

# VULNERABILIDADES Y SEGURIDAD EN EL SISTEMA OPERATIVO ANDROID.

Antolínez Ladino Andrea.  
ananla2181@gmail.com  
Universidad Piloto de Colombia

**Abstract**—*This document shows the main features of the Android operating system and its architecture, explaining the most important factors of the security model on this platform and identifying some vulnerabilities.*

**Resumen**—El presente documento muestra las características principales del sistema operativo Android y su arquitectura, explicando los factores más importantes del modelo de seguridad en esta plataforma e identificando algunas vulnerabilidades.

**Índice de Términos**—Android, seguridad, vulnerabilidad.

## I. INTRODUCCIÓN

Desplazando al PC de escritorio como herramienta básica en los hogares, inclusive, comenzado a arrinconar los equipos corporativos, Android ha logrado integrar diferentes servicios online, comunicación en entornos virtuales, envío de correos, edición de documentos, entre otros.

Estas ventajas son aprovechadas tanto por los usuarios que tienen un mayor poder adquisitivo como los que tienen pocas opciones económicas permitiendo que sea el sistema operativo móvil más utilizado en el mundo, con una presencia en el mercado del 88% al terminar el 2018 [4].

Android Inc. era una pequeña compañía de Palo Alto, California fundada en 2003, que desarrollaba software para teléfonos móviles. Fue en el 2005 cuando Google la adquiere, y en 2007, junto con la alianza comercial Open Handset Alliance (OHA), un consorcio de 47 empresas de hardware, software y telecomunicaciones dedicadas al fomento de

estándares abiertos para dispositivos móviles, presentan la primer versión Android 1.0 Apple Pie.

Al ser Android un software de código abierto, y de libre descarga, su creciente demanda, y la falta de conciencia de los usuarios al implementar seguridad,

lo hace vulnerable a diferentes tipos de ataques, generando una afectación tanto en los diferentes servicios como como en la integridad de los datos, factores que lo convierten en el objetivo número uno en cuanto amenazas.

## II. SISTEMA OPERATIVO ANDROID

Android es un sistema operativo Open Source basado en el Kernel de Linux. Es un sistema libre, gratuito y multiplataforma, con gran capacidad de adaptación a todo tipo de dispositivos lo que le confiere un gran potencial de desarrollo.

### A. Características

Dentro de las principales características están:

- **Framework de aplicaciones:** permite el reemplazo y la reutilización de los componentes.
- **Navegador integrado:** basado en el motor open Source Webkit.
- **SQLite:** base de datos para almacenamiento estructurado que se integra directamente con las aplicaciones.
- **Multimedia:** Soporte para medios con formatos comunes de audio, video e imágenes planas (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- **Máquina virtual Dalvik:** Base de llamadas de instancias muy similar a Java.
- **Telefonía GSM:** dependiente del terminal.
- **Bluetooth, EDGE, 3g y Wifi:** dependiente del terminal.

- **Cámara, GPS, brújula y acelerómetro:**  
Dependiente del terminal

**B. Arquitectura**

A continuación, se describen las capas de la arquitectura empleada en Android. Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez los suyos propios a las capas de niveles superiores.

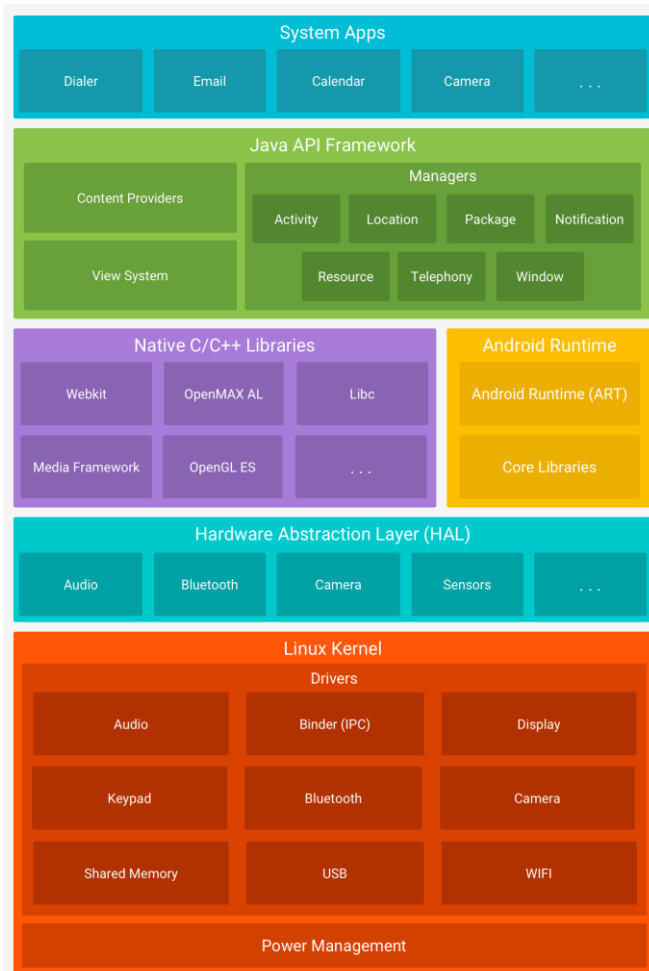


Fig 1. Arquitectura Android. [5]

