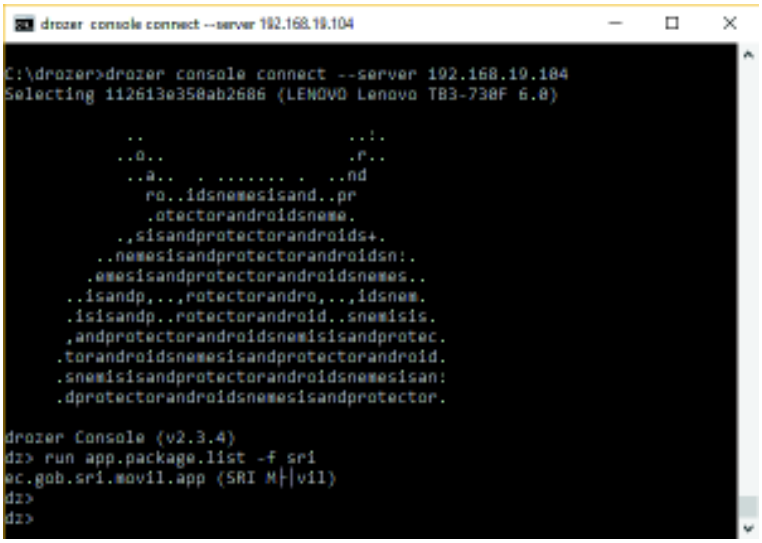

Figura 3.27. Ejecución de comandos para análisis de vulnerabilidades en Drozer

  - Listar los componentes que presentan exposición a algún tipo de vulnerabilidad:

Activities: dz> run app.activity.info -a ec.gob.sri.movil.app > c:\DRZ_EXT\6.5.2.ACTIVITIES_SRI.txt

- Realizar pruebas de seguridad al aplicativo y explotar los componentes que presentan incidentes de seguridad.

  Para el programa SRI Móvil únicamente se presentan incidentes de seguridad en *Activities*:

  - Explotar *Activities*:

    dz> run app.activity.start --component ec.gob.sri.movil.app ec.gob.sri.movil.app.MainActivity

    dz> run app.activity.start --component ec.gob.sri.movil.app com.google.zxing.client.android.encode.EncodeActivity

    dz> run app.activity.start --component ec.gob.sri.movil.app com.google.zxing.client.android.HelpActivity

- Desactivación del agente Drozer en el dispositivo.

En esta etapa, se pudo validar que existen incidentes de seguridad en tres componentes de aplicación del tipo *Activities,* pero en este caso, los componentes no están entregando o procesando ninguna información sensible de los usuarios. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 3.1.

3.3.1 ANÁLISIS DE INFORMACIÓN RECOLECTADA Y AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DEL APLICATIVO SRI MÓVIL - SRI ECUADOR FINANZAS

De acuerdo a la información obtenida en los diferentes procedimientos del ítem 3.2 referentes al análisis de vulnerabilidades presentes en el aplicativo, se procede a la implementación del ítem 2.1.2.3 para la evaluación de la afectación a la privacidad de los usuarios de acuerdo a siguientes las fases.

Fase 1: Hallar los Activos de Información y Amenazas (Th) como se muestra en la Tabla 3.2, y Nivel de Identificación de los Usuarios (IdL) Tabla 3.3.

Fase 2: Establecer los valores de Nivel de Severidad (SeL).

Para el caso de análisis, se considera un nivel de severidad: 4 – Máximo, para todos los activos de información que gestiona el aplicativo, debido a que la aplicación

gestiona información correspondiente a datos fiscales que son extremadamente privados y críticos.

Tabla 3.2. Mapeo de permisos de la Aplicación para establecer Activos de Información y Amenazas

Categoría	Activo de Información	Permiso Android	Amenazas de Privacidad				
			T1	T2	T3	T4	T5
Información Personal Gestionada.	Datos fiscales del usuario.	INTERNET					✔
Sensor / Localización.	Video Audio Localización	CAMERA	✔	✔			
		ACCESS_COARSE_ LOCATION	✔		✔		
		ACCESS_FINE_ LOCATION	✔		✔		
		ACCESS_NETWORK_ STATE	✔		✔		
Almacenamiento externo.	Cualquier tipo de archivos del usuario.	WRITE_EXTERNAL_ STORAGE		✔	✔		✔
		READ_EXTERNAL_ STORAGE		✔	✔		✔

T1. Seguimiento / Vigilancia.

T2. Interceptación / escucha a escondidas.

T3. Perfil del usuario / seguimiento de las actividades del usuario.

T4. *Phishing* / Adquirir credenciales de forma fraudulenta.

T5. Divulgación de Información Personal como documentos, archivo multimedia, etc.

Tabla 3.3. Mapeo de nivel de identificación de usuarios

Nivel de Identificación		Permisos
1	Insignificante	ACCESS_COARSE_LOCATION
		ACCESS_FINE_LOCATION
		ACCESS_NETWORK_STATE
2	Limitado	CAMERA
		READ_EXTERNAL_STORAGE
		WRITE_EXTERNAL_STORAGE
4	Máximo	INTERNET

Fase 3: Análisis de vulnerabilidad en base a los lineamientos de la sección 2.1.1. y ponderación de las vulnerabilidades en relación con la sección 2.1.2.1.6.

A continuación, se muestran los resultados en la Tabla 3.4, donde se encuentran las ponderaciones de vulnerabilidad para los controles definidos en la sección 2.1.1.

Tabla 3.4. Ponderación de vulnerabilidades para la aplicación SRI Móvil

C1. Información recopilada del dispositivo (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Se puede conocer características del sistema operativo o aplicaciones instaladas, sea desde otras aplicaciones o mediante accesos remotos al sistema operativo.
3–Significativa	No existe.
4–Critica	No existe.
C2. Métodos de autenticación y cifrado (4 - Critica)	
1–Insignificante	No existe.
2–Moderada	La aplicación permite ataques de fuerza bruta sobre variables de autenticación.
3–Significativa	Es posible capturar variables de autorización mediante Burpsuite.
4–Critica	Es posible capturar variables de autenticación almacenadas en Logs del sistema operativo.
C3. Información transferida entre dispositivos (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Existe información transferida con algún tipo de cifrado, pero puede ser leída usando *proxys*.
3–Significativa	No existe.
4–Critica	No existe.
C4.1. Análisis del código fuente de la aplicación (4 - Critica)	
1–Insignificante	No existe.
2–Moderada	• Androbugs: Tres incidents [Warning]. • Sandroid: Obtener aplicaciones instaladas en el dispositivo. • Sandroid: Existe permisos no utilizados. • Sandroid: Acceso a cámara
3–Significativa	El código fuente de la aplicación puede ser descompilado y alterado.
4–Critica	• Androbugs: [Critical] Riesgos de seguridad MODE_WORLD_READABLE, WRITEABLE. • Androbugs: [Critical] URL que no utiliza SSL (Total:9).
C4.2. Análisis de vulnerabilidades en los componentes de la aplicación (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	El aplicativo puede acceder o invocar componentes de otras aplicaciones u recursos del dispositivo.
3–Significativa	No existe.
4–Critica	No existe.

La ponderación del nivel de vulnerabilidad se calcula mediante el promedio de la mayor ponderación de cada lineamiento, para la aplicación SRI el valor de VL es 3.

Fase 4: Obtener los valores de Riesgo(R) y Nivel de Riesgo(RL) asociado al uso del aplicativo Android Analizado en base a la sección 2.1.2.1.7, tal como se muestra en la Tabla 3.5.

Tabla 3.5. Nivel de Riesgo asociado a la aplicación SRI

AMENAZA	ACTIVO	IdL(SRI)	SeL(SRI)	Im	ImL(SRI)	VL(SRI)	R	RL(SRI)
T1	S1	2	4	6	3	3	6	3
T1	S3	1	4	5	2	3	5	2
T2	S1	2	4	6	3	3	6	3
T2	A1	2	4	6	3	3	6	3
T3	S3	1	4	5	2	3	5	2
T5	I1	4	4	8	4	3	7	3
T5	A1	2	4	6	3	3	6	3
T3	A1	2	4	6	3	3	6	3
								2.75

De acuerdo con la información obtenida en la Tabla 3.5, se puede observar que para la aplicación SRI Móvil, se obtiene un nivel de riesgo **2.75,** que se categorizaría como **3 –Significativa.**

La información completa relacionada con la evaluación de la afectación a la privacidad de la aplicación SRI Móvil se encuentran en en el Anexo 13.

3.4 IMPLEMENTACIÓN DEL ANÁLISIS DE VULNERABILIDADES SOBRE EL APLICATIVO CNT MÓVIL

CNT Móvil es una aplicación gratuita que sirve para gestionar la suscripción de telefonía prepago o post pago, el cual tiene entre sus principales funcionalidades el consultar la cantidad de datos disponible, revisar saldos de plan y voz promocional, realizar recargas de saldo con modalidad débito bancario o cargo a factura, consultas de valores a cancelar, con acceso desde cualquier lugar del país.

De acuerdo con el procedimiento establecido en la sección 2.1.2.2 relacionado al análisis de vulnerabilidades presentes en las aplicaciones Android, se realiza el análisis de la aplicación CNT Móvil, de acuerdo con las siguientes fases.

Fase 0: Configurar los ambientes donde se desarrollará el análisis de vulnerabilidades, en base al procedimiento mostrado en la sección 3.2, con la finalidad de obtener un ambiente limpio para pruebas, que sirva para realizar el análisis de vulnerabilidades a la aplicación.

Fase 1: Recopilación de información del dispositivo móvil.

A continuación, se procede a recopilar toda la información posible del dispositivo a analizar, para lo cual se aplica el siguiente procedimiento:

- Ejecutar el siguiente comando: nmap -p 1-65535 -T4 -A -v 192.168.19.104 y nmap -sS -sU -T4 -A -v 192.168.19.104, para rastrear puertos presentes en el dispositivo móvil.

Los resultados de este comando se encuentran en el Anexo 10.2.1. y 10.2.2.

- Encender el módulo servidor “Embedded Server” del agente Drozer instalado en el dispositivo.
- Realizar la conexión entre la aplicación Drozer y shell de comandos, con la finalidad de ejecutar los comandos listados a continuación, que sirven para obtener la información básica del dispositivo móvil, y el listado de aplicaciones presentes en el mismo:

  dz> run information.deviceinfo > c:\DRZ_EXT\2.3.1.APK_CNT.txt

  dz> run app.package.list > c:\DRZ_EXT\2.3.2.APK_CNT.txt

  Los resultados de este procedimiento se encuentran en el Anexo 3.2.
- Desactivación del programa Drozer en el dispositivo.

