PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE ESMERALDAS

CARRERA:

INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

PREVIO AL GRADO ACADÉMICO DE INGENIERÍA DE TECNOLOGÍA DE LA INFORMACIÓN Y COMUNICACIÓN

TEMA DE INVESTIGACIÓN:

ANÁLISIS DE VULNERABILIDADES CRÍTICAS DEL SISTEMA OPERATIVO MÓVIL ANDROID MEDIANTE PENTESTING.

LÍNEA DE INVESTIGACIÓN:

ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE SOFTWARE

PREVIO A LA OBTENCIÓN DE TÍTULO DE:

INGENIERÍA DE TENOLOGÍAS DE LA INFORMACIÓN

AUTOR:

LADY LISETH VARGAS SANTANA

ASESOR:

MGT. JAIME SAYAGO HEREDIA

ESMERALDAS, 2023

TRIBUNAL DE GRADUACIÓN

Título: Mapeo sistemático de herramientas y plataformas para microservicios web.

Autor(a): Lady Liseth Vargas Santana

Mgt. Jaime Sayago Heredia f. _____

Asesor

Mgt. José Luis Carbajal f. _____

Lector #1

Mgt. Xavier Quiñonez Ku f. _____

Lector #2

Mgt. Xavier Quiñonez Ku f. _____

Coordinador de Carrera

AUTORÍA

Yo, Lady Liseth Vargas Santana con número de cédula de identidad 0850300948 manifiesto que mediante la presente investigación sobre el tema "ANÁLISIS DE VULNERABILIDADES CRÍTICAS DEL SISTEMA OPERATIVO MÓVIL ANDROID MEDIANTE PENTESTING." los resultados obtenidos como tesis de grado, previo a la obtención del título de "INGENIERO EN TECNOLGÍAS DE LA INFORMACIÓN" son de total responsabilidad del autor, y que se ha respetado las fuentes de información consultadas, realizando las citas correspondientes y los resultados alcanzados son totalmente legítimos. Al mismo tiempo declaro que todo el contenido incluyendo resultados, discusión, conclusiones, recomendaciones y otros efectos legales y académicos que se desglosan, son y serán exclusiva responsabilidad legal y académica del autor y de la PUCESE.

_____

Vargas Santana Lady Liseth

C.I 2100673546

AGRADECIMIENTOS

En primer lugar, deseo expresar mi agradecimiento a nuestro creador por la vida personal y profesional que he llevado.

De igual manera a mis padres, ustedes han sido siempre el motor que impulsa mis sueños y esperanzas, quienes estuvieron siempre a mi lado en los días y noches más difíciles durante mis horas de estudio. Siempre han sido mis mejores guías de vida. Hoy cuando concluyo mis estudios, les dedico a ustedes este logro amado padres, como una meta más conquistada. Orgullosa de haberlos elegido mis padres y que estén a mi lado en este momento tan importante.

A mi compañero de vida y esposo Oswaldo Leon por estar conmigo en todo este camino apoyándome y siempre creyendo en mí.

A mis docentes y tutor Mgt. Jaime Sayago, por sus virtudes, paciencia, y constancia a lo largo de este camino de conocimientos. Sus consejos fueron siempre útiles cuando no salían de mi pensamiento las ideas para escribir lo que hoy he logrado.

Mi amigo y compañero de viaje Adonis Benitez, no pudimos culminar hoy esto juntos, pero no puedo dejar de recordar cuantas tardes y horas de trabajo nos juntamos a lo largo de nuestra formación, por eso no puedo dejar de agradecerle por su apoyo y constancia, al estar en las horas más difíciles, por compartir horas de estudio. Gracias por estar siempre allí.

DEDICATORIA

Este trabajo es dedicado a mis padres por apoyarme y ayudarme a crecer personal y profesionalmente, por siempre pensar en mi regalándome lo mejor que pudieron que son los estudios. Gracias por su amor, por su sacrificio y por enseñarme a nunca rendirme ante los obstáculos de la vida.

Este logro es también suyo.

ÍNDICE DE CONTENIDOS

AUTORÍA III
AGRADECIMIENTOS IV
DEDICATORIA V
RESUMEN XI
ABSTRACT XII
INTRODUCCIÓN 13
Planteamiento del problema 13
Justificación 15
OBJETIVOS 16
General 16
Específicos 16
CAPÍTULO I: MARCO TEÓRICO 17
1.1 Bases teóricas – científicas 17
1.1.1 Características generales de los dispositivos móviles 17
1.1.2 Sistema operativo Android 17
1.1.3 ¿Qué es Android? 18
1.1.4 El inicio de Android 18
1.1.5 Características 20
1.1.6 Arquitectura del Sistema Operativo Android 21
1.1.7 Versiones de Android 26
1.1.8 Características y especificaciones actuales 40
1.1.9 Entorno de programación Android Studio 41
1.1.10 Requisitos del sistema 42
1.1.11 Pruebas 43
1.1.12 Clasificación de pruebas 43
1.1.13 Principales pruebas 45
1.1.14 Metodología de prueba de penetración 45
1.1.15 Pruebas de penetración 47
1.1.16 Fase de recopilación de información 48
1.1.17 Modelado de amenazas 49
1.1.18 Análisis de vulnerabilidad 50
1.1.19 Herramientas de análisis de vulnerabilidades 50
1.1.20 Explotación 51
1.1.21 Pentesting 52
1.1.22 Herramientas de pentesting 56
1.2 Antecedentes de la investigación 59
1.3 Fundamentación legal 61

CAPÍTULO II: METODOLOGÍA 63
2.1 Delimitación 63
2.2 Tipos de investigación 63
2.2 Métodos y técnicas 64
2.3 Técnicas de recolección de datos 64
2.4 Población y muestra 64
2.5 Descripción de instrumentos 65
2.6 Técnicas de procesamiento y análisis de datos 65
2.7 Normas éticas 66
CAPÍTULO III: RESULTADOS 67
3.1 Descripción del estudio 67
3.2 Instalación de Kali Linux 68
3.3 Instalación de Android Studio 69
3.4 Proceso del pentesting 70
3.4.1 Recolección de Información 70
3.4.2 Análisis de Vulnerabilidad 71
3.4.3 Análisis de riesgos 72
3.4.4 Fase de explotación de las vulnerabilidades 73
3.4.5 Reportes 75
CAPÍTULO IV: DISCUSIÓN 77
CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES 78
Conclusiones 78
Recomendaciones 78
REFERENCIAS BIBLIOGRÁFICAS 80
ANEXOS 84

ÍNDICE DE ILUSTRACIONES

Ilustración 13 Andy el robot, logo de la compañía Android Inc [33] 18
Ilustración 14 El HTC Dream, el primer teléfono inteligente en utilizar el sistema operativo Android [36] 20
Ilustración 15 Arquitectura por capa del sistema operativo Android 22
Ilustración 16 Funcionamiento Dalvik [41] 23
Ilustración 17 Inicio o Launcher en Android Lollipop [45] 26
Ilustración 18 Logo versión 1.0 Apple Pie [38] 27
Ilustración 19 Logo Versión 1.1 Banana Bread [46] 28
Ilustración 20 Logo versión 1.5 Cupcake [47] 29
Ilustración 21 Logo versión 1.6 Donut [48] 30
Ilustración 22 Logo versión 2.0 Eclair [49] 31
Ilustración 23 Logo versión 2.2 Froyo [50] 32
Ilustración 24 Logo versión 2.3 Gingerbread [38] 33
Ilustración 25 Logo versión 3.0 Honeycomb [51] 34
Ilustración 26 Logo versión 4.0 Ice Cream Sandwich [52] 35
Ilustración 27 Logo versión 4.1 Jelly Bean 3 [46] 36
Ilustración 28 Logo versión 4.4 Kit Kat [46] 37
Ilustración 29 Logo versión 5.0 Lollipop [38] 38
Ilustración 30 Logo versión 6.0 Marshmallow [55] 39
Ilustración 31 Porcentaje de versiones en dispositivos Android 40
Ilustración 32 Metodología de prueba de penetración 47
Ilustración 33 Pasos de modelado de amenazas [63] 49
Ilustración 34 Ejemplo de pivote [67] 51
Ilustración 35 Tipos de Pent Testing [70] 54
Ilustración 36 Características del dispositivo móvil 67
Ilustración 37 Características del sistema de la maquina 68
Ilustración 38 Características del almacenamiento de la maquina 68
Ilustración 39 Entorno de Kali Linux 69
Ilustración 40 Selección de SDK 69
Ilustración 41 Características de la virtualización del dispositivo móvil 70
Ilustración 42 Escaneo de servicios 71
Ilustración 43 Recopilación de información 72
Ilustración 44 Matriz de riesgo 73
Ilustración 45 Ataque robo de credenciales 74
Ilustración 46 Validación de tablas ip 74

Ilustración 47 Burp Siute Scan.....74
Ilustración 48 Configuracion de sniffer.....75
Ilustración 49 Vulnerabilidades encontradas.....76

ÍNDICE DE TABLAS

Tabla 1 Requisitos del sistema 42
Tabla 2 Características de dispositivo para la práctica 70
Tabla 3 Resultado de la prueba de penetración 75
Tabla 4 Tabla detallada de las vulnerabilidades 76
Tabla 5 Cronograma de actividades para el desarrollo de la investigación 84
Tabla 6 Modelo calidad en producto 84
Tabla 7 Modelo calidad de uso 84

RESUMEN

Los dispositivos móviles se han convertido en gran necesidad para las personas en ámbitos laborales y personales. El sistema operativo Android se ha convertido en el más utilizado al nivel mundial, por este motivo crecen de manera rápida millones de aplicación disponibles en la Play Store, esto provoca que los usuarios realicen varias descargas sin conocer los riegos que estas apps pueden traer.

El principal propósito del presente trabajo es la muestra de un diagnóstico de las vulnerabilidades que se presentan en los dispositivos Android mediante la penetración de pentesting. Para llevar a cabo dicha investigación fue necesario implementar una elección de herramientas para llevar a cabo las 5 fases de prueba. La evaluación de seguridad se llevó a cabo utilizando herramientas de software libre, ampliamente reconocidas en el entorno de seguridad informática y pentesting, sobre dispositivos móviles Smartphone físicos y otro usando emuladores.Se utilizó una metodología cualitativa, bibliográfica y descriptiva, el método analítico, análisis y la técnica para la recolección de datos fue SMS para la elección de las mejores herramientas para el proceso del pentesting. Obteniendo como conclusión que ninguna red es segura pero que se trata de poner trabas evitando la vulnerabilidad de estas buscando que sean lo más segura posible

Palabras claves: Pentesting, Seguridad, Android, Dispositivos Móviles, Herramientas.

ABSTRACT

Mobile devices have become a great necessity for people in work and personal environments. The Android operating system has become the most used worldwide, for this reason millions of applications available in the Play Store are growing rapidly, this causes users to make several downloads without knowing the risks that these apps can bring.

The main purpose of this work is to show a diagnosis of the vulnerabilities that are present in Android devices by means of penetration pentesting. To carry out this research it was necessary to implement a choice of tools to carry out the 5 phases of testing. The security assessment was carried out using free software tools, widely recognized in the computer security and pentesting environment, on physical Smartphone mobile devices and another using emulators.

Keywords: Pentesting, Security, Android, Mobile Devices, Tools.

INTRODUCCIÓN

Planteamiento del problema

Podría decirse que la capacidad de descargar una cantidad infinita de aplicaciones útiles es el aspecto más importante de los dispositivos inteligentes. Google es una gran corporación que creó el sistema operativo Android como una plataforma de código abierto que proporciona un sistema operativo, middleware y aplicaciones fundamentales para todos y cada uno de los dispositivos que pueden conectarse a Internet.

Esto permitió la creación de la ubicua PlayStore, desde la cual los usuarios de dispositivos con Android pueden descargar e instalar cualquier cantidad de aplicaciones, esto hace que los usuarios se vean obligados a registrarse para obtener una cuenta en Play Store; sin hacerlo, no tendrán acceso a las funciones de la plataforma y, en cambio, se verán obligados a descargar APK de Internet, una opción mucho más arriesgada. Como resultado, después de que los usuarios pasan por la molestia de registrarse para obtener una cuenta, descargan una serie de aplicaciones que almacenan datos confidenciales, dejándolos vulnerables al robo. Además, Android permite el desarrollo de aplicaciones vía SDK para desarrollar nuevas aplicaciones para la plataforma en Java, y no diferencia entre sus aplicaciones primarias y las nuevas aplicaciones desarrolladas con SDK [1].

Los teléfonos móviles actuales almacenan los datos biométricos de su propietario con el fin de proporcionar "seguridad" para el acceso a dispositivos móviles, sin embargo, los usuarios suelen pasar por alto el hecho de que otras personas pueden acceder a estos datos si existe una falla de seguridad. Las técnicas biométricas que miden la actividad cinética en las manos proporcionan un modelo del comportamiento de un usuario basado en sus interacciones con un dispositivo, incluida la tasa de pulsación, la velocidad de movimiento de los dedos y la orientación del dispositivo [2].

Basado en los datos más actualizados de la base de datos nacional de vulnerabilidades de EE. UU., TheBestVPN ha publicado un análisis que revela los sistemas operativos y las tecnologías más vulnerables en un período de 19 años, desde 1999 hasta 2019. Habiendo demostrado constantemente que Android está en segundo lugar con un total de 2.563 vulnerabilidades a lo largo de los años, está claro que este es un sistema muy riesgoso en varios sentidos. Además,

los datos presentados por TheBestVPN indican que Android ha liderado el grupo durante tres años consecutivos como el sistema operativo principal más vulnerable, en 2016, 2017 y 2019 [3].

El hecho de que necesite una conexión a Internet para completar cualquier tipo de descarga en un dispositivo móvil para uso del usuario significa que sus datos ya están en riesgo debido a múltiples puntos de entrada no seguros. Esto se debe a que Internet es una red en la que se almacena cualquier tipo de información que ingrese para que pueda recuperarla fácilmente. Las vulnerabilidades no se limitan a los dispositivos móviles; se pueden encontrar en automóviles, electrodomésticos y otros lugares donde se usa Android. Esto se debe a que cada dispositivo obtiene su propia capa de personalización cuando el sistema operativo se adapta para trabajar con él.

Entonces si es así, ¿Cómo mitigar este tipo de errores, ataque y vulnerabilidades dentro del sistema operativo Android?

Justificación

Las vulnerabilidades en un dispositivo móvil Android serán reveladas en la investigación presentada. A pesar del amplio conocimiento de la gente moderna sobre los dispositivos móviles, saben sorprendentemente poco sobre seguridad. Por esta razón, es crucial que las personas que usan estos dispositivos puedan detectar y evitar peligros potenciales. Esto se debe a que la forma en que este sistema operativo se implementa en diversos dispositivos tecnológicos ha cambiado con el tiempo, es decir, está evolucionando, ya que sus usuarios confían cada vez más en estos dispositivos tanto para sus necesidades personales como profesionales.

Por lo tanto, es crucial que todos los usuarios al menos estén al tanto de las amenazas o vulnerabilidades presentes en sus dispositivos con Android. Las personas podrían proteger mejor sus datos para que no sean espiados, tendrían una mejor comprensión de los tipos de ataques que enfrentarían y el daño que resultaría de ellos, y verían una instalación menos generalizada de software malicioso. Cuando un usuario desconoce los riesgos que plantea incluso un dispositivo móvil aparentemente inofensivo debido a sus vulnerabilidades inherentes, ese usuario corre el riesgo de que su seguridad y privacidad se vean comprometidas.

OBJETIVOS

General

Analizar los problemas de vulnerabilidad en un dispositivo móvil Android través del pentesting para conocer los riesgos al usuario.

Específicos

a) Revisar la literatura existente acerca del sistema operativo Android para identificar las ventajas y desventajas de cada versión.
b) Describir las mejores herramientas para el desarrollo de pentesting
c) Realizar un pentesting a dispositivos Smartphone con sistema operativo Android para determinar sus vulnerabilidades.
d) Describir las principales políticas de seguridad que ayuden a los usuarios tener una mejor protección en la seguridad de sus datos.

CAPÍTULO I: MARCO TEÓRICO

1.1 Bases teóricas – científicas

1.1.1 Características generales de los dispositivos móviles

Como se menciona anteriormente, la telefonía móvil ha ido aumentando aceleradamente, sobre todo por la llegada de los dispositivos móviles inteligentes "Smartphones" [3]. Cuando se habla de dispositivos móviles solo se piensa en teléfonos móviles y Smartphones, sin embargo, un dispositivo móvil puede ser cualquier dispositivo que cuente con las características [4]:

- El tamaño del sonido es pequeño.
- Capacidades especiales de procesamiento
- Conexión inalámbrica
- Memoria Interna y externa
- Interacción con pantalla o teclado
- Es empleado para una función principal contacto con otras funciones secundarias
- Uso individual

1.1.2 Sistema operativo Android

Los teléfonos móviles de hace diez años eran muy diferentes a lo que son hoy; se usaban principalmente para hacer y recibir llamadas telefónicas. Pero con el paso del tiempo, se agregaron nuevas funciones, como la capacidad de enviar y recibir mensajes de texto y correo electrónico y jugar juegos con gráficos más complejos. Sin embargo, estos teléfonos todavía no tienen su propio sistema operativo. Nokia fue uno de los primeros en adoptar el sistema operativo Symbian para sus teléfonos móviles, lo que causó revuelo debido a la amplia disponibilidad de aplicaciones útiles del sistema y su proceso de instalación relativamente simple. A pesar de su éxito comercial, Nokia no pudo innovar y Symbian finalmente se convirtió en un sistema operativo lento e ineficaz. A Symbian le siguieron otros sistemas operativos menos potentes [5].