1) *El núcleo* del sistema operativo Android está basado en el kernel de Linux versión 2.6, similar al que puede incluir cualquier distribución de Linux, como Ubuntu, solo que adaptado a las características del hardware en el que se ejecutará Android, es decir, para dispositivos móviles.

El núcleo actúa como una capa de abstracción entre el hardware y el resto de las capas de la arquitectura.

El desarrollador no accede directamente a esta capa, sino que debe utilizar las librerías disponibles en capas superiores. Si se necesita hacer uso de la cámara, el sistema operativo se encarga de utilizar la que incluya el teléfono, sea cual sea. Para cada elemento de hardware del teléfono existe un controlador (o driver) dentro del kernel que permite utilizarlo desde el software.

El kernel también se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo en sí: procesos, elementos de comunicación (networking), etc.

2) *HAL (Hardware Abstraction Layer)*

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

3) *Librerías*

La siguiente capa que se sitúa justo sobre el kernel la componen las bibliotecas nativas de Android, también llamadas librerías. Están escritas en C o C++ y compiladas para la arquitectura hardware específica del teléfono. Estas normalmente están hechas por el fabricante, quien también se encarga de instalarlas en el dispositivo antes de ponerlo a la venta. El objetivo de las librerías es proporcionar funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se llevan a cabo de la forma “más eficiente”.

Entre las librerías incluidas habitualmente se encuentran OpenGL (motor gráfico), Bibliotecas multimedia (formatos de audio, imagen y video), Webkit (navegador), SSL (cifrado de comunicaciones), FreeType (fuentes de texto), SQLite (base de datos), entre otras.

#### 4) *Entorno de Ejecución*

No se considera una capa en sí mismo, dado que también está formado por librerías. Aquí se encuentran las librerías con la funcionalidad habitual de Java así como otras específicas de Android. El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik. Las aplicaciones se codifican en Java y son compiladas en un formato específico para que esta máquina virtual las ejecute. La ventaja de esto es que las aplicaciones se compilan una única vez y de esta forma estarán listas para distribuirse con la total garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera la aplicación.

#### 5) *Java API Framework*

Todo el conjunto de funciones del SO Android está disponible mediante API escritas en el lenguaje Java. Estas API son los cimientos que se necesitan para crear apps de Android simplificando la reutilización de componentes del sistema y servicios centrales y modulares, como los siguientes:

- **Un sistema de vista** enriquecido y extensible que se puede usar para compilar la IU de una app; se incluyen listas, cuadrículas, cuadros de texto, botones e incluso un navegador web integrable.
- **Administrador de recursos** que brinda acceso a recursos sin código, como strings localizadas, gráficos y archivos de diseño.
- **Administrador de notificaciones**, permite que todas las apps muestren alertas personalizadas en la barra de estado.
- **Administrador de actividad** que administra el ciclo de vida de las apps y proporciona una pila de retroceso de navegación común.
- Y por último **Proveedores de Contenido** que permiten que las apps accedan a datos desde otras apps, como la app de Contactos, o compartan sus propios datos.

Los desarrolladores tienen acceso total a las mismas API del framework que usan las apps del sistema Android.

#### 6) *System Apps*

Este nivel incluye tanto las incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan los servicios, las API y librerías de los niveles anteriores.

#### C. *Versiones*

Android ha visto numerosas actualizaciones desde su liberación inicial. Estas actualizaciones al sistema operativo base típicamente arreglan fallos y agregan nuevas funciones.

Las versiones tienen nombre de postres en inglés y cada versión que cambia, continúa de forma incremental en el alfabeto, es decir que si el primer nombre inicio con A, el siguiente con B, el siguiente C y así sucesivamente. Por ejemplo:

Versión 1.0 Apple Pie - septiembre del 2008.

Versión 1.1 Banana Bread - febrero 2009.