Fase 2: Análisis de vulnerabilidades al dispositivo móvil utilizando la herramienta OpenVAS

A continuación, se procede a ejecutar un análisis de vulnerabilidades presente en el dispositivo móvil, con la finalidad de conocer fallos de seguridad presentes en el OS Android, para lo cual se aplica el siguiente procedimiento:

- Creación de nuevo *Target.* La información relacionada se puede revisar en el Anexo 15.7.
- Creación de nuevo *Task.* La información relacionada se puede revisar en el Anexo 15.8.
- Ejecución de *Task y revisión del reporte.* La información relacionada se puede revisar en el Anexo 15.9.

En esta etapa, el aplicativo OpenVAS no encontró ningún incidente relacionado con vulnerabilidades presentes en el dispositivo móvil.

Fase 3: Instalación del aplicativo CNT Móvil.

A continuación, se procede a realizar la instalación del aplicativo CNT Móvil, tal como se muestra en la Figura 3.28, con la finalidad de realizar posteriormente el respectivo análisis de vulnerabilidades sobre el aplicativo instalado y el dispositivo móvil.

Figura 3.28. Instalación de aplicación CNT Móvil.

Después de instalado el aplicativo, se realiza una nueva ejecución de los procedimientos de la Fase 2, primer procedimiento, y Fase 3, para validar si después de instalada la aplicación existen cambios sobre el OS Android, o algún fallo de seguridad. Los resultados de esta fase se encuentran en el Anexo 10.2.3, y 10.2.4.

Fase 4: Análisis de los permisos del aplicativo instalado CNT Móvil.

A continuación, se procede a obtener los permisos requeridos para el funcionamiento de la aplicación instalada con la finalidad de descubrir los activos de información que pueden ser susceptibles de amenazas, para lo cual se aplica el siguiente procedimiento:

- Encender el módulo servidor “Embedded Server” del agente Drozer.

- Ejecución de comandos Drozer.
  - Conexión al Shell de comandos.
    drozer console connect --server 192.168.19.104
  - Enumerar todos los paquetes instalados en el dispositivo.
    dz> run app.package.list
  - Búsqueda de algún paquete instalado con la clave “cnt”.
    dz> run app.package.list -f cnt
  - Obtener información básica sobre el paquete encontrado CNT.
    run app.package.info -a ec.gob.cnt.apps.plus >
    c:\DRZ_EXT\5.2.INFO_CNT.txt

Los resultados relacionados con esta fase se encuentran en el Anexo 3.2.3.

Fase 5: Valoración de la Vulnerabilidad (V).

A continuación, se procede a analizar las vulnerabilidades presentes en el dispositivo y en la aplicación CNT Móvil, con la finalidad de obtener información que permita conocer la presencia de vulnerabilidades que podrían afectar la privacidad de los usuarios, para lo cual se aplica el siguiente procedimiento de acuerdo a cada uno de los controles establecidos en la sección 2.1.1.

- **C1. Recopilación de información**. La información fue recolectada en las fases 1, 2 y 4 del procedimiento, y servirá posteriormente para realizar la de ponderación de nivel de vulnerabilidad para este control, en base a los lineamientos de la sección 2.1.2.1.6.
- **C2. Métodos de autenticación y cifrado**
  - Ataque tipo MITM utilizando Zanti.
    A continuación, se procede a ejecutar el paquete Zanti para realizar la captura de contraseñas y variables de sesión, donde no se obtuvo capturas de contraseñas ni variables de sesión utilizadas en el programa CNT Móvil, sin embargo, se pudo identificar que el aplicativo permitía ataques de fuerza bruta sobre la autenticación de la aplicación. La información relacionada con este procedimiento se puede revisar en el Anexo 8.
  - Configuración de la conexión proxy en el dispositivo móvil.
    A continuación, se procede a configurar en el dispositivo móvil el uso del proxy BurpSuite configurado en la Fase 0, con la finalidad de interceptar

historial de registros http, peticiones entre dispositivos y tráfico SSL relacionado con autenticación, para lo cual se aplica el siguiente procedimiento:

- Configuración de las direcciones y parámetros del proxy en el dispositivo móvil. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas> Proxy > Configuración Manual.
- Instalación del certificado digital de la herramienta BurpSuite. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.
- Revisión de métodos POST con claves de palabras referentes a autenticación en el historial de registros usando BurpSuite. Los resultados relacionados con esta fase se encuentran en el Anexo 2.2

La información relacionada con este procedimiento se puede revisar en los Anexos 15.10, 15.11 y 2.2.7.

En esta etapa se obtuvieron capturas de variables de autenticación que se transfieren usando protocolo https, pero que pueden ser visualizadas en texto plano mediante el uso de la herramienta BurpSuite.

- Revisión de paquetes de red transferidos entre el dispositivo móvil y servidor relacionado con autenticación, utilizando la herramienta Wireshark.

  En esta etapa se obtuvieron capturas de información transferidas entre el dispositivo y el servidor, pero no se logró capturar variables relacionadas con autenticación. Los resultados relacionados con esta fase se encuentran en el Anexo 6.2.1.

- Revisión del código fuente de la aplicación usando DEX2JAR y JDGUI, donde se puede observar el código de las clases, especialmente aquellas relacionadas con autenticación y cifrado.

  En esta etapa no se obtuvieron errores detectados en código que podrían afectar a los métodos de autenticación, ni tampoco claves quemadas en el

código fuente de la aplicación. Los resultados relacionados con esta fase se encuentran en el Anexo 16.2.

- Captura de registros log mediante Logcat.
  A continuación, se procede a extraer los registros generados por la aplicación CNT Móvil que se almacenan en los archivos log de Android, utilizando la herramienta Logcat.

  En esta etapa no se detectaron variables de autenticación almacenadas en los logs de Android. Los resultados relacionados con esta fase se encuentran en el Anexo 4.2.

➢ **C3. Información transferida entre dispositivos**

- Revisión de métodos GET y POST utilizando BurpSuite.
  A continuación, se procede a revisar las peticiones y respuestas generados por los métodos GET y POST en el historial de registros de la herramienta, donde se detectó que la información sensible gestionada por el aplicativo CNT Móvil se transfiere usando protocolo https, pero puede ser visualizada en texto plano mediante el uso del proxy BurpSuite. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 2.2.

- Captura de paquetes Utilizando Wireshark.
  A continuación, se procede a analizar los paquetes transferidos entre el aplicativo CNT Móvil y el servidor utilizando Wireshark, donde se logró capturar información relacionada con el dispositivo móvil que contiene el aplicativo CNT Móvil, pero no se pudo observar información sensible gestionada por el aplicativo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 6.2.1.

➢ **C4.1. Análisis del código fuente de la aplicación**

- Extracción del archivo apk usando APK Extractor.
  A continuación, se procede a extraer el archivo apk del aplicativo CNT Móvil, para lo cual se utiliza el aplicativo APK Extractor instalado en el mismo dispositivo donde se va a realizar la extracción.

- Análisis estático de la aplicación utilizando AndroBugs.
  A continuación, se procede a realizar un análisis de vulnerabilidades sobre el archivo apk del aplicativo CNT Móvil, utilizando la herramienta AndroBugs, teniendo como resultado un archivo de texto donde se pueden observar las incidencias encontradas durante el análisis. El comando ejecutado en este procedimiento es: C:\AndroBugs_Framework>androbugs.exe -f /APK/CNTMovil_ec.gob.cnt.apps.plus.apk

  En esta etapa, se pudo validar en el resultado generado la presencia de varias incidencias del tipo crítico, en donde una de las más críticas mencionaba el uso de urls sin SSL para transferencia de información usada en el aplicativo, la posibilidad de extraer información la identificación del dispositivo y la toma de backup de la información del equipo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 1.2.

- Análisis de vulnerabilidades del archivo apk utilizando SandDroid.
  A continuación, se procede a realizar la carga del archivo apk del aplicativo CNT Móvil en el portal de la herramienta SandDroid, donde se ejecuta un análisis de vulnerabilidades sobre la aplicación, generando un resultado dentro del portal [47].

  En esta etapa, se pudo validar en la página de resultado, la presencia de varias incidencias, entre las más notables se tiene que es posible obtener información de la red y de la línea telefónica conectada en el dispositivo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 5.1.

➢ **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación**

- Análisis de componentes de la aplicación Android utilizando Drozer.
  A continuación, se procede a realizar la ejecución del análisis de componentes presentes en el aplicativo CNT Móvil, con la finalidad de realizar una prueba de penetración sobre el aplicativo, donde muestre

[47] http://sanddroid.xjtu.edu.cn/report?apk_md5=ADC664BEF0FBABF0FCACE3AA3A53AF1C.

incidentes de seguridad presentes en la aplicación, para lo cual se utiliza el Shell Drozer para ejecutar los siguientes comandos:

- Conexión al Shell de comandos.

drozer console connect --server 192.168.19.104

- Búsqueda de algún paquete instalado con la clave “cnt”.

dz> run app.package.list -f cnt

- Obtener información sobre el archivo manifest del aplicativo CNT Móvil.

dz> run app.package.manifest ec.gob.cnt.apps.plus > c:\DRZ_EXT\6.5.1.MANIFEST_CNT.txt

- Realizar una prueba de penetración sobre el aplicativo CNT Móvil para obtener un detalle de sus componentes.

dz> run app.package.attacksurface ec.gob.cnt.apps.plus > c:\DRZ_EXT\6.5.1.ATTSURFACE_CNT.txt

- Listar los componentes que presentan exposición a algún tipo de vulnerabilidad:
  - Activities: dz> run app.activity.info -a ec.gob.cnt.apps.plus > c:\DRZ_EXT\6.5.2.ACTIVITIES_CNT.txt

- Realizar pruebas de seguridad al aplicativo y explotar los componentes que presentan incidentes de seguridad.

  Para el programa CNT Móvil únicamente se presentan incidentes de seguridad en *Activities*:
  - Explotar *Activities*:

    dz> run app.activity.start --component ec.gob.cnt.apps.plus ec.gob.cnt.apps.plus.MainActivity

- Desactivación del agente Drozer en el dispositivo.

En esta etapa, se pudo validar que existen incidentes de seguridad en un componente de aplicación del tipo *Activities,* pero en este caso, los componentes no están entregando o procesando ninguna información sensible de los usuarios. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 3.2.

3.4.1 ANÁLISIS DE INFORMACIÓN RECOLECTADA Y AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DEL APLICATIVO CNT MÓVIL

De acuerdo a la información obtenida en los diferentes procedimientos del ítem 3.3 referentes al análisis de vulnerabilidades presentes en el aplicativo, se procede a la implementación del ítem 2.1.2.3 para la evaluación de la afectación a la privacidad de los usuarios de acuerdo a las siguientes fases.

Fase 1: Hallar los Activos de Información y Amenazas (Th), tal como se muestra en la Tabla 3.6 y Nivel de Identificación de los Usuarios (IdL), Tabla 3.7.