Aunque Andy Rubin inicialmente desarrolló Android como un sistema operativo móvil en el año 2000, Google finalmente adquirió Android Inc. en 2005. El lanzamiento del primer iPhone de Apple en 2007 fue un momento decisivo, que alteró para siempre las concepciones de las personas sobre lo que un teléfono móvil podía y debía hacer. Desde el año posterior a la presentación del primer teléfono inteligente con Android, el HTC Dream, hasta el día de hoy, ambas plataformas han experimentado actualizaciones significativas que han culminado en el estado de Android como un sistema altamente estable, seguro, funcional, innovador y, quizás lo más importante, dispositivo indispensable. lo que es más importante, un ecosistema de desarrolladores distribuido globalmente [6].

1.1.3 ¿Qué es Android?

Es una plataforma de telefonía móvil gratuita basada en el sistema operativo GNU/Linux y la Licencia Pública GNU (GPL). Construido sobre una distribución modificada del sistema operativo central de Linux. Google y otros miembros de Open Handset Alliance trabajaron juntos para crear y lanzar Android. Incluso con la versión 10, el Plan de código abierto de Android (AOSP) está a cargo del mantenimiento y desarrollo de Android [6].

Por primera vez en diez años, Android superó a Symbian como la plataforma más popular para teléfonos inteligentes en el cuarto trimestre de 2010. El logotipo de Android es el robot llamado Andy.

Ilustración 1 Andy el robot, logo de la compañía Android Inc [33]

1.1.4 El inicio de Android

Android Inc. fue fundada en octubre de 2003 por los estadounidenses Andy Rubin, Rich Miner, Nick Sears y Chris White en Palo Alto. Una de las primeras contrataciones destacadas es Andy

McFadden, que ha trabajado en WebTV y MOXI, y Chris White, que ha trabajado en el diseño y la interfaz de WebTV antes de ayudar en el desarrollo de Android [7].

Los miembros fundadores de la empresa, incluido Rubin (cofundador de Danger Inc., más conocido por crear la línea de teléfonos inteligentes Sidekick para T-Mobile), Miner (cofundador de Wildfire Communications Inc. y vicepresidente de tecnología e innovación de Orange, una empresa de telecomunicaciones británica), entre las otras nuevas incorporaciones, todos tenían una amplia práctica en la industria inalámbrica. A pesar del evidente éxito de sus fundadores y primeros empleados, Android Inc. operaba en secreto y solo admitía que la empresa estaba desarrollando software para teléfonos móviles.

Cuando Android Inc. era todavía una pequeña empresa, Google la compró en agosto de 2005. Casi todo el personal de Android Inc. permaneció en la empresa adquirente después de la adquisición, y eso incluye a los fundadores originales de la empresa, Andy Rubin, Rich Miner, y Chris White. Aunque se sabía muy poco sobre el trabajo de Android Inc. en el momento de la adquisición, se supuso que Google tenía el propósito de ingresar al mercado de teléfonos móviles [8].

Cuando Rubin estaba en Google, supervisó el desarrollo de la plataforma móvil basada en Linux. Google lanzó una plataforma con estas características con la intención de brindar un sistema flexible y actualizable para fabricantes de dispositivos móviles y proveedores de servicios de red. Google ha señalado a los operadores de red que está abierto a diversos grados de cooperación y ha formado asociaciones con varias empresas de hardware y software [9].

En diciembre de 2006, aumentó la especulación de que pronto se lanzaría el sistema operativo móvil de Google, Android. Según informes de BBC32 y The Wall Street Journal, Google está trabajando activamente para llevar su plataforma de búsqueda y aplicaciones móviles a los teléfonos inteligentes. La noticia de que Google estaba desarrollando un producto llamado Google se difundió rápidamente en varias publicaciones impresas y en línea.

En 2007, se anunció la Open Handset Alliance 13 con la intención de crear estándares abiertos para dispositivos móviles. La participación de Google como miembro fundador de Open Handset Alliance convierte a Android en un sistema operativo gratuito y de código abierto. En 2008, se puso a disposición la primera versión estable del kit de desarrollo de software de

Android. En octubre de ese año, HTC comenzó a vender el G1, también conocido como HTC Dream. Corría sobre el sistema operativo Android desarrollado por Google y Open Handset Alliance (ver Ilustración 14). Cuenta con una pantalla táctil capacitiva de 3.2 pulgadas con una resolución de 320x480, un teclado QWERTY físico, una cámara de 3.15 megapíxeles, un procesador Qualcomm MSM7201A de 528 MHz, 192 MB de RAM y 256 MB de almacenamiento interno expandible a 16 GB a través de microSD.

Ilustración 2 El HTC Dream, el primer teléfono inteligente en utilizar el sistema operativo Android [36]

Había alrededor de 20 dispositivos diferentes en uso en 2009 que usaban Android como sistema operativo. Se lanzaron las versiones 1.5 Cupcake, 1.6 Donut y 2.0 Eclair. Android se convirtió en la segunda plataforma móvil más popular en 2010 con la inclusión de más de 60 dispositivos que ejecutan el sistema operativo. Lanzamiento de Froyo 2.2 [10].

1.1.5 Características

Debido al uso generalizado de los teléfonos inteligentes, han surgido varios sistemas operativos móviles (incluidos iOS, Windows Phone, Firefox OS, Blackberry, Ubuntu y otros). Cada uno de estos sistemas operativos tiene su propio conjunto de características que lo distinguen de la competencia. Aquí, se presentan algunas de las características que han ayudado a hacer de Android el sistema operativo móvil más utilizado en todo el mundo [11]:

1. En primer lugar, es una plataforma abierta, ya que el código fuente del Proyecto de código abierto de Android está disponible gratuitamente para que cualquiera lo use o modifique. Entorno de desarrollo de software de código abierto basado en Linux

2. El diseño de la interfaz de usuario (UI): el uso de XML para el diseño de la UI permite que las aplicaciones se ejecuten en dispositivos con diferentes resoluciones de pantalla y un código más limpio.
3. La tercera característica clave es que es compatible con Java y con una amplia variedad de tipos de archivos multimedia.
4. Gran cantidad de servicios, sensores y periféricos Permite a los desarrolladores crear programas con los que los usuarios pueden interactuar con el mundo que los rodea.
5. Quinto nivel de seguridad: la noción heredada de Linux de ejecutar programas en sus propios contenedores hace posible mantener los programas individuales aislados unos de otros. Además, cada aplicación tiene su propio conjunto de permisos que rigen su funcionalidad.
6. Seis, una plétora de aplicaciones prácticas.
7. Séptimo, está optimizado para baja potencia de procesamiento y memoria; Comenzando con Android 4.4 Kitkat, Google ha incluido nuevas funciones de ahorro de batería, incluida la máquina de realidad artificial ART, para reducir el consumo de dispositivos móviles.
8. SQLite se utiliza para el almacenamiento de datos. 8.
9. Imágenes y audio de alta calidad; estos juegos cuentan con gráficos 3D basados en OpenGL y animaciones vectoriales fluidas. Admite todos los códecs de audio y video estándar.
10. Hay muchos tipos de aplicaciones disponibles en la tienda Google Play.
11. Compatibilidad con HTML5, Adobe Flash y XHTML, versión 11.

1.1.6 Arquitectura del Sistema Operativo Android

Las partes de la aplicación se ejecutan en un tiempo de ejecución de objetos Java (JVM) que se compiló sobre la máquina virtual Dalvik (VM), utilizando las bibliotecas principales de Java. Algunas de las bibliotecas C estándar incluyen un administrador de interfaz gráfica (administrador de superficie), OpenCore, SQLite, la API OpenGL ES 2.0, Blink, el motor de gráficos SGL, el cifrado SSL y la biblioteca Bionic.

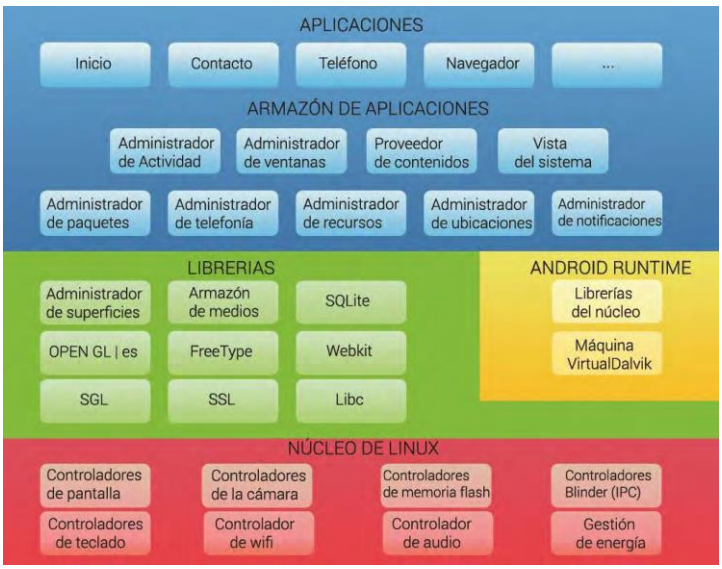

Ilustración 3 Arquitectura por capa del sistema operativo Android
Fuente: Reedición propia.

El sistema operativo contiene 12 000 000 líneas de código, incluidas 3 000 000 líneas de XML, 2 800 000 000 líneas de C y 2 100 000 000 líneas de Java [12].
Núcleo de Linux [13]:

- El sistema operativo subyacente de la plataforma es Linux 2.6. Usted es responsable de asegurarse de que el software y el hardware del teléfono inteligente sean compatibles entre sí. Las funciones principales, aunque no exclusivas, del dispositivo son:
- Gestión de memoria para todos los procesos y aplicaciones en ejecución.
- Multiproceso.
- Seguridad.
- Soporte de controlador de dispositivo: soporte de controladores para dispositivos.
- Gestión del tiempo de CPU utilizado por los programas y procesos en ejecución.
- Son los encargados de tener acceso a los periféricos del smartphone.
- Sirve como una capa de abstracción entre el hardware subyacente y el resto de la pila.

Android Runtime

Comparte el mismo concepto de "máquina virtual" que Java. Debido a las limitaciones de hardware en los dispositivos necesarios para ejecutar Android (poca RAM y una CPU lenta),

una máquina virtual Java estándar no era factible. Ante estas limitaciones, Google creó una nueva máquina virtual a la que llamaron Dalvik.

La máquina virtual Dalvik tiene algunas características que ayudan en la optimización de recursos. Diseñado para ejecutar archivos guardados en el formato ejecutable Dalvik de uso eficiente de la memoria (dex). Además, se basa en registros reales. Cada aplicación se ejecuta en su propio subproceso de Linux, completo con una copia de la máquina virtual Dalvik.

Algunas funciones se transfieren al kernel de Linux, como la gestión de procesos y la gestión de memoria de bajo nivel. Además de las bibliotecas estándar de Java, Android Runtime compila la gran mayoría de las bibliotecas disponibles para el lenguaje de programación Java en un solo paquete unificado [14].

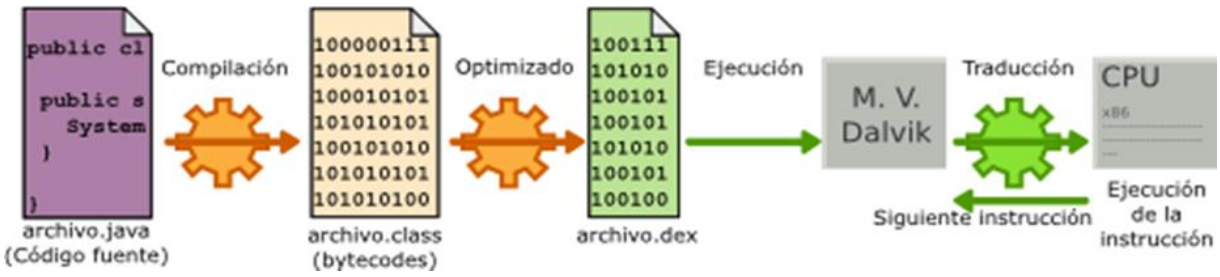

Ilustración 4 Funcionamiento Dalvik [41]

Librerías Nativas

Las bibliotecas nativas de Android (o bibliotecas) constituyen la siguiente capa de Android, que se encuentra sobre el kernel. Están escritos en C o C++ y compilados para su ejecución en la arquitectura de hardware única de un dispositivo móvil. En su mayor parte, los fabrica el fabricante y se instalan en el producto antes de que salga a la venta. El objetivo de las bibliotecas es agregar funcionalidad a las aplicaciones para tareas realizadas con frecuencia, evitando que los desarrolladores tengan que codificar estas tareas desde cero cada vez y asegurando que se completen de la manera "más eficiente" posible. Aquí hay una lista de libreros:

Biblioteca System C: un derivado de la biblioteca estándar BSD C (libc) que se ha adaptado para dispositivos Linux integrados [15]:

- Admite muchos formatos de archivos de audio/video, incluidos MPEG4, H.264, MP3, AAC, AMR, JPG y PNG; marco de medios basado en OpenCORE de Packet Video.
- Surface Administrator controla el acceso al subsistema de representación de gráficos en 2D y 3D.
- Admite un navegador web de última generación utilizado por el navegador Android y el visor Webview. Esta biblioteca es idéntica a la utilizada por los navegadores web Google Chrome y Safari de Apple.
- SGL es un motor de gráficos 2D.
- Las fuentes de renderizado vectorial y de mapa de bits también están disponibles en FreeType.
- SQLite es un sistema de gestión de bases de datos relacionales potente y ligero que se puede utilizar en cualquier aplicación.
- Secure Sockets Layer (SSL) es un servicio de cifrado que garantiza una conexión segura entre dos ordenadores.

Entorno de Aplicación

Teniendo en cuenta que se basa en las libertades individuales, esto no califica como una prioridad máxima. Aquí encontrará bibliotecas Java genéricas y específicas de Android. Dalvik, la máquina virtual de Android es una parte crucial del entorno de ejecución de Android. Las aplicaciones se codifican en Java y luego se compilan en un formato único para que la máquina virtual pueda ejecutarlas. Por lo tanto, las aplicaciones solo deben compilarse una vez antes de que estén listas para su distribución, y se garantiza que se ejecutarán en cualquier dispositivo Android que ejecute al menos la versión mínima del sistema operativo requerida por la aplicación [16].

Debe quedar claro que Dalvik es una versión de la máquina virtual de Java y, por lo tanto, es incompatible con el código de bytes de Java. Ni Java ni las aplicaciones de Java se pueden ejecutar en Android porque el kit de desarrollo de software (SDK) de Android crea ejecutables con la extensión dex, que es exclusiva de Dalvik.

Proporciona una plataforma de desarrollo gratuita para crear aplicaciones avanzadas (incluidos sensores, seguimiento de ubicación, descubrimiento de servicios, una barra de notificaciones

y más). Nombre alternativo para el kit de desarrollo de software de Java. Se están desarrollando arquitecturas para facilitar el reciclaje y la reutilización de materiales. Las capacidades de las aplicaciones pueden hacerse públicas y luego ser utilizadas por otras aplicaciones (sujetas a las restricciones de seguridad). Los usuarios pueden cambiar las piezas giratorias utilizando el mismo mecanismo.

El marco de aplicación de Android es un punto fuerte porque se basa en el popular lenguaje de programación Java. Aunque el SDK de Android aún no ofrece una compatibilidad completa con JRE, funciona con un subconjunto considerable de JRE. Los servicios más significativos que incluye son [17]:

- Vistas, las vistas forman la porción visual del todo.
- El acceso a los recursos no codificados está disponible a través del Administrador de recursos.
- Administrador de Actividades (Administrador de Actividades): Controla el ciclo de vida de las aplicaciones y proporciona una forma de cambiar entre ellas.
- El Administrador de notificaciones/Administrador de notificaciones permite que los programas muestren notificaciones personalizadas en la barra de estado.
- Proveedores de contenido: un mecanismo fácil de usar para acceder a datos de otras aplicaciones (como los contactos).

Aplicaciones

La capa final consta de todas las aplicaciones instaladas en el dispositivo, tanto aquellas con y sin interfaz de usuario, nativas y a nivel de sistema, escritas en C/C++ y Java, así como aquellas que el usuario instala por sí mismo.

La aplicación central del sistema se encuentra en este nivel. Inicio (Ilustración 17) es responsable de iniciar otras aplicaciones a través de un menú y muestra una variedad de espacios de trabajo en blanco donde los usuarios pueden agregar accesos directos a otras aplicaciones o widgets (también conocidas como "aplicaciones de inicio").

Android proporciona una configuración robusta para desarrollar aplicaciones que cumplan con todos los requisitos del sistema. Con Android, no hay funciones ocultas; en cambio, siempre

puede jugar con las aplicaciones de su teléfono para encontrar la mejor manera de hacer las cosas. La promesa de Android radica en la total libertad que permite a los usuarios personalizar sus teléfonos inteligentes según sus propias especificaciones [18].

Ilustración 5 Inicio o Launcher en Android Lollipop [45]
Fuente: Edición propia.

1.1.7 Versiones de Android

Desde que se lanzó la primera versión de Android el 23 de septiembre de 2008, los desarrolladores han lanzado constantemente actualizaciones que corrigen errores y agregan nuevas funciones en versiones posteriores. Esto ha ayudado a garantizar que el sistema no se atasque, como ocurría con las plataformas más antiguas que no recibían actualizaciones periódicas y, por lo tanto, se volvían lentas y obsoletas. Después de eso, se enumeran las muchas versiones lanzadas.

Versión 1.0 Apple Pie

La primera versión de Android se lanzó el 23 de septiembre de 2008 y el primer dispositivo móvil que presentó esta versión fue el HTC Dream (que se muestra en la Ilustración 18). Sin

embargo, esta versión carecía de muchas de las funciones que ahora son estándar en los teléfonos inteligentes, no se comercializó, entre sus principales características se encuentran [11]:

- Incorporación de la tienda de aplicaciones Android Market.
- Necesita un navegador web que pueda mostrar páginas web HTML y XHTML.
- El soporte para la cámara es mínimo. Producir nuevas alfombras o moquetas.
- Conéctese a servidores de correo basados en Internet que admitan el Protocolo de oficina de correos versión 4, el Protocolo de acceso a mensajes de Internet versión 4 y el Protocolo simple de transferencia de correo versión 5. Integración con otros servicios de Google, como Gmail, Calendario de Google y Contactos de Google. IM, servicio de mensajes cortos y servicio de mensajería multimedia.
- En otras palabras, soporte para fondos de escritorio y widgets.

