MAESTRÍA EN AUDITORIA DE TECNOLOGÍA DE LA INFORMACIÓN

MARCO DE REFERENCIA PARA EL DESARROLLO DE UNA AUDITORÍA DE SEGURIDAD INFORMÁTICA EN APLICACIONES ANDROID.

Propuesta de artículo presentado como requisito para la obtención del título:

Magíster en Auditoría de Tecnologías de la Información.

Por la estudiante:
Santiago Fabricio LEÓN CABRERA.

Bajo la dirección de:
Marco Vinicio SOTOMAYOR SANCHEZ.

Universidad Espíritu Santo
Maestría en Auditoría de Tecnología de la Información
Samborondón - Ecuador
Noviembre del 2018.

Marco de referencia para el desarrollo de una auditoría de seguridad informática en Aplicaciones Android.

Framework for the development of a computer security audit in Android Applications.

Santiago Fabricio LEÓN CABRERA[1]
Marco Vinicio SOTOMAYOR SANCHEZ [2]

Resumen

El presente trabajo está enfocado en proponer un marco de referencia para el desarrollo de una auditoría de seguridad informática en aplicaciones Android, que especifique procedimientos y ajuste nomenclaturas que facilite a los auditores definir el impacto, la probabilidad de que una vulnerabilidad se materialice y el nivel de seguridad de las salvaguardas encontradas dentro de las vulnerabilidades frecuentes en aplicaciones Android. Para esto se realiza un análisis bibliométrico de los principales conceptos sobre las fases de una auditoría enfocadas a las tecnologías de la información, tratamientos de riesgos enfocados en tecnologías de la información y metodologías enfocadas al análisis de fallos en aplicaciones móviles, con el resultado obtenido se seleccionaron tres artículos académicos los cuales fueron escogidos debido a que están basados en metodologías reconocidas y probadas: la guía de auditoría y aseguramiento de sistemas de la información planteada por la Isaca, la metodología de análisis y gestión de riesgos de los Sistemas de Información (Magerit) y la metodologías de análisis de seguridad *Owasp Mobile Security Project*, las cuales sirven de base para la construcción del marco propuesto con lo que se exponen las etapas y procedimientos que se deben seguir dentro de una auditoría, igualmente con la información obtenida de la elaboración del marco de referencia se obtiene una lista de chequeo con los fallos más frecuentes que se debería analizar con la opción de clasificar su probabilidad de ocurrencia y especificar el nivel de implementación de las salvaguardas encontradas. La validación de este trabajo se realiza a través de la técnica de grupo focal con un enfoque cualitativo, la cual facilitó la participación de cinco profesionales en el área los cuales proporcionaron sus opiniones y puntos de vista acerca de la factibilidad del uso del marco propuesto. Al final se obtuvo la aceptación del cien por ciento de los participantes considerándolo una alternativa viable para realizar una auditoría de seguridad en aplicaciones Android.

Palabras clave: Marco de Auditoría de TI, Vulnerabilidades Aplicaciones, Tratamiento de riesgos, Android.

Abstract

The present work is focused on proposing a frame of reference for the development of an IT security audit in Android applications, specifying procedures and adjusting nomenclatures that facilitate the auditors to define the impact, the probability that a vulnerability will materialize and the level security of the safeguards found within the frequent vulnerabilities in Android applications. For this, a bibliometric analysis of the main concepts on the phases of an audit focused on information technologies, risk treatments focused on information technologies and methodologies focused on the analysis of failures in mobile applications is carried out, with the result obtained They selected three academic articles which were chosen because they are based on recognized and proven methodologies: the audit and information systems assurance guide proposed by the Isaca, the risk analysis and management methodology of the Information Systems (Magerit) and the Owasp Mobile Security Project security analysis methodologies, which serve as the basis for the construction of the proposed framework, which exposes the stages and procedures that must be followed within an audit, also with the information obtained from the elaboration of the frame of reference you get a li This is the checklist with the most frequent failures that should be analyzed with the option to classify their probability of occurrence and specify the level of implementation of the safeguards found. The validation of this work is done through the technique of a focus group with a qualitative approach, which facilitated the participation of five professionals in the area who provided their opinions and points of view about the feasibility of using the proposed framework. In the end the acceptance of one hundred percent of the participants was obtained considering it a viable alternative to perform a security audit in Android applications.

Key words IT Audit Framework, Vulnerabilities Applications, Risk Management, Android.

[1] Estudiante de Maestría en Auditoría de Tecnología de Información, Universidad Espíritu Santo – Ecuador. E-mail sfleon@uees.edu.ec.
[2] Máster en tecnologías de la información, E-mail: mvinicio@uees.edu.ec.

INTRODUCCIÓN

A finales de la década de los noventa los dispositivos móviles empezaron a ganar mercado a nivel mundial. Poco a poco estos terminales ofrecían más funciones optimizando tiempos y simplificando las tareas tanto personales como laborales, siguiendo el crecimiento de la tecnología se incorporaron sistemas operativos robustos que en conjunto con la creación de aplicaciones móviles fueron progresando enormemente debido a que se añadieron gran variedad de funcionalidades que facilitaron la experiencia de los usuarios.

Gartner (2016) señala que actualmente las empresas que están liderando los sistemas operativos utilizados en dispositivos móviles son Android, IOS y Windows *Phone*, siendo el más usado Android que pertenece a la empresa Google posicionándose en el primer lugar con un 86% del mercado, esto debido a que está ampliamente comercializado en dispositivos móviles de diferentes marcas.

Según Ferreira, dos Santos, & Choren (2016) se descargaron en el año 2017 aproximadamente 267.78 mil millones de aplicaciones desde Google Play, de todas esas descargas el 90% de las aplicaciones tiene algún tipo de vulnerabilidad.

A la par de los avances tecnológicos se acarrean grandes problemas de seguridad dado que las aplicaciones se vuelven más sofisticadas y requieren acceder a información almacenada en los dispositivos para brindar una experiencia personalizada al usuario y en muchos casos estas no son desarrolladas tomando en cuenta normas de seguridad de software provocando un gran problema debido a la fuga de información que puede existir derivando esto en pérdidas económicas a los usuarios y a las empresas que utilizan dichas aplicaciones (Arxan Technologies, Inc, 2016).

Debido a esto es importante que las aplicaciones sean auditadas dado que ayudará en gran medida a minimizar las vulnerabilidades que pueden existir evitando con esto inconvenientes que pueden poner en peligro información confidencial que manejan (Santos , 2012).

De acuerdo a lo investigado se ha evidenciado que si bien es cierto existen varios estudios los cuales están direccionados al aseguramiento de las aplicaciones que van desde: El presentado por Berthomé, Fécherolle, & Guilloteau (2012) los cuales elaboran una auditoría para evidenciar si una aplicación es sospecha de solicitar permisos para acceder a datos confidenciales; de la misma manera Shinichi & Kouichi (2013) basándose en un análisis estático obtienen las posibles filtraciones de información confidencial que se pueden presentar en una aplicación. Por otra parte, Ehrenreich, Acharya, & Marc (2015) han realizado una verificación de fallos de seguridad basándose en metodologías de análisis de seguridad como *Owasp Mobile* Project. Dentro del mismo contexto Hernández, Toro, & Vargas (2015) utilizando la metodología NIST *Special Publication* 800-163 realizan un guía para comprobar si una aplicación tiene fallos de seguridad que puedan repercutir en el correcto funcionamiento de una aplicación. Otros estudios como el desarrollado por Lu, Li, Wu, Lee, & Jiang (2012) y SounthiraraJ, Sahs, Greenwood, Lin, & Khan (2014) han desarrollado programas para automatizar la búsqueda de errores en las aplicaciones con el propósito de disminuir sus vulnerabilidades. Sin embargo, como señala Purser (2014) existen inconvenientes al momento de obtener resultados de una auditoría de seguridad puesto que se dificulta clasificar el grado de protección que se obtiene dentro de un análisis ocasionado por los diferentes conceptos y terminologías que se pueden utilizar según la profundidad de la auditoría. Así también Quirolgico, Voas, Karygiannis, Michael, & Scarfone (2015) concuerdan en que cada auditor puede interpretar de diferente manera aspectos fundamentales como: la gravedad de los fallos de seguridad, la semántica y nomenclatura dado que si bien es cierto se apoyan en metodologías para el análisis de vulnerabilidades cada uno puede tener distintos criterios sobre un mismo

fallo provocando un conflicto al momento de tomar decisiones con informes de diferentes analizadores. Con todo lo citado previamente se identifica que en la actualidad no existe un marco de referencia que unifique procedimientos de análisis dentro de una auditoría de seguridad en aplicaciones Android el cual permita que el o los informes presentados manejen los mismo procedimientos y criterios de calificación.

El presente trabajo de investigación tiene como objetivo proponer un marco de referencia para el desarrollo de una auditoría de seguridad informática en aplicaciones Android que permita minimizar la diferencia de criterios entre analizadores. Para lograr este propósito primero se realizará un análisis bibliométrico el cual permita obtener información relevante que servirá de apoyo para elaboración de esta investigación. Dentro de los puntos a plantear en este marco están: 1) Presentar las etapas de la auditoría tomando como referencia la guía de auditoría y aseguramiento de sistemas de la información de Isaca descritas por Cooke (2018); 2) Exponer las vulnerabilidades más frecuentes y su impacto negativo para ello se basa en el trabajo de investigación realizado por Ferreira, dos Santos, & Choren (2016)*;* 3) Luego teniendo como referencia el trabajo realizado por Vega, Arroyo, & Yoo (2017) los cuales manejan la metodología de análisis y gestión de riesgos (Magerit) se realiza la clasificación del impacto y probabilidad de ocurrencia de las vulnerabilidades estandarizando procedimientos y nomenclaturas, por último se elabora una lista de chequeo tomando como base lo presentado en los puntos dos y tres. La validación de esta investigación se la realiza con la intervención de un grupo focal que mediante sus opiniones y puntos de vista se verifica la factibilidad del uso del marco propuesto.

MARCO TEÓRICO.

Estudios Previos.

Dentro de los estudios relacionados se encuentra que entre los primeros autores que abordaron esta temática están Enck, Octeau, & McDaniel (2011) los cuales realizaron un estudio de las propiedades de la seguridad de un conjunto específico de aplicaciones Android con el propósito de obtener su código fuente y analizarlo en busca de fallos de seguridad. Posteriormente Berthomé, Fécherolle, & Guilloteau (2012) realizaron un trabajo direccionado en auditar aplicaciones sospechosas que solicitan permisos para acceder a datos privados y saber si se ha producido el acceso, para dicho trabajo ejecutan el reempaquetado de la aplicación y verifican su funcionamiento.