Tabla 3.6.Mapeo de permisos de la Aplicación para establecer Activos de Información y Amenazas

Categoría	Activo de Información	Permiso Android	Amenazas de Privacidad				
			T1	T2	T3	T4	T5
Envío de Información.	I1. Información de usuarios gestionada por la aplicación.	INTERNET					✔
Sensor/ Localización.	S1. Audio S2. Localización	RECORD_AUDIO	✔	✔			
		ACCESS_COARSE _LOCATION	✔		✔		
		ACCESS_FINE _LOCATION	✔		✔		
		ACCESS_NETWORK _STATE	✔		✔		
Almacenamiento externo.	A1. Archivos del usuario.	WRITE_EXTERNAL _STORAGE		✔	✔		✔
		READ_EXTERNAL _STORAGE		✔	✔		✔
Identidad.	ID1. Identidad del usuario.	READ_PHONE_STATE			✔	✔	✔

T1. Seguimiento / Vigilancia.

T2. Interceptación / escucha a escondidas.

T3. Perfil del usuario / seguimiento de las actividades del usuario.

T4. *Phishing* / Adquirir credenciales de forma fraudulenta.

T5. Divulgación de Información Personal como documentos, archivos multimedia, etc.

Fase 2: Establecer los valores de Nivel de Severidad (SeL).

Para el caso de análisis, se considera un nivel de severidad: 4 – Máximo, para todos los activos de información que gestiona el aplicativo, debido a que la aplicación gestiona información correspondiente a suscripción de telefonía que son de carácter privado.

Tabla 3.7. Mapeo de nivel de identificación de usuarios

Nivel de Identificación		Permisos
1	Insignificante	ACCESS_COARSE_LOCATION
		ACCESS_FINE_LOCATION
		ACCESS_NETWORK_STATE
2	Limitado	READ_EXTERNAL_STORAGE
		RECORD_AUDIO
		WRITE_EXTERNAL_STORAGE
4	Máximo	INTERNET
		READ_PHONE_STATE

Fase 3: Análisis de vulnerabilidad en base a los lineamientos de la sección 2.1.1. y ponderación de las vulnerabilidades en relación con la sección 2.1.2.1.6.

A continuación, se muestra los resultados en la Tabla 3.8, donde se encuentran las ponderaciones de vulnerabilidad para los controles definidos en la sección 2.1.1

Tabla 3.8. Ponderación de vulnerabilidades para la aplicación CNT Móvil – 1 de 2

C1. Información recopilada del dispositivo (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Se puede conocer características del sistema operativo o los aplicativos instalados desde otras aplicaciones o de forma remota.
3–Significativa	No existe.
4–Critica	No existe.
C2. Métodos de autenticación y cifrado (4 - Critica)	
1–Insignificante	No existe.
2–Moderada	• La aplicación permite ataques de fuerza bruta sobre variables de autenticación. • Se puede capturar nombres de usuarios para autenticación en la aplicación.
3–Significativa	Es posible capturar variables de sesión o autorización mediante el uso de métodos proxy.
4–Critica	Es posible capturar variables de autenticación mediante el uso de métodos proxy.

Tabla 3.8 Ponderación de vulnerabilidades para la aplicación CNT Móvil - 2 de 2

C3. Información transferida entre dispositivos (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Existe información transferida con algún tipo de encriptación, pero puede ser leída mediante el uso de proxy.
3–Significativa	No existe.
4–Critica	No existe.
C4.1. Análisis del código fuente de la aplicación (4 - Critica)	
1–Insignificante	No existe.
2–Moderada	Androbugs: [*Notice*] ADB Backup seta habilitado.
3–Significativa	• El código fuente de la aplicación puede ser descompilado y alterado. • Androbugs: [*Warning*] Obtener *information* IMEI y *Device* ID.
4–Critica	• Androbugs: [*Critical*] URLs que no utiliza SSL (Total:9). • Sandroid: Obtener información de la red y línea telefónica conectada en el dispositivo. • Sandroid: Obtener la celda a la cual está conectado el dispositivo.
C4.2. Análisis de vulnerabilidades en los componentes de la aplicación (2 - Moderada):	
1–Insignificante	No existe.
2–Moderada	El aplicativo puede acceder o invocar componentes de otras aplicaciones u recursos del dispositivo.
3–Significativa	No existe.
4–Critica	No existe.

La ponderación del nivel de vulnerabilidad se calcula mediante el promedio de la mayor ponderación de cada lineamiento, para la aplicación CNT Móvil es VL= 3.

Fase 4: Obtener los valores de Riesgo(R) y Nivel de Riesgo(RL) asociado al uso del aplicativo Android Analizado en base a la sección 2.1.2.1.7, tal como se muestra en la Tabla 3.9.

De acuerdo con la información obtenida en la Tabla 3.9, se puede observar que para la aplicación CNT Móvil, se obtiene un nivel de riesgo **3.00,** que se categorizaría como **3 –Significativa.**

La información completa relacionada con la evaluación de la afectación a la privacidad de la aplicación CNT Móvil se encuentra en se encuentra en formato digital en el Anexo 13.

Tabla 3.9. Nivel de Riesgo asociado a la aplicación CNT Móvil

AMENAZA	ACTIVO	IdL(CNT)	SeL(CNT)	Im	ImL(CNT)	VL(CNT)	R	RL(CNT)
T1	S1	2	4	6	3	3	6	3
T1	S2	1	5	6	3	3	6	3
T2	S1	2	6	8	4	3	7	3
T2	A1	2	7	9	4	3	7	3
T3	S2	1	8	9	4	3	7	3
T3	A1	2	9	11	4	3	7	3
T3	ID1	4	10	14	4	3	7	3
T4	ID1	4	11	15	4	3	7	3
T5	A1	2	12	14	4	3	7	3
T5	ID1	4	13	17	4	3	7	3
T5	I1	4	14	18	4	3	7	3
								3.00

3.5 IMPLEMENTACIÓN DEL ANÁLISIS DE VULNERABILIDADES SOBRE EL APLICATIVO ECUADOR SEGURO

Ecuador Seguro es una aplicación para dispositivos móviles que tiene como propósito informar y educar a la población sobre fenómenos naturales y/o de origen humano. que pueden afectar al ciudadano y su entorno, mediante las siguientes funcionalidades: envío de notificaciones a la población frente a cambios de alerta de riesgos decretados por las autoridades, geo-referenciación de zonas de riesgos y puntos seguros para que la población se encuentre siempre informada.

De acuerdo con el procedimiento establecido, se genera el análisis de la aplicación Ecuador Seguro, relacionado con la sección 2.1.2.2 relacionado al análisis de vulnerabilidades presentes en las aplicaciones Android, se realiza el análisis de la aplicación Ecuador Seguro, de acuerdo con las siguientes fases.

Fase 0: Configurar los ambientes donde se desarrollará el análisis de vulnerabilidades, en base al procedimiento mostrado en la sección 3.2,

con la finalidad de obtener un ambiente limpio para pruebas, que sirva para realizar el análisis de vulnerabilidades a la aplicación.

Fase 1: Recopilación de información del dispositivo móvil.

A continuación, se procede a recopilar toda la información posible del dispositivo a analizar, para lo cual se aplica el siguiente procedimiento:

- Ejecución del comando: nmap -p 1-65535 -T4 -A -v 192.168.19.104 y nmap -sS -sU -T4 -A -v 192.168.19.104, para rastrear puertos presentes en el dispositivo móvil.

Los resultados de este comando se encuentran en el Anexo 10.3.1. y 10.3.2.

- Encender el módulo servidor “Embedded Server” del agente Drozer instalado en el dispositivo.
- Conexión a la herramienta Drozer y Shell de comandos, con la finalidad de ejecutar los siguientes comandos, con lo que se obtendrá información básica acerca del dispositivo móvil, y el listado de aplicaciones presentes en el mismo
  dz> run information.deviceinfo > c:\DRZ_EXT\2.3.1.APK_ ECSEG.txt
  dz> run app.package.list > c:\DRZ_EXT\2.3.2.APK_ ECSEG.txt
  Los resultados de este procedimiento se encuentran en el Anexo 3.3.
- Desactivación del programa Drozer en el dispositivo.

Fase 2: Análisis de vulnerabilidades al dispositivo móvil utilizando la herramienta OpenVAS.

A continuación, se procede a ejecutar un análisis de vulnerabilidades presente en el dispositivo móvil, con la finalidad de conocer fallos de seguridad presentes en el OS Android, para lo cual se aplica el siguiente procedimiento:

- Creación de nuevo *Target*. La información relacionada se puede revisar en el Anexo 15.7.
- Creación de nuevo *Task.* La información relacionada se puede revisar en el Anexo 15.8.
- Ejecución de *Task y revisión del reporte.* La información relacionada se puede revisar en el Anexo 15.9.

En esta etapa, el aplicativo OpenVAS no encontró ningún incidente relacionado con vulnerabilidades presentes en el dispositivo móvil.

Fase 3: Instalación del aplicativo Ecuador Seguro.

A continuación, se procede a realizar la instalación del aplicativo Ecuador Seguro, tal como se muestra en la Figura 3.29, con la finalidad de realizar posteriormente el respectivo análisis de vulnerabilidades sobre el aplicativo instalado y el dispositivo móvil.

Figura 3.29. Instalación de aplicación Ecuador Seguro.

Después de instalado el aplicativo, se realiza una nueva ejecución de los procedimientos de la Fase 2, primer procedimiento, y Fase 3, para validar si después de instalada la aplicación existe cambios sobre Android, o algún fallo de seguridad. Los resultados de esta fase se encuentran en el Anexo 10.3.3, y 10.3.4.

Fase 4: Análisis de los permisos del aplicativo instalado Ecuador Seguro.

A continuación, se procede a obtener los permisos requeridos para el funcionamiento de la aplicación instalada con la finalidad de descubrir los activos de información que pueden ser susceptibles de amenazas, para lo cual se aplica el siguiente procedimiento:

- Encender el módulo servidor “Embedded Server” del agente Drozer.
- Ejecución de comandos Drozer.
  - Conexión al Shell de comandos.

    drozer console connect --server 192.168.19.104

- Enumerar todos los paquetes instalados en el dispositivo.
  dz> run app.package.list
- Búsqueda de algún paquete instalado con la clave “Ecuador”.
  dz> run app.package.list -f Ecuador
- Obtener información básica sobre el paquete encontrado Ecuador Seguro.
  dz> run app.package.info -a activities.riesgo.yacare.com.ec.appriesgo > c:\DRZ_EXT\5.2.INFO_ECSEG.txt

Los resultados relacionados con esta fase se encuentran en el Anexo 3.3.3.

Fase 5: Valoración de la Vulnerabilidad (V).