Ilustración 6 Logo versión 1.0 Apple Pie [38]

Versión 1.1 Banana Bread

El 9 de febrero de 2009, se lanzó una actualización menor del HTC Dream, cuyo logotipo es un pan de plástico; corrigió varios errores en la versión anterior y se agregaron nuevas características [19]:

- Se agregaron detalles y reseñas de negocios y atracciones locales a Google Maps.
- Cambiar la pantalla de llamada, una pantalla diferente para cuando tienes manos libres, y la opción de mostrar u ocultar el teclado numérico.
- El almacenamiento de archivos adjuntos en los mensajes es una opción.

Ilustración 7 Logo Versión 1.1 Banana Bread [46]

Versión 1.5 Cupcake

Está basado en Linux 2.6.27 y fue lanzado el 30 de abril de 2009. Su logo es un poco pastel (Ilustración 20). La interfaz de usuario vio numerosas adiciones y actualizaciones, mejorando la usabilidad con las siguientes características [20]:

- Una cámara de video que puede grabar y reproducir clips.
- La capacidad de subir videos a YouTube e imágenes a Picasa directamente desde el teléfono.
- Compatibilidad con Bluetooth A2DP y AVRCP
- Capacidad de conexión automática para el emparejamiento de auriculares Bluetooth hasta un cierto rango.
- animaciones de transición de pantalla
- Predicción de texto y teclado QWERTY en pantalla.
- Gadgets de oficina o "widgets".
- Desktop Widget Software Development Kit (SDK) para desarrolladores externos.
- Una gama más amplia de funciones portapapel.

Ilustración 8 Logo versión 1.5 Cupcake [47]

Versión 1.6 Donut.

Publicado el 15 de septiembre de 2009. Utiliza el kernel Linux 2.6, su logo es una dona glaseada (Ilustración 21). En esta actualización se incluyó [21]:

- Actualización en Android Market Cámara, video y galería integrados Los usuarios ahora pueden elegir varias fotos a la vez para eliminarlas
- Búsqueda por voz mejorada que responde más rápido y está más estrechamente integrada con las aplicaciones nativas, como la opción de etiquetar contactos.
- Puede buscar marcadores, historiales, contactos y sitios web directamente desde la pantalla de inicio gracias a nuestra experiencia de búsqueda mejorada.
- Se agregó soporte para resoluciones de pantalla más altas (incluyendo WVGA) y velocidades más rápidas (especialmente en la cámara y aplicaciones de búsqueda), y ahora se admiten tecnologías más antiguas (incluyendo CDMA/EVDO, 802.1x y texto a voz).
- Marco de navegación gratuito paso a paso de Google GestureBuilder.

Ilustración 9 Logo versión 1.6 Donut [48]

Versión 2.0 / 2.1 Eclair

Basado en el kernel de Linux 2.6.29 liberado el 26 de octubre de 2009. Su logo es un pan alargado cubierto de chocolate (Ilustración 22). Los cambios incluyeron [22]:

- Velocidad de hardware optimizada
- Soporte para más tamaños de pantalla y resoluciones
- Interfaz de usuario renovada
- Nuevo interfaz de usuario en el navegador y soporte para HTML5
- Una mejor relación de contraste para los fondos
- Mejoras en Google Maps 3.1.2
- Soporte para Microsoft Exchange
- Soporte integrado de flash para la cámara
- Zoom digital
- MotionEvent mejorado para captura de eventos multi-touch
- Teclado virtual mejorado
- Bluetooth 2.1
- Fondos de pantalla animados.

El SDK 2.0.1 fue liberado el 3 de diciembre de 2009 y el SDK 2.1 fue liberado el 12 de enero de 2010.

Ilustración 10 Logo versión 2.0 Eclair [49]

Versión 2.2 Froyo.

Liberado el 20 de mayo del 2010, basado en el kernel de Linux 2.6.32. Su logo es un vaso con helado de yogurt (Ilustración 23). El nexus one fue el primer smartphone en contar con esta actualización, los cambios incluyeron [23]:

- Mejoras en el rendimiento general, el uso de la memoria y la velocidad de Android.
- Hay nueva galería de fotos ahora.
- El rendimiento de la aplicación se ha mejorado con la adición de JIT53.
- La aplicación del navegador se ha actualizado para ejecutarse en el motor JavaScript V8 de Google Chrome.
- Lanzador de aplicaciones mejorado que integra el navegador web y las funciones de un teléfono.
- Hay un punto de acceso WiFi y opciones de carga USB disponibles.
- Es posible desactivar el tráfico de datos basado en el operador.
- Grabación de voz Bluetooth y uso compartido de contactos.
- Pantallas para bloquear códigos e ingresar PIN; soporte de instalación de tarjetas de memoria; Soporte de tarjeta de memoria intercambiable.
- Se permiten pantallas de alta definición (como 720p).

Ilustración 11 Logo versión 2.2 Froyo [50]

Versión 2.3 Gingerbread

El 6 de diciembre de 2010, fue liberado. Basado en el kernel de Linux 2.6.35.7. Su logo es una galleta de jengibre con la forma de Andy el robot (Ilustración 24) Los cambios incluyeron [11]:

- Soporte para pantallas muy grandes de WXGA y resoluciones superiores;
- Compatibilidad con teléfono VoIP nativo (SIP); Diseño de interfaz de usuario nuevo y mejorado
- Soporte para reproducir videos WebM/VP8 y decodificar audio AAC
- Los nuevos efectos de audio incluyen reverberación, ecualización, simulación de auriculares y mejora de graves.
- Soporte para Tecnología NFC (NFC).
- La funcionalidad de cortar, copiar y pegar está disponible en todo el sistema.
- La Revisión de la Cobertura Multitextual (teclado).
- Mayor eficiencia en los procesos de entrada de datos, sonido y gráficos para los desarrolladores de juegos.
- Reunir varios elementos competitivos para impulsar el rendimiento.
- Soporte nativo para una gama más amplia de sensores (incluidos giroscopios y barómetros).
- Posibilidad de descargar archivos de gran tamaño con un gestor de descargas.

Ilustración 12 Logo versión 2.3 Gingerbread [38]

Versión 3.0 / 3.1 / 3.2 Honeycomb

Lanzada en febrero del 2011. Actualización únicamente para tablets, la Tablet Xoom de Motorola fue la primera en contar con esta actualización. Su logo es una abeja con características de Andy el robot (Ilustración 25), llamada panal de miel. Sus características son [24]:

- Compatibilidad con alta resolución y pantalla panorámica (WXGA y superior);
- Soporte nativo para teléfonos VoIP (SIP); Diseño de interfaz de usuario actualizado
- Ayuda para reproducir videos WebM/VP8 y decodificar audio AAC
- Algunos de los nuevos efectos de audio son reverberación, ecualización, simulación de auriculares y mejora de graves.
- Respaldamos la tecnología NFC (NFC).
- La capacidad de recortar, copiar y pegar está disponible en cualquier parte del sistema.
- Redescubrimiento de Textos Múltiples con Edits (teclado).
- Los procesos de entrada de datos, sonido y gráficos se han mejorado para los desarrolladores de juegos.
- Un reequilibrio de los factores competitivos para impulsar el rendimiento.
- Compatibilidad integrada con una gama más amplia de sensores (incluidos giroscopios y barómetros)
- La capacidad de descarga de archivos grandes del administrador de descargas.

Ilustración 13 Logo versión 3.0 Honeycomb [51]

Versión 4.0 Ice Cream Sandwich.

Interfaz estilo Honeycomb, en cualquier dispositivo, se busca la homogeneidad entre teléfonos, televisiones, tablets y netbooks. Su logo es un sándwich de helado con la forma de Andy el robot [25].

- "Nueva fuente tipográfica Roboto
- Interfaz Holo
- Sistema de gestión de notificaciones mejorado
- Multitarea mejorada
- Sugerencias y diccionarios para el teclado virtual
- Nuevo diseño y funcionalidades para la pantalla "Home"
- Android Beam, funcionalidad para transferir datos entre dos dispositivos vía NFC
- Función de desbloqueo mediante el rostro
- Nuevas funciones para la visualización y gestión del consumo de datos
- Nuevas aplicaciones de correo y calendario
- Herramienta integrada de captura de (botones de volumen y encendido simultáneamente)
- Soporte MKV
- Soporte Stylus (lápiz táctil)".

Ilustración 14 Logo versión 4.0 Ice Cream Sandwich [52]

Versión 4.1 Jelly Bean

El principal cambio de esta versión, lanzada en julio de 2012, es la eliminación de la compatibilidad con Flash Player de Adobe. Su logo es un recipiente con la forma de Andy el robot, el cual en su interior tiene grageas dulces. Rendimiento del sistema y gráfico mejorado gracias a Project Butter [26]:

- Lanzamiento del asistente de voz inteligente de Google, Google Now;
- Introducción del navegador web Google Chrome
- Motor de búsqueda superior basado en voz
- Revisitando el tipo de letra Roboto
- Mejores controles ortográficos y gramaticales para un trabajo grabado con voz.

Con el mes de noviembre llegó Android 4.2, una actualización menor que mantuvo el mismo nombre que su predecesor (Jelly Bean) [19].

- Nuevo panel de control
- Acceso a widgets y cámara fotográfica desde la pantalla de bloqueo
- Soporte para Miracast (función de streaming de vídeo y audio desde el terminal)
- Soporte para varios perfiles de usuario
- Photosphere, captura de fotografías panorámicas de 360º
- Gestual Mode para personas invidentes

El 24 de julio del 2013 Google anuncia Android 4.3:

- Soporte multilingüe y de perfil actualizado;
- Compatibilidad con OpenGL ES 3.0;
- Compatibilidad con TRIM;
- servicios mejorados de posicionamiento Wi-Fi.

Ilustración 15 Logo versión 4.1 Jelly Bean 3 [46]

Versión 4.4 Kit Kat

Fecha de publicación inicial: 31 de octubre de 2013 KitKat mejora a su predecesor, Android 4.4, al presentar nuevas funciones y corregir su defecto más evidente: la plétora de versiones diferentes. Para la versión en español de SO, Google ha llegado a un acuerdo con Nestlé para utilizar el nombre de uno de los productos de Nestlé. Su logo es un KitKat con la forma de Andy el robot [19].

- Reversión de requisitos de hardware para solucionar problemas de versiones
- Aceptable para su uso en máquinas con 512 MB de RAM o más
- La optimización de los sensores permite reducir el consumo de energía.
- Se incluyen los servicios integrados de almacenamiento en la nube de Google Drive, Box y QuickOffice.
- Soporte de receptor de infrarrojos. Hace uso del dispositivo móvil como control remoto de TV.

- Compatibilidad con el perfil de dispositivo de interfaz humana Bluetooth a través del GATT y el perfil de acceso a mensajes Bluetooth
- Grabación de pantalla en formato de vídeo.

Al mismo tiempo que se anunció el nuevo Nexus 5, Google también presentó KitKat 4.4 para sus otros teléfonos inteligentes que fueron desarrollados conjuntamente con LG y se basaron en el modelo G2 de gama alta. En 2014, Android amplía su alcance al hacerse compatible con una amplia variedad de dispositivos. Nexus Player, Android Auto, Android Wear y otros dispositivos de entretenimiento. (Ilustración 28)

Ilustración 16 Logo versión 4.4 Kit Kat [46]

Versión 5.0 Lollipop

La versión 5.0 de Lollipop se lanzó el 12 de noviembre de 2014 y una de sus características más notables es la incorporación de Material Design, un nuevo lenguaje de diseño que estandarizará la experiencia del usuario en todos los dispositivos. El logo es Andy sosteniendo una paleta de dulce [27].

- La próxima iteración de Material Design apunta a un flujo de trabajo más fluido.
- Una interfaz flexible que se ajusta al tamaño de pantalla de cualquier dispositivo
- Sistema de notificación inteligente rediseñado
- Mejora de la eficiencia energética.
- En español: Función Smart Lock para Dispositivos Android.
- Al Modo Invitado.

- ART (Android RunTime) es ahora el entorno de ejecución de aplicaciones predeterminado.
- Soporte para sistemas operativos de 64 bits.

La actualización 5.1 Lollipop se lanzó el 9 de marzo de 2015. La administración mejorada de la batería significa tiempos de ejecución más prolongados sin tener que recargar y un mejor rendimiento de la CPU en general. Otra mejora importante que trae esta actualización es el soporte nativo para múltiples tarjetas SIM, así como un nuevo sistema antirrobo [11].

- Después de una breve ausencia en la versión 5.0, se agregó un modo silencioso.
- Gestión mejorada de la memoria de acceso aleatorio (RAM).
- Soluciona el problema de las aplicaciones que se cierran inesperadamente.
- Mejor gestión de la batería.
- Se solucionó el problema con el uso excesivo del ancho de banda de la red WiFi.

Ilustración 17 Logo versión 5.0 Lollipop [38]

Versión 6.0 Marshmallow

Lanzado el 5 de octubre de 2015, es compatible con Nexus 5, Nexus 6, Nexus 7 y Nexus. Finalmente, con Marshmallow, se redujo drásticamente el consumo de batería del teléfono, incluso cuando no estaba en uso. El logo de esta versión es Andy sosteniendo un malvavisco (Ilustración 30). Entre las características principales se encuentran las siguientes [28]:

- La batería se ha actualizado.
- Para administrar la RAM, encontrará opciones relevantes en el menú de administración de RAM.
- Permite una gestión completa de volumen, medios y alarmas.
- Ayuda para la identificación de huellas dactilares.
- Android Pay basado en NFC y Host Card Emulation.
- Reenviar a la nube a través de Google Drive.
- Incluye un "Menú de Impresión"
- Cambia el menú de configuración en una nueva dirección. - Menú de ajuste alternado.
- Mejoras en la conservación de energía.
- Permite el acceso a Google Now desde la pantalla de bloqueo.
- Tenemos el especial "Función Now On Tap".
- Eliminar los programas de software de oficina.
- El Administrador de archivos permite a los usuarios buscar, copiar, compartir, filtrar y eliminar archivos.

Ilustración 18 Logo versión 6.0 Marshmallow [55]

Los desarrolladores de aplicaciones móviles deben familiarizarse con los niveles de API disponibles antes de comenzar a trabajar en una aplicación. Si están interesados en usar Android Beam, deben saber que la transferencia de datos a través de Android Beam se introdujo en Android 4.0 (Ice Cream Sandwich). Conocer el porcentaje de usuarios en cada versión y nivel de API es esencial para asegurarse de que la aplicación tenga un amplio alcance y se pueda usar en una variedad de dispositivos. (Ilustración 31)

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.6%
4.1.x	Jelly Bean	16	6.0%
4.2.x		17	8.3%
4.3		18	2.4%
4.4	KitKat	19	29.2%
5.0	Lollipop	21	14.1%
5.1		22	21.4%
6.0	Marshmallow	23	15.2%

Datos recopilados durante un período de 7 días hasta el 1 de agosto de 2016.
No se muestran versiones con una distribución inferior al 0,1%.

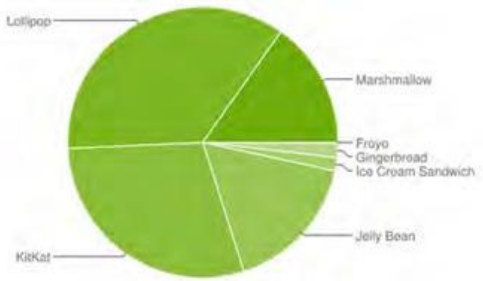

Ilustración 19 Porcentaje de versiones en dispositivos Android
Fuente: Extraída de https://www.android.com/.

1.1.8 Características y especificaciones actuales

Desde el lanzamiento de la primera versión de Android Apple Pie, cada vez es más común que los teléfonos inteligentes tengan biseles más delgados, pantallas más grandes y otras mejoras de hardware. Mejores sensores, cámaras, podómetros y software de mayor calidad: mejoras en la funcionalidad en todos los ámbitos, incluida la extensión de la duración de la batería en Android 6.0 Marshmallow.

Firebase Cloud Messaging ahora es una función del servicio de notificaciones automáticas de Android, que se une al almacenamiento de datos basado en SQLite, tecnologías de conectividad 3G y 4G, mensajes de texto SMS y MMS, y más [29].

El navegador web de Android se basa en el renderizador de código abierto WebKit y está optimizado para su uso con el motor JavaScript V8 que se encuentra en Google Chrome. Este navegador obtiene una puntuación de 93 sobre 100 en la evaluación de Acid3.

Soporte para Java a pesar de que la plataforma no incluye una máquina virtual Java (JVM) para ejecutar programas Java. El código no se está ejecutando en este momento. Todo está integrado

en un Dalvik ejecutable y se ejecuta en una máquina Dalvik virtual. Es una máquina virtual específica de Android construida a partir de dispositivos móviles que funcionan con batería y hardware de bajo consumo. Las aplicaciones de terceros, como J2ME MIDP Runner, se pueden actualizar para incluir compatibilidad con J2ME. Soporte completo de descarga progresiva RTP/RTSP y HTML5 (3GPP PSS, ISMA) [19].

1.1.9 Entorno de programación Android Studio

Tener un entorno de programación completo que le permita manipular elementos básicos y no tan básicos como botones, etiquetas, diseños y controles de zoom es esencial para desarrollar aplicaciones móviles, al igual que tener la capacidad de purgar y ejecutar esas aplicaciones en una variedad de Dispositivos físicos y digitales.