Siguiendo el enfoque del tema anteriormente citado Lu, Li, Wu, Lee, & Jiang (2012) proponen un método llamado CHEX el cual realiza un análisis estático de las aplicaciones Android enfocados en detectar y alertar vulnerabilidades específicas como: la pérdida de permisos, el acceso a datos no autorizados, la suplantación de identidad, basándose en el análisis de flujo de datos. Dentro de la misma orientación Shinichi & Kouichi (2013) presentan una herramienta para la verificación de vulnerabilidades en aplicaciones Android utilizando el análisis estático, para ello analizan el código fuente y determinan la posibilidad de filtraciones de información privada.

Así también Dhama (2014) presenta un análisis de los desafíos de seguridad sobre los permisos en aplicaciones Android, para ello el autor desarrollo una aplicación de prueba para mostrar el uso y el nivel de protección de los permisos que se deberían manejar en las aplicaciones Android basándose en el análisis de varios permisos y sus niveles de protección. Mientras que SounthiraraJ, Sahs, Greenwood, Lin, & Khan (2014) presentan una implementación llamada SMV-Hunter el cual está enfocado en determinar si la certificación estándar del protocolo SSL/TLS que es utilizada por los desarrolladores de las aplicaciones es segura, para esto realiza un análisis estático y dinámico en busca de procedimientos de validaciones personalizadas que desencadenen en un ataque *Man-in-the-Middle.*

Ehrenreich, Acharya, & Marc (2015) proponen una metodología basándose en las buenas prácticas del proyecto *Owasp Mobile* para la implementación de una evaluación de aplicaciones médicas que manejen almacenamiento de datos. Así mismo empleando la norma NIST 800 – 163 Hernández, Toro, & Vargas (2015) generan una guía para la validación de aplicaciones móviles enfocado en verificar sus vulnerabilidades aplicando controles basados en la norma ISO 27001. Continuando la misma línea de investigación Ramírez, Edward, & Cifuentes (2016) direccionan su estudio al análisis de las vulnerabilidades detectadas en aplicaciones médicas móviles según las vulnerabilidades definidas por *OWASP* versión 2014.

Dentro de trabajos relacionados a este tema en el Ecuador Moreira, García, & Moreira (2016) realizan una evaluación de protocolos de seguridad de aplicaciones Android con el propósito de determinar la seguridad de ciertas aplicaciones enfocadas al uso de las redes sociales. Así mismo, Erreyes (2017) presenta una metodología para el escogimiento de protocolos y herramientas que pueda utilizar los usuarios para optimizar la seguridad en dispositivos móviles tomando como referencia el proyecto *Owasp Mobile* y Cobit 5. Mientras que Argudo, López, & Sánchez (2017) proponen una evaluación para el análisis de vulnerabilidades en aplicaciones móviles para ello escogen controles del Proyecto de Seguridad Móvil de OWASP enfocándose en la recopilación de información, inicio de sesión y cifrado y transferencia de datos entre dispositivos.

Por otra parte, Faysal, Syeda , & Anindya (2017) realizaron un estudio empírico sobre un conjunto específico de aplicaciones y en base a ellas detectaron las vulnerabilidades que las afectan utilizando tres herramientas de calidad reportando los resultados detectados y los detalles acerca de los fallos encontrados. Mientras que Gao, Li, Kong, & Klein (2018) realizan un trabajo enfocado en presentar la evolución de vulnerabilidades en diferentes versiones que históricamente se lanzaron para la misma aplicación y con el apoyo de reconocidos escáneres de fallos se presentaron los hallazgos más relevantes y las diferencias que existen entre las versiones analizadas con el propósito de minimizar futuros problemas.

Dentro de los estudios previos expuestos anteriormente se puede apreciar que los autores han realizado diferentes enfoques hacia el aseguramiento de aplicaciones Android, estos han ido desde minimizar las vulnerabilidades a través de la busca de fallos de seguridad, pasando por la utilización de diferentes programas con el objetivo de disminuir las vulnerabilidades. También se han empleado ciertas metodologías enfocadas en análisis de fallos de seguridad para encontrar vulnerabilidades que puedan afectar su seguridad, y otros autores han direccionado sus estudios en desarrollar aplicaciones con el propósito de realizar análisis automatizados de fallos de seguridad. Sin embargo, a pesar que se han realizado varios enfoques hacia temas relacionados al aseguramiento de aplicaciones, no se ha desarrollado un marco de referencia orientado a una auditoría de seguridad informática en aplicaciones Android el cual presente etapas, procedimientos y estandarice nomenclaturas de evaluación que ayuden a minimizar la diferencia de criterios que pueden tener los analizadores sobre una misma aplicación.

Entorno de Aplicaciones Android.

Mullis (2017) menciona que las aplicaciones Android se desarrollan con Android Software Development Kit (Android SDK) utilizando el lenguaje de programación Java combinado según el caso con C / C++. La aplicación empaquetada de Android (APK) es la encargada de guardar el contenido y a su vez es el archivo que se usa para instalarla, posteriormente a esto Android genera una máquina virtual por cada aplicación ejecutada.

Dentro de una aplicación Android Chen, Qian, & Mao (2014) describen cuatro tipos de componentes, Actividades: Representa una pantalla con interfaz de usuario cada una es

independiente pero trabajan en conjunto, tiene tres estados (Activa, Pausada, Detenida); Proveedor de Contenidos: Administra un conjunto de datos compartidos de una aplicación para que otras aplicaciones puedan consultar o realizar modificaciones; Receptor de mensajes: Responde a los intentos de transmisión de mensajes emitidos por el sistema; y Servicios: Se ejecutan en segundo plano sin interfaz gráfica son los encargados de realizar operaciones largas.

García del Moral (2016) describe la estructura de una aplicación de la siguiente manera: Android *Manifest***:** Es un fichero XML ubicado en la raíz del fichero APK el cual está encargado de brindar información esencial al sistema con el objetivo de identificar las acciones que podrá realizar basándose en los permisos declarados; *Classes*: Contiene el código Java compilado en formato DEX para que la máquina virtual *Dalvik* (DVM) o su vez Android *Runtime* (ART) lo interpreten; *Resources.arsc*: Abarca recursos multimedia almacenados; Meta-Inf: Dentro de este directorio se encuentren ficheros como: el *Manifest.mf* encargado de enumerar los ficheros en conjunto con su hash SHA1; RSA: Es donde se incluye la firma del fichero *Cert.sf* y su certificado; *Assets:* Es un directorio que contiene recursos sin procesar; Res*:* Es el directorio encargado de guardar los recursos multimedia de la aplicación que no fueron compilados en el fichero resources.arsc; Directorio Lib: Almacena y administra las librerías compartidas, compiladas y distribuidas en subdirectorios basándose en la arquitectura del procesador. En la figura 1 se presenta la estructura completa que tiene una aplicación Android.

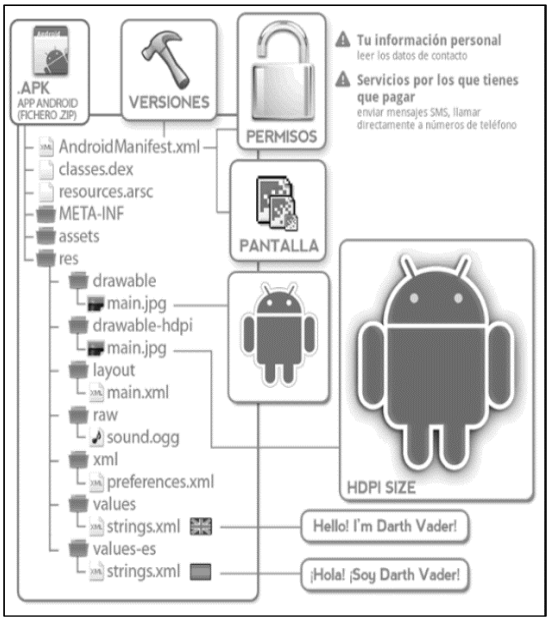

Figura 1: Estructura de una Aplicación.
Fuente: (Sanchez, 2016)

Amenazas y Vulnerabilidades en Aplicaciones.

El crecimiento acelerado que tiene la plataforma Android han acarreado un aumento en la infección de malware mediante la publicación de aplicaciones infectadas en tiendas online. Un método común de propagarse es mediante el re empaquetamiento de aplicaciones legítimas o la creación de aplicaciones infectadas, para ello se inserta código malicioso manteniendo sus operaciones invisibles para el usuario el tiempo suficiente para poner en peligro la información confidencial que interactúe con la aplicación (Khanmohammadi, Rejali , & Hamou-Lhadj, 2015).

Según reportes de Unuche, Sinitsyn, Parinov, & Liskin (2017) investigadores de kaspersky en el tercer trimestre del año 2017 detectaron: 1.598.196 aplicaciones maliciosas; y 19.748 paquetes de instalación de troyanos bancarios para dispositivos móviles; dentro de los malware más peligrosos expuestos por estos investigadores están: *DangerousObject Multi.Generic* (67,14%), *Trojan.AndroidOS.Boogr.gsh* (7,52%) y Trojan.AndroidOS.Hiddad.ax *malware* (4,56%).

Para Kuma Jha, Lee, & Lee (2017) cada vez las aplicaciones manejan información de mayor relevancia y no siempre se la gestiona de manera correcta, una de las principales razones que provocan fallos de seguridad dentro de las aplicaciones son los errores de configuración. Un archivo de suma importancia es *Manifest* dado que la mayor parte de sus parámetros de configuración son altamente comprometedores y una mala distribución de funciones puede tener graves consecuencias.

Auditoría enfocada a las tecnologías de la información.

Porter & Burton (1983) definen a la auditoría como una verificación de la información realizada por una tercera persona ajena a quien la realizó, con el propósito de confirmar su veracidad y posteriormente dar a conocer los resultados obtenidos. En el mismo contexto Aumatell (2003) señala que la auditoría es un proceso sistemático e independiente que tiene como propósito brindar un diagnóstico general de la situación de la información comprobando si las actividades, objetivos y resultados propuestos cumplen las disposiciones establecidas por la empresa.

En un ámbito orientado a las tecnologías de la información OLACEFs (2011) describe a la auditoría como un examen objetivo independiente, crítico y sistemático de las políticas, procesos e informes de una entidad con el propósito de brindar una opinión imparcial sobre la validez, efectividad; y optimización de los recursos tecnológicos. Mientras que Weber (1988) define a la auditoría como una función que ha sido creada para asegurar las salvaguardas de los activos de los sistemas informáticos manteniendo la integridad de los datos siempre enfocado en cumplir los objetivos establecidos previamente. De la misma manera Echenique (2001) manifiesta que la auditoría de sistemas de información está dirigida a la revisión y evaluación de los riegos con el fin de establecer si los recursos analizados protegen la integridad y la disponibilidad de datos minimizando los eventos no deseados.

Mejores prácticas de una auditoría en sistemas de información.