A continuación, se procede a analizar las vulnerabilidades presentes en el dispositivo y en la aplicación Ecuador Seguro, con la finalidad de obtener información que permita conocer la presencia de vulnerabilidades que podrían afectar la privacidad de los usuarios, para lo cual se aplica el siguiente procedimiento de acuerdo a los controles establecidos en la sección 2.1.1:

- **C1. Recopilación de información**. La información fue recolectada en las fases 2, 3 y 5 del procedimiento, y servirá posteriormente para realizar la de ponderación de nivel de vulnerabilidad para este control, en base a los lineamientos de la sección 2.1.2.1.6.
- **C2. Métodos de autenticación y cifrado**
  - Ataque tipo MITM utilizando Zanti.
    A continuación, se procede a ejecutar el paquete Zanti para realizar la captura de contraseñas y variables de sesión, donde no se obtuvo capturas de contraseñas ni variables de sesión utilizadas en el programa Ecuador Seguro, sin embargo, se pudo identificar que el aplicativo permitía ataques de fuerza bruta sobre la autenticación de la aplicación. La información relacionada con este procedimiento se puede revisar en el Anexo 8.
  - Configuración de la conexión proxy en el dispositivo móvil.
    A continuación, se procede a configurar en el dispositivo móvil el uso del proxy BurpSuite, con la finalidad de interceptar historial de registros http, peticiones entre dispositivos y tráfico SSL relacionado con autenticación, para lo cual se aplica el siguiente procedimiento:

- Configuración de las direcciones y parámetros del proxy en el dispositivo móvil. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas> Proxy > Configuración Manual.
- Instalación del certificado digital de la herramienta BurpSuite. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.
- Revisión de métodos POST con claves de palabras referentes a autenticación en el historial de registros usando BurpSuite. Los resultados relacionados con esta fase se encuentran en el Anexo 2.3.

La información relacionada con este procedimiento se puede revisar en los Anexos 15.10, 15.11 y 2.3.8.

En esta etapa se obtuvo capturas de nombres de usuario que se transfieren entre equipos, que pueden ser visualizadas en texto plano mediante el uso de la herramienta BurpSuite.

- Revisión de paquetes de red transferidos entre el dispositivo móvil y servidor relacionado con autenticación, utilizando la herramienta Wireshark.

  En esta etapa se obtuvo capturas de información transferidas entre el dispositivo y el servidor, pero no se logró capturar variables relacionadas con autenticación. Los resultados relacionados con esta fase se encuentran en el Anexo 6.3.1.
- Revisión del código fuente de la aplicación usando DEX2JAR y JDGUI, donde se puede observar el código de las clases, especialmente aquellas relacionadas con autenticación y cifrado.

  En esta etapa no se obtuvo errores detectados en código que podrían afectar a los métodos de autenticación, ni tampoco claves quemadas en el código fuente de la aplicación. Los resultados relacionados con esta fase se encuentran en el Anexo 16.3.

- Captura de registros log mediante Logcat.

A continuación, se procede a extraer los registros generados por la aplicación Ecuador Seguro que se almacenan en los archivos log de Android, utilizando la herramienta Logcat.

En esta etapa no se detectaron variables de autenticación almacenadas en los logs de Android. Los resultados relacionados con esta fase se encuentran en el Anexo 4.2.

- **C3. Información transferida entre dispositivos**
  - Revisión de métodos GET y POST utilizando BurpSuite.
    A continuación, se procede a revisar las peticiones y respuestas http generado por los métodos GET y POST en el historial de registros de la herramienta BurpSuite.

    En esta etapa, se detectó que la información sensible gestionada por el aplicativo Ecuador Seguro se transfiere usando protocolo https, pero puede ser visualizada en texto plano mediante el uso del proxy BurpSuite. Los resultados relacionados con esta fase se encuentran en el Anexo 2.3.

  - Captura de paquetes Utilizando Wireshark.
    A continuación, se procede a analizar los paquetes transferidos entre el aplicativo Ecuador Seguro y el servidor utilizando Wireshark.

    En esta etapa, se logró capturar información relacionada con el dispositivo móvil que contiene el aplicativo Ecuador Seguro, pero no se pudo observar información sensible gestionada por el aplicativo. Los resultados relacionados con esta fase se encuentran en el Anexo 6.3.1.

- **C4.1. Análisis del código fuente de la aplicación**
  - Extracción del archivo apk usando APK Extractor.
    A continuación, se procede a extraer el archivo apk del aplicativo Ecuador Seguro, para lo cual se utiliza el aplicativo APK Extractor instalado en el mismo dispositivo donde se va a realizar la extracción.

  - Análisis estático de la aplicación utilizando AndroBugs.
    A continuación, se procede a realizar un análisis de vulnerabilidades sobre el archivo apk del aplicativo Ecuador Seguro, utilizando la herramienta

AndroBugs, teniendo como resultado un archivo de texto donde se pueden observar las incidencias encontradas durante el análisis.

El comando ejecutado en este procedimiento es: C:\AndroBugs_Framework>androbugs.exe -f /APK/Ecuador Seguro_activities.riesgo.yacare.com.ec.appriesgo.apk

En esta etapa, se pudo validar en el resultado generado la presencia de varias incidencias del tipo crítico, en donde una de las más críticas mencionaba el uso de urls sin SSL para transferencia de información usada en el aplicativo, y permitir el uso de certificados SSL auto firmados, entre otros. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 1.3.

- Análisis de vulnerabilidades del archivo apk utilizando SandDroid.

  A continuación, se procede a realizar la carga del archivo apk del aplicativo Ecuador Seguro en el portal de la herramienta SandDroid, donde se ejecuta un análisis de vulnerabilidades sobre la aplicación, generando un resultado dentro del portal [48].

  En esta etapa, se pudo validar en la página de resultado, la presencia de varias incidencias, entre las más notables se tiene que mediante el aplicativo es posible tomar capturas de imágenes con la cámara. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 5.1.

➢ **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación**

- Análisis de componentes de la aplicación Android utilizando Drozer.

  A continuación, se procede a realizar la ejecución del análisis de componentes presentes en el aplicativo Ecuador Seguro, con la finalidad de realizar una prueba de penetración sobre el aplicativo, donde muestre incidentes de seguridad presentes en la aplicación, para lo cual utilizamos el Shell Drozer para ejecutar los siguientes comandos:

  - Conexión al Shell de comandos.

  drozer console connect --server 192.168.19.104

[48] http://sanddroid.xjtu.edu.cn/report?apk_md5=90F4B4F24C5DB7D2D7C7ECD855275FFD

- Búsqueda de algún paquete instalado con la clave “Ecuador”.

dz> run app.package.list -f Ecuador

- Obtener información sobre el archivo manifest del aplicativo Ecuador Seguro.

dz> run app.package.manifest activities.riesgo.yacare.com.ec.appriesgo > c:\DRZ_EXT\6.5.1.MANIFEST_ECSEG.txt

- Realizar una prueba de penetración sobre el aplicativo Ecuador Seguro para obtener un detalle de sus componentes.

dz> run app.package.attacksurface activities.riesgo.yacare.com.ec. appriesgo > c:\DRZ_EXT\6.5.1.ATTSURFACE_ECSEG.txt

- Listar los componentes que presentan exposición a algún tipo de vulnerabilidad:
  - Activities: dz> run app.activity.info -a activities.riesgo.yacare.com.ec.appriesgo > c:\DRZ_EXT\6.5.2.ACTIVITIES_ECSEG.txt
  - Broadcast *Receiver*: dz> run app.broadcast.info --package activities.riesgo.yacare.com.ec.appriesgo > c:\DRZ_EXT\6.5.2.BREC_ECSEG.txt

- Realizar pruebas de seguridad al aplicativo y explotar los componentes que presentan incidentes de seguridad.

  Para el programa Ecuador Seguro se presentan incidentes de seguridad en *Activities y Broadcast Receiver*:

  - Explotar *Activities*:

    dz> run app.activity.start --component activities.riesgo.yacare.com.ec.appriesgo activities.riesgo.yacare.com.ec.appriesgo.general.SplashActivity

  - *Explotar Broadcast Receiver:*

    dz> run app.broadcast.send –action com.google.android.c2dm.intent.RECEIVE --component activities.riesgo.yacare.com.ec.appriesgo com.microsoft.windowsazure.notifications.NotificationsBroadcastReceiver --extra string message "ENVIO BROADCAST MESSAGE"

dz> run app.broadcast.send --action com.google.android.c2dm.intent.RECEIVE --component activities.riesgo.yacare.com.ec.appriesgo com.google.android.gms.gcm.GcmReceiver --extra string from epn.app.andro@gmail.com

- Desactivación del agente Drozer en el dispositivo.

3.5.1 ANÁLISIS DE INFORMACIÓN RECOLECTADA Y AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DEL APLICATIVO ECUADOR SEGURO

De acuerdo a la información obtenida en los diferentes procedimientos del ítem 0 referentes al análisis de vulnerabilidades presentes en el aplicativo, se procede a realizar la implementación del ítem 2.1.2.3 para la evaluación de la afectación a la privacidad de los usuarios de acuerdo a los siguientes ítems:

Fase 1: Hallar los Activos de Información y Amenazas (Th), tal como se muestra en la Tabla 3.10 y Nivel de Identificación de los Usuarios (IdL), Tabla 3.11.

Tabla 3.10. Mapeo de permisos de la Aplicación para establecer Activos de Información y Amenazas

Categoría.	Activo de Información	Permiso Android.	Amenazas de Privacidad.				
			T1	T2	T3	T4	T5
Envío de Información.	I1. Información del usuario gestionada por la aplicación.	INTERNET					✔
		BLUETOOTH					✔
Sensor/ Localización.	S1. Localización. S2. Información de configuraciones bluetooth	ACCESS_COARSE_ LOCATION	✔		✔		
		ACCESS_FINE_ LOCATION	✔		✔		
		BLUETOOTH_ ADMIN	✔		✔		
		ACCESS_NETWORK_ STATE	✔		✔		
Almacenamiento externo.	A1. Archivos del usuario.	WRITE_EXTERNAL_ STORAGE		✔	✔		✔
		READ_EXTERNAL_ STORAGE		✔	✔		✔
Identidad.	ID1. Cuentas del Usuario.	GET_ACCOUNTS			✔	✔	✔
Credenciales.	C1. Credenciales de autenticación.	USE_CREDENTIALS				✔	

T1. Seguimiento / Vigilancia.

T2. Interceptación / escucha a escondidas.

T3. Perfil del usuario / seguimiento de las actividades del usuario.

T4. *Phishing* / Adquirir credenciales de forma fraudulenta.

T5. Divulgación de Información Personal como documentos, archivos multimedia, etc.

Tabla 3.11. Mapeo de nivel de identificación de usuarios

Nivel de Identificación		Permisos
1	Insignificante	ACCESS_COARSE_LOCATION
		ACCESS_FINE_LOCATION
		BLUETOOTH_ADMIN
		ACCESS_NETWORK_STATE
2	Limitado	READ_EXTERNAL_STORAGE
		WRITE_EXTERNAL_STORAGE
3	Significativo	GET_ACCOUNTS
		USE_CREDENTIALS
4	Máximo	INTERNET
		BLUETOOTH

Fase 2: Establecer los valores de Nivel de Severidad (SeL).