Versión 1.5 Cup Cake- abril 2009

Versión 1.6 Donut - septiembre 2009

Versión 2.0 Eclair - octubre 2009

Versión 2.2 Froyo - mayo 2010

Versión 2.3 Gingerbread - diciembre 2010

Versión 3.0 Honeycomb - febrero 2011

Versión 4 .0 Ice Cream Sandwich - octubre 2011

Versión 4.1 Jelly Bean - Salió en julio 2012

Versión 4.4 KitKat - Salió en octubre 2013

La reiterada aparición de nuevas versiones que, en muchos casos, no llegan a funcionar correctamente en el hardware diseñado para versiones previas, hacen que Android sea considerado uno de los elementos promotores de la obsolescencia programada.

Sin embargo, las primeras versiones de Android tenían elementos de seguridad que fueron mejorando a medida que iban surgiendo nuevas actualizaciones. Además de los beneficios comentados como la implementación de nuevas funciones y arreglo de fallos, cada última versión de Android trae varias mejoras de seguridad que refuerzan la seguridad del sistema.

### III. ESQUEMA DE SEGURIDAD DE ANDROID

La implementación del modelo de seguridad de Android se lleva a cabo a lo largo de toda la arquitectura del sistema. A continuación, se detalla los aspectos más importantes de dicho modelo.

#### 1) Application Sandbox

Android implementa el principio de *mínimo privilegio* haciendo que cada aplicación se ejecute en, como se lo denomina, un sandbox (arenero); es decir, forzando a que cada aplicación sólo pueda tener acceso irrestricto a sus propios recursos. Por defecto, ninguna aplicación puede acceder a otras partes del sistema (por ejemplo, cámara fotográfica del teléfono móvil, libreta de contactos, otra aplicación); para ello se debe obtener el permiso correspondiente.

El mecanismo de Application Sandbox es implementado a nivel de kernel, asignando un identificador único de usuario UID para cada aplicación con pocos privilegios.

Se configuran todos los archivos creados en el almacenamiento interno para que, por defecto, solo puedan ser accedidos por el UID de la aplicación que los creó. Los archivos creados en almacenamiento externo, como tarjetas SD, pueden ser leídos y modificados por cualquier aplicación con la sola restricción de tener el permiso adecuado para escribir en almacenamiento externo, de necesitar hacerlo.

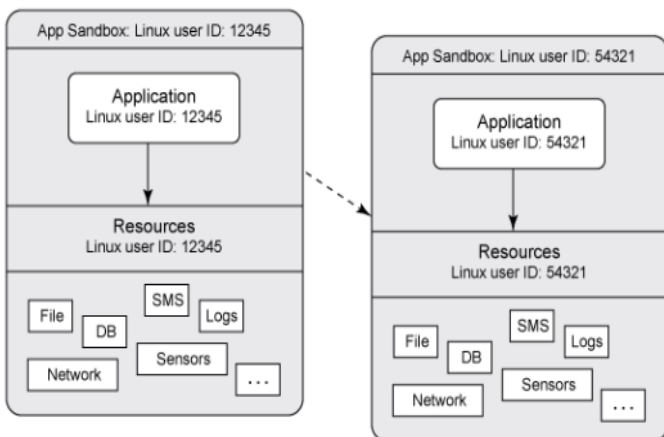


Fig.2. Dos aplicaciones en diferentes procesos con diferentes UID. [7]

Adicionalmente, se ejecuta cada aplicación como un proceso separado del resto, con su propio espacio de direcciones. Debido a esto, valiéndose de los mecanismos de protección de los sistemas Linux, Android garantiza que ninguna aplicación puede, al menos por defecto, acceder a los recursos de otra (ya que tienen distintos UIDs) ni a los recursos del sistema (ya que el UID asignado a cada aplicación tiene pocos privilegios) y, dado que cada aplicación corre en un proceso diferente, la única forma de interacción entre ellas es a través de mecanismos de IPC (Inter Process Communication).

#### 2) Application Signing

Todas las aplicaciones Android deben estar firmadas digitalmente de forma tal que sus claves privadas solo sean conocidas por sus respectivos desarrolladores. Además, se deben incluir certificados que identifiquen el origen de sus claves públicas. Dichos certificados no necesitan estar firmados por una entidad de certificación; de hecho, la práctica más frecuente es que estén firmados por los propios desarrolladores (self-signed certificates).

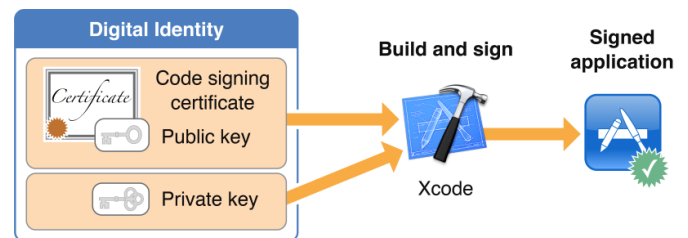


Fig 3. Cómo firmar una App. [7]

Los certificados son utilizados por Android para distinguir cuando dos aplicaciones distintas fueron hechas por el mismo desarrollador. Esta información se torna relevante para el sistema a la hora de decidir si conceder permisos de tipo signature o autorizar a dos aplicaciones a tener el mismo UID.