Netbeans, Motodev, Eclipse y Android Studio son solo algunos de los fantásticos IDE disponibles en la actualidad para el desarrollo de Android. Eclipse fue ampliamente utilizado por los desarrolladores de aplicaciones de Android porque sus extensiones y bibliotecas se podían instalar fácilmente, creando un entorno de desarrollo productivo. El enfoque de esta tesis será Android Studio porque este entorno de desarrollo creado por Google a menudo se considera la mejor opción.

El 16 de mayo en la edición 2013 de Google/IO, Google anunció un nuevo IDE: Android Studio, en su versión beta 0.1. Luego de que se le hicieran varias mejoras, el 8 de diciembre de ese año, Google lanzó la versión estable 1.0, anunciando lo que se convertiría en el entorno de desarrollo oficial de Android al incluir nuevas funciones como el soporte para Android Wear [30]:

- Para ayudar con la creación de aplicaciones Android Wear.
- Herramientas Lint, que detectan código incompatible con la arquitectura o código que el compilador no puede entender.
- Al exportar a APK, usar ProGuard para optimizar y comprimir el código del proyecto es una excelente opción para dispositivos de gama baja con limitaciones de memoria.
- Como herramienta de gestión y automatización de proyectos, Gradle facilita las pruebas, la compilación y el empaquetado.

- Se ha agregado una interfaz de desarrollo de Android adicional.
- En otras palabras, puede importar proyectos de Eclipse que empleen el compilador ANT en lugar de Gradle de Android Studio.
- Acceder a un repositorio (en un formato como Mercurial, Git, Github o Subversion) permite el control de versiones.
- Advertencias en tiempo real sobre problemas sintácticos, de compatibilidad y de rendimiento antes de la compilación de la aplicación.
- Vista previa en una variedad de dispositivos y resoluciones.
- Compatibilidad con Google Cloud Platform, lo que permite el uso de muchos servicios basados en la nube de Google.
- Un editor de diseño que le muestra una vista previa de su trabajo en tiempo real a medida que realiza cambios en el archivo xml de origen.

1.1.10 Requisitos del sistema

Ya sea que esté usando Linux, Mac OS X o Windows, querrá asegurarse de que su computadora cumpla con los requisitos mínimos para Android Studio (consulte la Tabla 1) antes de instalarlo.

Tabla 1 Requisitos del sistema

Linux	Mac Os	Windows
Escritorio GNOME o KDE, la librería GNU C glibc en su versión 2.11 o superior.	Mac OS X 10.8.5 o superior	Microsoft Windows 8/7/Vista/2003 (32 o 64 bit)
Mínimo 2GB de RAM como mínimo		
400 MB de espacio libre en el disco duro		
1GB extra para Android SDK, Imágenes del sistema para emulador, etc.).		
Resolución de pantalla de 1280×800 o superior.		
Oracle Java Development Kit (JDK) 7 o superior.		

Fuente: Elaboración propia

1.1.11 Pruebas

Las pruebas de software son un proceso de ejecución de un programa o aplicación con la intención de encontrar los errores del software. También se puede definir como el proceso de validación y verificación de que un programa o aplicación o producto:

- Cumple los requisitos empresariales y técnicos que guiaron su diseño y desarrollo.
- Funciona según lo esperado
- Puede implementarse con las mismas características.

Las pruebas y la revisión de las aplicaciones son diferentes del análisis y el desarrollo de estas. Con esto se quiere decir que se está construyendo o desarrollando aplicaciones y se está trabajando positivamente para resolver los problemas durante el proceso de desarrollo y hacer que el producto se ajuste a las especificaciones del usuario.

Sin embargo, al probar o revisar un producto buscamos los defectos o fallos de este. Por lo tanto, la creación de software requiere una mentalidad diferente a la de las pruebas de software.

1.1.12 Clasificación de pruebas

Las pruebas de software deben realizarse en distintos niveles del desarrollo de software con un objetivo específico en cada nivel. Hay una gran variedad de tipos de pruebas de software para comprobar las distintas características del software elegido en función del escenario del proyecto. Sin embargo, las pruebas de software pueden ser manuales o automatizadas.

Los tipos de pruebas de software pueden clasificarse en pruebas estáticas y pruebas dinámicas. En las pruebas estáticas, se revisan los documentos del proyecto de software para identificar los errores. La revisión puede ser informal, formal o de inspección, técnica o durante una reunión.

Un analista de control de calidad puede realizar una revisión informal en cualquier momento del proyecto. Es una forma poco costosa de realizar pruebas y los beneficios dependen de la persona que realiza las pruebas de software.
La revisión formal o inspección la planifica y controla el moderador (un jefe de pruebas puede

desempeñar el papel de moderador). Los analistas de control de calidad revisan los documentos del proyecto de software asignados antes de la reunión de revisión. Durante la reunión de revisión, se debaten los errores y el escriba (función asignada a uno de los evaluadores durante la reunión de revisión) documenta el debate. Los revisores entregan el informe de revisión al moderador al final de la reunión de revisión.

En las pruebas dinámicas, el software se prueba durante la ejecución. Las pruebas dinámicas pueden ser de caja blanca o de caja negra. Los dos tipos de pruebas de caja blanca son las pruebas unitarias y las pruebas de integración. Las pruebas de caja blanca también se denominan pruebas estructurales. La prueba unitaria o de componentes consiste en probar cada componente del programa de forma aislada. La prueba de integración consiste en probar las interfaces entre los componentes del programa.

Los desarrolladores realizan las pruebas de caja blanca utilizando herramientas del entorno de desarrollo. El otro tipo de pruebas dinámicas son las pruebas de caja negra, también denominadas pruebas basadas en especificaciones. Las pruebas del sistema y las pruebas de aceptación del usuario (UAT) se clasifican dentro de las pruebas de caja negra. Las pruebas del sistema sobre el software las realiza un equipo independiente de analistas de control de calidad para establecer la calidad del software. El equipo de control de calidad comprueba tanto las características funcionales como las no funcionales del software.

Las pruebas del sistema se realizan con distintas herramientas de pruebas de software en el entorno de pruebas. Tras las pruebas del sistema, los usuarios pueden comprobar la usabilidad del software en las pruebas de aceptación del usuario. Los analistas de control de calidad pueden ayudar a los usuarios en las pruebas de aceptación del usuario. Los usuarios pueden realizar la UAT en el entorno de TI y se denomina prueba alfa. Si los usuarios prueban el software en un entorno real o de preproducción, se denomina prueba beta.

1.1.13 Principales pruebas

Planificación y control de pruebas

Es un proceso fundamental que define el objetivo y la meta del proceso de pruebas.
Utilizando la especificación de requisitos, el plan de pruebas será más detallado. Esto es un proceso continuo y se realiza en todos los ciclos de vida. Principalmente se programa el análisis de pruebas y el proceso de diseño.

Implementación y ejecución de pruebas

Es un proceso de prueba en el que se realiza el trabajo real en que se ejecutan casos de prueba con datos de prueba, ejecutando los procedimientos de prueba utilizando herramientas de ejecución de pruebas y verificando la n Notificación de errores y creación de informes de incidencias.

1.1.14 Metodología de prueba de penetración

Una prueba de penetración es similar a cualquier otra prueba en que sigue una cierta metodología para verificar la eficiencia y confiabilidad del sistema operativo. Esto a menudo comienza con la identificación de la información disponible públicamente, como el sistema operativo que se usa, la versión del software que se usa, los parches y módulos actualmente habilitados y, a veces, incluso alguna información interna, como la estructura del sistema de archivos o la dirección IP.

Una vez que el evaluador tiene una teoría sobre qué software podría estar ejecutándose en el objetivo, debe confirmar esa teoría. La información disponible para el probador se puede combinar y comparar con vulnerabilidades conocidas, y luego esas vulnerabilidades se pueden probar para ver si los resultados respaldan o contradicen la información recopilada previamente [31].

En la mayoría de los casos, las pruebas de penetración comienzan antes de que se firme un contrato, con una discusión de por qué es necesaria una prueba de penetración para el buen funcionamiento del sistema de información. El valor de los resultados depende de la

minuciosidad con la que el auditor defina el alcance de la prueba, los sistemas que pretende auditar y cualquier parámetro relevante que pueda encontrarse a lo largo del examen.

Una vez que el pentester ha decidido el alcance, el formato del informe y otros detalles, la prueba real puede comenzar en la ubicación del pentester o en las instalaciones del cliente. La primera etapa es recopilar la mayor cantidad de datos posible sobre el cliente y el sistema. Un hacker ético, o pentester, buscará en Internet cualquier información disponible públicamente sobre el negocio de su objetivo y buscará cualquier posible punto de entrada a sus redes. La siguiente es la fase de modelado de amenazas, durante la cual se calcula el valor de la amenaza utilizando los datos acumulados de la primera fase. Todos los descubrimientos se analizan en esta etapa para determinar si representan o no una amenaza para el cliente o si podrían permitir que un atacante obtenga acceso no autorizado al sistema. Dado que el pentester ya ha encontrado las vulnerabilidades conocidas del sistema y está buscando más que puedan explotarse, este proceso se conoce comúnmente como análisis de vulnerabilidad. En este punto, el pentester está preparado para crear un plan de acción con pasos claramente definidos [32].

El pentester está listo para comenzar a atacar el sistema una vez que haya completado los preparativos mencionados anteriormente. Esta es la fase de exploración, durante la cual el prospector dedica tanto o tan poco tiempo a explorar como sea factible en el tiempo asignado. Cuando una explotación tiene éxito, puede allanar el camino para una segunda fase más avanzada de explotación en la que se extraen datos confidenciales, otros sistemas y otra información útil de la vulnerabilidad resultante.

Como paso final, el pentester debe anunciar el resultado de la prueba y detallar todo el proceso en un informe, independientemente de si la prueba fue exitosa o no. Los hallazgos son resumidos por el pentester y presentados a ejecutivos y profesionales técnicos para consulta sobre arreglos y configuraciones adicionales.

Ilustración 20 Metodología de prueba de penetración

1.1.15 Pruebas de penetración

Una prueba de penetración, a menudo conocida como prueba de penetración, es un ataque autorizado a un sistema de TI para evaluar el nivel de seguridad de ese sistema. El objetivo de una prueba de penetración de recursos es aumentar la seguridad de los informáticos que se están probando al encontrar todas las vulnerabilidades posibles que un atacante puede aprovechar. En muchos casos, a un probador de penetración se le daría cierto acceso a nivel de usuario y, el objetivo sería elevar el estado de la cuenta o usuario y obtener acceso a información adicional a la que un usuario de ese nivel no debería tener acceso. Por lo tanto, las pruebas de penetración tienen más énfasis en obtener el mayor acceso posible. El principal Lo que separa a un probador de penetración de un atacante es el permiso en el sistema [33].

Las vulnerabilidades del sistema pueden surgir debido a una instalación y configuración incorrectas de los parámetros del sistema por errores de codificación que nunca se descubrieron durante las pruebas. Los usuarios con intenciones maliciosas pueden intentar explotar las debilidades de un sistema para obtener datos y obtener beneficios económicos. La forma en que un sistema responde a un ataque, incluida la facilidad con la que se pueden violar sus defensas y la información que se puede extraer de él, se puede mostrar a través de una prueba de penetración.

La única vez que vale la pena una prueba de penetración es si el sistema de destino realmente contiene información confidencial que un atacante quisiera tener en sus manos. Una prueba penetrante puede identificar con precisión las posibles vulnerabilidades del sistema, lo que permite a sus propietarios protegerlo mejor para garantizar el funcionamiento seguro y sin

problemas de todo el sistema de información. Dada la experiencia técnica que generalmente se requiere, este es el enfoque preferido para las grandes corporaciones y las agencias gubernamentales que buscan proteger sus redes para lograr sus objetivos comerciales de manera efectiva.

Los profesionales de la seguridad de la información han estado advirtiendo a los gobiernos y las empresas desde 1965 que la proliferación de la capacidad de las computadoras para intercambiar datos a través de las redes conducirá inevitablemente a intentos de intrusión. Esto, presumiblemente, resaltó la necesidad de desarrollar un enfoque sistemático para garantizar la integridad de los sistemas. La comprensión de cómo diseñar un sistema infalible ha sido un tema candente de estudio durante décadas. Numerosas herramientas altamente especializadas existen ahora para realizar una prueba de penetración. Las plataformas particulares también están diseñadas con herramientas de penetración incorporadas. Kali Linux es solo un ejemplo de una distribución utilizada para análisis forense digital y pruebas de penetración [34].

1.1.16 Fase de recopilación de información

El primer paso de un pentest es recopilar la mayor cantidad de datos posible sobre el sistema de destino. Existen dos métodos de recopilación de datos: activo y pasivo [32].

En la recopilación de datos activa, la sonda intenta determinar la identidad del sistema operativo, los servicios que se ejecutan actualmente en el sistema y los puertos que están abiertos. Este tipo de datos es esencial y facilita la prueba porque la mayoría de las vulnerabilidades, por ejemplo, en el software del sistema operativo, se enumeran y pueden explotarse con poco esfuerzo. El problema con esta estrategia es que las técnicas activas de recolección de datos son bastante ruidosas y pueden ser fácilmente detectadas por los sistemas de detección de intrusos, los sistemas de prevención de intrusos y los firewalls.

En la recopilación pasiva de datos, el pentester recopila la mayor cantidad de datos posible a través de medios como motores de búsqueda, redes sociales, sitios web e ingeniería social. Las pautas y las mejores prácticas para pentesting a menudo recomiendan métodos pasivos de recopilación de datos porque son menos intrusivos para los sistemas. Aunque aparentemente lleva mucho tiempo, este enfoque puede generar una gran cantidad de información sobre el

tipo de sistema operativo, los puertos abiertos y las direcciones IP dominantes. si el pentester comenzará o no la prueba utilizando las variables recibidas depende en última instancia del cliente y de los términos acordados [35].

1.1.17 Modelado de amenazas

La fase de modelado de amenazas comienza con la información aprendida durante la fase de recopilación de datos. El objetivo del pentester es crear estrategias y ataques basados en los datos recopilados. El modelado de amenazas es una técnica para evaluar la vulnerabilidad de un sistema ante posibles ataques. El enfoque está estructurado de manera que los riesgos de seguridad asociados con el sistema puedan identificarse, cuantificarse y tratarse [26]. Hay cinco pasos en este proceso [36]:

- Revisar los objetivos de seguridad de la organización Este paso implica una revisión de los objetivos generales de seguridad de la organización.
- Una inspección del sistema es el primer paso para averiguar qué partes tiene y si está o no conectado a otras redes.
- El primer paso es desmontar todo el sistema para identificar todas las partes que pueden afectar la seguridad del sistema, como el módulo de inicio de sesión.
- Amenazas identificadas: este paso identifica posibles amenazas externas a las que se enfrenta el sistema. Esto se dirige principalmente a aquellos que son accesibles al público en los sitios web.
- Determine si el sistema es vulnerable o no a las amenazas identificadas realizando un análisis de las vulnerabilidades identificadas.

Ilustración 21 Pasos de modelado de amenazas [63]

1.1.18 Análisis de vulnerabilidad

La identificación de vulnerabilidades es la etapa final del modelado de amenazas. Estos agujeros de seguridad pueden causar serios problemas a un sistema operativo durante su fase operativa. El análisis de vulnerabilidades generalmente se realiza con una herramienta de explotación automatizada, y luego el pentester tiene que estudiar y analizar las vulnerabilidades descubiertas. La identificación, cuantificación y priorización (o clasificación) de las vulnerabilidades del sistema a menudo se logran mediante escaneo automatizado, análisis dirigido e investigación manual [37].

1.1.19 Herramientas de análisis de vulnerabilidades

Los escáneres de aplicaciones web son herramientas automatizadas que escanean sitios web en busca de fallas de seguridad como Cross-Site Scripting e inyección SQL desde el exterior. Extracción de datos insegura, inyección de comandos, búsqueda de rutas y configuración del servidor. El término "herramientas de prueba de seguridad de aplicaciones dinámicas" se usa comúnmente para referirse a esta categoría de software [38]. Las herramientas de escaneo de vulnerabilidades pueden venir con características variables como [39]:

- Algoritmos de escaneo inteligente
- Escaneos automáticos e instantáneos
- Tecnología de escaneo profundo
- Escaneo rápido
- Eliminación de malware
- Respuesta a tiempo
- Escaneo remoto de malware
- Informes avanzados
- Lista negra de sitios web
- Monitoreo de integridad de archivos
- Acciones de seguridad posteriores al hackeo
- Eliminación de malware
- Respuesta a tiempo.

Según la profundidad del análisis y los recursos disponibles, un profesional puede encontrar una variedad de herramientas en línea o por una tarifa. Muchos entornos de prueba comerciales dependen únicamente de tales herramientas de escaneo.

1.1.20 Explotación

Después de realizar una evaluación exhaustiva de la vulnerabilidad, se debería haber formulado un conjunto de objetivos. La fase de explotación de las pruebas de penetración se ocupa únicamente de establecer el acceso a un sistema o recurso evadiendo las medidas de seguridad.

En la fase de explotación, los piratas informáticos intentan activamente aprovechar las fallas de seguridad que han descubierto en la fase de prueba de penetración. Los exploits se desarrollan para recopilar datos confidenciales o para dar a un pentester la capacidad de comprometer un sistema y obtener permisos elevados.

Por ejemplo, si un sistema comprometido no puede interactuar con sistemas internos a los que no se puede acceder desde Internet, el pentester ahora tiene acceso a más objetivos potenciales que no estaban disponibles antes y, por lo tanto, intentará obtener acceso a más sistemas. Para lograr cualquier objetivo nuevo, debemos volver a las etapas de reconocimiento y catalogación para recopilar datos sobre estos sistemas novedosos y usarlos con pleno efecto [32]. “En algunos casos, las vulnerabilidades, como el uso de contraseñas predeterminadas es un ejemplo de una explotación fácil y rápida, otros son más complicados” [40].