Para el caso de análisis, se considera un nivel de severidad: 4 – Máximo, para todos los activos de información que gestiona el aplicativo, debido a que se trata de información crítica de los usuarios.

Fase 3: Análisis de vulnerabilidad en base a los lineamientos de la sección 2.1.1. y ponderación de las vulnerabilidades en relación con la sección 2.1.2.1.6.

A continuación, se muestra los resultados en la Tabla 3.12, donde se encuentran las ponderaciones de vulnerabilidad para los controles definidos en la sección 2.1.1.

Tabla 3.12. Ponderación de vulnerabilidades para la aplicación Ecuador Seguro 1 de 3

C1. Información recopilada del dispositivo (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Se puede conocer características del sistema operativo o los aplicativos instalados desde otras aplicaciones o de forma remota.
3–Significativa	No existe.
4–Critica	No existe.

Tabla 3.12 Ponderación de vulnerabilidades para la aplicación Ecuador Seguro – 2 de 3

C2. Métodos de autenticación y cifrado (2 - Moderada)	
1–Insignificante	No existe.
2–Moderada	Se puede capturar nombres de usuarios para autenticación en la aplicación.
3–Significativa	No existe.
4–Critica	No existe.
C3. Información transferida entre dispositivos (3 - Significativa)	
1–Insignificante	No existe.
2–Moderada	No existe.
3–Significativa	Existe información transferida sin ninguna encriptación de canal o datos.
4–Critica	No existe.
C4.1. Análisis del código fuente de la aplicación (4 - Critica)	
1–Insignificante	No existe.
2–Moderada	Sanddroid**:** Encriptación o Decriptación de información a los que tiene acceso.
3–Significativa	• El código fuente de la aplicación puede ser descompilado y alterado. • Androbugs: [*Warning*] carga dinámica de código (DexClassLoader). • *Sandroid*: Permite toma de capturas de imágenes con la cámara.
4–Critica	• Androbugs: [*Critical*] Riesgos de seguridad "MODE_WORLD_READABLE", "MODE_WORLD_WRITEABLE". • Androbugs: [*Critical*] Permite que verificadores de nombres de host autodefinidos acepten todos los nombres comunes (CN), lo cual puede permitir ataques MiTM. • Androbugs: [*Critical*] URLs que no utiliza SSL (Total:14). • Androbugs: [*Critical*] la aplicación NO comprueba la validación de certificados SSL, permite certificados auto firmados. • Androbugs: [*Critical*] vulnerabilidad en WebView "addJavascriptInterface".

Tabla 3.12 Ponderación de vulnerabilidades para la aplicación Ecuador Seguro – 3 de 3

C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación (3 - Significativa)	
1–Insignificante	No existe.
2–Moderada	El aplicativo puede acceder o invocar componentes de otras aplicaciones u recursos del dispositivo.
3–Significativa	Existen componentes expuestos que, implementando alguna restricción, pero pueden ser vulnerados de alguna forma (Broadcast Receivers).
4–Critica	No existe.

La ponderación del nivel de vulnerabilidad se calcula mediante el promedio de la mayor ponderación de cada lineamiento, para la aplicación EC Seguro es VL= 3.

Fase 4: Obtener los valores de Riesgo(R) y Nivel de Riesgo(RL) asociado al uso del aplicativo Android Analizado en base a la sección 2.1.2.1.7, tal como se muestra en la Tabla 3.13.

Tabla 3.13. Nivel de Riesgo asociado a la aplicación Ecuador Seguro

AMENAZAS	ACTIVO	IdL(ES)	SeL(ES)	Im	ImL(ES)	VL(ES)	R	RL(ES)
T1	S1	1	4	5	2	3	5	2
T1	S2	1	4	5	2	3	5	2
T2	A1	2	4	6	3	3	6	3
T3	S1	1	4	5	2	3	5	2
T3	S2	1	4	5	2	3	5	2
T3	A1	2	4	6	3	3	6	3
T3	ID1	3	4	7	4	3	7	3
T4	ID1	3	4	7	4	3	7	3
T4	C1	3	4	7	4	3	7	3
T5	I1	4	4	8	4	3	7	3
T5	A1	2	4	6	3	3	6	3
T5	ID1	3	4	7	4	3	7	3
								2.67

De acuerdo con la información obtenida en la Tabla 3.13, se puede observar que para la aplicación Ecuador Seguro, se obtiene un nivel de riesgo **2.67,** que se categorizaría como **3 –Significativa.**

La información completa relacionada con la evaluación de la afectación a la privacidad de la aplicación Ecuador Seguro se encuentra en formato digital en el Anexo 13.

3.6 IMPLEMENTACIÓN DEL ANÁLISIS DE VULNERABILIDADES SOBRE EL APLICATIVO ECU911

La aplicación móvil ECU 911 permite reportar emergencias en línea desde un dispositivo móvil, para lo cual es necesario realizar el ingreso de información personal del usuario y aceptar los términos y condiciones de uso, los mismos que son de carácter sensible. Es recomendado el uso responsable de la aplicación debido a que la atención de emergencias es un derecho ciudadano, y el reportar emergencias reales puede salvar vidas.

Toda información solicitada en la aplicación será utilizada exclusivamente con el propósito para el cual fue requerido, y deberá ser verídica, a fin de brindar una asistencia oportuna.

Para reportar una emergencia se debe seguir los siguientes pasos:
Paso 1.- Escoger la Institución para atender la emergencia.
Paso 2.- Escoger el incidente que describa la emergencia.
Paso 3.- Confirmar el envío de la emergencia.

De acuerdo con el procedimiento establecido en la sección 2.1.2.2 relacionado al análisis de vulnerabilidades presentes en las aplicaciones Android, se realiza el análisis de la aplicación ECU 911, de acuerdo con las siguientes fases:

Fase 0: Configurar los ambientes donde se desarrollará el análisis de vulnerabilidades, en base al procedimiento mostrado en la sección 3.2, con la finalidad de obtener un ambiente limpio para pruebas, que sirva para realizar el análisis de vulnerabilidades a la aplicación.

Fase 1: Recopilación de información del dispositivo móvil.

A continuación, se procede a recopilar toda la información posible del dispositivo a analizar, para lo cual se aplica el siguiente procedimiento:

- Ejecución del comando: nmap -p 1-65535 -T4 -A -v 192.168.19.104 y nmap -sS -sU -T4 -A -v 192.168.19.104, para rastrear puertos presentes en el dispositivo móvil. Los resultados de este comando se encuentran en el Anexo 10.4.1. y 10.4.2.
- Encender el módulo servidor "Embedded Server" del agente Drozer instalado en el dispositivo.
- Conexión a la herramienta Drozer y Shell de comandos, con la finalidad de ejecutar los siguientes comandos, con lo que se obtendrá información básica acerca del dispositivo móvil, y el listado de aplicaciones presentes en el mismo.

  dz> run information.deviceinfo > c:\DRZ_EXT\2.3.1.APK_ECU911.txt

  dz> run app.package.list > c:\DRZ_EXT\2.3.2.APK_ ECU911.txt

Los resultados de este procedimiento se encuentran en el Anexo 3.4.

- Desactivación del programa Drozer en el dispositivo.

Fase 2: Análisis de vulnerabilidades al dispositivo móvil utilizando la herramienta OpenVAS.

A continuación, se procede a ejecutar un análisis de vulnerabilidades presente en el dispositivo móvil, con la finalidad de conocer fallos de seguridad que pueda tener Android, para lo cual se aplica el siguiente procedimiento:

- Creación de nuevo *Target*. La información relacionada se puede revisar en el Anexo 15.7.
- Creación de nuevo *Task.* La información relacionada se puede revisar en el Anexo 15.8.
- Ejecución de *Task y revisión del reporte.* La información relacionada se puede revisar en el Anexo 15.9.

En esta etapa, el aplicativo OpenVAS no encontró ningún incidente relacionado con vulnerabilidades presentes en el dispositivo móvil.

Fase 3: Instalación del aplicativo ECU911.

Luego, se procede a realizar la instalación del aplicativo ECU 911, tal como se muestra en la Figura 3.30, con la finalidad de realizar posteriormente el respectivo análisis de vulnerabilidades sobre el aplicativo instalado y el dispositivo móvil.

Figura 3.30. Instalación de aplicación ECU911.

Después de instalado el aplicativo, se realiza una nueva ejecución de los procedimientos de la Fase 2, primer procedimiento, y Fase 3, para validar si después de instalada la aplicación existe cambios sobre Android, o algún fallo de seguridad. Los resultados de esta fase se encuentran en el Anexo 10.4.3, y 10.4.4.

Fase 5: Análisis de los permisos del aplicativo instalado ECU 911.

A continuación, se procede a obtener los permisos requeridos para el funcionamiento de la aplicación instalada con la finalidad de descubrir los activos de información que pueden ser susceptibles de amenazas, para lo cual se aplica el siguiente procedimiento:

- Encender el módulo servidor “Embedded Server” del agente Drozer.
- Ejecución de comandos Drozer.
  - Conexión al Shell de comandos.
    drozer console connect --server 192.168.19.104
  - Enumerar todos los paquetes instalados en el dispositivo.
    dz> run app.package.list
  - Búsqueda de algún paquete instalado con la clave “911”.
    dz> run app.package.list -f 911
  - Obtener información básica sobre el paquete encontrado Ecuador Seguro.
    dz> run app.package.info -a com.walpana.ecu911 > c:\DRZ_EXT\5.2.INFO_ECU911.txt

Los resultados relacionados con esta fase se encuentran en el Anexo 3.4.3.

Fase 5: Valoración de la Vulnerabilidad (V).

A continuación, se procede a analizar las vulnerabilidades presentes en el dispositivo y en la aplicación ECU 911, con la finalidad de obtener información que permita conocer la presencia de vulnerabilidades que podrían afectar la privacidad de los usuarios, para lo cual se aplica el siguiente procedimiento de acuerdo a cada uno de los controles establecidos en la sección 2.1.1:

- **C1. Recopilación de información**. La información fue recolectada en las fases 1, 2 y 4 del procedimiento, y servirá posteriormente para realizar la de ponderación de nivel de vulnerabilidad para este control, en base a los lineamientos de la sección 2.1.2.1.6.
- **C2. Métodos de autenticación y cifrado**
  - Ataque tipo MITM utilizando Zanti.
    A continuación, se procede a ejecutar el paquete Zanti para realizar la captura de contraseñas y variables de sesión, donde no se obtuvo capturas de contraseñas ni variables de sesión utilizadas en el programa ECU 911. La información relacionada con este procedimiento se puede revisar en el Anexo 8.
  - Configuración de la conexión proxy en el dispositivo móvil.
    A continuación, se procede a configurar en el dispositivo móvil el uso del proxy BurpSuite, con la finalidad de interceptar historial de registros http, peticiones entre dispositivos y tráfico SSL relacionado con autenticación, para lo cual se aplica el siguiente procedimiento:
    - Configuración de las direcciones y parámetros del proxy en el dispositivo móvil. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > [RED WIFI] > Modificar configuración de red > Mostrar Opciones Avanzadas> Proxy > Configuración Manual.
    - Instalación del certificado digital de la herramienta BurpSuite. Para la configuración de este procedimiento se debe seguir la siguiente ruta: Dispositivo > Configuración > Wifi > Avanzados >Instalar Certificados.