Varias aplicaciones firmadas con el mismo certificado pueden pedir tener el mismo UID y, de esta forma, tener la posibilidad de compartir sus recursos entre sí, como también ejecutarse en el mismo proceso

### 3) *Permisos*

Se necesita algún mecanismo para que una aplicación pueda acceder a los recursos que necesita para llevar a cabo su objetivo y, al mismo tiempo, que se tenga un cierto control sobre a quién se le permite el acceso a los mismos. La solución que provee Android es un sistema de permisos que constituye una parte principal de su modelo de seguridad. El funcionamiento básico del sistema de permisos en Android es el siguiente: una aplicación declara estáticamente el conjunto de permisos que necesita para obtener las capacidades adicionales que no tiene por defecto. Al momento de instalar una aplicación se otorgan o no los permisos solicitados basándose, dependiendo del tipo de permiso, o bien en el certificado de la misma o, con mayor frecuencia, en la autorización directa del usuario.



Fig 4. Solicitud de permiso al usuario desde la App Messenger.[7]

También existen permisos que son otorgados automáticamente por el sistema al ser solicitados, sin necesidad de pedir la autorización explícita del usuario. Por otro lado, si alguno de los permisos no es concedido, típicamente, la aplicación no puede instalarse; Android no cuenta con un mecanismo que otorgue permisos dinámicamente (en tiempo de ejecución)

Cada permiso se identifica con un nombre/texto y, en general, se puede distinguir dos clases de ellos: los definidos por las mismas aplicaciones con el propósito de auto-protección y los que están predefinidos por Android, que controlan el acceso a recursos y servicios del sistema (por ejemplo, acceso a libreta de contactos, envío de SMS). Todo permiso tiene asignado un nivel de protección que caracteriza

el riesgo potencial asociado al mismo.

Dependiendo del nivel de protección de un permiso dado, el sistema define el procedimiento a seguir para determinar si se concede dicho permiso a una aplicación solicitante.

### 4) *Delegación de Permisos*

Android provee, básicamente, dos mecanismos que permiten a una aplicación dada delegar sus propios permisos a otra; posibilitando, de esta forma, que esta última pueda realizar una determinada acción para la que no cuenta, originalmente, con los permisos necesarios. Estos mecanismos son: pending intents y URI permissions.

El concepto de pending intent es bastante simple: un desarrollador define un intent para realizar una determinada acción (por ejemplo, iniciar una actividad) como se hace normalmente. Sin embargo, en lugar de pasar dicho intent al método que realiza la acción (por ejemplo, `startActivity()`), este se pasa a un método especial que crea un objeto

`PendingIntent` asociado a la acción deseada. Dicho objeto no es más que una referencia que puede ser concedida a otra aplicación para que ésta invoque, en el momento que disponga, la acción en cuestión, contando, al hacerlo, con los permisos e identidad de la aplicación original. Por otro lado, si una aplicación crea un objeto `PendingIntent`, a pesar de que ésta deje de ejecutarse, todas las aplicaciones que lo recibieron podrán seguir usándolo. Sin embargo, el mismo solo podrá ser cancelado por la aplicación que lo creó y, de suceder esto, todas las aplicaciones que estén usando dicho objeto deben dejar de hacerlo.

### 5) *Android Manifest*

Toda aplicación Android debe incluir en su directorio raíz un archivo XML llamado `AndroidManifest`. En este archivo se declaran todos los componentes que forman parte de la aplicación en cuestión, junto con algunas características de los mismos que se definen de forma estática. Además, se identifican los permisos que solicitará la aplicación al momento de instalarse y los que serán exigidos por la misma. Adicionalmente, en este archivo se especifica

si se autoriza el mecanismo de delegación conocido como URI permissions, sobre los content providers de la aplicación. Uno de los elementos más importantes del cuerpo de un AndroidManifest es, en donde, además de especificar algunas características de la aplicación en cuestión, se incluyen los elementos que describen los componentes de la misma. Cada componente, dependiendo de su tipo, se declara utilizando alguno de los siguientes elementos: <activity>, <service>, <provider> y <receiver>. En los mismos se especifican las propiedades de cada componente perteneciente a la aplicación, tanto en sus atributos como en los elementos que a su vez pueden llegar a contener.

#### IV. VULNERABILIDADES EN ANDROID

Como se observó en el apartado anterior, Android implementa herramientas que vienen con el sistema operativo para brindar la posibilidad de proteger los dispositivos móviles. Sin embargo, la naturaleza abierta de Android, la facilidad con que se pueden crear aplicaciones y la amplia variedad de mercados de aplicaciones (no oficiales) influyen en la seguridad.