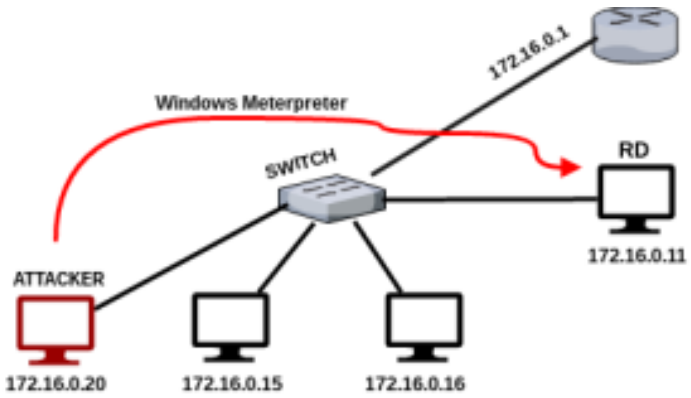

Ilustración 22 Ejemplo de pivote [67]

Explotación posterior

En la fase posterior a la explotación, el pentester recopila información valiosa sobre el sistema atacado y busca archivos intrigantes y datos críticos que demostrarán las debilidades del sistema y los medios por los cuales es posible el acceso no autorizado.

La revisión es el paso final en cualquier prueba penetrante. Incluso si un pentest no termina en un ataque exitoso, cada paso se registra meticulosamente para garantizar que no se pierda nada. Esto garantiza que todo se pueda reconstruir en detalle después de la prueba, aclarando por qué la explotación era imposible.

Esta información se utiliza para generar un informe individual al final del pentest, que aclara los resultados de la prueba para el lector. Todo en el informe está escrito por el pentester que realizó la prueba, por lo que sabe que refleja con precisión los resultados de la pentest e incluye toda la información relevante sobre sus hallazgos específicos [41].

1.1.21 Pentesting

Definición de pruebas de penetración

La prueba de penetración (también llamada "prueba de penetración") es la práctica de probar sistemas informáticos, redes o aplicaciones web para encontrar vulnerabilidades que los atacantes puedan aprovechar. Las pruebas de penetración pueden automatizarse mediante aplicaciones de software o realizarse manualmente. En cualquier caso, el proceso implica recopilar información sobre el objetivo antes de la prueba (reconocimiento), identificar posibles puntos de entrada, intentar una intrusión (virtual o en vivo) e informar los resultados [42].

El objetivo principal de las pruebas de penetración es identificar vulnerabilidades de seguridad. Las pruebas de penetración también se pueden usar para probar el cumplimiento de las políticas de seguridad de una organización, la conciencia de seguridad de los empleados y la capacidad de una organización para detectar y responder a incidentes de seguridad.

Las pruebas de penetración a veces se denominan "ataques de sombrero blanco" porque los buenos intentan entrar.

Tipos de pruebas

Hay tres tipos de pruebas de penetración en función de su alcance: caja negra, caja gris y caja blanca. Aunque estas pruebas son diferentes entre sí, tienen el objetivo común de encontrar vulnerabilidades de seguridad.

Caja negra o *black box*

Este es un intento de comprometer un sistema informático sin conocimiento previo. Esta prueba revela errores de aplicación o vulnerabilidades de seguridad que pueden ser aprovechadas por ciberdelincuentes que atacan desde el exterior sin acceso al sistema. Solo se proporciona la URL o IP de la aplicación. Los casos de prueba están limitados porque no se aprovecha la funcionalidad interna de la aplicación. [43]

Caja gris o *grey box*

Esta prueba proporciona información confidencial sobre la aplicación, como contraseñas de acceso y descripción general de la arquitectura. Esto ayuda a expandir el número de casos de prueba para ejecutar. Por lo que suele encontrar violaciones de seguridad más graves e importantes. Ciertas partes de la aplicación son atacadas de manera muy específica. Tiene todas las ventajas de las pruebas de caja negra. Sin embargo, lleva más tiempo ya que se realizan ataques externos e internos simulando el rol de un usuario autenticado. [43]

Caja blanca o *white box*

Un pentest de caja blanca proporciona total confidencialidad de su aplicación y sistema. Esto incluye diseños arquitectónicos, credenciales y, lo que es más importante, el código fuente se comparte para una revisión completa y encontrar aún más vulnerabilidades. Esta es la prueba más exhaustiva, ya que proporciona un análisis de seguridad completo de su sistema. Sin embargo, debido a su alta complejidad, lleva más tiempo desarrollarse. [43]

Ilustración 23 Tipos de Pent Testing [70]

Actividades que comprende

Cada prueba de penetración tiene diferentes fases o etapas que se desarrollan con el tiempo. Los profesionales de la ciberseguridad deben seguir protocolos para planificar y ejecutar de manera óptima cada prueba. Esto nos permite verificar y garantizar la seguridad de la información que se encuentra en nuestro sistema.

Se explica de forma detallada en qué consiste cada una de las fases y cuál es su utilidad en los sistemas de las organizaciones [44]:

Reconocimiento

Esta es la fase en la que un atacante intenta recopilar toda la información necesaria sobre el sistema o la red que se está analizando para lanzar un ataque exitoso. Tenga en cuenta que, en esta etapa, el probador de penetración no está tratando de ingresar al sistema en sí, sino recopilando información del exterior. La información recopilada en esta etapa incluye direcciones IP (para obtener especificaciones de firewall), datos personales de los empleados de la empresa (nombre, apellido) y, por supuesto, direcciones de correo electrónico.

Escaneo

Esta fase está destinada a verificar de manera proactiva si algo encontrado durante la fase de detección tiene vulnerabilidades relacionadas con los servicios encontrados. Esto ayuda a

definir el nivel de probabilidad de intrusión. De hecho, esta fase de pentesting es muy importante desde el punto de vista del análisis de ciberseguridad ya que nos permite verificar el nivel de seguridad del sistema. Una vez que tenga una descripción general de sus puntos de acceso, puede ingresar al sistema a través de ellos en la siguiente etapa de pentesting.

Explotación

Una vez que haya encontrado vulnerabilidades o brechas de seguridad descubiertas desde la fase anterior, es hora de probarlas. Esto significa que la persona responsable de las pruebas de penetración debe intentar penetrar en el sistema a través de puntos de entrada previamente identificados. Además, los programadores continúan buscando un posible acceso a los niveles de privilegio del sistema si pueden obtener acceso al sistema y aprovechar la vulnerabilidad. El objetivo es obtener la mayor cantidad de información posible y mostrar el daño que pueden causar los ciberdelincuentes.

El propósito es reconocer los puntos más débiles del sistema, qué acciones se pueden tomar para fortalecer estas vulnerabilidades y comprender su importancia en términos de seguridad de la información del sistema.

Borrado de rastro

Después de que se hayan realizado todas las pruebas de penetración, puede dejar algunos rastros o rastros que pueden servir como guías para posibles ataques futuros. Por lo tanto, en este punto, cualquier "rastro" que quede debe eliminarse por completo. Si no se hace correctamente, se considera una vulnerabilidad de alto riesgo en su sistema y compromete completamente su ciberseguridad. En este sentido, las pruebas de penetración periódicas nos permiten actualizar nuestros sistemas y conocer nuevas vulnerabilidades antes de que sean explotadas con fines maliciosos.

Beneficios de un pentesting

El objetivo del pentesting es ayudar a las empresas a identificar vulnerabilidades que pueden haber pasado desapercibidas, facilitando así los ataques de los ciberdelincuentes. En resumen, las pruebas de penetración o penetración son útiles para: Cuán probable es que un ciberataque

tenga éxito, cuáles son las vulnerabilidades de mayor y menor riesgo y cómo pueden poner en riesgo a su organización Determinar qué vulnerabilidades son casi imposibles de detectar.

1.1.22 Herramientas de pentesting

Nmap

En un proceso completo de pruebas de penetración, existen instancias previas a ejecutar esta herramienta, pero, para dar los primeros pasos, probablemente sea la mejor forma de comenzar. Nmap es una herramienta de escaneo de redes que permite identificar qué servicios se están ejecutando en un dispositivo remoto, así como la identificación de equipos activos, sistemas operativos en el equipo remoto, existencia de filtros o firewalls, entre otros [45].

En palabras sencillas, cuando se va a atacar un servidor o dispositivo, el atacante podrá realizar distintas arremetidas en función del servicio: no es lo mismo dañar un servidor web, un servidor de base de datos o un router perimetral. Por lo tanto, en cualquier despliegue, el primer paso será identificar los servicios en la infraestructura, para decidir cómo avanzar y, considerando que en una prueba de penetración se "imitan" los pasos de un atacante, también se iniciará de la misma manera.

Nmap es una herramienta de línea de comandos (existen algunas interfaces gráficas, pero, personalmente, no las recomiendo, aunque es una cuestión de gustos) donde se debe indicar cuál será el o los objetivos y la serie de parámetros que afectarán la forma en que se ejecuten las pruebas y los resultados que se obtienen. Puede instalarse tanto en Linux, Windows, Mac u otros sistemas operativos.

Nessus

Una vez que se tienen identificados los servicios que se están ejecutando, se puede comenzar el uso de las herramientas que sirven para identificar vulnerabilidades en los servicios. En este campo, la mejor herramienta para introducirse en este mundo es Nessus, otra aplicación gratuita (solo para uso hogareño, suficiente para los fines de este artículo; en el caso de fines profesionales es necesario usar la versión de pago) que, por su base de datos y su facilidad de

uso, es la preferida en este aspecto [46].

Aunque posee una línea de comandos, considero que su interfaz gráfica, muy completa e intuitiva, es una forma sencilla de comenzar a probar esta herramienta. Nessus posee una extensa base de datos de vulnerabilidades conocidas en distintos servicios y, por cada una de éstas, posee plugins que se ejecutan para identificar si la vulnerabilidad existe (o no) en determinado equipo objetivo. En resumen, al ejecutarse Nessus sin parámetros específicos, se probarán miles de vulnerabilidades y se obtendrá como resultado un listado de las vulnerabilidades que fueron identificadas.

La lógica de Nessus es similar a Nmap: hay que indicar el objetivo, en este caso la o las direcciones IP y los parámetros. Estos permiten limitar el campo de búsqueda, especialmente si en una etapa anterior se identificaron los servicios: no tiene sentido buscar vulnerabilidades conocidas en Linux en un equipo que tiene instalado Windows.

Metasploit Framework

Una vez identificados los servicios y sus vulnerabilidades, el paso siguiente sería la explotación de las vulnerabilidades. Es decir, primero se tiene que probar si realmente las vulnerabilidades identificadas permiten a un atacante causar algún daño. Después se intenta conocer cuál sería ese daño. A pesar de que se haya identificado una vulnerabilidad en la instancia anterior, podría ser que, al momento de intentar explotarla, existan otras medidas de control que no hayan sido consideradas, otras capas de seguridad o distintas variables que podrían hacer más complicada la explotación de la misma. Asimismo, si se logra explotar la vulnerabilidad, podría comprobarse y dimensionar cuál podría ser el daño hacia la organización, en función de la información o sistemas que estuvieran “detrás” de dicha vulnerabilidad [44] .

Para este fin, Metasploit es la herramienta ideal para hacer estas pruebas. Mientras Nessus posee una base de datos de vulnerabilidades, Metasploit posee una base de exploits que podrían aprovecharlas. En otras palabras, en lugar de revisar si hay una vulnerabilidad en un equipo remoto, directamente se intenta la ejecución de un exploit y se simulan las consecuencias posteriores, en caso de que éste se ejecutara con éxito. Al igual que Nessus, su versión de línea de comandos, msfconsole, es la tradicional, incluso recomendable para la automatización.

Sin embargo, su interfaz gráfica es muy conveniente para dar los primeros pasos y tener una mayor comprensión.

DVL-DVMA

Para probar las tres herramientas anteriores, es necesario definir un sistema objetivo, un sistema en el que se harán las pruebas. Una pésima costumbre de quienes inician en este ámbito es realizar sus primeros pasos y pruebas en sistemas públicos de Internet, en un entorno real. Esto podría acarrear problemas legales y no es la forma correcta (ni ética) de realizarlo. Para aprender a usar estas herramientas, se debe utilizar un entorno de pruebas, es decir, un escenario de investigación en donde uno pueda tener acercamientos sin riesgos de afectar algún entorno en producción.

Para ello, existen dos herramientas excelentes: Damn Vulnerable Linuxy (DVL) y Damn Vulnerable Web Application (DVWA). Aunque el primero está descontinuado, aún se puede conseguir en Internet para hacer los primeros pasos y primeras pruebas. Se trata de un sistema operativo y una aplicación web que poseen todo tipo de vulnerabilidades, de tal forma que, la persona que los utiliza puede intentar explotarlas y experimentar [35].

También es posible "construir" nuestro propio sistema de pruebas: tan solo instala cualquier sistema operativo (desactiva las actualizaciones o instala una versión antigua) y sobre él comienza a instalar servicios en versiones anteriores a la última. De esta forma, tendrás tu propio sistema vulnerable para hacer pruebas. Este entorno es el correcto para dar tus primeros pasos en Penetration Testing.

Kali Linux (backtrack)

Finalmente, hay una distribución de Linux diseñada exclusivamente para Penetration Testing. Las herramientas antes descritas (Nmap, Nessus, Metasploit) están disponibles y, no solo eso, también hay muchas más herramientas para continuar practicando. Por ejemplo, Kali (antes conocida como Backtrack) es una distribución que posee todo tipo de herramientas preinstaladas que sirven para realizar Penetration Testing [42] .

El orden en que se presentaron las herramientas no es aleatorio, es lo recomendable para

comenzar a experimentar. Primero hay que probarlas de forma aislada y luego, abocarse completamente a Kali Linux. Puede ser descargada como imagen ISO o directamente para VMWare. Una vez que inicias un sistema Kali Linux, verás un menú muy extenso con más de 300 herramientas para pentesters. Nmap y Metasploit Framework están incluidos en esta lista, entre otros.

1.2 Antecedentes de la investigación

A continuación, se mostrará información de investigaciones realizadas referente al tema "Análisis de vulnerabilidades críticas del sistema operativo móvil android" que ayudó a obtener más conocimientos y detalles específicos y generales del análisis del tema de investigación.

La investigación y búsqueda de referentes artículos científicos se realizó de manera rigurosa y especifica por medio de distintas bases de documentos como IEEE XPLORE, SCIELO y WEB OF SCIENCE, teniendo un total de cuatro seleccionados para reforzar el análisis del tema. Estos estudios han sido publicados entre el periodo 2014-2020 y serán detallados a continuación.

La primera investigación titula "Análisis de vulnerabilidades y seguridad de dispositivos móviles con sistema operativo iOs 6.1.4", presenta una serie de antecedentes acorde con el riesgo de los Smartphones, basándose en diferentes plataformas móviles para generar un documento que contenga recomendaciones o prácticas de salvaguardar la información almacenada de los dispositivos. La metodología de carácter cualitativo que elabora conclusiones referentes a las investigaciones propuestas. Tomando en cuenta los testeos realizados se sabe que tanto los usuarios como las organizaciones son cada vez más dependientes de internet y esto le genera más riegos y amenazas en todos los ámbitos entonces adicional se debería de tener presente que es necesario estar informados sobre la seguridad del uso de las aplicaciones o medios de comunicación que se puede generar desde cualquier dispositivo. Esta enlista los diferentes dispositivos y propone diferentes etapas para analizar dichos riegos, primero realizan una revisión del documental del sistema operativo iOS, luego revisa los riegos en smartphone y Usuarios, teniendo esto explora las vulnerabilidades que presenta y por ultimo las extrae [47].

La segunda investigación presentada lleva por título “Estado del arte vulnerabilidades de seguridad en sistemas operativos móviles andrioid y ios” se basa en realizar un estudio investigativo acerca de las vulnerabilidades identificadas en dispositivos móviles con sistemas operativos Android y IOs [48]. Para realizar esta investigación se utilizó la metodología para la revisión bibliográfica y la gestión de información de temas científicos basándose en documentales, artículos científicos, revistas u otras fuentes relacionadas al tema de investigación. Al final para llegar a desarrollar ámbitos cotidianos relacionados con el tema de investigación, se realizó una rigurosa búsqueda de información de distintos lugares. Este estudio aporta a la investigación ideas de seguridad y ataques a los dispositivos móviles.

La tercera investigación titulada “Análisis de riegos, de vulnerabilidades y auditorías de dispositivos” presenta un análisis de amenazas de seguridad para dispositivos móviles con plataforma Android para diseñar un modelo de informe que recoge los detalles presentados y los presenta en la investigación [49]. La metodología que se propone es una explicativa de una auditoria forense de sistemas Android en donde se enumeran y definen las fases y recursos necesarios para complementar la información y realizar un diseño de modelo de informe. El estudio analizado es de gran aporte para la investigación a realizar por motivos que describe los distintos riegos que pueden existir dentro de una plataforma Android por medio de un sistema operativo.

La cuarta investigación es un artículo que lleva por título “Detectando aplicaciones maliciosas en Smartphone con sistema Android a través del uso de una aplicación” este presenta herramientas para identificar aplicaciones maliciosas y a su vez desarrolla una herramienta que permita visualizar los riegos que puede tener un dispositivo móvil con sistema operativo Android. El objetivo de este artículo es presentar una aplicación (app) para la detección oportuna a potenciales aplicaciones maliciosas presentes en Smartphones, particularmente aquellos con sistema operativo Android [50]. La metodología utilizada es una experimental e informativa, en donde busca las ventajas y desventajas de las apps al momento de ser utilizada para mitigar los riegos. El análisis de este articulo escogido, es de gran aporte en esta investigación porque demuestra diferentes App que se encargan de investigar las vulnerabilidades de los sistemas operativos Android en Smartphone.

1.3 Fundamentación legal

La presente investigación se basa en la aplicación y regimientos de las siguientes normas y leyes: Constitución de la República del Ecuador [49], y la Ley Orgánica de Protección de Datos Personales [50].