Dentro de las mejores prácticas de una auditoría de sistemas de información Isaca (2012) señala que es importante establecer un plan de trabajo que sirva de guía en el desarrollo de una auditoría para ello exponen las siguientes fases: La primera etapa es la planificación en la cual se define el contexto del negocio, los objetivos y el plan a seguir durante la auditoría; la segunda etapa es la ejecución donde se realiza la recopilación de información evaluando los riesgos y la planificación mediante un cronograma que facilite la evaluación y pruebas de cumplimiento; La tercera etapa es el informe en el cual se emite un juicio basándose en las evidencias obtenidas en las fases anteriores adicionalmente se presenta las limitaciones encontradas. A continuación, se presenta en la figura 2 las fases de una auditoría informática.

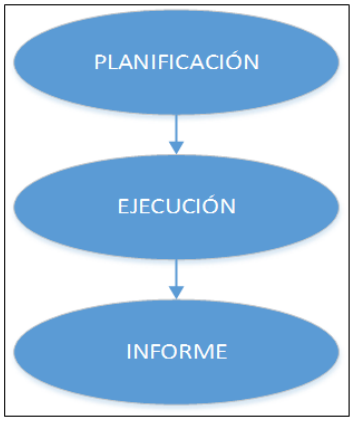

Figura 2: Fases de una Auditoría informática.
Fuente: Elaboración propia - & Isaca (2012).

De la misma forma dentro de Isaca (2012) COBIT 5 se presenta como un modelo para auditar la gestión y control de los sistemas de información integrando la administración y la tecnología relacionada con la organización, con el propósito de minimizar los riegos y los impactos negativos que estos puedan causar (Nugroho, 2014). Así mismo el proceso de administración de sistemas de información permite fortalecer a la organización para el cumplimiento de los objetivos de gobernanza y

con ello la creación de valor del negocio (Ataya, 2013).

Por otra parte, ISO 19011 (2011) Brinda una guía para la dirección de una auditoría basándose en la planeación y realización de una auditoría a sistemas de gestión relacionando tanto el riesgo de que el proceso auditado no alcance los objetivos, como el potencial de que la auditoría interfiera con las actividades y proceso revisados; para ello dispone de fases las cuales ayudan al cumplimiento del objetivo y alcance planteados. A continuación, en la figura 3 se presenta las fases de una auditoría según la ISO 19011 - 2011.

Figura 3: Fases de una Auditoría.
Fuente: Elaboración propia - (ISO 19011 , 2011)**.**

Riesgos en las tecnologías de la información.

Para Areito (2008) el riesgo es el impacto negativo que puede causar una vulnerabilidad considerando tanto su impacto como la probabilidad de ocurrencia, identificando y priorizando los riesgos inherentes que puede tener un sistema, producto u organización para con ello establecer un nivel aceptable de riesgo para la empresa. De la misma manera Benaroch (2002) indica que estos riegos deben ser registrados y según la gravedad de los mismos pueden ser manejados en conjunto mediante subcontratación externa o a su vez escalando secuencialmente responsabilidades a medida que las incertidumbres se solventan.

Ramirez & Ortiz (2011) señalan que es importante realizar el análisis de riesgo dado que permite tomar decisiones sobre las acciones a realizar con las amenazas detectadas y aplicar medidas correctivas antes de empezar un servicio o cuando se encuentra en funcionamiento. Por lo tanto, se requiere analizar los activos relevantes de la organización para que las amenazas no produzcan daños significativos sobre un servicio.

Metodologías para la Gestión de Riesgos en tecnologías de la información.

En cuanto a las metodologías utilizadas para la gestión del riesgo *NIST* 800-30 (2002) señala que tiene como objetivo brindar una base para el desarrollo de la gestión del riesgo mediante aseguramiento de los sistemas de información optimizándolos a partir del resultado de un análisis previo, para ello realiza una asignación de un nivel de riesgo según la vulnerabilidad, pudiendo ser: alta, media y baja.

Por otra parte, Magerit es una metodología de análisis y gestión de riegos enfocada en las tecnologías de la información y comunicación (TIC), la misma está dirigida en concientizar a los encargados de las organizaciones a la existencia de los riesgos y la importancia de gestionarlos, realizando un análisis de los fallos permitiendo una toma de decisiones apropiadas mediante una serie de pasos los cuales ayudan a minimizar la improvisación y los criterios variados entre analistas, por lo cual se trabaja mediante el conocimiento del impacto que puede causar una amenaza sobre los activos y la probabilidad de que esto puede suceder (Candau & Amutio, 2015).

Dentro de esta metodología para realizar un análisis de riesgo es importante tener en cuenta los siguientes pasos: el primero es determinar los activos relevantes para la empresa, segundo es importante determinar las amenazas a los que están expuestas dichos activos, tercero

verificar las contramedidas que están implementadas, cuarto estimar el impacto provocado por el daño al activo producido por la ejecución de la amenaza, y quinto estimar el riesgo determinado por el impacto ponderado con la probabilidad de ocurrencia de la amenaza (Candau & Amutio, 2015). En la figura 4 se detalla los elementos del análisis de riesgos.

factor	nivel	significado
0%	L0	inexistente
	L1	inicial / ad hoc
	L2	reproducible, pero intuitivo
	L3	proceso definido
	L4	gestionado y medible
100%	L5	optimizado

Figura 4: Elementos del Análisis de Riesgos.
Fuente: (Candau & Amutio, 2015)**.**

Así también Vanegas (2017) señala que dentro de Magerit es importante verificar si se incorporan salvaguardas y que tan efectivas son, para ello implementa un rango que va del 0% para las situaciones en las que no existen salvaguardas y un 100% para aquellas salvaguardas que están correctamente implementadas y controladas; entre los puntos a tomar en cuenta para calificar el nivel de las contramedidas se especifican: Verificar si las contramedidas están correctamente implementadas y configuradas; Confirmar el cumplimiento de parámetros para hacer frente a los riesgos y por último validar la existencia de controles que alerten posibles fallos del sistema.

Soleto, Torres, & Rivera (2012) indican que existen diferentes maneras de manejar los riesgos dependiendo del tratamiento que se quiera dar, entre los que señalan: Evitar los escenarios que lo provocan; mitigar el riesgo o minimizar la probabilidad de ocurrencia, compartir o transferir el riesgo y por último aceptar que pueda existir un riesgo mínimo dado que no existe una seguridad absoluta. En la figura 5 se presenta las formas de tratamiento de los riegos.

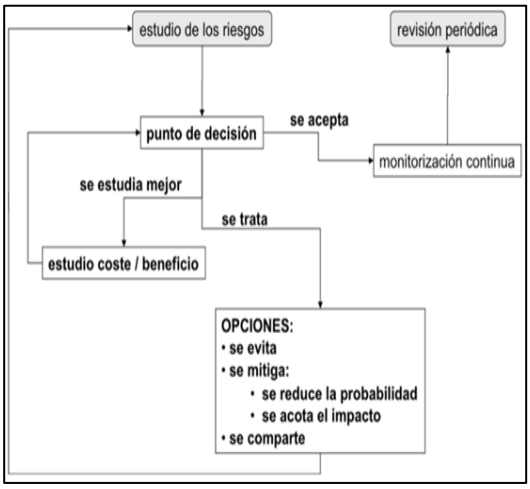

Figura 5: Formas de tratamiento de los riesgos.
Fuente: (Candau & Amutio, 2015)**.**

Metodologías enfocadas al análisis de vulnerabilidades de aplicaciones móviles.

Para el análisis de aplicaciones existen una cantidad reducida de metodologías que se pueden tomar como referencia las cuales se presentan a continuación:

NIST *Special Publication* 800-163.

Quirolgico, Voas, Karygiannis, Michael, & Scarfone (2015) desarrollaron la NIST *Special Publication* 800-163 la cual tiene el respaldo del Instituto Nacional de Normas y Tecnología de los Estados Unidos que tiene como objetivo comprender el proceso para verificar la seguridad de aplicaciones móviles señalando la importancia, función e interacción que tiene un auditor con el personal encargado del desarrollo y análisis de las aplicaciones

De la misma manera Bello (2017) señala que esta metodología involucra cuatro enfoques para la revisión de una aplicación, las cuales están destinadas a proporcionar pautas para detectar un error o vulnerabilidad que pueda ocasionar la pérdida o fuga de información, 1) Pruebas de corrección: Orientada a la identificación de los errores enfocado en el mejoramiento de la calidad; 2) Código fuente o

Código Binario: Permite el análisis del código fuente de la aplicación en busca de fallos, cuando el código fuente no está disponible se realiza ingeniería inversa para el análisis del código; 3) Pruebas manuales y automáticas: Se basa en simuladores, accesos remotos al dispositivo y el ingreso de datos inválidos o aleatorios en las entradas de datos; 4) Análisis estático o dinámico: El análisis estático está enfocado en examinar el código fuente y buscar posibles errores en la etapa de desarrollo, por el contrario el análisis dinámico verifica el comportamiento que tiene una aplicación en tiempo de ejecución.

Cabe señalar que la NIST *Special Publication* 800-163 expone las vulnerabilidades en aplicaciones Android que bajo su criterio se presenta con mayor frecuencia y pueden ocasionar mayores peligros para la información las mismas se presentan en la tabla 1.

Tabla 1: Listado de Vulnerabilidades frecuentes según NIST SP 800-163.

METODOLOGÍA NIST SP 800-163	
VULNERABILIDAD	**DESCRIPCIÓN**
Permiso incorrecto.	Fallo en la entrega excesiva de permisos.
Comunicaciones expuestas.	Fallos en los protocolos de comunicación.
Funcionalidad potencialmente peligrosa.	Fallo en el acceso a recursos críticos del sistema o información de usuario.
Colusión de aplicaciones.	Intercambio de información más allá del alcance declarado entre aplicaciones.
Ofuscación.	Fallo en el ocultamiento de Código fuente y librerías externas.
Consumo de energía excesivo.	Funcionalidad excesivas no deseadas.
Vulnerabilidades de software tradicional.	Fallo en la autenticación, control de acceso, y cifrado.

Fuente: Elaboración Propia - (Quirolgico, Voas, Karygiannis, Michael, & Scarfone, 2015)**.**

OASAM (Metodología de evaluación de seguridad de Android).

Es una metodología enfocada en la evaluación de vulnerabilidades en aplicaciones Android, la cual cataloga las vulnerabilidades que consideran más riesgosas con el propósito de apoyar a los desarrolladores de aplicaciones a tener presente las fallas que se pueden presentar al momento de su desarrollo o ejecución (Medianero & Villagrá, 2017).

Para iniciar el análisis Martínez (2014) menciona que OASAM hace énfasis en la importancia de realizar una recopilación de información debido a que con ello se define la superficie de los posibles fallos pudiendo conocer errores de configuración o de implementación de la aplicación analizada, posteriormente se inicia con la verificación de los controles en las vulnerabilidades más frecuentes que menciona esta metodología, las cuales se muestran en la tabla 2.

Tabla 2: Listado de Vulnerabilidades frecuentes según OASAM.