En la era de la tecnología móvil, la seguridad se ha convertido en un aspecto a tener en cuenta cuando se navega por internet o se descargan aplicaciones de las tiendas de Google, Apple o Amazon mediante estos dispositivos.

Por ello, es importante minimizar los riesgos de seguridad para proteger la información confidencial en los dispositivos móviles, tanto si es para uso personal o uso de los dispositivos para el trabajo diario.

La mayoría de las aplicaciones Android se encuentran disponibles en Google Play, antes conocida como Android Market, aunque también existen desarrolladores que crean aplicaciones que puedes encontrar en sitios web independientes. En este último caso, es recomendable estar seguro de

que el sitio web en cuestión es confiable para evitar la instalación de virus o malware en general en los dispositivos Android.

Los usuarios son menos conocedores de la seguridad sobre el malware en los dispositivos móviles. Los ciber-atacantes intentan engañar a los usuarios para que descarguen aplicaciones maliciosas o con técnicas de ingeniería social para engañar a los usuarios a hacer clic en vínculos cuestionables.

Para Android, en el año 2018 se publicaron 517 fallos de seguridad (Fig.5), experimentando un decrecimiento del 39% respecto al total de vulnerabilidades reportadas para esta plataforma en 2017, año en que la cantidad de CVE marcó un pico histórico alcanzando los 842 fallos publicados.



Figura 5. Vulnerabilidades de criticidad alta en Android. [4]

Durante la segunda mitad del año 2018, importantes vulnerabilidades han sido descubiertas, como RAMPAGE [11], una variación de Rowhammer capaz de modificar los datos almacenados en memoria, que afectó a millones de dispositivos, pudiendo causar pérdida de información o acceso no autorizado.

Sin embargo, el porcentaje de vulnerabilidades graves con criticidad igual o mayor a siete también ha decrecido, constituyéndose como el 48% del total (un decremento del 53% desde el año 2017). De esos fallos graves, el 20% permitiría la ejecución de código por parte de un atacante [4].

Además de haber tenido un menor número de fallos, las detecciones de malware para Android también han

experimentado un decrecimiento del 23% respecto a 2017. Afortunadamente, esta disminución no se plantea como un fenómeno aislado, sino como una tendencia sostenida desde 2016.

Los países donde se registró la mayor cantidad de detecciones de malware para Android, fueron Rusia (16%), Irán (14%) y Ucrania (8%). El primer país latinoamericano en aparecer dentro del ranking internacional es México (3%) en el sexto puesto, seguido por Perú (2%) en el décimo lugar. Si se toma en cuenta solamente detecciones en países latinoamericanos, en 2018 los países con mayores detecciones fueron México (26%), Perú (17%) y Brasil (11%) [Fig. 6].

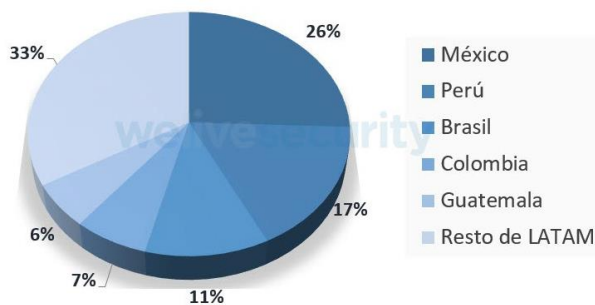


Fig 6. Detecciones de malware para Android en Latinoamérica en 2018. [4]

Si se considera el porcentaje de detecciones de Android por sobre el total de detecciones en cada país, se observa que el país latinoamericano más afectado fue Guatemala (0,95%), seguido de Nicaragua (0,76%) y Honduras (0,68%).

Para el año en curso (2019) se sigue observando dicha tendencia en la disminución de detecciones de vulnerabilidades. En el primer semestre se han registrado 84 vulnerabilidades [Fig.7], de las cuales 4 son de Negación de Servicio (Denial of Service), 24 de Código de Ejecución (Execute Code), 4 de Desbordamiento (Overflow), 10 de Corrupción de Memoria (Memory Corruption), 9 Omisión de Restricción (Bypass Something), 5 para obtener información (Gain Information), 1 para obtención de privilegios (Gain Privileges), entre otros; la mayoría con un puntaje de criticidad alta ( $\geq 7$ ) [12].