De acuerdo con la Constitución de la República del Ecuador en el Titulo VII de Régimen de buen vivir, Capitulo Primero de Inclusión y equidad, Sección octava de Ciencia, tecnología, innovación y saberes ancestrales; que en el Art. 385, menciona que "Desarrollar tecnologías e innovaciones que impulsen la producción nacional, eleven la eficiencia y productividad, mejoren la calidad de vida y contribuyan a la realización del buen vivir" [49].

Además, está la Ley de protección de datos, es el conjunto de técnicas jurídicas e informáticas encaminadas a garantizar los derechos de las personas sobre el control de su información personal y sobre la confidencialidad, integridad y disponibilidad de esta. el 26 de mayo de 2021 se publicó la Ley Orgánica de Protección de Datos Personales. Tiene por objeto garantizar el derecho a la protección de datos personales, que incluye el acceso y decisión sobre la información y datos de este carácter, así como su correspondiente protección [49].

Toda persona tiene derecho a que sean rectificados, actualizados y, cuando corresponda, suprimidos o sometidos a confidencialidad los datos personales de los que sea titular, que estén incluidos en un banco de datos. En esta investigación, al interceptar, examinar, retener, datos personales sin autorización, damos cumplimiento al artículo 178 de la constitución república del Ecuador, con pena privativa de libertad de uno a tres años, por otro lado, no es aplicable para la persona que divulgue grabaciones de audio y vídeo en las que interviene personalmente. La reprogramación o modificación de los terminales móviles, da el cumplimiento del artículo 191 de nuestra constitución, con pena privativa de libertad de uno a tres años. El intercambio, comercialización o ya sea compra de información situadas en terminales móviles, da cumplimiento al artículo 192 de nuestra constitución, con pena privativa de libertad de uno a tres años.

La modificación de etiquetas de los terminales móviles que describe la información de este da cumplimiento al artículo 193 de nuestra constitución, con pena privativa de libertad de uno a tres años. La comercialización de terminales móviles sin la autorización competente da

cumplimiento al artículo 194 de nuestra constitución, con pena privativa de libertad de uno a tres años.

Poseer programas, equipos, bases de datos o etiquetas que permitan reprogramar, modificar o alterar la información de identificación de un equipo terminal móvil, da cumplimiento al artículo 195 de nuestra constitución, con pena privativa de libertad de uno a tres años. En esta investigación, la intercepción de datos del terminal móvil, con objetivo de obtener información registrada o disponible, da cumplimiento al artículo 230 de la constitución, con pena privativa de libertad de tres a cinco años. Violar la integridad de cualquier sistema informático, da cumplimiento al artículo 230 de nuestra constitución, con pena privativa de libertad de tres a cinco años.

Por otro lado, el artículo 45 de la Ley mencionada indica que, Promover y supervisar el uso efectivo y eficiente del espectro radioeléctrico y demás recursos limitados o escasos de telecomunicaciones y garantizar la adecuada gestión y administración de tales recursos, sin permitir el oligopolio o monopolio directo o indirecto del uso de frecuencias y el acaparamiento.

Desde el ámbito internacional, está la Ley 11 de telecomunicaciones, el objeto de esta ley es la regulación de las telecomunicaciones, que comprende la instalación y explotación de las redes de comunicaciones electrónicas, la prestación de los servicios de comunicaciones electrónicas, sus recursos y servicios asociados, los equipos radioeléctricos y los equipos terminales.

CAPÍTULO II: METODOLOGÍA

2.1 Delimitación

Este estudio se basa en un examen de las fallas de seguridad del sistema operativo móvil Android utilizando la práctica de Pentesting. Esta investigación ha buscado proyectos o artículos indexados en bibliotecas digitales para un análisis más profundo.

Dado que el proyecto propuesto se estudia e implementa en dispositivos móviles con Android, sus límites espaciales no se limitan a ninguna ubicación física. En cuanto a las limitaciones de tiempo, se espera que el estudio esté terminado en un plazo de 5 meses, momento en el cual se habrá presentado la investigación, el análisis y la propuesta proporcionada por los artículos en consideración (octubre-2021 a febrero-2022). Por otro lado, Se implementó el ambiente de pruebas y luego el proceso del pentesting en un periodo de 5 meses.

2.2 Tipos de investigación

Este estudio es de tipo bibliográfico en donde se identificaron otros estudios que dan sustento directo al tema del proyecto; al mismo tiempo, es de tipo experimental es un tipo de investigación cuantitativa. Se basa en un protocolo de control, la presencia de variables, la manipulación de dichas variables y la observación de resultados cuantificables [51], porque se probaron herramientas de explotación para determinar cuáles son las más efectivas.

El tema de investigación propuesto es un estudio híbrido, ya que emplea muchos métodos para explotar las vulnerabilidades y recopila datos para la evaluación de riesgos. En consecuencia, el estudio puede clasificarse como un enfoque de investigación híbrido cualitativo y cuantitativo. La investigación cualitativa puede entenderse como un examen de varias técnicas y herramientas para encontrar fallas de seguridad en los dispositivos, considerando artículos y proyectos que han sido cuidadosamente seleccionados debido a la gran cantidad de información que incluyen.

En cuanto al aspecto cuantitativo, las herramientas analizadas y seleccionadas se utilizaron para la experimentación en el contexto de laboratorio, y se evaluaron su eficacia o no en el momento de realizar las pruebas.

2.2 Métodos y técnicas

Métodos de investigación

Para llevar a cabo la investigación se utilizó el método de la ciencia del diseño (SMS en adelante) y los hallazgos brindarán pautas específicas para la evaluación e implementación del proyecto. Por lo tanto, se utilizó el método de investigación SMS (systematic mapping study), ya que su enfoque principal está en obtener una descripción completa sobre un tema de investigación y recopilar evidencia para la eficacia de los artefactos propuestos del estudio. Además, el método ayudó a clasificar las principales herramientas que se utilizaron para explotar las vulnerabilidades y realizar el análisis en la literatura, la evaluación de la calidad y los resultados. Entonces se puede realizar una evaluación precisa de la herramienta.
Otra técnica que se utilizó es una analítica que permitió examinar las características de cada herramienta que se recopiló mediante el método SMS y luego elegir la mejor herramienta o ejecución con la que realizar el pentesting.

2.3 Técnicas de recolección de datos

En cuanto a los métodos de investigación, se implementó la observación experimental en la que se realizó un estudio preliminar de las variantes que finalmente fueron analizadas y utilizadas en las pruebas. Se monitoreó la respuesta de cada versión a las descargas de APK y aplicaciones, los permisos y el acceso. También se utilizó documentación analítica, con informes y otros registros pertinentes al tema que se analiza. Estos detalles se recopilan mediante la investigación analítica de muchos recursos, como libros, revistas y artículos. Por lo tanto, se buscó, eligió y analizó información que contribuya al tema en cuestión, revisando los documentos existentes que sirven como bases de datos, informes u otra documentación, teniendo en cuenta que todo debe estar relacionado con el tema en cuestión.

2.4 Población y muestra

La población está constituida para las versiones del sistema operativo Android; sin embargo, solo se muestra las versiones de Android que aún son compatibles y tienen soporte: Android 12 como la más actual y Android 5 (Lollipop) como la más antigua. Estas dos variantes fueron elegidas para una comparación de vulnerabilidades y un examen de las mejoras presentadas.

2.5 Descripción de instrumentos

Los instrumentos elegidos por su capacidad para aportar información se utilizaron en el estudio actual de las muchas técnicas ya mencionadas. En cuanto a la observación experimental, se utilizó una ficha de análisis de datos para registrar y evaluar los parámetros del proceso. Además, se presentó un análisis documental en el que los artículos científicos, gráficos e informes que brinden información sobre el tema servirán como fuente principal. Esta información se recopiló utilizando una base de datos científica. Todos estos datos se recopilados, luego se clasificaron y analizaron en Excel según una serie de criterios.

Herramientas como Wireshark para el rastreo de redes, Kali Linux para pruebas de penetración, Burp Suite para detección de vulnerabilidades y Nmap para escanear puertos abiertos en un servidor se utilizaron en la implementación de la identificación de vulnerabilidades en dispositivos móviles Android, luego de ser evaluados a través de un análisis personalizado. Todas estas herramientas son útiles durante cada etapa del proceso de pentesting.

2.6 Técnicas de procesamiento y análisis de datos

Los parámetros, categorías e indicadores se utilizan para clasificar los resultados del procesamiento de datos y análisis. El análisis y procesamiento de datos que resultó del uso de estos métodos y herramientas arrojó los hallazgos que se discutirán en las siguientes secciones:

- La información para el análisis de las herramientas que se utilizaron durante las fases de pentesting se recopiló de una variedad de bases de datos, incluidas Scopus, Scielo, Web of Science e IEEE Xplore.
- La hoja electrónica fue útil para compilar los diversos trabajos, artículos e informes obtenidos, y luego fueron clasificador de acuerdo con varios criterios. Además, se utilizó para la elaboración de la tabla de observación.
- El seguimiento experimental se realizó mediante el uso de herramientas seleccionadas para las etapas del procedimiento de pentesting. Específicamente, BurpSuite se usó en un entorno de red para rastrear la red y Nmap se usó para escanear los puertos del servidor. Esto permitió llevar a cabo las fases de la metodología.

  Kali Linux también se usó para analizar cualquier violación de datos o ataques que ocurran durante las próximas sesiones de práctica.

2.7 Normas éticas

La autenticidad, originalidad y profesionalismo del tema propuesto en este documento se basan en la investigación que presenta datos e información personal auténticos de acuerdo con los lineamientos establecidos por el Reglamento de Grados de la Pontificia Universidad Católica del Ecuador Sede Esmeraldas. Se aplicó la información, mostrando nuevos conceptos e innovaciones a medida que surjan de los datos de investigación procesados mientras se protegen los derechos de propiedad intelectual.

CAPÍTULO III: RESULTADOS

3.1 Descripción del estudio

El proceso de esta prueba se realizó mediante una prueba de penetración. El dispositivo elegido para esta prueba es una virtualización con sistema operativo Android 6.0 Mashmallow.

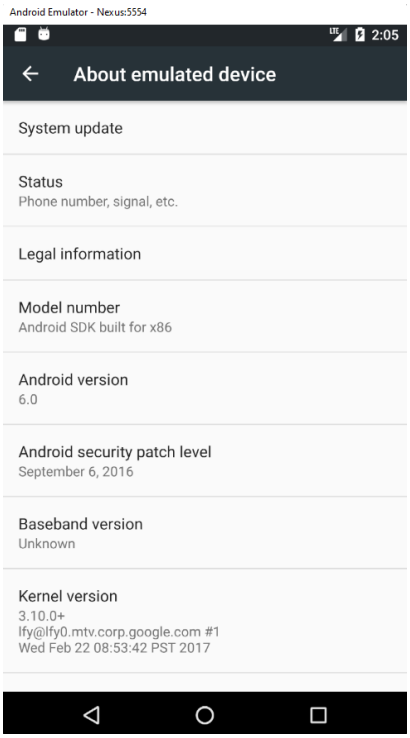

Ilustración 24 Características del dispositivo móvil
Fuente propia

El pestenting desarrollado se hizo en máquinas virtuales, el sistema operativo a utilizar fue Kali Linux, que es basada a Ubuntu en el ambiente de Linux en donde vienen algunas de las herramientas ya preinstaladas para el comienzo de la penetración. El proceso fue aplicado mediante sus 5 fases para el comienzo del ataque, estas son sus fases:

- Recolección de información
- Análisis de vulnerabilidades
- Análisis de riesgo
- Explotación de vulnerabilidades
- Reporte

3.2 Instalación de Kali Linux

Se descargó la versión 2023 W07 de Kali Linux y el proceso de la creación de esta máquina se realizó de la siguiente manera:

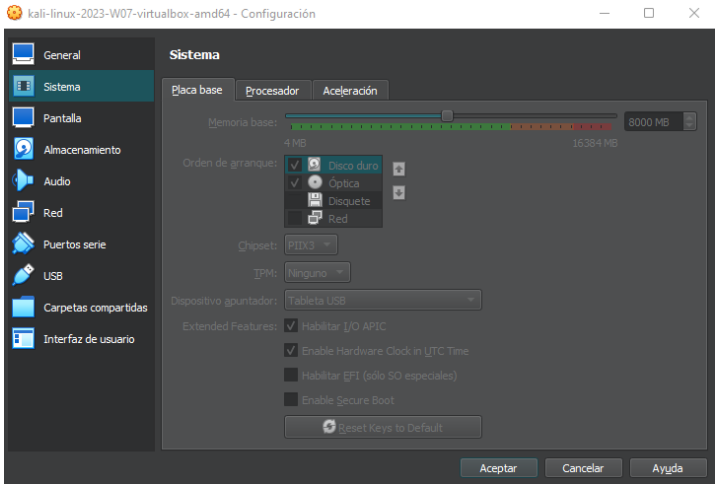

Ilustración 25 Características del sistema de la maquina
Fuente propia

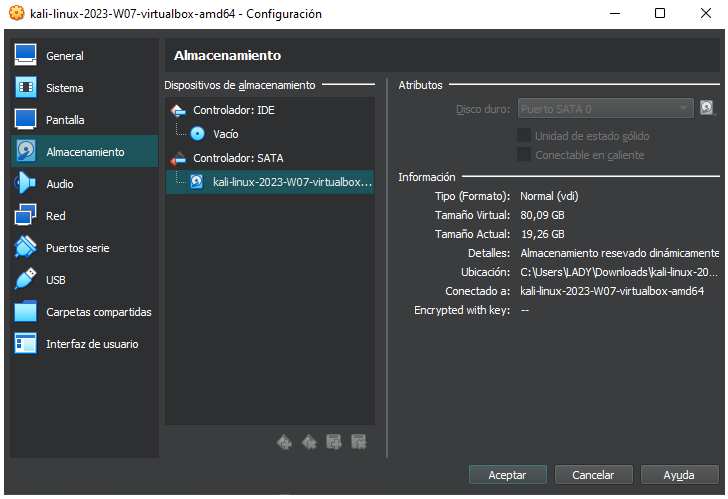

Ilustración 26 Características del almacenamiento de la maquina
Fuente propia

Ilustración 27 Entorno de Kali Linux

3.3 Instalación de Android Studio

Teniendo ya el sistema Kali instalado, se procede a emular el dispositivo móvil que se utilizó para la prueba, en este caso se empleó Android Studio para obtener esta virtualización. Se comienza descargado el SDK que se va a manejar, en este caso será el Android 6.0 (Marshmello) y el Android 12.0, que son las versiones seleccionadas para el pentesting.

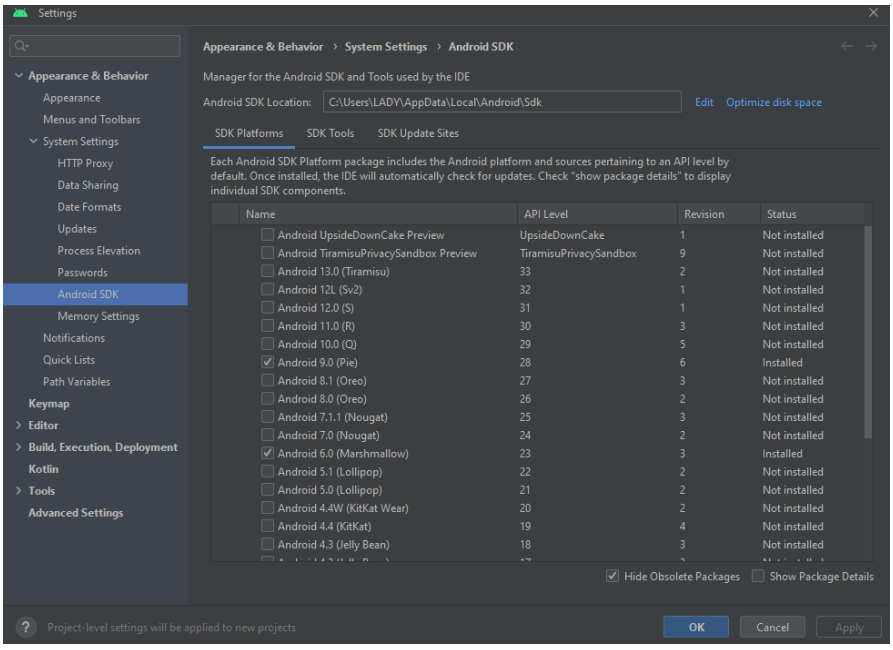

Ilustración 28 Selección de SDK
Fuente propia

Para no presentar problemas en el rendimiento del dispositivo móvil se agregaron los recursos presentados en la imagen para que no se presenten problemas en el procedimiento. Finalizada la instalación se procede a ejecutar el APK y comenzar con la prueba.

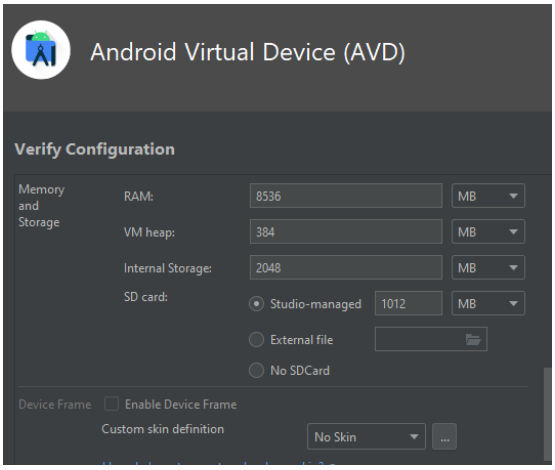

Ilustración 29 Características de la virtualización del dispositivo móvil
Fuente propia

3.4 Proceso del pentesting

3.4.1 Recolección de Información

En esta primera fase se recopiló los siguientes datos:

Tabla 2 Características de dispositivo para la práctica

Descripción	**Detalle**	**Cantidad**
Tipo de dispositivo	smarphone	1
Marca	Nexus 6	N/A
Versión del sistema operativo	Android 6.0 Mashmallow	N/A
Kernel	Linux 3.10.0	N/A
Blurtooth	Si	N/A
Dirección ip	192.168.1.31	N/A
Puertos	8080	N/A
Wi-Fi	Si	N/A
Anti-Virus	No	N/A
Navegador	Browser	N/A

Para realizar un escaneo de los puertos se utilizó la herramienta wireshark para identificar los servicios expuestos en la red conectada con el dispositivo móvil.