METODOLOGÍA OASAM	
VULNERABILIDAD	**DESCRIPCIÓN**
OASAM-INFO	Recopilación de Información.
OASAM-CONF	Error en la configuración de la aplicación.
OASAM-AUTH	Fallo de Autenticación.
OASAM-CRYPT	Fallo de Criptografía.
OASAM-LEAK	Fuga de información confidencial.
OASAM-DV	Validación de datos.
OASAM-IS	Intento de suplantación
OASAM-UIR	Recibo de componentes no autorizados.
OASAM-BL	Problemas inesperados de diseño que afectan el rendimiento.

Fuente: Elaboración Propia - (Medianero & Villagrá, 2017).

Owasp Mobile Security Project.

Carter & Thakur (2017) lideran el proyecto Owasp Mobile Security Project el cual está enfocado en analizar la seguridad y brindar los recursos necesarios para construir y mantener aplicaciones móviles seguras; para lograr un nivel considerable de seguridad esta metodología maneja tres factores: El primero está orientado a las personas las cuales deben estar en constante capacitaciones; El segundo es el manejo de los procesos a través del administración y verificación de vulnerabilidades

y por último la tecnología aplicando herramientas adecuadas para el desarrollo y análisis de errores (Arroyo, 2016).

Uno de sus principales objetivos de esta metodología está orientado a la clasificación de riesgos más frecuentes y con ello proporcionar controles necesarios para asegurar la información. Su análisis abarca desde la capa de aplicación hasta la transmisión con el servidor que se comunica la aplicación móvil; centrándose en la integración de la aplicación móvil, los servicios de autenticación remota y funciones específicas de la aplicación para minimizar los daños que puedan causar las vulnerabilidades (The OWASP Foundation, 2017). En la tabla 3 se detalla las vulnerabilidades.

Tabla 3: Listado de Vulnerabilidades frecuentes**.**

METODOLOGÍA OWASP MOBILE PROJECT	
VULNERABILIDAD	**DESCRIPCIÓN**
Uso inadecuado de la plataforma.	Mal uso de las características y permisos de la plataforma.
Almacenamiento inseguro de datos.	Pérdida o exposición de los datos confidenciales.
Comunicación insegura.	Fallos que facilitan el robo de información entre dispositivos.
Autenticación no segura.	Debilidades en la gestión incorrecta de sesiones.
Criptografía insuficiente.	Fallo al cifrar los datos que permite que la información se vea comprometida.
Autorización insegura.	Fallos del lado del servidor durante el procedimiento de autenticación.
Calidad del código del cliente.	Fallos a nivel de código que afectan a la aplicación.
Manipulación de Código.	Fallo por la modificación del recurso binario o local de la aplicación.
Ingeniería inversa.	Fallos en el núcleo binario del código fuente, bibliotecas, algoritmos.
Funcionalidad extraña.	Fallo producido por los desarrolladores al incluir funciones ocultas como puerta trasera u otros controles internos de seguridad

Fuente: Elaboración propia - (The OWASP Foundation, 2017)**.**

Tipos de pruebas en Aplicaciones Móviles.

Las tres metodológicas expuestas anteriormente coinciden que para la ejecución de una prueba de seguridad en aplicaciones móviles se realizan dos tipos de análisis: El primero se trata de un Análisis estático el cual está enfocado en analizar el código fuente de una aplicación intentando deducir los comportamientos y posibles errores que podría presentarse cuando la aplicación se ejecute. Para realizar este análisis se utiliza un desensamblador o descompilador para recuperar el código fuente, uno de los principales archivos que se debe analizar es el "*AndroidManifest.xml*" dado que contiene información relevante sobre cómo está estructurada y los componentes con los que interactúa, así también se encuentra código Java en el cual se consiguen ficheros como el "*classes.dex*" el cual es importante considerar, adicionalmente se obtienen librerías compartidas o distribuidas como parte de los "*Assets*" donde se obtienen ficheros con extensiones como: *Class, Jar, Dex, Apk, Html* entre otros. (Garcia del Moral, 2016).

Dentro del mismo contexto, Garcia del Moral (2016) señala que el análisis dinámico está basado en verificar el funcionamiento del código de la aplicación en tiempo de ejecución, para ello se utiliza un depurador o emulador según el campo de análisis; esto se lo realiza con el propósito de encontrar solicitudes indebidas, comunicaciones establecidas, autenticaciones, interacciones entre componentes y configuraciones hacia conexiones remotas, entre otros.

METODOLOGÍA.

Para la elaboración de este trabajo se escogió una metodología con enfoque cualitativo con un alcance de tipo descriptivo. Para ello se realiza un análisis bibliométrico para obtener tipos de auditorías dirigidas a las tecnologías de la información, metodologías enfocadas en el análisis de vulnerabilidades y metodologías enfocadas al tratamiento de riesgos sobre tecnologías de la información y con ello estructurar el marco de referencia propuesto.

Pregunta de Investigación.

La poca información dirigida hacia la elaboración de una auditoría sobre aplicaciones Android derivada de la variedad de conceptos y diferentes criterios que pueden presentar los auditores, da origen a la interrogante: ¿Qué lineamientos se deben seguir para generalizar conceptos y procedimientos al momento de realizar una auditoría informática sobre aplicaciones Android?. Para esto se realiza un análisis bibliométrico que permita obtener artículos académicos basados en metodologías reconocidas y probadas las cuales sirven de base para la construcción del marco propuesto.

Procedimiento de la Investigación.

Como punto de partida se define el propósito de la investigación la cual está enfocada en especificar las fases que debe seguir una auditoría de aplicaciones Android, haciendo énfasis en delimitar y clasificar las vulnerabilidades y generalizar la nomenclatura que debe ser utilizada. De la misma forma es importante establecer los atributos deseables de esta investigación, los cuales son: Confiable: para ello se buscará para la construcción de este marco de referencia metodologías reconocidas las cuales hayan sido comprobadas y validadas; Ágil: Se buscará simplificar el proceso de selección de vulnerabilidades y la manera de analizarlas.

A continuación, se realiza un análisis bibliométrico el cual ayude a cumplir los objetivos y atributos del marco de referencia, para ello se tomaron en cuenta las fuentes de información: Scopus, Google Académico y la Biblioteca digital de ACM para obtener una elección de trabajos de investigación que expongan conceptos y criterios acerca de: Fases de una auditoría en tecnologías de la información, gestión de riesgos en sistemas de información y metodologías enfocadas en la verificación de seguridad en aplicaciones móviles, todo esto con el propósito de establecer definiciones y procedimientos que serán utilizados para el desarrollo del marco de referencia.

Para obtener la información relevante que será utilizado para crear este marco, como primer paso se detalla en la Tabla 4 que aspectos se toman en cuenta para realizar la búsqueda de fuentes de información.

Tabla 4. Protocolo de búsqueda de información.

Protocolo de búsqueda de fuentes de información	
Lenguajes de Búsqueda:	Ingles - Español
Periodo:	2013 - 2018
Términos de Búsqueda:	**Individual:** Audit process, security, Risk Management, Android Applications, assessment methodology, information technology.
	Combinado: Búsqueda 1: audit Information. Búsqueda 2: Risk management methodology. Búsqueda 3: methodologies Vulnerability Android applications.
Recursos de información	SCOPUS, GOOGLE ACADEMICO, ACM DIGITAL LIBRARY
Estrategias de búsqueda:	Se realizaron tres búsquedas con combinaciones de diferentes palabras clave: • **Búsqueda 1:** "audit process" AND "Information technology " • **Búsqueda 2:** "methodology" AND "information security risk" AND "risk management" • **Búsqueda 3:** "assessment methodology" AND "Android application" **Los resultados fueron refinados sucesivamente considerando:** 1. Año de publicación: de 2013 a 2018 2. Relación de publicaciones con ciencias de la computacion e informática 3. Artículos relevantes enfocados en resolver el vacío de la investigación.

Fuente: Elaboración Propia.

Una vez señalado como se maneja el proceso de búsqueda, se presenta un Diagrama de flujo de la estrategia de refinamiento de búsqueda, para ello se ejecutó tres búsquedas: La primera se compone de los términos: *"Audit proccess information technology"*, la cual esta direccionado en obtener procesos para realizar una auditoría sobre tecnologías de la información; La segunda se basa en: "*methodology information security risk management"* y está enfocada en obtener maneras de gestionar los riesgos sobre sistemas de información, y la tercera se fundamenta en: *"assessment methodology Android application*", la cual está direccionada para obtener metodologías enfocadas en analizar fallos en aplicaciones Android, La figura 6 detalla el refinamiento de búsqueda realizada.

Figura 6: Diagrama de flujo de la estrategia de refinamiento de búsqueda.
Fuente: Elaboración Propia.

Para el proceso de depuración del diagrama se realizó una inspección de fuentes de información detallado en la Tabla 5 con el cual se realizó en primera instancia un filtrado de acuerdo a la fecha de publicación y la relevancia de los títulos para reducir la lista. Posteriormente se realizó una segunda reducción eliminando los artículos duplicados y por la relevancia del contenido de los resúmenes, dando como resultado una lista de 38 documentos. Estos artículos fueron analizados teniendo como resultado 12 artículos los cuales se ordenaron dependiendo de sus contribuciones para resolver el vacío de la investigación. El resultado fue una lista de tres documentos los cuales se utilizan para la construcción del marco de referencia propuesto.

En la Tabla 5 se presentan los protocolos de inspección de información para la selección de los artículos.

Tabla 5. Protocolo de inspección de información.

Protocolo de inspección de fuente de información	
Reglas de Selección:	El orden de selección es el siguiente: 1. Revisión del título 2. Inspección del resumen. 3. Si la información es relevante para el tema de investigación, el documento es revisado.
Criterio de exclusión:	El orden de inspección es el siguiente: 1. Información obsoleta 2. Información duplicada. 3.Información no relacionada con el tema.
Criterios de inclusión:	1. Información relevante y relacionada con el tema de investigación.

Fuente: Elaboración Propia.

A continuación, se presenta a detalle el resultado de las búsquedas realizadas cabe indicar que para ciertos casos se modificó el orden de las palabras clave para optimizar los resultados de las búsquedas.

Búsqueda 1.

Scopus: El primer filtro es el año desde 2013 hasta 2018 y se limita a estudios relacionados con la ingeniería obteniendo 44 artículos luego eliminando duplicados se obtienen 11 y verificando la relevancia y relación con el tema de investigación se seleccionan 5 documentos.
Google Académico: El primer filtro desde el año 2013 hasta 2018 se obtienen 26 artículos luego relacionando con el tema de investigación se seleccionan 6 artículos.
ACM: El primer filtro del año 2013 hasta 2018 se obtienen 2 artículos luego relacionando al tema de investigación se selecciona 1 artículo.

Búsqueda 2.

Scopus: El primer filtro es desde el año 2013 hasta 2018 donde se obtienen 34 investigaciones posteriormente relacionando con el tema de investigación se selecciona 4 documentos.
Google Académico: El primer filtro es desde el año 2013 hasta 2018 y tomado en cuenta los documentos enfocados a sistemas de información se consigue 32 artículos y

relacionando con el tema de investigación se seleccionan 8 artículos.
ACM: El primer filtro es desde el año 2013 hasta 2018 con un filtrado por documentos enfocados a sistemas de información se consiguen 23 artículos y luego se los revisa para obtener los que están enfocados al tema de investigación obteniendo 6 artículos.