- Revisión de métodos POST con claves de palabras referentes a autenticación en el historial de registros usando BurpSuite. Los resultados relacionados con esta fase se encuentran en el Anexo 2.4.

La información relacionada con este procedimiento se puede revisar en los Anexos 15.10, 15.11 y 2.4.6.

En esta etapa se obtuvieron capturas de nombres de usuario que se transfieren entre equipos, que pueden ser visualizadas en texto plano mediante el uso de la herramienta BurpSuite.

- Revisión de paquetes de red transferidos entre el dispositivo móvil y servidor relacionado con autenticación, utilizando la herramienta Wireshark.

  En esta etapa se obtuvieron capturas de información transferidas entre el dispositivo y el servidor, pero no se logró capturar variables relacionadas con autenticación. Los resultados relacionados con esta fase se encuentran en el Anexo 6.4.1.

- Revisión del código fuente de la aplicación usando DEX2JAR y JDGUI, donde se puede observar el código de las clases, especialmente aquellas relacionadas con autenticación y cifrado.

  En esta etapa no se obtuvieron errores detectados en código que podrían afectar a los métodos de autenticación, ni tampoco claves quemadas en el código fuente de la aplicación. Los resultados relacionados con esta fase se encuentran en el Anexo 16.3.

- Captura de registros log mediante Logcat.
  A continuación, se procede a extraer los registros generados por la aplicación ECU 911 que se almacenan en los archivos log de Android, utilizando la herramienta Logcat.

  En esta etapa no se detectaron variables de autenticación almacenadas en los logs de Android. Los resultados relacionados con esta fase se encuentran en el Anexo 4.2.

➢ **C3. Información transferida entre dispositivos**

- Revisión de métodos GET y POST utilizando BurpSuite.

Luego, se procede a revisar las peticiones y respuestas http generado por los métodos GET y POST en el historial de registros de la herramienta, donde se detectó que la información sensible gestionada por el aplicativo ECU 911, se transfiere usando protocolo https, pero puede ser visualizada en texto plano mediante el uso del proxy BurpSuite. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 2.4.

- Captura de paquetes Utilizando Wireshark.

  A continuación, se procede a analizar los paquetes transferidos entre el aplicativo ECU 911 y el servidor utilizando Wireshark, donde se logró capturar información relacionada con el dispositivo móvil que contiene el aplicativo ECU 911, pero no se pudo observar información sensible gestionada por el aplicativo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 6.4.1.

➢ **C4.1. Análisis del código fuente de la aplicación**

- Extracción del archivo apk usando APK Extractor.

  A continuación, se procede a extraer el archivo apk del aplicativo ECU 911, para lo cual se utiliza el aplicativo APK Extractor instalado en el mismo dispositivo donde se va a realizar la extracción.

- Análisis estático de la aplicación utilizando AndroBugs.

  A continuación, se procede a realizar un análisis de vulnerabilidades sobre el archivo apk del aplicativo ECU 911, utilizando la herramienta AndroBugs, teniendo como resultado un archivo de texto donde se pueden observar las incidencias encontradas durante el análisis. El comando ejecutado en este procedimiento es: C:\AndroBugs_Framework>androbugs.exe -f /APK/Ecuador ECU911_com.walpana.ecu911.apk

  En esta etapa, se pudo validar en el resultado generado la presencia de varias incidencias del tipo crítico, en donde una de las más críticas mencionaba el uso de urls sin SSL para transferencia de información usada en el aplicativo. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 1.4.

- Análisis de vulnerabilidades del archivo apk utilizando SandDroid.

A continuación, se procede a realizar la carga del archivo apk del aplicativo ECU 911 en el portal de la herramienta SandDroid, donde se ejecuta un análisis de vulnerabilidades sobre la aplicación, generando un resultado dentro del portal [49].

En esta etapa, se pudo validar en la página de resultado, que no existía la presencia de incidencias graves. Los resultados relacionados con la ejecución de esta fase se encuentran en el Anexo 5.1.

- **C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación**

  - Análisis de componentes de la aplicación Android utilizando Drozer.

    A continuación, se procede a realizar la ejecución del análisis de componentes presentes en el aplicativo ECU 911, con la finalidad de realizar una prueba de penetración sobre el aplicativo, donde muestre incidentes de seguridad presentes en la aplicación, para lo cual utilizamos el Shell Drozer para ejecutar los siguientes comandos:

    - Conexión al Shell de comandos.

    drozer console connect --server 192.168.19.104

    - Búsqueda de algún paquete instalado con la clave “911”.

      dz> run app.package.list -f 911

    - Obtener información sobre el archivo manifest del aplicativo ECU 911.

      dz> run app.package.manifest com.walpana.ecu911 > c:\DRZ_EXT\6.5.1.MANIFEST_ECU911.txt

    - Realizar una prueba de penetración sobre el aplicativo ECU 911 para obtener un detalle de sus componentes.

      dz> run app.package.attacksurface com.walpana.ecu911 > c:\DRZ_EXT\6.5.1.ATTSURFACE_ECU911.txt

  - Listar los componentes que presentan exposición a algún tipo de vulnerabilidad:

    - Activities: dz> run app.activity.info -a com.walpana.ecu911 > c:\DRZ_EXT\6.5.2.ACTIVITIES_ECU911.txt

[49] http://sanddroid.xjtu.edu.cn/report?apk_md5=E4247522BA50AEDFCD69A7465B317801

- Realizar pruebas de seguridad al aplicativo y explotar los componentes que presentan incidentes de seguridad.

  Para el programa ECU 911 se presentan incidentes de seguridad únicamente en *Activities.*

  - Explotar *Activities*:

  dz> run app.activity.start --component com.walpana.ecu911
  com.walpana.ecu911.ListCategoryActivity

- Desactivación del agente Drozer en el dispositivo.

3.6.1 ANÁLISIS DE INFORMACIÓN RECOLECTADA Y AFECTACIÓN A LA PRIVACIDAD DE LOS USUARIOS DEL APLICATIVO ECU 911 MÓVIL

De acuerdo a la información obtenida en los diferentes procedimientos del ítem 3.6 referentes al análisis de vulnerabilidades presentes en el aplicativo, se procede a la implementación del ítem 2.1.2.3 para la evaluación de la afectación a la privacidad de los usuarios de acuerdo a las siguientes fases.

Fase 1: Hallar los Activos de Información y Amenazas (Th), tal como se muestra en la Tabla 3.14 y Nivel de Identificación de los Usuarios (IdL), Tabla 3.15.

Tabla 3.14. Mapeo de permisos de la Aplicación para establecer Activos de Información y Amenazas

Categoría	Tipo de activo de información	Permiso Android	Amenazas de privacidad				
			T1	T2	T3	T4	T5
Envío de Información.	I1. Información del usuario gestionada por la aplicación.	INTERNET					✔
Sensor/ Localización	S1. Localización	ACCESS_FINE_ LOCATION	✔		✔		
Identidad.	ID1. Información del dispositivo y llamadas	READ_PHONE_ STATE			✔	✔	✔

T1. Seguimiento / Vigilancia.

T2. Interceptación / escucha a escondidas.

T3. Perfil del usuario / seguimiento de las actividades del usuario.

T4. *Phishing* / Adquirir credenciales de forma fraudulenta.

T5. Divulgación de Información Personal como documentos, archivos multimedia, etc.

Tabla 3.15. Mapeo de nivel de identificación de usuarios

Nivel de identificación		Permisos
1	Insignificante	ACCESS_FINE_LOCATION
4	Máximo	READ_PHONE_STATE
		INTERNET

Fase 2: Establecer los valores de Nivel de Severidad (SeL).

Para el caso de análisis, se considera un nivel de severidad: 4 – Máximo, para todos los activos de información que gestiona el aplicativo, debido a que se trata de información crítica de los usuarios.

Fase 3: Análisis de vulnerabilidad en base a los lineamientos de la sección 2.1.1. y ponderación de las vulnerabilidades en relación con la sección 2.1.2.1.6.

A continuación, se muestra los resultados en la Tabla 3.16, donde se encuentran las ponderaciones de vulnerabilidad para los controles definidos en la sección 2.1.1.

Tabla 3.16. Ponderación de vulnerabilidades para la aplicación ECU 911 – 1 de 2

C1. Información recopilada del dispositivo (2 - Moderada):	
1–Insignificante	No existe.
2–Moderada	Se puede conocer características del sistema operativo o los aplicativos instalados desde otras aplicaciones o de forma remota.
3–Significativa	No existe.
4–Critica	No existe.
C2. Métodos de autenticación y cifrado (2 - Moderada):	
1–Insignificante	No existe.
2–Moderada	Se puede capturar nombres de usuarios para autenticación en la aplicación.
3–Significativa	No existe.
4–Critica	No existe.
C3. Información transferida entre dispositivos (3 - Significativa):	
1–Insignificante	No existe.
2–Moderada	Existe información transferida con algún tipo de encriptación, pero puede ser leída por cualquier método.
3–Significativa	Existe información transferida sin ningún tipo de encriptación de canal o de datos.
4–Critica	No existe.

Tabla 3.16. Ponderación de vulnerabilidades para la aplicación ECU 911 – 2 de 2

C4.1. Análisis del código fuente de la aplicación (4 - Critica):	
1–Insignificante	No existe.
2–Moderada	No existe.
3–Significativa	• El código fuente de la aplicación puede ser descompilado y alterado. • Androbugs: [Warning] la aplicación tiene acceso a los códigos device ID, IMEI de GSM y MEID de ESN o ESN de dispositivos CDMA.
4–Critica	• Androbugs: [Critical] URLs que no utiliza SSL (Total:9).
C4.2. Análisis de las vulnerabilidades en los componentes de la aplicación (2 - Moderada):	
1–Insignificante	No existe.
2–Moderada	El aplicativo puede acceder o invocar componentes de otras aplicaciones u recursos del dispositivo.
3–Significativa	No existe.
4–Critica	No existe.

La ponderación del nivel de vulnerabilidad se calcula mediante el promedio de la mayor ponderación establecida para cada lineamiento, para la aplicación ECU911 es VL= 3.