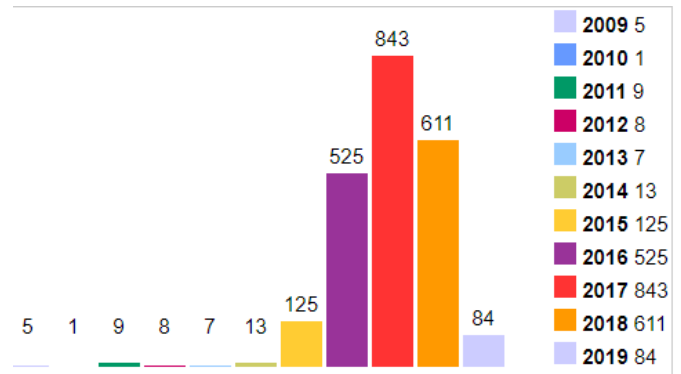


Fig. 7. Vulnerabilidades por año. [12]

En lo que lleva el segundo semestre se han detectado 62 vulnerabilidades adicionales [11].

Dentro de los objetos maliciosos más usados detectados en teléfonos Android pueden dividirse en tres grupos principales:

- Virus troyanos de SMS
- Módulos de publicidad
- Exploits para ganar acceso al sistema raíz de los teléfonos

La fiebre por las criptomonedas también plantea preocupaciones en términos de seguridad desde el punto de vista de las vulnerabilidades que puedan contener las apps de billeteras electrónicas.

Otro tipo de códigos maliciosos que ha dominado el panorama móvil, es el malware bancario.

En los últimos años, el malware bancario para Android no ha dejado de incrementarse. Estos troyanos no solo se propagan mediante sitios no oficiales, sino que también pueden encontrarse en tiendas oficiales de apps. Algunos de estos códigos están equipados para realizar ataques a gran escala, apuntando a capturar datos bancarios de un amplio abanico de entidades financieras a lo largo del mundo.

Además del malware en la Play Store, otro factor que incrementa el riesgo que corren los usuarios de Android es la decisión de algunos desarrolladores de remover sus productos de la Play Store, como fue el caso del videojuego Fortnite. Este enfoque es peligroso porque puede volver a los usuarios más susceptibles de caer en estafas.

Fortnite tiene más de 125 millones de usuarios en el mundo, lo que resulta un gran incentivo para que cibercriminales creen campañas de propagación simulando ser este juego.

## V. SEGURIDAD EN EQUIPOS MÓVILES CORPORATIVOS

La práctica de la administración de vulnerabilidades para mitigar los riesgos de seguridad en equipos de escritorio puede, y debe, extenderse a terminales móviles.

Uno de los desafíos constantes a los que se enfrentan los equipos de seguridad de empresas hoy en día es la falta de visibilidad sobre su panorama de amenazas, que incluye malware, amenazas de red, riesgos de aplicaciones y vulnerabilidades de los sistemas operativos, un creciente punto débil de las empresas.

El programa Common Vulnerabilities and Exposure (CVE) identifica cientos de vulnerabilidades de los sistemas operativos móviles cada año, algunas de ellas críticas. Apple y Google lanzan actualizaciones de seguridad o parches mensualmente, pero los usuarios de dispositivos móviles no siempre son conscientes de las actualizaciones; e incluso cuando lo están, no siempre los instalan o, si desean instalarlos, las actualizaciones no siempre están disponibles para su dispositivo. En muchos casos, las vulnerabilidades permanecen abiertas a la explotación porque los equipos de seguridad no son lo suficientemente conscientes o no tienen las herramientas para responder.

En la red y en los programas tradicionales de seguridad para terminales, los administradores han usado la práctica de toda la vida de la administración de vulnerabilidades para identificar, priorizar y remediar las vulnerabilidades de software. Entre otros beneficios, la administración de vulnerabilidades proporciona la capacidad básica de ver todas las vulnerabilidades conocidas y su gravedad, a las que está expuesta una organización. Una vez que hay una imagen más clara del panorama de vulnerabilidad, los administradores pueden priorizar y responder al riesgo de varias maneras, como instalar un parche de seguridad (reparación) o eliminar el sistema vulnerable de la red (mitigación).

Al igual que la administración de vulnerabilidades se ha convertido en un componente central de los programas de seguridad para endpoints, también debe usarse en la seguridad para dispositivos móviles. Después de todo, una terminal en dispositivo móvil sigue siendo un endpoint, por lo que la seguridad móvil es una parte integral de ella. Los programas de seguridad para empresas que no extienden la administración de vulnerabilidades a los dispositivos móviles básicamente asumen que estos dispositivos no son susceptibles a las vulnerabilidades del sistema operativo que amenazan los endpoints tradicionales.

En última instancia, cada sistema operativo es vulnerable al riesgo. En dispositivos móviles, el riesgo es aún mayor: estos dispositivos siempre están encendidos, siempre están conectados y los usuarios se comportan de forma menos segura con ellos (se conectan a más redes libremente, descargan aplicaciones que pueden ser de riesgo, y usan mucho el correo electrónico y las aplicaciones de mensajería instantánea, etc.). Solo se necesita explotar una vulnerabilidad del sistema operativo para dar a un atacante acceso completo al contenido de un dispositivo, a las redes a las que se conecta, a la nube y a los sistemas empresariales locales a los que accede.