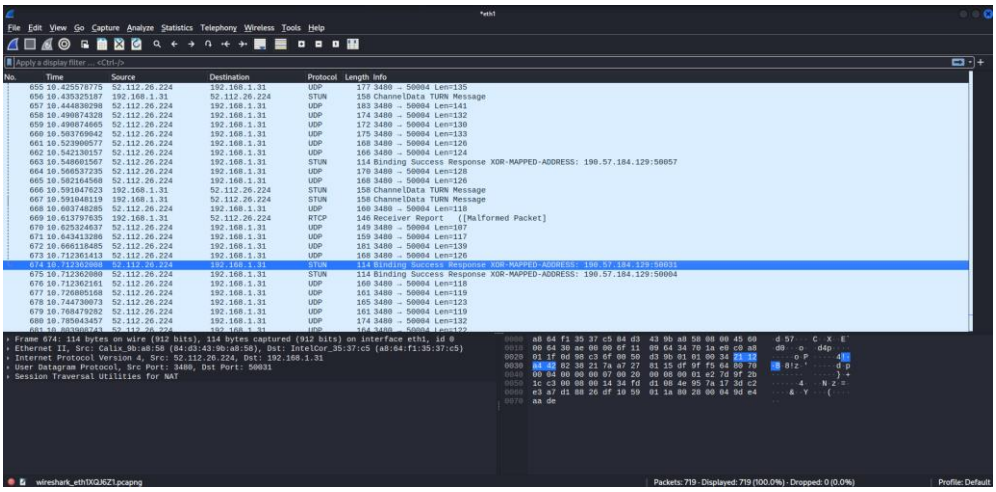

Ilustración 30 Escaneo de servicios

3.4.2 Análisis de Vulnerabilidad

Para realizar un escaneo de los puertos se utilizó la herramienta wireshark para identificar los servicios expuestos en la red conectada con el dispositivo móvil.

Las vulnerabilidades de los dispositivos móviles varían dependiendo de la marca móvil, pero se identifican las principales vulnerabilidades que generalmente están presentes en todos los dispositivos que son las siguientes:

- Desbordamiento de memoria
- Desbordamiento de Sppofing

Para comenzar con este procedimiento se usó una la herramienta Burp Siute, con la que se logró obtener vulnerabilidades asociadas con stagefringth al abrir una APK instalada, esta APK fue “Pinterest”.

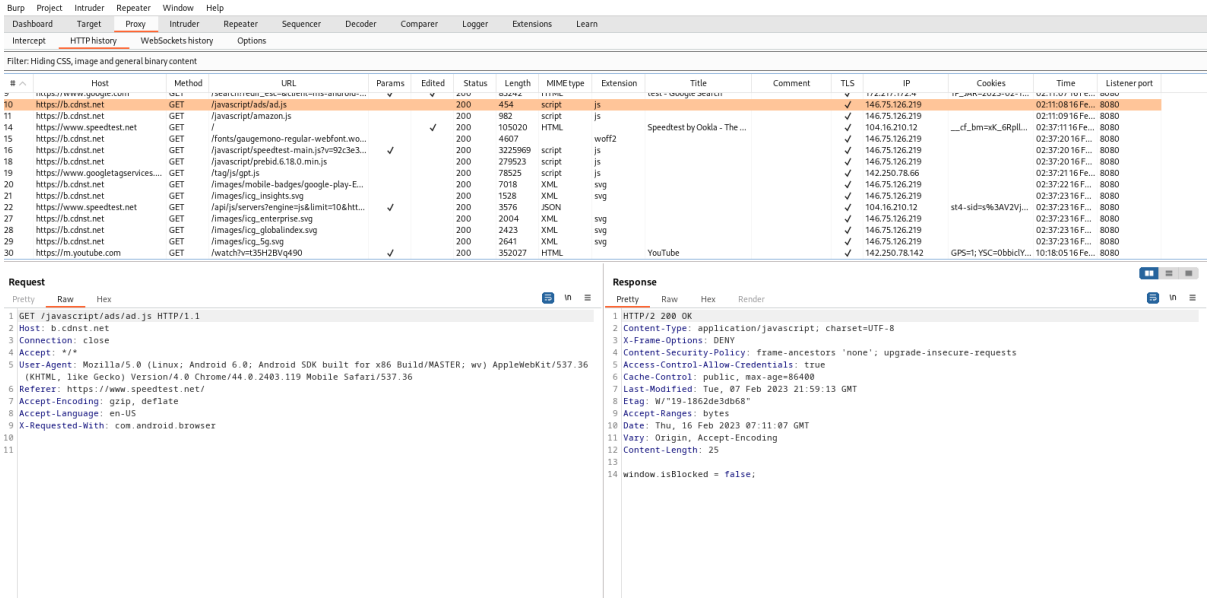

Ilustración 31 Recopilación de información
Fuente propia

3.4.3 Análisis de riesgos

En esta fase se procede a realizar un análisis para calificar las amenazas y vulnerabilidades que afecten al dispositivo móvil seleccionado. Para eso se realizó una matriz de riesgos donde se ve el impacto, el riesgo y el principio para la seguridad de la información del dispositivo móvil:

	RIESGOS	RIESGO INHERENTE			PRINCIPIO AFECTADO(SEGURRRIDAD DE LA INFORMACIÓ				
CÓDIGO DEL RIESGO	DESCRIPCIÓN	Probabilidad de ocurrenc	Impacto	Nivel de riesgo	Conf	Int	Disp	Priv	Aux
R1	Ausencia de un canal de comunicación seguro.	Muy alta	Superior		x	x	x	x	
R2	Instalación de programas con contenido maliciosos, en el dispositivo movil desde la SD CARD	Alta	Superior		x	x	x	x	
R3	Descarga der aplicaciones del market que contenga virus	Alta	Mayor	Alto		x	x		
R4	Afectación de el edispositivo con un Touchlogger	Baja	Importante			x	x		x
R5	Alteración, modificación o destrucción de datos, o registros con motivos intencionales o no y de forma no autorizada	Alta	Superior			x	x		x
R6	Explotar vulnerabilidades del sistema	Muy alta	Superior			x	x		
R7	Eliminación masiva de datos por ejecución de comandos	Muy alta	Mayor			x	x		x
R8	Control remoto del dispositivo	Muy alta	Superior		x	x	x	x	
R9	Propagación de virus por recursos de red compartidos a los demas equip	Muy alta	Mayor		x	x		x	
R10	Ausencia o definición errada de los estándares y/o patrones de seguridad para los dispositivo	Alta	Superior		x	x	x	x	
R11	Aplicación parcial o deficiente de parches de segundad liberados por el fabricante.	Alta	Mayor	Alto	x	x	x	x	x
R12	Ausencia de verificación de los registros de auditoría con el fin de verificar posible incidentes de seguridad	Alta	Mayor	Alto					x
R13	Suplantación de funcionarios del entomo financiero (En caso de perdida o robo del dispositivo)	Muy alta	Superior		x	x	x	x	
R14	Ulizacion del dispositivo para fines personales	Moderada	Importante		x		x		
R15	Conectar el disposito a diferentes redes inalambricas	Alta	Mayor	Alto		x			
R16	Inexistencia o Softw are antivirus desactualizado, que permita la propagación de troyanos, malw are o código malicioso.	Alta	Superior		x	x	x	x	x
R17	Ausencia de politicas y procedimientos implementados de clasificación de información basados en contenido, valor y riesgos asocidos a la información contenidad en la base de datos.	Muy alta	Superior		x	x	x	x	x
R18	El no uso de contraseñas fuertes para el acceso al dispositivo	Muy alta	Superior		x	x		x	
R19	Ausencia de personal encargado de la investigacion del nacimiento de nuevas cepas de mallw are	Alta	Importante	Alto		x	x		
R20	Asignación errada de privilegios de acceso a los usuarios a la red interna, por medio del dispositivo	Muy alta	Superior		x	x	x		

Ilustración 32 Matriz de riesgo
Fuente propia

3.4.4 Fase de explotación de las vulnerabilidades

Teniendo identificadas la amenazas y las vulnerabilidades, se procedió a realizar los ataques para causar impacto en el dispositivo móvil utilizando las APK o navegadores:

- Robo de credenciales
- Descarga e instalación de APK maliciosas
- Robo de información confidencial

En las siguientes imágenes se podrá observar las pruebas realizadas para llevar a cabo el ataque:

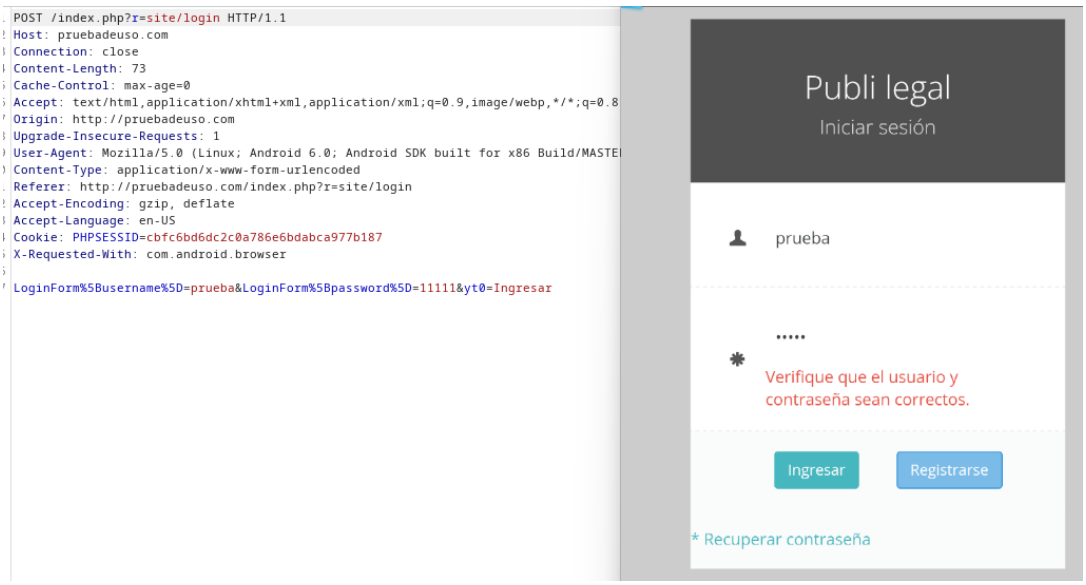

Ilustración 33 Ataque robo de credenciales

Para la interceptación del tráfico en el canal de comunicaciones se realiza una validación de las tablas ip para comenzar el ataque.

Ilustración 34 Validación de tablas ip

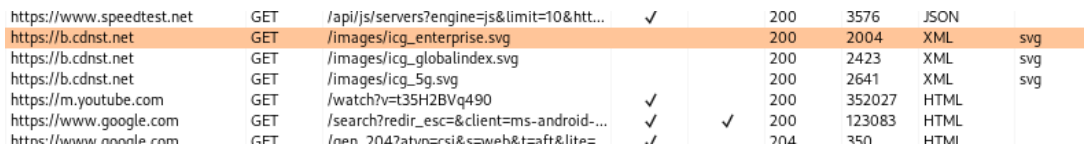

Ilustración 35 Burp Siute Scan

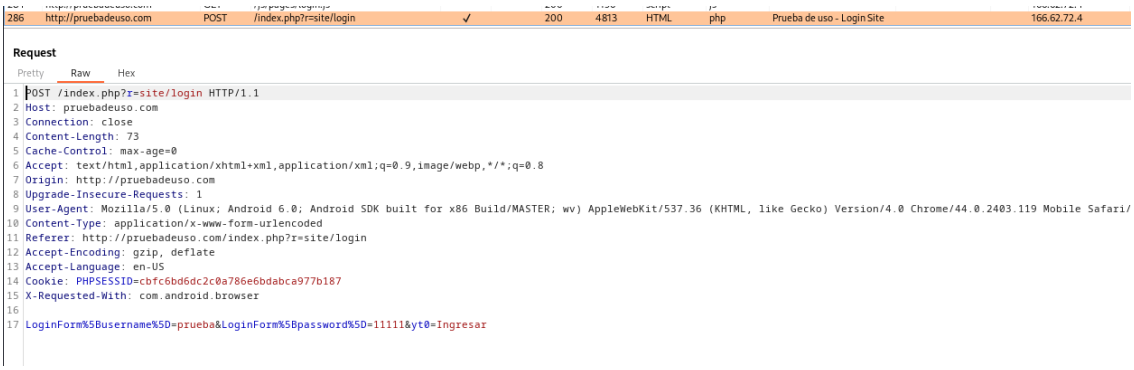

Ilustración 36 Configuracion de sniffer

Como resultado del ataque se logra obtener la descarga del archivo que se realizó, en este caso es una imagen con formato svg. Además, también se logró obtener las credencias de un ingreso mediante inicio de sesión.

3.4.5 Reportes

En la siguiente tabla se evidencia los resultados de la prueba de penetración.

Tabla 3 Resultado de la prueba de penetración

ATAQUE	*TECNICA*	*EFECTIVIDAD %*
Elevación de privilegio	Control Wireshark	80%
Descarga de APK maliciosa	Control Burp Siute	95%
Robo de credenciales	Control Burp Siute	95%

El análisis de acuerdo con Dradis dio como resultado que existen 4 vulnerabilidades con riesgo alto a explotación:

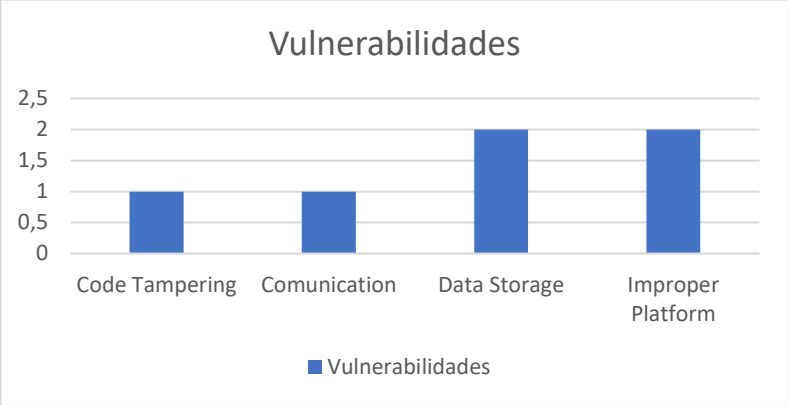

Ilustración 37 Vulnerabilidades encontradas

Luego de la ejecución de las fases del pentesting se muestran las vulnerabilidades encontradas con los detalles de cómo se realizó:

Tabla 4 Tabla detallada de las vulnerabilidades

N.º	*Vulnerabilidad*	*Descripción*	*Origen*	*Herramienta*	*Calificación*
1	Replica de sus archivos	La clonación o replica de sitios web permite la copia de este, pero no es absoluto ya que cambia de servidor web al subirse nuevamente a Internet.	Fase Escaneo	Wireshark	
2	Certificado SSL	Un certificado SSL/TLS asegura la transmisión encriptada de datos punto a punto.	Fase Explotación	Burp Siute	
3	Puertos Vulnerables	El servidor Apache permite es susceptible a ataque de denegación de servicios a través de una petición HTTP	Fase Explotación	Burp Siute	
4	Contraseñas no encriptadas	Las contraseñas son consideradas datos sensibles por los que deben ser encriptadas. Se recomienda utilizar encriptación superior a SHA1.	Fase Escaneo	Wireshark	

CAPÍTULO IV: DISCUSIÓN

De acuerdo con los resultados, se evidencia que existe concordancia con lo expuesto en el estudio realizado por Janosik [47], sobre, la seguridad de los dispositivos móviles, donde expone que los dispositivos son cada vez más dependientes de internet y esto le genera más riegos y amenazas en todos los ámbitos, por tanto, mediante esta investigación se logró comprobar mediante testeos y ataques existen muchas más vulnerabilidades presentes cuando el usuario no es informado sobre la seguridad del uso de las aplicaciones o medios de comunicación que se puede generar desde cualquier dispositivo, por ende se extrae información de manera más fácil y rápida con app y Archivos no seguros.

Por otro lado, mediante la recopilación de información de distintas fuentes para esta investigación, se determinó que Android, es el sistema operativo más propenso a ataques de diferentes tipos, como se puede comprobar en el artículo "Estado del arte vulnerabilidades de seguridad en sistemas operativos móviles andrioid y ios" [48], quienes exponen el sistema operativo Android y iOS para realizar una comparación de quien sería el más vulnerable, concluyendo que Android es un blanco fácil para un ataque y filtración de información de datos personales.

Android aumentó las vulnerabilidades hasta posesionarse en el puesto más alto de los sistemas operativos con más fallas en el año 2016 y 2017, como lo García Altarejos [49], en el artículo "Análisis de riegos, de vulnerabilidades y auditorías de dispositivos". Esto, junto con los datos expuestos dentro de los resultados dejando a relucir que las principales vulnerabilidades son: Data storage y Improper platform, siendo de igual manera estas dos vulnerabilidades más visibles dentro de este estudio.

Los usuarios Android tienen disponible la tienda oficial de Google para las descargas seguras de app, como lo menciona Mejía [50], en el artículo "Detectando aplicaciones maliciosas en Smartphone con sistema Android a través del uso de una aplicación", otros métodos de protección a tener en cuenta es, la instalación de un antivirus o que el usuario se informe con respecto al tema de la seguridad en dispositivos móviles, ya que la mayoría desconocen los riegos que representan los terminales.

CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES

Conclusiones

La temática de los dispositivos móviles con sistema operativo Android es amplio, abarca mucha información tanto la parte del software como hardware, constantemente aparecen nuevas informaciones ya sea de vulnerabilidades encontradas o los parches de actualización.

Las mejores herramientas para el pentesting que se usaron son; Nmap, Nessus, Metasploit Framework, DVL-DVMA y Kali Linux (backtrack), todas estas poseen diferentes características, pero que comparten algo en común, con diferentes formas y/o métodos, encontrar vulnerabilidades en la seguridad de los dispositivos.

Analizado los datos obtenidos se comprobó que, Overflow, Code Execution, Gain Privileges, son los tipos de vulnerabilidades más frecuente en el sistema operativo de Android. Los smartphones con Android, con acceso o sin acceso a internet están en constante peligro, puede sufrir daños físicos o ser víctima de un ataque por algún tipo de malware, incluso, el mismo usuario puede representar un riesgo al no tener los hábitos de buenas prácticas.

Mediante las pruebas se expuso como un smartphone, puede adquirir los permisos de superusuario, es decir, obtener el control del sistema desde la raíz, sin embargo, una consecuencia de ser root, es la disminución de la seguridad establecida por el sistema de Android; por otro lado, se demostró que un código malicioso instalado puede manipular a su antojo el terminal de manera remota.

Recomendaciones

Para mitigar los riesgos que pueden pasar al utilizar un dispositivo Android se presentan un conjunto de medidas de seguridad. Una de las principales medidas de seguridad es limitar el acceso a la información. Cuantas menos personas accedan a una información, menor será el riesgo de comprometerla. Por lo tanto, es necesario implantar en nuestra empresa un sistema que impida dar acceso a datos innecesarios, a un usuario, cliente, etc.

Poseer un sistema de copias de seguridad periódico permite que la empresa garantice que puede recuperar los datos ante una incidencia de carácter catastrófico, impidiendo la pérdida de estos y permitiendo la recuperación de la normalidad en el trabajo en apenas unos minutos.

El acceso a las distintas plataformas que utiliza la empresa (correo electrónico, servidor de copias de seguridad NAS, etc.) debe realizarse utilizando claves de seguridad (contraseñas) seguras, que impidan que puedan ser fácilmente descubiertas por piratas informáticos. El uso de contraseñas seguras es una de las medidas de seguridad informática más importantes en una empresa. Así como también tener los sistemas y/o aplicaciones actualizadas, garantizando el uso de las mejoras y parches de seguridad más recientes.

REFERENCIAS BIBLIOGRÁFICAS

[1] V. Navarro, «Desarrollo de aplicaciones en la nube,» Universitat Politècnica de València, 2017.

[2] V. Mantovani, «Autenticación de Múltiples factores,» Universidad de Buenos Aires, 2019.

[3] A. Ladino, «Vulnerabilidades y seguridad en el sistema operativo android.,» Universidad Piloto de Colombia, 2019.

[4] J. Muelaner, «El papel de los PLC en el control, la prueba y la medición industriales,» 2021.

[5] I. Adum, «La construcción de los mensajes y la influencia en los receptores,» UNIVERSIDAD COMPLUTENSE DE MADRID, 2018.

[6] A. Torres, «Sistema Operativo Android: ventajas y desventajas (2020),» 2020.

[7] F. Olmos, «El origen de Android (Características y Costos),» 2020.

[8] Á. Felguera, «GOOGLE:LAS CLAVES DEL ÉXITO Estudio de los aspectos determinantes del éxito empresarial,» 2018.

[9] J. Navarro, «Métodos de pago móvil: desarrollo y estudio comparativo,» 2020.

[10] D. Trujillo, «Reporte Final del Ejercicio Sabático ELABORACION DE MATERIALES, RECURSOS O AUXILIARES DIDACTICOS.,» 2017.

[11] M. Amine, «TransPack-App movil para la gestion del transporte de paquetes en android,» 2018.

[12] R. Invarato, «Aportes y mejoras al procesado de imágenes en android,» 2020.

[13] A. Albor, «Impacto de las aplicaciones móviles como herramienta de promoción y publicidad dentro del ITSMT,» 2019.

[14] J. Esquivel, «Las Ventajas y Desventajas de los Operadores de un Dispositivos Móvil, Según el S.O,» 2020.

[15] J. Argudo, «Estudio de la afectación a la privacidad de los usuarios que utilizan aplicaciones android desarrolladas para instituciones públicas del ecuador,» 2019.

[16] A. Pérez, «Soporte de Aplicaciones de Tiempo Real en Dispositivos Móviles,» 2020.

[17] J. Jaén, «Aplicación Android para la búsqueda de precios económicos de combustible y geolocalización de E.E.S.S.,» 2017.

[18] S. Luque, «Olvídate de esas apps que dicen optimizar tu móvil: cinco consejos para hacerlo tú mismo,» 2017. [En línea]. Available: https://www.xatakandroid.com/sistema-operativo/olvidate-de-esas-apps-que-dicen-optimizar-tu-movil-con-estos-trucos-puedes-hacerlo-tu-mismo.

[19] O. Fernández, «Juegos Adaptados,» Escuela Ténica Superior De Ingeniería Y Sistemas De Telecomunicación, 2017.

[20] C. Carvajal, «A qué distancia pierden conexión bluetooth los auriculares,» 2022 . [En línea]. Available: https://computerhoy.com/noticias/tecnologia/distancia-pierden-conexion-bluetooth-auriculares-1065329.

[21] I. Linares, «Fotos de calidad hasta en móviles baratos con la cámara Google Go: esta nueva Gcam es brutal,» 2022. [En línea]. Available: https://www.xatakandroid.com/aplicaciones-android/fotos-calidad-moviles-baratos-camara-google-go-esta-nueva-gcam-brutal.

[22] C. Torres, «La verdadera historia de Android – De Android 1.1 Banana Bread a Android 2.0 Eclair (2009),» 2021. [En línea]. Available: https://www.androidsis.com/la-verdadera-historia-de-android-de-android-1-1-banana-bread-android-2-0-eclair-2009/.

[23] J. Garcia, «Comparamos a fondo las capas de personalización de Android: así es el software de Samsung, Huawei, LG, Xiaomi, Google y más,» 2019. [En línea]. Available: https://www.xatakandroid.com/sistema-operativo/comparamos-a-fondo-capas-personalizacion-android-asi-software-samsung-huawei-lg-xiaomi-google.

[24] S. Méndez, «Aplicación móvil para el monitoreo de productos avícolas TESIS,» 2018.

[25] M. Palomino, «Aplicación android de apoyo al vuelo en globo aerostático.,» 2017.

[26] L. Sánchez, «Seguridad Informática,» Universidad del Rosario, 2020.

[27] I. Aliagas, «Análisis neuropsicofisiológico de la eficacia del emplazamiento de producto en videojuegos,» 2022.

[28] L. Torre, «Prototipo de aplicación móvil con realidad aumentada como apoyo en la adherencia a un tratamiento farmacológico,» 2018.

[29] H. González, «Aplicación android para el apoyo al diagnóstico y tratamiento del carcinoma de hígado,» Universidad De Valladolid, 2018.

[30] J. Nova, «Diseño y desarrollo de una aplicación para monitorear la concentración de CO y $\boldsymbol{C}$H4 en dispositivos móviles android.,» Universidad Pontificia Bolivariana, 2018.

[31] R. Menejías, N. Hidalgo, A. Marín y Y. Trujillo, «Procedimiento para evaluar seguridad a productos de software,» *Revista Cubana de Ciencias Informáticas,* vol. 15, nº 4, 2021.

[32] J. Guillén, «Introducción al pentesting,» Universitat de Barcelona, 2017.

[33] M. Torres, «Utilización de herramientas para pruebas de jrenetración en auditorías informáticas,» Universidad De Quintana Roo, 2019.

[34] J. Mattar y L. Cuervo, Planificación para el desarrollo en América Latina y el Caribe, CEPAL, 2017.

[35] L. Gil, «Estudio de los ataques y su defensa en la ingeniería social,» Máster Universitario en Ingeniería Informática, 2022.

[36] E. Lorenzon, Sistemas y organizaciones, Edilp, 2020.

[37] G. Pilleux, «Sistema de pruebas de penetración automatizadas para aplicaciones web,» Universidad De Chile, 2021.

[38] N. Traña, «Analisis comparativo de soluciones WAF,» Universidad Nacional de Ingeniería, 2018.

[39] M. Sosa, «Implementación de herramientas para el control y análisis de seguridad de una página web.,» Benemérita Universidad Autónoma de Puebla, 2022.

[40] A. Navarro y C. Londoño, «Análisis de caracterización de frameworks para detección de aplicaciones maliciosas en Android,» *June,* p. 12, 2014.

[41] L. Sanchez, «Propuesta de un marco de trabajo centrado en el usuario, utilizando patrones de interfaz responsiva y de interactividad en proyectos web,» 2017.

[42] C. Succi, «Tendencias actuales en el uso de dispositivos móviles,» *Salaona,* vol. 1, pp. 38-46, 2016.

[43] J. Santos, «¿Qué es el Pentesting? Tipos y cómo utilizarlo para prevenir ciberataques,» *Delta Protect,* pp. 1-2, 2023.

[44] S. Bortnik, «Pruebas de penetración para principiantes : 5 herramientas para empezar,» *DGTIC,* 2017.

[45] L. Ardila y V. Hugo, «Vulnerabilidades más importantes en plataformas Android,» *CPE,* pp. 1-8, 2016.

[46] M. Álvarez y H. Montoya, «Ciberseguridad en las redes móviles de telecomunicaciones y su gestión de riesgos,» *Ingeniería de Desarrollo ,* vol. 38, nº 2, 2021.

[47] Janosik, Steven M, «ANÁLISIS DE VULNERABILIDADES Y SEGURIDAD DE DISPOSITIVOS MÓVILES CON SISTEMA OPERATIVO iOS 6.1.4 JEYSON,» *NASPA Journal,* vol. 42, nº 4, p. 1, 2005.

[48] UNIVERSIDAD, YAMIR ASMIRIO MUÑOZ CACERES, «ESTADO DEL ARTE VULNERABILIDADES DE SEGURIDAD EN SISTEMAS OPERATIVOS MÓVILES ANDRIOID Y IOS,» p. 183, 2019.

[49] Constitución de la República, Régimen de Buen Vivir, Quito: Asamblea, 2008.

[50] Ley Orgánica de Protección de Datos Personales, Datos Personales, Quito: Asamblea, 2021.

[51] Begoña Bermejo Fraile, «Estudios experimentales,» de *Matronas,* 2008.

[52] J. Martínez y L. Rojas, «Vulnerabilidad en dispositivos móviles con sistema operativo Android,» *Cuadernos de Actulización,* vol. 7, nº 7, pp. 55-65, 2015.

[53] García Altarejos, Carlos, «Seguridad en Smartphones: Análisis de riesgos, de vulnerabilidades y auditorías de dispositivos,» p. 122, 2017.

[54] Mejía, Jezreel, «Detectando aplicaciones maliciosas en Smartphone con sistema Android a través del uso de una aplicación,» *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação*, nº 31, p. 82, 2019.

ANEXOS

Imágenes detalladas del cronograma de actividades que se desarrollaron para el proceso de desarrollo de la investigación.

Tabla 5 Cronograma de actividades para el desarrollo de la investigación

	Nombre de tarea	Fecha de inicio	Fecha final	Duración
		25/10/2022 09:00	20/02/2023 19:00	85d
1	**OBJETIVO 1: Examinar Información**	**25/10/2022 09:00**	21/11/2022 19:00	20d
1.1	TAREA 1: Busquedad de 50 articulos o revistas	25/10/2022 09:00	07/11/2022 19:00	10d
1.2	TAREA 2: Clasificación de herramientas a utilizar en la practica	08/11/2022 09:00	21/11/2022 19:00	10d
	Añadir una tarea | Añadir un hito			
2	**OBJETIVO 2: Identificar la informacion referente a las versiones propuestas**	**22/11/2022 09:00**	12/12/2022 19:00	15d
2.1	TAREA 1: Investigación de las caracteristicas de cada version de Android presentada	22/11/2022 09:00	28/11/2022 19:00	5d
2.2	TAREA 2: Identificar las ventajas y desventajas de ambas versiones	29/11/2022 09:00	05/12/2022 19:00	5d
2.3	TAREA 3: Identificar las vulnerabilidades presentadas mediante investigación	06/12/2022 09:00	12/12/2022 19:00	5d
	Añadir una tarea | Añadir un hito			
3	**OBJETIVO 3: Implementacion del pentensting con las herramientas elegidas**	**13/12/2022 09:00**	20/02/2023 19:00	50d
3.1	TAREA 1: Desarrollar el analisis de pentesting en la primera version	13/12/2022 09:00	26/12/2022 19:00	10d
3.2	TAREA 2: Desarrollar el analisis de pentesting en la segunda version	27/12/2022 09:00	09/01/2023 19:00	10d
3.3	TAREA 3: Realizar una comparativa de vulnerabilidades presentadas	10/01/2023 09:00	23/01/2023 19:00	10d
3.4	**OBJETIVO 4: Desarrollar Politicas de seguridad**	**24/01/2023 09:00**	20/02/2023 19:00	20d
3.4.1	TAREA 1: Identificar y recolectar informacion referente a los problemas encontrados durante el pentensting	24/01/2023 09:00	06/02/2023 19:00	10d
3.4.2	TAREA 2: Presentar las politicas de seguridad	07/02/2023 09:00	20/02/2023 19:00	10d

Tabla 6 Modelo calidad en producto

Modelo Calidad en producto											
#	*Herramienta*	*FASE*	*Funcionalidad*	*Rendimiento*	*Usabilidad*	*Fiabilidad*	*Seguridad*	*Compatibilidad*	*Mantenibilidad*	*Portabilidad*	*Total*
1	Kali Linux	Análisis/Explotación de vulnerabilidades	4,0	4	4,2	3,8	4	3,6	4,7	4	32,2
2	Maltego	Recopilación de información	3	3	4	3	4	4	4	4	27,7
3	TumbleRF	Recopilación de información	3	4	3	3	3	3	3	3	24,3
4	Snort	Post – Explotación	4	3	3,8	3,5	3,8	3,4	3,7	3	27,9
5	Burp Suite	Análisis/Explotación de vulnerabilidades	4,3	4,5	4,3	3,8	4	4,2	4,3	4	33,5
6	Empire	Post – Explotación	3	3,5	3,3	3,8	4	3,6	4	4	29,5
7	Netcat	Post – Explotación	4	4	3,3	3	4,2	3,4	4	3,5	29,1
8	Dnsmap	Recopilación de información	4	5	4	4	4	4	4	4	33,5
9	OWASP Zap Proxy	Análisis/Explotación de vulnerabilidades	4,0	4,5	4,2	3,8	4	4	4,3	4	32,8
10	Wireshark	Recopilación de información	5	4,5	4,2	4,3	4,4	4,4	4	4,5	34,9
11	Android Studio	Recopilación de información	3	3	3,3	3,3	3,2	3,4	3	4	26,4
12	Nessus	Análisis/Explotación de vulnerabilidades	4	3,5	4	4,3	4	4	3,7	3,5	31,3
13	Nmap	Recopilación de información	4	4,5	4,3	3,8	3,8	3,6	3,7	3,5	31,5
14	GNU Radio	Recopilación de información	3	3,5	4,3	3,8	3,6	3,8	3,7	4,5	30,5
15	Xarp	Análisis/Explotación de vulnerabilidades	4	4,5	4,5	3,8	3,8	3,2	4,3	2	29,8
16	METASPLOIT	Post – Explotación	4	4	4,3	4,3	4	4	4,3	3,5	32,8
17	VIRTUAL BOX	Análisis/Explotación de vulnerabilidades	3	3	4,3	4	3,8	3,6	3,7	3	28,7
18	Dradis	Reporte	4	4	4,2	4,3	4,2	4	4,3	4	33,3
19	Faraday	Reporte	3	3,5	3,8	3,5	3,6	3,6	4	3,5	28,5
20	Hydra	Análisis/Explotación de vulnerabilidades	4	3	3,7	3,8	3,4	3,2	3,7	3,5	27,9

Tabla 7 Modelo calidad de uso

Modelo Calidad de uso								
#	*Herramienta*	*FASE*	*Efectividad*	*Eficiencia*	*Satisfacion*	*Seguridad*	*Contexto Global*	*TOTAL*
1	Kali Linux	Análisis/Explotación de vulnerabilidades	4	4	4	4	4	20
2	Maltego	Recopilación de información	3	4	4	3	3	17
3	TumbleRF	Recopilación de información	3	3	3	3	3	15
4	Snort	Post – Explotación	3	4	4	3	4	18
5	Burp Suite	Análisis/Explotación de vulnerabilidades	5	5	4	5	4	23
6	Empire	Post – Explotación	4	4	3	3	3	17
7	Netcat	Post – Explotación	4	4	4	5	4	21
8	Dnsmap	Recopilación de información	4	3	3	3	4	17
9	OWASP Zap Proxy	Análisis/Explotación de vulnerabilidades	4	3	3	3	3	16
10	Wireshark	Recopilación de información	5	4	4	4	5	22
11	Android Studio	Recopilación de información	3	3	3	3	3	15
12	Nessus	Análisis/Explotación de vulnerabilidades	4	4	3	4	3	18
13	Nmap	Recopilación de información	4	4	3	3	3	17
14	GNU Radio	Recopilación de información	3	3	4	4	4	18
15	Xarp	Análisis/Explotación de vulnerabilidades	3	3	3	3	4	16
16	METASPLOIT	Post – Explotación	5	5	5	5	4	24
17	VIRTUAL BOX	Análisis/Explotación de vulnerabilidades	4	3	3	4	3	17
18	Dradis	Reporte	5	5	5	5	4	24
19	Faraday	Reporte	4	3	3	3	4	17
20	Hydra	Análisis/Explotación de vulnerabilidades	4	4	4	3	4	19