Búsqueda 3.

Scopus: En el primer filtro del año 2013 hasta 2018 se obtienen 6 investigaciones luego relacionando con el tema de investigación se seleccionan 4 documentos.
Google Académico: El primer filtro es desde el año 2013 hasta 2018 se obtienen 32 documentos luego se filtra y se relaciona con el tema de investigación seleccionando 3 artículos.
ACM: El primer filtro es desde el año 2013 hasta 2018 se obtienen 12 artículos y relacionado al tema de investigación se obtiene 1 documento.

Como siguiente punto, se detalla en la Tabla 6 los artículos por año de publicación (2013 - 2018) los mismos son los que están relacionados al tema de investigación.

Tabla 6. Artículos por año de Publicación.

AÑO	BÚSQUEDA 1	BÚSQUEDA 2	BÚSQUEDA 3	TOTAL
2013	2	0	0	2
2014	1	3	2	6
2015	2	4	1	7
2016	2	5	2	9
2017	3	4	1	8
2018	2	2	2	6
TOTAL:	12	18	8	38

Fuente: Elaboración Propia.

En la Tabla 7 se detallan los artículos que luego de realizar el refinamiento de búsqueda y aplicando el protocolo de inspección de información son los tres artículos relacionados considerados como base para constituir el marco propuesto.

Tabla 7. Documentos seleccionados.

DOCUMENTO	AUTOR (ES)	RESUMEN
Innovation in the IT audit process	(Cooke, 2018)	Tiene como propósito proporcionar los pasos necesarios para desarrollar programas integrales de auditoría que documenten de manera clara y consistente los procedimientos que se utilizarán para probar los controles y recopilar datos de respaldo.
Experience in applying the analysis and risk management methodology called MAGERIT to identify threats.	(Vega, R.G., Arroyo, R., Yoo, S.G. 2017)	El presente trabajo se centra en compartir la experiencia en la aplicación de la metodología MAGERIT para identificar amenazas y vulnerabilidades que podrían materializarse en riesgos y afectar a la empresa.
Vulnerabilities Classification for Safe Development on Android. Journal of Information Systems Engineering and Management.	(Ferreira, R. L. D., dos Santos, A. F., & Choren, R. 2016)	Este estudio presenta una clasificación de vulnerabilidades, con el propósito de permitir la identificación de posibles puntos de fallo, permitiendo al desarrollador corregir las brechas identificadas.

Fuente: Elaboración Propia.

Criterios de selección de los artículos.

Para la selección del artículo elaborado por Cooke (2018) se buscaron auditorías enfocadas hacia las tecnologías de la información con la cual luego de realizar el filtrado fue seleccionado este trabajo debido a que está basado en una norma ampliamente probada como lo es la guía de auditoría y aseguramiento de sistemas de la información planteada por la *Information Systems Audit and Control Association* (Isaca) la cual se toma como base para cumplir uno de los objetivos de este marco

el cual es presentar las fases que debe seguir una auditoría en aplicaciones Android.

El trabajo desarrollado por Vega, R.G., Arroyo, R., Yoo, S.G. (2017) fue seleccionado luego de buscar investigaciones que se basen en metodologías para gestionar los riegos en tecnologías de la información, por lo cual se lo seleccionó dado que está basado en una metodología reconocida y probada como lo es la metodología de análisis y gestión de riesgos de los Sistemas de Información (Magerit), el mismo servirá de base para cumplir otro objeto del marco propuesto el cual es gestionar y clasificar los riesgos para generalizar nomenclaturas.

La investigación realizada por Ferreira, R. L. D., dos Santos, A. F., & Choren, R (2016) fue seleccionado debido a que está basada en una metodología reconocida y probada como lo es Owasp Mobile Security Project y a su vez su estudio expone las vulnerabilidades más relevantes de esta metodología las cuales serán tomadas como base para el desarrollo de este marco con el cual se cumplirá otro objeto que esta direccionado en identificar las vulnerabilidades que se debe analizar.

Adicionalmente con los trabajos seleccionados se cumple con los atributos deseables planteados dado que al estar basados en metodologías reconocidas y probadas internacionalmente se cumple con el atributo de confiablidad de los resultados que se obtengan, de la misma manera al compactar y clasificar las vulnerabilidades a ser analizadas y la manera que deben ser gestionadas hacen que se minimice los tiempos de desarrollo de la auditoría haciendo un marco de referencia ágil.

Estructura de la Auditoría.

Una vez detallados los criterios por los que fueron seleccionados los artículos se procede a establecer la estructura que llevará la auditoría, para ello. En primera instancia, seleccionando la investigación de Cooke (2018) el cual se basa en la guía de auditoría y aseguramiento de sistemas de la información planteada por Isaca se establece las tres fases de la auditoría que serán aplicadas en este marco: Planeación, Ejecución e Informe; Luego, en base al trabajo de investigación realizado por Ferreira, dos Santos, & Choren (2016) se obtienen las seis *v*ulnerabilidades más frecuentes en aplicaciones Android las cuales serán aplicadas en este trabajo.

Posteriormente, tomando como base el trabajo planteado por Ferreira, dos Santos, & Choren (2016) se indica el impacto negativo que pueden provocar cada vulnerabilidad; Una vez obtenida esta información basado en el trabajo de Vega, R.G., Arroyo, R., Yoo, S.G. (2017) se establece la clasificación de impacto y la probabilidad de ocurrencia para establecer la atención al riesgo. De igual manera basado en el mismo trabajo se estructura las contramedidas que están implementadas con el propósito de conocer y determinar qué tan efectivas son.

Posteriormente se estructura una lista de chequeo en la que se consideran las seis vulnerabilidades presentadas y los posibles fallos que deben ser analizados basado en el trabajo de Ferreira, dos Santos, & Choren (2016) el cual está basado en *Owasp Mobile Security Project*; a continuación, se presenta la nomenclatura para calificar la probabilidad que se materialice una amenazada y las contramedidas encontradas para esto se basa en el trabajo elaborado por Vega, R.G., Arroyo, R., Yoo, S.G. (2017) basado en la metodología de análisis y gestión de riesgos de los Sistemas de Información (Magerit), esto en conjunto dan lugar a la lista de chequeo el cual tiene como propósito facilitar al auditor la verificación y calificación de las fallas que pueden presentarse en una aplicación

Por último, Para validar el marco de referencia propuesto enfocado a una auditoría en aplicaciones Android se realiza un grupo focal con la participación de 5 profesionales los cuales brindan sus apreciaciones sobre la factibilidad de uso del marco propuesto. Dentro de un enfoque profesional los participantes tienen una experiencia de más de 12 años como

desarrolladores y auditores de software y alrededor de 8 años de experiencia en el desarrollo y testeo de aplicaciones móviles, así mismo entre sus perfiles profesionales poseen títulos como: Ingenieros de Sistemas y Magíster en gestión de proyectos.

Marco de referencia Planteado.

A continuación, se propone el marco de referencia para el desarrollo de una auditoría de seguridad informática en aplicaciones Android basado una revisión literaria exhaustiva.

Como primer aspecto se indica que para el proceso de auditoría se toma como base las fases de: Planeación, Ejecución e Informe que es expuesta por Cooke (2018) el cual basa su trabajo en la Guía de Auditoría y Aseguramiento de sistemas de la información de Isaca.

Fase Uno - Planeación.

Basados en un estudio previo de Cooke (2018) el cual señala que en esta etapa se debe determinar: Como primer paso el objetivo de la auditoría la cual identifique el propósito de la auditoría; Como siguiente punto se debe definir el alcance esto hace referencia en delimitar las actividades que serán incluida en la revisión, luego es importante realizar la planificación previa de la auditoría donde se establezca el equipo de trabajo.

Para finalizar esta fase es importante determinar los procedimientos iniciales, esto va direccionado en que en medida de lo posible se realice un acuerdo de trabajo y confidencialidad con el propósito de tener un respaldo formal del análisis que se realizará en la auditoría, esto se debe efectuar entre las partes involucradas dejando constancia de todo lo expuesto en los puntos anteriores y con ello evitar que existan disconformidades o que la auditoría no cumpla las expectativas esperadas (Cooke,2018).

Fase Dos - Ejecución.

Se detalla los fundamentos y pasos que se deben tomar en cuenta para realizar una auditoría de seguridad informática en aplicaciones Android. Cabe señalar que para esta fase se tomará como punto primordial las vulnerabilidades a ser analizadas debido a que en base a esto se desarrollarán los siguientes pasos.

Como primer paso en esta fase se exponen los fallos frecuentes dentro de una aplicación Android; para ello, de la investigación elaborada por Ferreira, dos Santos, & Choren (2016) los cuales tomaron como base la metodología de análisis de seguridad de aplicaciones móviles *Owasp Mobile Security Project* y determinaron los aspectos de seguridad definidos por la norma ISO/IEC 27002 que se infringirían si se explotara el riesgo y se usara un vector de ataque. Dando como resultado seis vulnerabilidades las cuales se presentan en la Tabla 8.

Tabla 8: Listado de Vulnerabilidades en Aplicaciones Android**.**

VULNERABILIDADES EN APLICACIONES ANDROID	
VULNERABILIDAD	**DESCRIPCIÓN**
Ingeniería Inversa	Visualización del código fuente de una aplicación
Almacenamiento deficiente de datos.	Exposición de datos confidenciales.
Comunicaciones inseguras.	Fallos en protocolos de comunicación.
Autenticación insegura.	Administración fallida de sesiones de ingreso.
Criptografía insuficiente	Errores de cifrado que exponen información confidencial.
Manipulación de Código	Errores de programación y uso equivocado de los permisos y características de la aplicación.

Fuente: Elaboración propia – (Ferreira, dos Santos, & Choren, 2016).

Posteriormente se formaliza el tratamiento de los riesgos (vulnerabilidades) para ello tomando como base el trabajo de: Ferreira, dos Santos, &

Choren (2016) los cuales para su investigación utilizaron la metodología para el análisis de vulnerabilidades de aplicaciones móviles *Owasp Mobile Security Project*, se obtiene la clasificación del impacto negativo que puede ocasionar cada vulnerabilidad con respecto al correcto funcionamiento de una aplicación estableciendo dos tipos de impactos: Moderado: La falla puede ocasionar un funcionamiento defectuoso de la aplicación y Crítico: La vulnerabilidad produce un fallo catastrófico poniendo en peligro la información que maneja la aplicación. A continuación, se presenta la Tabla 9 en la que se muestra de manera ascendente el impacto que puede provocar cada vulnerabilidad establecida.