Además, los hackers pueden obtener acceso a los sensores de un dispositivo, lo que les permite espiar a las víctimas 24/7 sin que ellos lo sepan. La premisa es

clara: si los administradores utilizan la administración de vulnerabilidades para reducir el riesgo de puntos vulnerables de software en las terminales Windows o Mac, deben utilizarla sin dudar para reducir el riesgo en todo sistema por más avanzado que esté presenta vulnerabilidades sean pocas o muchas y que como usuarios y especialistas en seguridad informática se deben tomar medidas preventivas y sensibilizar a las personas en nuestro entorno para ser consciente sobre el uso seguro para este tipo de dispositivos. dispositivos iOS y Android.

El problema es que los programas de administración de vulnerabilidades no tienen la misma supervisión y control sobre el diverso panorama de los dispositivos móviles, como con las terminales tradicionales. En los dispositivos móviles, son los usuarios finales (y, a menudo, el operador de telefonía móvil o el fabricante que están utilizando), no el departamento de TI, los que deciden qué firmware usar y cuándo instalar las actualizaciones de seguridad. Los administradores de TI no pueden realizar evaluaciones de vulnerabilidad activa en dispositivos móviles y no pueden forzar actualizaciones o parches de seguridad.

El riesgo de las amenazas en dispositivos móviles está creciendo y, así como las organizaciones utilizan la administración de vulnerabilidades para proteger sus endpoints tradicionales de las vulnerabilidades de software, harían bien en usarlas para sus terminales modernas. Las herramientas de administración de vulnerabilidades para dispositivos móviles pueden proporcionar visibilidad sobre las vulnerabilidades conocidas que afectan a los dispositivos iOS y Android, pero las herramientas que también brindan una evaluación de riesgos y posibilidades de acción serán más efectivas para garantizar la reducción continua del riesgo.

## VI. BYOD: BRING YOUR OWN DEVICE

Bring your own device, (Trae tu propio dispositivo), es una política empresarial que consiste en que los empleados lleven sus propios dispositivos personales (portátiles, tabletas, móviles...) a su lugar de trabajo para tener acceso a recursos de la empresa tales como correos electrónicos, bases de datos y archivos en servidores así como datos y aplicaciones personales. También se le conoce como “Bring Your Own Technology BYOT, (trae tu propia tecnología), ya que de esta manera se expresa un fenómeno mucho más amplio puesto que no sólo cubre al equipo sino que también cubre al software.



Fig. 8. Protección de la Información en dispositivos tipo BYOD. [13]

BYOD está haciendo grandes cambios en el mundo de los negocios ya que los beneficios para la empresa son una disminución de costes en equipos y una mayor productividad porque el empleado maneja un equipo que ya conoce y domina, y porque acceder a la infraestructura tecnológica de la empresa desde sus propios equipos puede facilitar la adopción de horarios de trabajo más flexibles a través del teletrabajo. Desde el punto de vista de los empleados, utilizan el equipo que han elegido, no necesitan cargar con un teléfono y/o tableta adicional a los suyos y utilizan las aplicaciones a las que están acostumbrados.

De no tomarse controles, esta práctica puede ser muy perjudicial para la empresa ya que puede dejar fisuras donde se puede filtrar la información o introducir aplicaciones malignas a la red [Fig. 8]. Por ejemplo: si un empleado utiliza un Smartphone para

acceder a la red de la compañía y luego lo pierde, la información confidencial guardada en el teléfono podría llegar a manos no confiables.

Uno de los inconvenientes más grandes que existen con BYOD consiste en rastrear y controlar acceso a redes privadas y corporativas. Los dispositivos a conectarse por BYOD, tienen que tener configurados un protocolo seguridad inalámbrica para evitar accesos no deseados. Principalmente WPA2-Enterprise, el cual es el único nivel de seguridad de conexión inalámbrica que permite las tres formas de seguridad: cifrado de información en tránsito, autenticidad de usuario y autenticidad de red.

## VII. CONCLUSIONES

Actualmente la mayoría de usuarios no implementan seguridad, lo que se refleja en la cantidad de amenazas que se vienen presentando y que en muchas de las ocasiones tienen éxito.

El uso tan extendido de los dispositivos móviles, en cuanto a uso de comunicación y entretenimiento, ha hecho que los ciberdelincuentes lo vean como un nicho de mercado a explotar, y al día de hoy, los dispositivos móviles se han convertido en uno de los focos principales de ataques informáticos.

Las vulnerabilidades que se presentan en los dispositivos móviles con Android, son en su mayoría por la falta de conocimiento de los usuarios y por la poca precaución que tienen al instalar aplicaciones.