Fase 4: Obtener los valores de Riesgo(R) y Nivel de Riesgo(RL) asociado al uso del aplicativo Android Analizado en base a la sección 2.1.2.1.7, tal como se muestra en la Tabla 3.17.

Tabla 3.17. Nivel de Riesgo asociado a la aplicación ECU911

AMENAZAS	ACTIVO	IdL(911)	SeL(911)	Im	ImL(911)	VL(911)	R	RL(911)
T1	S1	1	4	5	2	3	5	2
T3	S1	1	4	5	2	3	5	2
T3	ID1	4	4	8	4	3	7	3
T4	ID1	4	4	8	4	3	7	3
T5	I1	4	4	8	4	3	7	3
T5	ID1	4	4	8	4	3	7	3
								2.67

De acuerdo con la información obtenida en la Tabla 3.17, se puede observar que para la aplicación ECU 911, se obtiene un nivel de riesgo **2.67,** que se categorizaría como **3 –Significativa.**

La información completa relacionada con la evaluación de la afectación a la privacidad de la aplicación ECU 911 se encuentra en formato digital en el Anexo 13.

CAPÍTULO 4
CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

Luego de analizar las metodologías relacionadas con la evaluación de seguridad OWASP y OASAM, se observó que en estas metodologías hay ciertas similitudes que sirven principalmente para realizar un control efectivo acerca de los desarrollos y lineamientos necesarios para el diseño y codificación de aplicaciones menos riesgosas. Sin embargo, no hacen gran énfasis sobre la clasificación de información privada de los usuarios que gestionan estas aplicaciones, y pueden ser accedidos por entes no autorizados e inclusive la misma aplicación utilizada para fines que no guardan relación con la lógica de negocio para la cual fue diseñada. Inclusive, en ciertos casos sin que los usuarios tengan conocimiento de esto. Por lo cual, fue necesario revisar también aspectos relacionados al análisis del riesgo que puede existir al usar estos aplicativos.

Mediante la revisión de las metodologías de análisis de riesgo CNIL [29] y la evaluación de los riesgos de privacidad en Android [30], se logró realizar un análisis más específico relacionado al riesgo que representa gestionar información privada de los usuarios desde aplicaciones móviles, logrando identificar aquellos activos de información propios del usuario que pueden servir para identificar a los individuos o sus preferencias, y que podrían estar siendo mal utilizadas por los dueños de las aplicaciones o por terceros.

Con la información de metodologías de vulnerabilidades y de riesgos se pudo establecer un procedimiento para la prueba de seguridad de aplicaciones móviles Android mediante una estructura que toma en cuenta la información gestionada del usuario y las vulnerabilidades presentes en el dispositivo y aplicación. Lo cual sirvió para ponderar el riesgo asociado al uso de una aplicación, en este caso aquellas que son públicas y distribuidas por el gobierno del Ecuador.

En relación con el uso de herramientas computacionales, se seleccionaron aquellas que generaron una gran cantidad de información y que lo entregaban de alguna

forma fácil y ágil de analizar para el respectivo estudio. A pesar de que la mayoría de las herramientas funcionaron sin mayor problema, existieron algunas que tenían intermitencias o fallos recurrentes, que a pesar de que sus desarrolladores podían brindar soporte, este era de difícil acceso, y por este motivo algunas de estas no fueron seleccionadas para implementarse en la guía de análisis de la afectación de privacidad de los usuarios.

Después de haber realizado el análisis de vulnerabilidades encontradas en Android y sus aplicaciones, se puede concluir que la mayoría o la totalidad de las vulnerabilidades criticas encontradas, se presentan por el desarrollo incorrecto de aplicaciones Android, debido a que de todos los incidentes encontrados ninguno guardaba relación directa con fallos de seguridad o de funcionamiento del sistema operativo Android, más bien estaban alineados al uso de librerías de terceros riesgosas, transferencia de información que puede ser visualizada en texto plano, incluyendo información confidencial, inclusive en ciertos casos variables de autenticación, como sucedió en los registrados en el archivo log generado al realizar el análisis de la aplicación SRI Móvil.

Después de realizar el respectivo análisis de vulnerabilidad a las aplicaciones públicas distribuidas por el estado del Ecuador, el mayor tipo de vulnerabilidades encontradas están relacionadas con la captura de variables de autenticación como usuarios y claves, captura de variables de sesión, trasferencia de datos en texto plano que se pueden capturar con algún método de intrusión tipo MiTM y requerimiento de permisos innecesarios por parte del aplicativo estatal.

Con base en todos los análisis realizados se puede concluir que uno de los problemas de codificación más relevantes es la no ofuscación del código de todas las aplicaciones analizadas, debido a que son vulnerables a ingeniería inversa que podría ser capaz de recompilar la aplicación inyectando código malicioso indetectable y poder distribuir la aplicación haciéndose pasar como la original.

De acuerdo con la evaluación de seguridad asociada a las aplicaciones, se refleja que la aplicación que representa un mayor nivel de riesgo para la privacidad de los usuarios es CNT Móvil.

4.2 RECOMENDACIONES

Al momento de desarrollar aplicaciones Android, es recomendado seguir los lineamientos que nos recomienda las diferentes metodologías que existen acerca de pruebas de seguridad, tal como las revisadas en el presente documento en la sección 1.4, inclusive con prioridad en aquellas proporcionadas por Google, con la finalidad de tener aplicativos cuyo nivel de vulnerabilidad sea menor.

Es necesario desarrollar aplicaciones que eviten el uso innecesario de componentes. Además, no exponer alguno especialmente si no se tiene algún método de autenticación ni autorización para ser invocado por el sistema operativo o por otra aplicación presente en el dispositivo móvil.

Para aquellos aplicativos que utilizan algún tipo de autenticación para su uso, se recomienda que estas implementen algún tipo de método, en el cual, si se capturan paquetes y se los descifra por algún agente ajeno al proceso, este no pueda descubrir con facilidad los nombres de usuario y contraseña de usuarios.

En la transferencia de datos, es necesario contar con algún tipo de encriptación de la data que se transfiere por el canal de comunicaciones, debido a que en ocasiones el cifrar el canal de transmisión no es suficiente debido a que si el usuario utiliza alguna comunicación que implementa algún tipo de proxy, la información que viaja puede ser totalmente descubierta y la data personal de los usuarios podría ser fácilmente leída.

En el momento del desarrollo es fundamental el ofuscamiento del código, ya que con esto se limita a que algún atacante pueda aplicar ingeniería inversa sobre los archivos APK de los aplicativos con la finalidad de inyectar código malicioso. Además, implementando este procedimiento se puede implementar procesos que permitan cifrar, codificar o restringir el acceso a la información que se transfiere entre dispositivos y servidores, y disminuir el nivel de vulnerabilidad de la aplicación.

Se recomienda a los usuarios de Android no instalar bajo ningún concepto en sus dispositivos aplicaciones que no estén publicadas en sitios seguros como Google

Play, debido a que al instalar versiones desconocidas es muy probable que su información personal y privada va a ser accedida y filtrada, tal como sucede cuando el canal de acceso a Internet se realiza mediante algún tipo de proxy.

REFERENCIAS BIBLIOGRÁFICAS

[1] «Open Handset Alliance». [En línea]. Disponible en: http://www.openhandsetalliance.com/. [Accedido: 21-ene-2017].

[2] «Industry Leaders Announce Open Platform for Mobile Devices | Open Handset Alliance». [En línea]. Disponible en: http://www.openhandsetalliance.com/press_110507.html. [Accedido: 21-ene-2017].

[3] Statista, «Smartphone OS sales share worldwide 2011-2016 | Statistic», *Statista*. [En línea]. Disponible en: https://www.statista.com/statistics/236035/market-share-of-global-smartphone-os-shipments-by-mobile-operating-system-per-quarter/. [Accedido: 21-ene-2017].

[4] Christian González León, «Análisis y explotación de vulnerabilidades en Android.», *Instituto Politécnico Nacional*, nov-2015. [En línea]. Disponible en: http://premios.eset-la.com/universitario/pdf/analisis_y_explotacion_de_vulnerabilidades_en_android.pdf. [Accedido: 28-sep-2016].

[5] R. Calero Asencios, «Modelo de Seguridad para mitigar los problemas derivados de las vulnerabilidades en dispositivos móviles Android con respecto a los principios de Integridad, Confidencialidad y Disponibilidad», *Pontificia Universidad Javeriana*, 2013. [En línea]. Disponible en: http://pegasus.javeriana.edu.co/~CIS1310SD03/Articulo.pdf. [Accedido: 21-sep-2016].

[6] H. N. Security, «A look into the current state of mobile security», *Help Net Security*, 12-feb-2016. .

[7] I. H. Morrison, «¿Cómo Google controla y evita malware en tu smartphone? |». .

[8] A. N. Gonzalez, «¿Qué es Android?», *Xataka Android*, 08-feb-2011. [En línea]. Disponible en: https://www.xatakandroid.com/sistema-operativo/que-es-android. [Accedido: 22-ene-2017].

[9] «Android, the world's most popular mobile platform | Android Developers». [En línea]. Disponible en: https://developer.android.com/about/index.html. [Accedido: 22-ene-2017].

[10] Android Developers, «Introducción a Android», *Introducción a Android*. [En línea]. Disponible en: https://developer.android.com/guide/index.html. [Accedido: 21-ene-2017].

[11] «Las versiones de Android y niveles de API - Diploma de Especialización en desarrollo de aplicaciones para Android». [En línea]. Disponible en: http://www.androidcurso.com/index.php/tutoriales-android-fundamentos/31-unidad-1-vision-general-y-entorno-de-desarrollo/146-las-versiones-de-android-y-niveles-de-api. [Accedido: 09-abr-2017].

[12] «Paneles de control | Android Developers». [En línea]. Disponible en: https://developer.android.com/about/dashboards/index.html. [Accedido: 15-dic-2017].

[13] «2.2. Arquitectura Android - Software de Comunicaciones». [En línea]. Disponible en: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android. [Accedido: 09-abr-2017].

[14] «Arquitectura de la plataforma | Android Developers». [En línea]. Disponible en: https://developer.android.com/guide/platform/index.html?hl=es. [Accedido: 09-abr-2017].

[15] «Los tres pilares de la seguridad en Android - Diploma de Especialización en desarrollo de aplicaciones para Android». [En línea]. Disponible en: http://www.androidcurso.com/index.php/tutoriales-android/41-unidad-7-seguridad-y-posicionamiento/280-los-tres-pilares-de-la-seguridad-en-android. [Accedido: 09-abr-2017].

[16] «Centro de Políticas de Desarrolladores». [En línea]. Disponible en: https://play.google.com/about/intl/es_ALL/about/developer-content-policy/index.html. [Accedido: 29-may-2017].