Tabla 9: Impacto que causa las Vulnerabilidades**.**

IMPACTO DE LAS VULNERABILIDAES	
VULNERABILIDAD	**IMPACTO**
Ingeniería Inversa	MODERADO
Almacenamiento deficiente de datos.	CRÍTICO
Comunicaciones inseguras.	CRÍTICO
Autenticación insegura.	CRÍTICO
Criptografía insuficiente	CRÍTICO
Manipulación de Código	CRÍTICO

Fuente: Elaboración propia.

Como siguiente paso luego de señalar las vulnerabilidades que se deben analizar y el impacto que pueden ocasionar, tomado como base el estudio realizado por Vega, R.G., Arroyo, R., Yoo, S.G. (2017) los cuales para su investigación utilizaron la metodología de análisis y gestión de riesgos de los Sistemas de Información - Magerit para establecer de manera cualitativa la probabilidad de ocurrencia del riesgo con base en el impacto que puede ocasionar, se establece tres probabilidades de ocurrencia de las vulnerabilidades, las cuales son: baja, media y alta. Los cuales se presentan en la tabla 10.

Tabla 10: Clasificación del impacto y la probabilidad de ocurrencia.

Fuente: Elaboración propia – (Vega, R.G., Arroyo, R., Yoo, S.G, 2017).

A continuación, se plantea la atención al riesgo el cual toma como base la clasificación del impacto y la probabilidad de ocurrencia presentado en la Tabla 10 y con ello se presenta en la Tabla 11 el rango y probabilidad de ocurrencia de una amenaza.

Tabla 11: Atención de la Vulnerabilidad según su riesgo.

ATENCIÓN DEL RIESGO		
RANGO	**PROBABILIDAD**	**DESCRIPCIÓN**
1-2	BAJA	La amenaza esta en un rango aceptable, por lo que no genera un peligro considerable.
3-5	MEDIA	Se requiere una atención pronta, realizando una depuración completa del fallo.
6-9	ALTA	Se debe tomar acciones inmediatas para reducir la probabilidad que se materialice la amenaza.

Fuente: Elaboración propia – (Vega, R.G., Arroyo, R., Yoo, S.G, 2017)

Otro aspecto a tratar está enfocado en verificar si existen contramedidas implementadas y si son lo suficientemente efectivas para minimizar los fallos de seguridad en la aplicación, para esto se toma como base la investigación realizada por Vega, Arroyo, & Yoo (2017) los mismo que utilizan en su investigación la

metodología Magerit la cual permite categorizar las contramedidas en una auditoría, a continuación se presenta la Tabla 12 en la que se identifica el factor, nivel y estado en las cuales se pueden encontrar las contramedidas que ya están implementadas cuando se realiza una auditoría.

Tabla 12: Contramedidas encontradas.

CONTRAMEDIDAS ENCONTRADAS.			
FACTOR	**NIVEL**	**ESTADO**	**DESCRIPCIÓN**
0%	L0	**INEXISTENTE**	No existen contramedidas implementadas que reduzcan los fallos.
50%	L3	**MODERADO**	Dispone de salvaguardas básicas.
100%	L5	**ÓPTIMO**	Presenta salvaguardas actualizadas y configuradas correctamente.

Fuente: Elaboración propia.

Posteriormente se presenta como anexo la Tabla 13, para esto se toma como base la Tabla 7 y el trabajo previo desarrollado por Ferreira, dos Santos, & Choren (2016) los cuales utilizan la Metodología *Owasp Mobile Project* para exponer puntos que se deben verificar dentro de un análisis de vulnerabilidades, con lo cual se obtiene una lista de chequeo que contiene 32 ítems presentando el fallo que provoca y la configuración correcta que debería tener cada ítem recomendando el tipo de prueba que se puede realizar. Dentro de la misma Tabla 13 basándose en la Tabla 11 se exponen la probabilidad de que se materialice el fallo presentando las tres opciones específicas de selección (Baja, Media y Alta). Para finalizar, basándose en la Tabla 12 se expone la efectividad de las contramedidas que se encuentran ya implementadas en la aplicación, pudiendo escoger una de tres opciones (Inexistente, Moderado y Óptimo).

Fase Tres – Informe.

En esta fase de la auditoría basado en el estudio realizado por Cooke (2018 se recomienda reunir los requisitos que tendrá el informe, para esto se utiliza los resultados obtenidos en la Fase 2 y tomando como base la lista de chequeo de la Tabla 11 la cual ayudará a especificar los resultados. En la tabla 14 se presenta los fallos encontrados con los cuales se realizará el Reporte Borrador.

Tabla 14: Fallos encontrados.

FALLOS ENCONTRADOS.				
ID	**SITUACIÓN ENCONTRADA**	**CAUSA**	**CONTRAMEDIDA ENCONTRADA**	**RECOMENDACIÓN**
Identificador - Lista de Chequeo	Descripción de lo encontrado	Motivo que causo el fallo	Salvaguardas encontradas.	Sugerencias del auditor para minimizar el fallo

Fuente: Elaboración propia.

Posteriormente de la misma manera basado en Cooke (2018) es importante presentar el Informe Final para ello se basa en el reporte borrador y se detalla los fallos encontrados que se deben corregir por parte de la empresa y a su vez se presenta de ser el caso las recomendaciones pertinentes.

Resultados de la revisión Literaria.

Después del análisis exhaustivo, se detalla las contribuciones de cada documento, para ello en la Tabla 15 se establece la información con respecto a qué documentos proporcionan información para cada uno de las fases del marco propuesto.

Tabla 15. Contribuciones de los documentos para el desarrollo del Marco Propuesto.

DOCUMENTO	1.PLANIFICACIÓN	2. EJECUCIÓN	3. INFORME
Innovation in the IT audit process (Cooke, I. 2018)	X	X	X
Experience in applying the analysis and risk management methodology called MAGERIT to identify threats (Vega, R.G., Arroyo, R., Yoo, S.G. 2017)		X	X
Vulnerabilities Classification for Safe Development on Android. Journal of Information Systems Engineering and Management (Ferreira, R. L. D., dos Santos, A. F., & Choren, R. 2016)		X	X

Fuente: Elaboración propia.

ANÁLISIS DE RESULTADOS.

Para el análisis de resultados se conformó un grupo focal con cinco profesionales con los cuales se realizó un conversatorio para validar la fiabilidad del marco propuesto. Para un mejor análisis y clasificación de resultados se utilizó un instrumento de encuesta, para ello se tomó como base el elaborado por el Instituto de Auditores Internos por sus siglas en inglés (IIA) el cual divide las preguntas en 4 etapas (Planeación, Ejecución, Reporte y Desempeño del equipo de Auditoría). El mismo se lo orientó hacia el marco de referencia planteado con lo cual se lo segmenta a tres etapas debido que esta investigación consta de las fases (Planeación, ejecución e informe) de la misma manera se utiliza la escala Likert (1 - Totalmente en desacuerdo, 2 - En desacuerdo, 3 - Ni de acuerdo ni en desacuerdo, 4 - De acuerdo, 5 - Totalmente de acuerdo) para delimitar las posibles respuestas.

Posteriormente se seleccionan y modifican las preguntas con el propósito de que guarden relación con la investigación; quedando siete preguntas las cuales son: Ítem 1 = ¿ Está usted de acuerdo que en la fase de Planeación se sugiera determinar el objetivo, alcance y Procedimientos Iniciales de la auditoria?; Ítem 2 = ¿Considera relevantes las vulnerabilidades expuestas en este marco de referencia?; Ítem 3= ¿Considera adecuada la clasificación del impacto asignada a cada vulnerabilidad?; Ítem 4= ¿Le parece apropiado la clasificación del riesgo presentado en este trabajo?; Ítem 5= ¿Considera correcta la clasificación de las salvaguardas encontradas?, Ítem 6= ¿La lista de chequeo presentada facilita la búsqueda de vulnerabilidad y ayuda a la estandarización criterios de respuesta?, Ítem 7= ¿Considera que el marco de referencia propuesto para el desarrollo de una auditoría de seguridad informática en aplicaciones Android puede ser aplicado?.

Entre las consideraciones en la que los participantes se basaron para realizar la valoración de la encuesta está el cumplimiento de los atributos deseables de esta investigación los cuales son la confiablidad (Uso de metodologías reconocidas y probadas) y la agilidad (delimitar procesos de selección, clasificación y análisis de vulnerabilidades). Con ello quedó a criterio de cada participante la calificación de cada ítem.

Para la validación del instrumento de encuesta se aplica el Alfa de Cronbach.

Tabla 16. Consolidado del Instrumento de Encuesta.

ALFA DE CRONBACH								
ENCUESTA	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	TOTAL
1	5	5	4	4	4	4	5	31
2	5	4	4	4	4	4	4	29
3	4	5	5	4	4	5	5	32
4	4	4	4	3	3	5	4	27
5	4	4	4	3	3	4	4	26
VARIANZA	0,30	0,30	0,20	0,30	0,30	0,30	0,30	6,50

Fuente: Elaboración propia.

En la Tabla 16 se visualiza el consolidado de las respuestas de cada uno de los profesionales participantes del grupo focal en donde mediante la escala de Likert proporcionaron su criterio con respecto a las preguntas planteadas.

Con la información obtenida se calculó la confiablidad, para ello se tomaron: el número de Ítems – preguntas (“7”), la suma de la varianza de cada Ítem (“2.00”), se obtuvo la varianza total (“6.50”), estos datos aplicando la fórmula de Alfa Cronbach adjudico el valor: “0.81” con lo que consigue una confiabilidad alta en el instrumento de encuesta (Bojórquez, Lopez, Hernández, & Jiménez, 2013).

Para cerrar el análisis se presentan los criterios de los participantes acerca de la aplicabilidad del marco propuesto para el desarrollo para una auditoría de seguridad informática en aplicaciones. Obteniendo como resultado que el 100% de los participantes aprueban la utilización de este marco señalando que al tener como base estudios previos y al delimitarlas de la interpretación de cada analizador ayudaría en gran medida a normalizar resultados en las

auditorías en donde sea aplicado. En la tabla 17 se exponen los criterios de cada participante con respecto a la factibilidad de utilizar este marco de referencia.

Tabla 17. Criterio de los participantes.

Participante	Opinión
1	"En mi opinión considero que es factible aplicar este marco, una de sus ventajas más notables es que nos presenta una guía especifica de cómo se puede analizar las vulnerabilidades enmarcado en el impacto que pudiera tener."
2	"También considero la aplicabilidad de este marco ya que encierra los aspectos más relevantes del manejo de las vulnerabilidades. Un punto a mejorar seria que se especifique con más detalle la etapa de planeación, se pudiera detallar con cuadros lo que se indica, por el resto creo que este marco se puede aplicar sin problema."
3	"Considero que el factor diferenciador de este marco se basa en delimitar y proporcionar pautas para clasificar y tratar las vulnerabilidades, de manera que no existan mayores diferencias entre los resultados de diferentes analizadores, por lo que veo apropiada su aplicación."
4	"Pienso que es un marco bien direccionado y apropiado para el objetivo propuesto, aunque su enfoque se centra en el cómo se debe trata las vulnerabilidades también detalla las etapas que debe seguir una auditoría."
5	"En términos generales el marco maneja pautas claras de cómo manejar una auditoría de vulnerabilidades en aplicaciones, señalaría como un punto importante en este trabajo la delimitación que se hace en cuestión de la nomenclatura a ser utilizada. Un aspecto que se podría mejorar seria detallar un poco más las pruebas descritas en la lista de chequeo para el análisis de vulnerabilidades."