Todo sistema por más avanzado que esté presenta vulnerabilidades sean pocas o muchas y que como usuarios y especialistas en seguridad informática se deben tomar medidas preventivas y sensibilizar a las personas en nuestro entorno para ser consciente sobre el uso seguro para este tipo de dispositivos.

La seguridad informática debe ser considerada como una inversión; se trata de tiempo y esfuerzo bien invertidos, ya que previene riesgos y mitiga los

efectos nocivos que acarrea ser víctima de cibercriminales.

Ninguna empresa que busque ser competitiva puede privarse del uso de la tecnología para dar acceso a la información que requieran sus empleados en cualquier lugar y tiempo, pero también deben garantizar de forma efectiva la protección móvil para mitigar los riesgos que conlleva entregar información de la empresa en dispositivos en lo que no se tiene pleno control.

## REFERENCIAS

- [1] GSMA. “Calidad de los Servicios Móviles”. 2015. Disponible en: <https://www.gsma.com/latinamerica/wp-content/uploads/2015/02/calidad-servicios-moviles-latam-2015.pdf>.
- [2] GSMA. “Seguridad, Privacidad y Protección del Ecosistema Móvil”. 2017. Disponible en: <https://www.gsma.com/latinamerica/wp-content/uploads/2017/06/Seguridad-privacidad-y-proteccion%CC%81n-del-ecosistema-mo%CC%81vil.pdf>.
- [3] GSMA. “Seguridad y Privacidad en las Redes Móviles”. 2018. Disponible en: <https://www.gsma.com/latinamerica/wp-content/uploads/2018/04/Seguridadyprivacidad.pdf>.
- [4] D, Giusto Bilić. “Seguridad en Dispositivos Móviles: Resumen de lo que fue el 2018”. Welivesecurity.com by ESET. 2018. Disponible en: <https://www.welivesecurity.com/la-es/2018/12/21/seguridad-dispositivos-moviles-resumen-2018/>.
- [5] D.Y, Londoño Arenas y J.F, Hurtado Rivera. “Esquema de Seguridad para Protección de Dispositivos Móviles con el Sistema Operativo

Android”. Universidad de San Buenaventura Seccional Medellín. 2014. Disponible en <https://pdfs.semanticscholar.org/6235/56264df6fb749d347ef069ee4d8d5646129d.pdf>.

[6] C. Madero García. “Controles y Seguridad bajo Entorno Android”. Universidad Carlos III de Madrid. 2013. Disponible en: [https://e-archivo.uc3m.es/bitstream/handle/10016/18102/PFC\\_Cristian\\_Madero\\_Garcia.pdf?sequence=1&isAllowed=y](https://e-archivo.uc3m.es/bitstream/handle/10016/18102/PFC_Cristian_Madero_Garcia.pdf?sequence=1&isAllowed=y).

[7] A. V. Romano. “Descripción y Análisis Formal del Modelo de Seguridad de Android”. Universidad Nacional de Rosario, Argentina 2014. Disponible en: <https://www.fing.edu.uy/inco/grupos/gsi/docs/proyectos/informefinal.pdf>.

[8] C. A. Ortegón Serna. “Amenazas, Vulnerabilidades, Factores de Riesgo y Defensa en Profundidad en Aplicaciones Web”. Universidad Piloto de Colombia. Disponible en: <http://repository.unipiloto.edu.co/bitstream/handle/20.500.12277/4913/00005093.pdf?sequence=1&isAllowed=y>.

[9] O Betancur Jaramillo y S. E. Eraso Hanrryr. “Seguridad en Dispositivos Móviles Android”. Universidad Nacional Abierta y a Distancia - UNAD. 2015. Disponible en: <https://stadium.unad.edu.co/preview/UNAD.php?url=/bitstream/10596/3614/1/59836994.pdf>.

[10] <https://nvd.nist.gov/vuln/detail/CVE-2018-9442>

[11] <https://cve.mitre.org/>

[12] [https://www.cvedetails.com/product/19997/Google-Android.html?vendor\\_id=1224](https://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224)

[13] Instituto Nacional de Ciberseguridad. España. “Uso de Dispositivos Móviles Corporativos”. Disponible en: [https://www.incibe.es/sites/default/files/contenidos/politicas/documentos/uso-dispositivos-moviles-](https://www.incibe.es/sites/default/files/contenidos/politicas/documentos/uso-dispositivos-moviles-corporativos.pdf)

[corporativos.pdf](#)

[14] D.S. Espitia. “Protección Móvil”. Reporte Digital. Julio, 2019. Disponible en: <https://reportedigital.com/especial-movistar/proteccion-movil-seguros-dispositivos-moviles-empleados/>