[17] «El esquema de permisos en Android - Diploma de Especialización en desarrollo de aplicaciones para Android». [En línea]. Disponible en: http://www.androidcurso.com/index.php/tutoriales-android/41-unidad-7-seguridad-y-posicionamiento/282-el-esquema-de-permisos-en-android. [Accedido: 09-abr-2017].

[18] «Manifest.permission | Android Developers», 02-feb-2017. [En línea]. Disponible en:

https://developer.android.com/reference/android/Manifest.permission.html. [Accedido: 02-feb-2017].

[19] «Componentes de la aplicación | Android Developers». [En línea]. Disponible en: https://developer.android.com/guide/components/index.html?hl=es-419. [Accedido: 28-may-2017].

[20] «Procesos y subprocesos | Android Developers». [En línea]. Disponible en: https://developer.android.com/guide/components/processes-and-threads.html?hl=es. [Accedido: 03-feb-2017].

[21] «Ficheros y carpetas de un proyecto Android - Diploma de Especialización en desarrollo de aplicaciones para Android». [En línea]. Disponible en: http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/148-elementos-de-un-proyecto-android. [Accedido: 04-feb-2017].

[22] «Tecnologia_y_desarrollo_en_dispositivos_moviles_(Modulo_6).pdf». .

[23] «Los tipos de malware». [En línea]. Disponible en: http://support.kaspersky.com/sp/viruses/general/614. [Accedido: 15-feb-2017].

[24] Delfin, «Infosec 101: Superficie de ataque», *Delfín Abzueta*, 04-feb-2015. .

[25] «Projects/OWASP Mobile Security Project - Security Testing - OWASP». [En línea]. Disponible en: https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Security_Testing. [Accedido: 05-feb-2017].

[26] «OWASP Mobile Security Project - OWASP». [En línea]. Disponible en: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Top_10_Mobile_Controls. [Accedido: 05-feb-2017].

[27] P. I. L. publicado 26 F. 2014- 03:34PM, «Top 10 de OWASP de vulnerabilidades en aplicaciones móviles», *WeLiveSecurity*, 26-feb-2014. [En línea]. Disponible en: http://www.welivesecurity.com/la-es/2014/02/26/top-10-owasp-vulnerabilidades-aplicaciones-moviles/. [Accedido: 16-feb-2017].

[28] OASAM, «OASAM - Open Android Security Assessment Methodology», 15-oct-2016. [En línea]. Disponible en: http://oasam.org/es/oasam. [Accedido: 15-oct-2016].

[29] «CNIL-ManagingPrivacyRisks-Methodology.pdf». .

[30] «Assessing Privacy Risks in Android: A User-Centric Approach», en *ResearchGate*.

[31] A. Warren, R. Bayley, C. Bennett, A. Charlesworth, R. Clarke, y C. Oppenheim, «Privacy Impact Assessments: International experience as a basis for UK Guidance», *Comput. Law Secur. Rev.*, vol. 24, n.º 3, pp. 233-242, 2008.

[32] «Instalación de Android Studio | Android Studio». [En línea]. Disponible en: https://developer.android.com/studio/install.html. [Accedido: 17-feb-2017].

[33] «Genymotion – Fast & Easy Android Emulator», *Genymotion – Fast & Easy Android Emulator*. [En línea]. Disponible en: https://www.genymotion.com/. [Accedido: 17-feb-2017].

[34] «The Best Android Emulator For PC & Mac | Andy Android Emulator», *Andy - Android Emulator for PC & Mac*. [En línea]. Disponible en: http://www.andyroid.net/. [Accedido: 17-feb-2017].

[35] «Guía de referencia de Nmap (Página de manual)». [En línea]. Disponible en: https://nmap.org/man/es/index.html. [Accedido: 16-abr-2017].

[36] «OpenVAS - OpenVAS - Open Vulnerability Assessment System». [En línea]. Disponible en: http://www.openvas.org/. [Accedido: 16-abr-2017].

[37] «Wireshark · Go Deep.» [En línea]. Disponible en: https://www.wireshark.org/#learnWS. [Accedido: 16-abr-2017].

[38] «SandDroid». [En línea]. Disponible en: http://sanddroid.xjtu.edu.cn/. [Accedido: 16-abr-2017].

[39] «dexter». [En línea]. Disponible en: https://dexter.dexlabs.org/. [Accedido: 16-abr-2017].

[40] «AndroBugs/AndroBugs_Framework», *GitHub*. [En línea]. Disponible en: https://github.com/AndroBugs/AndroBugs_Framework. [Accedido: 16-abr-2017].

[41] «IBM Security AppScan Mobile Analyzer (SaaS) simplifies mobile application security testing and helps accelerate their time to market», 16-dic-2014. [En línea]. Disponible en: //www.ibm.com/products/us/en/. [Accedido: 16-abr-2017].

[42] «zANTI - Mobile Security Risk Assessment | Zimperium», 20-feb-2017. [En línea]. Disponible en: https://www.zimperium.com/zanti-mobile-penetration-testing. [Accedido: 20-feb-2017].

[43] «Drozer», *MWR Labs*. [En línea]. Disponible en: https://labs.mwrinfosecurity.com/tools/drozer/. [Accedido: 16-abr-2017].

[44] «linkedin/qark», *GitHub*. [En línea]. Disponible en: https://github.com/linkedin/qark. [Accedido: 16-abr-2017].

[45] «About Santoku · Santoku Linux». [En línea]. Disponible en: https://santoku-linux.com/about-santoku/. [Accedido: 16-abr-2017].

[46] «androguard/androguard», *GitHub*. [En línea]. Disponible en: https://github.com/androguard/androguard. [Accedido: 16-abr-2017].

[47] «Androlyze — Androguard 1.9 documentation». [En línea]. Disponible en: http://doc.androguard.re/html/androlyze.html. [Accedido: 21-feb-2017].

[48] «Burp Suite editions and features». [En línea]. Disponible en: https://portswigger.net/burp/. [Accedido: 16-abr-2017].

[49] ports, «dex2jar». [En línea]. Disponible en: http://tools.kali.org/reverse-engineering/dex2jar. [Accedido: 16-abr-2017].

[50] «OWASP Mobile Security Project - OWASP», 05-feb-2017. [En línea]. Disponible en: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Top_10_Mobile_Controls. [Accedido: 05-feb-2017].

[51] *The Leading Security Assessment Framework for Android.: mwrlabs/drozer*. MWR Labs, 2018.

ANEXOS

ANEXO 1. ANDROBUGS

- 1.0 com.android.insecurebankv2.txt
- 1.1 ec.gob.sri.movil.app.txt
- 1.2 ec.gob.cnt.apps.plus.txt
- 1.3 activities.riesgo.yacare.com.ec.appriesgo.txt
- 1.4 com.walpana.ecu911.txt

ANEXO 2. BURPSUITE

- 2.1 SRI
  - 2.1.1 COMPROBANTES.png
  - 2.1.2 LOGIN.png
  - 2.1.3 MAPS.png
  - 2.1.4 MATRICULACION_REQUEST.png
  - 2.1.5 MATRICULACION_RESPONSE.png
  - 2.1.6 TURNOS_WEB.png
  - 2.1.7 SRI
- 2.2 CNT
  - 2.2.1 AUTENTICACION.png
  - 2.2.2 DETALLE_PLAN.png
  - 2.2.3 REGISTRO_CUENTA.png
  - 2.2.4 TOKENS_SESIONES.png
  - 2.2.5 VARIABLES_DE_REGISTRO.png
  - 2.2.6 VARIABLES_DE_SESION.png
  - 2.2.7 CNT
- 2.3 EC SEGURO
  - 2.3.1 C2DM_REGISTRO_ANDROID.png
  - 2.3.2 CAPTURA_ALERTAS.png
  - 2.3.3 CAPTURA_ID.png
  - 2.3.4 CERTIFICADO_SEGURIDAD.png
  - 2.3.5 REGISTRO.png
  - 2.3.6 REGISTRO_WINDOWS.png
  - 2.3.7 RESPUESTA_REGISTRO_WINDOWS.png
  - 2.3.8 RIESGOS

3.4.3 [5.2.INFO_ECU911].txt

3.4.4 [6.5.1.ATTSURFACE_ECU911].txt

3.4.5 [6.5.1.MANIFEST_ECU911].txt

3.4.6 [6.5.2.ACTIVITIES_ECU911].txt

ANEXO 4. LOGCAT

4.0 log_DISPOSITIVO.txt

4.1 log_SRI.txt

4.2 log_CNT.txt

4.3 log_EC_SEGURO.txt

4.4 log_911.txt

ANEXO 5. SANDDROID

1.1 LINKS ANALISIS.xlsx

ANEXO 6. WIRESHARK

6.1 SRI

6.1.2 SRI.pcapng

6.6.1 METODO_GET.png

6.2 CNT

6.2.1 CNT.pcapng

6.3 EC_SEGURO

6.3.1 RIESGOS.pcapng

6.4 911

6.4.1 ECU911.pcapng

ANEXO 7. IBM MOBILE SCAN

7.1 InsecureBankv2.pdf

ANEXO 8. ZANTI

8.1 Mapeo de red.png

8.2 Dispositivo Móvil.png

8.3 Ataques MiTM.png

8.4 Captura de sesiones.png

8.5 Captura de contrase¤as.png

ANEXO 9. OPENVAS

9.1 SRI

9.2 CNT

9.3 EC SEGURO

9.4 ECU 911

ANEXO 10. NMAP

10.1 SRI

10.1.1 TCP_ANTERIOR_SRI.xml

10.1.2 UDP_ANTERIOR_SRI.xml

10.1.3 TCP_POSTERIOR_SRI.xml

10.1.4 UDP_POSTERIOR_SRI.xml

10.2 CNT

10.2.1 TCP_ANTERIOR_CNT.xml

10.2.2 UDP_ANTERIOR_CNT.xml

10.2.3 TCP_POSTERIOR_CNT.xml

10.2.4 UDP_POSTERIOR_CNT.xml

10.3 EC SEGURO

10.3.1 TCP_ANTERIOR_ECSEG.xml

10.3.2 UDP_ANTERIOR_ECSEG.xml

10.3.3 TCP_POSTERIOR_ECSEG.xml

10.3.4 UDP_POSTERIOR_ECSEG.xml

10.4 ECU 911

10.4.1 TCP_ANTERIOR_911.xml

10.4.2 UDP_ANTERIOR_911.xml

10.4.3 TCP_POSTERIOR_911.xml

10.4.4 UDP_POSTERIOR_911.xml

ANEXO 11. DEXTER

11.1 Información_General.png

11.2 Lista de clases.png

11.3 Diagrama de clases.png

11.4 Dependencia de clases.png

ANEXO 12. QARK

12.1 InsecureBankv2

ANEXO 13. Mapeo de Activos de información.xlsx

ANEXO 14. APKS

14.0 InsecureBankv2.apk