Fuente: Elaboración propia.

CONCLUSIONES.

En la actualidad existen metodologías que permiten a los auditores a realizar pruebas de seguridad sobre aplicaciones móviles en dispositivos Android los cuales son de mucha ayuda para conocer donde se producen los errores, adicionalmente algunos también presentan programas para ser utilizados los cuales ayudan en la depuración de las vulnerabilidades sirviendo como una guía para realizar un proceso de control de vulnerabilidades. Pero a su vez cuando se aplica cualquier metodología queda a criterio del auditor los procedimientos a seguir y que tan grave es un fallo encontrado siendo el razonamiento del analizador la base para tomar las acciones respectivas dentro de una empresa, el problema que se presenta es que las opiniones pueden variar de un auditor a otro.

Para minimizar este inconveniente se planteó este marco de referencia el cual fue elaborado tomando como base investigaciones previas seleccionadas luego de un análisis bibliométrico. Es así que, Como primer paso se plantearon tres fases de la auditoría: Planeación, Ejecución e Informe obtenidas del estudio realizado por Cooke (2018); La primera fase de Planeación está enfocado en delimitar y brindar pautas necesarias para iniciar de forma correcta y organizada una auditoría; En la segunda fase de Ejecución es donde se presenta en mayor medida el factor diferenciador que tiene este trabajo, para su elaboración se basó en las siguientes investigaciones obtenidas del análisis bibliométrico: Ferreira, dos Santos, & Choren (2016) y Vega, Arroyo, & Yoo (2017) con las cuales se desarrollaron diferentes pasos y procesos que sirven para manejar y delimitar las vulnerabilidades con el objetivo de minimizar las diferentes interpretaciones que pudiera provocar una misma vulnerabilidad; En la tercera fase de Informes se presenta de manera compacta como se deben presentar los resultados, basado en los trabajos realizados por Cooke (2018), Ferreira, dos Santos, & Choren (2016) y Vega, Arroyo, & Yoo (2017).

Al final se obtuvo como resultado el Marco de referencia para el desarrollo de una auditoría de seguridad informática en Aplicaciones Android, el cual cumplió con el objetivo de la investigación debido a que se presentaron fases específicas para realizar una auditoría definiendo: procesos, las vulnerabilidades a ser analizadas, delimitando la atención al riesgo mediante la clasificación del impacto y probabilidad de ocurrencia, así también la clasificación de las contramedidas encontradas; logrando con esto minimizar la diferencia de criterios que pudiera existir entre auditores.

La validación de este marco de referencia fue mediante la intervención de cinco profesionales con experiencia en el desarrollo y testeo de aplicaciones, que a través de un conversatorio mediante la técnica de grupo focal expusieron sus opiniones y sugerencias donde se pudo obtener información cualitativa que ayudo al análisis y validación de este marco.

Los resultados del estudio fueron obtenidos mediante un instrumento de encuesta la cual fue validada mediante la aplicación del Alfa de Cronbach con la que se obtuvo una confiabilidad alta del "0.81", así también con el conversatorio realizado los profesionales expusieron en base a su experiencia sus opiniones acerca de la aceptación del marco presentado considerado por el cien por ciento de los participantes como una alternativa viable a las diferentes maneras de realizar un análisis de vulnerabilidades en aplicaciones Android, debido a que su enfoque está direccionado a presentar procedimientos dentro de una auditoría y también a delimitar las diferentes interpretaciones que se pueden presentar entre auditores.

Una limitante encontrada durante el desarrollo de este trabajo fue la poca información encontrada en trabajos previos enfocadas a la calificación y procedimientos para la atención de vulnerabilidades encontradas en aplicaciones Android.

Como futuros trabajos se plantea: La aplicabilidad de este trabajo bajo la normativa de seguridad y legislación del país en el cual se utilice este marco de referencia. Otro trabajo podría enfocarse en la identificación y validación de las herramientas más idóneas para el desarrollo de cada proceso propuesto en este marco.

Referencias Bibliográficas.

Ali, S., & Green, P. (2012). Effective information technology (IT) governance mechanisms: An IT outsourcing perspective. *Information Systems Frontiers*, 179-193.

Areitio, J. (2008). *Seguridad de la informacion.* Madrid: Paraninfo.

Argudo, A., López, G., & Sánchez, F. (2017). Privacy vulnerability analysis for Android Applications. *Fourth International Conference on eDemocracy & eGovernment (ICEDEG).* Quito: IEEE.

Arroyo, M. (21 de 11 de 2016). *Hacking Etico.* Obtenido de OWASP Mobile Security Project: https://hacking-etico.com/2016/11/21/owasp-mobile-security-project/

Arxan Technologies, Inc. (2016). *Mobile App Protection Handbook.* Maryland.

Ataya, G. (2013). *Information Security, Risk Governance and Management Frameworks. An Overview of COBIT 5.* Brucelas.

Aumatell, C. (2003). *La Auditoría de la Información, componente clave de la gestión estratégica de la información.*

Bello, J. (2017). Cimentando conceptos para el desarrollo de "auditorías prácticas" de seguridad a aplicaciones móviles. *CACS Latin -Conceptos y elementos necesarios para el entendimiento de sistemas Android y IOs y auditar técnicamente una aplicación móvil.* San Jose.

Benaroch, M. (2002). *Managing Information Technology Investment Risk: A Real Options Perspective.*

Berthomé, P., Fécherolle, T., & Guilloteau, N. (2012). Repackaging Android Applications for Auditing Access to Private Data. *2012 Seventh International Conference on Availability, Reliability and Security.* Prague.

Bojórquez, J., Lopez, L., Hernández, M., & Jiménez, E. (2013). Utilización del alfa de Cronbach para validar la confiabilidad de un instrumento de medición de satisfacción del estudiante en el uso del software Minitab MISP. *Innovation in Engineering, Technology and Education for Competitiveness and Prosperity*, (págs. 14-16). Cancun.

Candau, J., & Amutio, M. (2015). *MAGERIT – Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información.* Madrid: administración electrónica.

Carter, J., & Thakur, M. (2017). *OWASP Mobile Security Project.* Obtenido de www.owasp.org/index.php/OWASP_Mobile_Security_Project

Chen, Q., Qian, Z., & Mao, M. (2014). Peeking into Your App without Actually Seeing it: UI State Inference and Novel Android Attacks. *USENIX Security Symposium*, (págs. 1037-1052). San Diego.

Cooke, I. (2018). Innovation in the IT Audit Process. *ISACA Journal*, 6-11.

Dhama, S. (2014). An Overview of Security Challenges of Android Apps Permissions. *International Journal of Information and Computation Technology.*, 373-380.

Echenique, J. (2001). *Auditoria Informatica.*

Ehrenreich, B., Acharya, S., & Marc, J. (2015). OWASP inspired mobile security. *IEEE International Conference on Bioinformatics and Biomedicine*, (págs. 782-784). Washington.

Enck, W., Octeau, D., & McDaniel, P. (2011). A Study of Android Application Security. *USENIX Security Symposium.*

Erreyes, D. (2017). *Metodología para la selección de herramientas eficientes y protocolos adecuados para mejorar la seguridad de los dispositivos móviles.* Cuenca.

Faysal, S., Syeda , A., & Anindya, I. (2017). Vulnerability detection in recent Android apps: An empirical study. *International Conference on Networking, Systems and Security, NSysS 2017*, (págs. 55-63).

Fernandez, A., & García, D. (2016). Complex vs. simple asset modeling approaches for information security risk assessment: Evaluation with MAGERIT methodology. *2016 6th International Conference on Innovative Computing Technology, INTECH 2016*, (págs. 542-549). Gijon.

Ferreira, R., dos Santos, A., & Choren, R. (2016). Vulnerabilities Classification for Safe

Development on Android. *Journal of Information Systems Engineering & Management*, 187-190.

Gao, J., Li, L., Kong, P., & Klein, J. (2018). On vulnerability evolution in Android apps. *40th International Conference on Software Engineering: Companion Proceeedings*, (págs. 276-277). Gothenburg, .

Garcia del Moral, M. (2016). *Malware en Android.* Madrid.

Gartner. (19 de 8 de 2016). Obtenido de http://www.gartner.com/newsroom/id/3415117

Hernández, M., Toro, C., & Vargas, A. (2015). *Guía para validar el nivel de seguridad de los permisos y uso de recursos de una aplicación móvil bajo plataformas Android.*

Isaca. (2012). *Guía de Auditoría y Aseguramiento de sistemas de la información.*

ISO 19011 . (2011). *Directrices para la auditoría de Sistemas de Gestión.*

Khanmohammadi, K., Rejali , M., & Hamou-Lhadj, A. (2015). Understanding the Service Life Cycle of Android Apps: An Exploratory Study. *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices* , (págs. 81-86). Denver.

Lu, L., Li, Z., Wu, Z., Lee, W., & Jiang, G. (2012). CHEX: statically vetting Android apps for component hijacking vulnerabilities. *ACM conference on Computer and communications security* , (págs. 229-240). Raleigh.

Martínez, A. (30 de 9 de 2014). *Certsi* . Obtenido de The Problem of Fragmentation on Mobile Platforms: https://www.certsi.es/en/blog/the-problem-of-fragmentation-on-mobile-platforms

Medianero, D., & Villagrá, V. (15 de 11 de 2017). *OASAM*. Obtenido de https://github.com/b66l/OASAM

Moreira, F., García, T., & Moreira, J. (2016). *Evaluación de protocolos de seguridad de las App de redes sociales en dispositivos móviles Android.*

Mullis, A. (2017). *How to install the Android SDK (Software Development Kit).*

NIST 800-30. (2002). *Risk Management Guide for Information Technology Systems.*

Nugroho, H. (2014). conceptual model of it governance for higher education based on cobit 5 framework. *Journal of Theoretical and Applied Information Technology*, 216-221.

OLACEFs. (2011). *Manual de Auditoría de Gestión a las Tecnologías de Información y Comunicaciones.*

Porter, W., & Burton, J. (1983). Auditoria un enfoque conceptual. Mexico: Limusa.

Purser, S. (2014). *A Practical Guide to Managing Information Security.*

Quirolgico, S., Voas, J., Karygiannis, T., Michael, C., & Scarfone, K. (2015). *Vetting the Security of Mobile Applications.*

Rajamäki, J., & Rajamäki, M. (2013). National security auditing criteria, KATAKRI: Leading auditor training and auditing process. *European Conference on Information Warfare and Security, ECCWS*, (págs. 217-223). Helsinki.

Ramirez, A., & Ortiz, Z. (2011). Gestión de Riesgos tecnológicos basadaen ISO 31000e ISO 27005y su aporte a lacontinuidad de negocios. *Red de Revistas Científicas de América Latina y el Caribe, España y Portugal*, 56-66.

Ramírez, L., Edward, G., & Cifuentes, Y. (2016). Analysis of Security Vulnerabilities for Mobile Health Applications. *Actas de Ingeniería*, 325-333.

Sanchez, E. (2016). Destripando Pokemon Go by Owasp. *Cyber Camp Cyber Security.*

Santos , M. (2012). Guia para la evaluacion de seguridad en un Sistema. *II Jornada Nacional de Seguridad Informática .*

Shinichi, M., & Kouichi, S. (2013). A proposal for the privacy leakage verification tool for android application developers. *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication, ICUIMC.*

Soleto, M., Torres, J., & Rivera, J. (2012). *Un proceso práctico de análisis de riesgos de activos de información.*

SounthiraraJ, D., Sahs, J., Greenwood, G., Lin, Z., & Khan, L. (2014). SMV-HUNTER: Large Scale, Automated Detection of SSL/TLS Man-in-the-Middle. *21st Annual Network and Distributed System Security Symposium.*

The OWASP Foundation. (27 de 4 de 2017). Recuperado el 12 de 10 de 2017, de https://www.owasp.org/index.php/OWASP_Mobile_Security_Project

The OWASP Foundation. (13 de 2 de 2017). Recuperado el 10 de 11 de 2017, de Mobile Top 10 2016-Top 10: https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

Tsai, W., Hsieh, C., Wang, C., & Chen, C. (2016). The impact of IT management process of COBIT 5 on internal control, information quality, and business value. *IEEE International Conference on Industrial Engineering and Engineering Management*, (págs. 631-634). Singapore.

Unuche, R., Sinitsyn, F., Parinov, D., & Liskin, A. (10 de 11 de 2017). *Kaspersky.* Obtenido de securelist: https://securelist.lat/it-threat-evolution-q3-2017-statistics/85714/

Vanegas, J. (2017). *Guia de auditoria basada en el análisis de riesgos a un centro de datos aplicando la metodología Magerit.*

Vega, R., Arroyo, R., & Yoo, S. (2017). Experience in applying the analysis and risk management methodology called MAGERIT to identify threats and vulnerabilities in an Agro-Industrial Company. *International Journal of Applied Engineering Research*, 6741-6750.

Weber, R. (1988). *Auditing conceptual foundations and practice.*

ANEXO.

Tabla 13: Lista de Chequeo para búsqueda de Vulnerabilidades.
Fuente: Elaboración propia.

Ingeniería Inversa								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
MODERADO	ID - 1	Nivel de Ofuscación de la Aplicación	Verificar la configuración del archivo AndroidManifest.xml y su nivel de ocultamiento de código	Visualización de código confidencial	Código visible no presenta información relevante.	Estático		
	ID - 2		Comprobar el valor asignado a minifyEnabled en la configuración del archivo *"build.gradle"*	Nivel bajo de Ofuscación	"TRUE"	Estático		
	ID - 3	Facilidad de Depuración de la Aplicación	Revisar dentro del archivo AndroidManifest.xml el valor asignado a la propiedad android:debuggable	Fuga de Datos involuntarios	"FALSE"	Estático		
	ID - 4		Verificar los componentes a los cuales brinda permisos de propiedad *"Uses permission"* dentro del archivo AndroidManifest.xml	Componentes pueden provocar accesos indebidos	Brindar permisos solo a componentes específicos	Estático		
	ID - 5		Revisar dentro del archivo AndroidManifest.xml la configuración de la opción *"exported"* de la *Activity*	Evasión de Autenticación (bypassing autenticación)	"FALSE"	Estático		

Manipulación de Código								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
CRÍTICO	Ck - 6	Validación de interacción de Aplicación	Buscar claves o *API-Keys* visibles en el código fuente.	Fuga de Datos (hardcoded Passwords)	No existen claves visibles en el código.	Estático		
	Ck - 7		Verificar si la aplicación está firmada por el desarrollador.	Inconsistencia de la originalidad de la aplicación	Permite comprobar la identidad del propietario	Estático / Dinámico		
	Ck - 8		Identificar que las entradas que interactúa el usuario estén validadas correctamente para evitar el ingreso de caracteres no deseados.	Inyección SQL	Los campos *"Input"* no permiten el ingreso de secuencia de caracteres	Estático / Dinámico		
	Ck - 9		Evidenciar que los componentes de terceros utilizados por la aplicación sean de fuentes seguras.	Intrusión de Malware.	Componentes externos son confiables.	Estático		
	Ck - 10	Proveedores de Contenido	Comprobar la configuración dentro de AndroidManifest.xml de la opción *"exported"* del *Content Provider.*	Fuga de datos por el proveedor de contenido	"FALSE"	Estático / Dinámico		

Almacenamiento deficiente de datos								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
CRÍTICO	Ck - 11	Preferencias Compartidas	Verificar las *Shared Preference* en archivos como *userinfo.xml*, y comprobar si usan opciones como "*mode_world_readable" y "mode_world_writeable"*	Visualización de Datos confidenciales	Uso exclusivo de la opción: "*Modo_Private"*	Estático / Dinámico		
	Ck - 12	Almacenamiento Interno	Verificar el uso de clases como "FileInputStream" - "FileOutputStream" y comprobar si usan opciones como "mode_world_readable" y "mode_world_writeable"	Exposición de datos a otras aplicaciones.	Uso exclusivo de la opción: "*Modo_Private"*	Estático / Dinámico		
	Ck - 13	Almacenamiento Externo	Verificar que la aplicación se instale en la ruta *"data/data/<package-name>/"*	Información sensible fuera del control de la aplicación una vez eliminada	Evitar instalaciones en SdCard	Estático / Dinámico		
	Ck - 14	Base de datos	Verificar en la ruta*"/data/data/<package-name>/databases/"* dentro de la propiedad SQLite Database el uso de contraseña de seguridad	Visualización en texto plano de la información de base de datos	Establecer una contraseña robusta para la interacción con la base de datos	Estático / Dinámico		
	Ck - 15	Respaldos	Verificar en el o los archivos de la ruta*"/data/apk/fichero. Apk"* no se guarde información confidencial	Visualización de Datos confidenciales	No guardar información confidencial en Backups.	Estático / Dinámico		

Comunicaciones Inseguras								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
CRÍTICO	Ck - 16	Seguridad de Protocolos	Verificar que protocolos de comunicación utiliza la aplicación	Visualización de Datos confidenciales	Manejar protocolos seguros Tls	Dinámico		
	Ck - 17	Seguridad de Webview	Comprobar el certificado del servidor WebView	Exposición de datos a otras aplicaciones.	Conexiones seguras de WebView hacia sitios remotos	Dinámico		
	Ck - 18	Conexiones de Host	Verificar nombre del host, buscar propiedades con "*HostnameVerifier NO_VERIFY*"	Permite la conexión de cualquier host	conexión a un nombre de host de confianza	Dinámico		
	Ck - 19	Bibliotecas	Comprobar que solo se interactúe con bibliotecas seguras.	Intrusión de Malware.	Uso de bibliotecas actualizadas.	Estático / Dinámico		
	Ck - 20	Aseguramiento de WebViews	Verificar configuración de la propiedad "setJavaScriptEnabled" de JavaScript dentro de los WebViews.	Ataques *Cross-Site Scripting* (XSS)	"FALSE"	Estático		
	Ck - 21		Comprobar que los WebViews brinde permisos solo a protocolos requeridos para su correcta comunicación.	Ataque Man in the Middle (MitM).	Protocolo HTTPS	Estático / Dinámico		

Criptografía insuficiente								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
CRÍTICO	Ck - 22	Algoritmo de Cifrado	Comprobar si la aplicación maneja algoritmos de cifrados obsoletos (DES, 3DES, RC2, RC4, MD4, MD5, SHA1).	Exposición de información confidencial	Uso de algoritmos de cifrado actualizados.	Estático / Dinámico		
	Ck - 23	Validación de longitud de claves	Verificar longitud de claves utilizadas en la aplicación.	Vulnerable a ataques de fuerza bruta	Longitud de clave superior a diez caracteres	Estático / Dinámico		
	Ck - 24	Algoritmos personalizados	Mencionar si existen métodos de cifrado realizados por los desarrolladores de la aplicación.	Exposición de información confidencial	Utilizar métodos de cifrado reconocidos internacionalmente	Estático / Dinámico		
	Ck - 25	Certificados confiables	De ser el caso verificar la configuración del Estándar de cifrado avanzado (AES)	Fallo en la integridad de los datos almacenados	Bloques de cifrado únicos y aleatorios mediante "*Cipher Block Chaining (CBC*)"	Estático / Dinámico		
	Ck – 26	Algoritmos de generación numérica	Identificar de ser el caso el tipo de algoritmo utilizado para generar números aleatorios como "*Dual_EC_DRBG*"	Exposición de los valores generados	Usar algoritmos de generación seguros.	Estático / Dinámico		

Autenticación insegura								
IMPACTO	**ID**	**DESCRIPCION**	**ANALIZAR**	**ERROR QUE PROVOCA**	**CONFIGURACION CORRECTA**	**TIPO PRUEBA**	**PROBABILIDAD (BAJA - MEDIA - ALTA)**	**SALVAGUARDAS IMPLEMENTADAS (LO - L3 - L5)**
CRÍTICO	Ck - 27	Conexiones Remotas	Verificar si existen autenticaciones remotas y si son realizadas con políticas de contraseñas.	Exposición de información confidencial	Configuraciones remotas de autenticación con políticas de seguridad	Estático / Dinámico		
	Ck - 28		Comprobar que desde una conexión remota se finalice sesión cuando desde la aplicación se realice la misma acción.	Exposición de información confidencial	Configurar cierre de sesión	Estático / Dinámico		
	Ck - 29		Verificar la existencia de un segundo factor de autenticación hacia conexiones remotas	Accesos indebidos a la información	Configurar un segundo factor de autenticación	Estático / Dinámico		
	Ck - 30	Contraseñas Seguras	Identificar la existencia de política de seguridad para las contraseñas.	Exposición de información confidencial	Aplicar de políticas para la robustez de contraseñas	Estático / Dinámico		
	Ck - 31	Actividad de Sesiones	Comprobar que las sesiones finalicen luego de un tiempo estimado de inactividad.	Accesos indebidos a la información	Configurar un tiempo máximo para el cierre de autenticación inactiva	Estático / Dinámico		
	Ck - 32	Factores de Autenticación	Verificar si existe un número máximo de intentos de autenticación.	Accesos indebidos a la información - Ataques de fuerza bruta/ diccionario	Limitar el intento de autenticaciones	Estático / Dinámico		