Estudio práctico de mecanismos de seguridad en dispositivos Android

Enric Jódar Ciurana
Departament d'Enginyeria Telemàtica
Universitat Politècnica de Catalunya
Email: enric.jodar@entel.upc.edu

Josep Pegueroles Vallés
Departament d'Enginyeria Telemàtica
Universitat Politècnica de Catalunya
Email: josep.pegueroles@upc.edu

Juan Vera del Campo
Departament d'Enginyeria Telemàtica
Universitat Politècnica de Catalunya
Email: juanvi@entel.upc.edu

***Resumen*—Android climbed up to 80 percent of the smartphone and mobile devices market share. One of the key aspects for the acceptance of a mobile OS is the security degree the user perceives from the system. In this article, we explore some of the important security mechanisms implemented in Google Android through the study of several recent vulnerabilities. Particularly, we discuss a recent security issue in WhatsApp, the dangers of connecting devices to external machines and the security of current mechanisms for access control. We describe these vulnerabilities through in-lab proof-of-concepts. The experience learned from these cases is used to propose better practices for improving the security of the system.**

Palabras clave*—Android, acceso *(access control)*, cifrado *(encryption)*, revisión *(survey)*, seguridad *(security)

I. Introducción

El sistema operativo (SO) *Android*, presente tanto en *smarpthones* como en tabletas, ha sufrido un gran crecimiento en el número de usuarios provocando también que la comunidad de desarrolladores haya crecido [1]. De la misma forma las posibilidades para utilizar la plataforma *Android* con fines técnicos-comerciales también han crecido, se hace necesario realizar un estudio por parte de la comunidad sobre qué nivel de seguridad se puede garantizar en los dispositivos actuales.

Las prestaciones del primer dispositivo, el *HTC Dream* más conocido como *T-Mobile G1*, no tienen nada que ver con las prestaciones de los terminales actuales. Así pues, es razonable pensar que tanto las funcionalidades disponibles para el usuario como las herramientas del sistema hayan mejorado de forma considerable, permitiendo una multitud de aplicaciones que a su vez conllevan implicaciones a nivel de privacidad y seguridad de los datos del usuario.

Android está construido sobre el kernel de *Linux* y su código es de tipo abierto o *open-source*. Esto ha permitido identificar vulnerabilidades del sistema que posteriormente han sido subsanadas, ya sea a partir de acciones de los usuarios o bien por la determinación de Google para mejorar su sistema.

La falta de documentación técnica actualizada sobre dispositivos *Android* relacionada con la seguridad y el análisis de ésta es el motivo por el cual hemos decidido realizar este artículo. El análisis se ha hecho con ejemplos reales y recientes de algunas vulnerabilidades del sistema.

El artículo está organizado de la siguiente manera. La sección II hace referencia a anteriores artículos donde se revisa el estado del arte de la seguridad en dispositivos *Android* así como la percepción sobre el nivel de seguridad por parte de los usuarios. En la sección III se han estudiado los principales elementos de seguridad sobre los que va a tratar el artículo. En el apartado IV desarrollamos en detalle el análisis de seguridad mediante ejemplos concretos y en la sección V se hacen diversas propuestas que permitan mejorar la seguridad de nuestro dispositivo. Finalmente, este artículo acaba con las conclusiones de nuestro trabajo y referencias para consultar.

II. Trabajo relacionado

La elaboración de este artículo parte de un análisis realizado en el año 2009 [2] en el que se destacan los principales mecanismos de seguridad en el sistema operativo Android, así como se describen algunos cambios y propuestas concretas que permitirían mejorar en este aspecto. Aunque en el análisis realizado se hace referencia al móvil *HTC Dream*, cuyas prestaciones no se asemejan a la de los móviles actuales, el artículo nos han permitido obtener una visión más concreta sobre la construcción del sistema.

En [3] hemos podido analizar una encuesta hecha a 60 usuarios donde se estudia comparativamente las acciones que el usuario realiza en un ordenador personal frente a las que realiza en un *smartphone*. En el artículo se comprueba que los usuarios prefieren utilizar el ordenador al realizar operaciones con datos sensibles como por ejemplo introducir el número de la seguridad social, el número de cuenta bancaria, efectuar procesos de compra o bien intercambiar información personal relativa a la salud. El artículo muestra que un 60 % de los usuarios reticentes a utilizar el *smartphone* para realizar este tipo de operaciones argumentan motivos relacionados con la seguridad del dispositivo. En el estudio también se observan los criterios seguidos por parte de los usuarios al instalar aplicaciones *Android*. Los criterios más valorados son el precio, la popularidad y las críticas recibidas por otros usuarios. Sin embargo, los permisos, la política de privacidad y las condiciones de uso de la aplicación son los criterios que menos se valoran por parte de los usuarios.

III. Descripción del sistema Android

En esta sección se describen los principales mecanismos que intervienen en la seguridad de los dispositivos.

ISBN: 978-84-9717-323-0

III-A. Mecanismos de cifrado

Android desde la versión 2.3.4 tiene soporte para el cifrado del sistema de ficheros aunque ésta opción se ofreció a los usuarios a partir de la versión 3.0, para ello se utiliza *dm-crypt*, herramienta presente en el *kernel*. El cifrado, según se describe en [4], se realiza con un algoritmo AES de 128 bits con el modo CBC. La *master key* se cifra con otra clave AES de 128 bits, para ello se utiliza la función PBKDF2, implementada en *OpenSSL*. La *sal* se genera a partir de una secuencia de números o contraseña establecida por el usuario.

Éste es un proceso irreversible en el cual se cifra por completo el sistema de ficheros, sólo se puede revertir en caso de realizar un borrado al estado de fábrica, perdiendo así los datos almacenados. A nivel lógico se sigue utilizando el sistema de ficheros de manera transparente para el usuario. Sin embargo, a nivel físico los datos se encuentran cifrados y no podrían ser extraídos por otro dispositivo sin la clave correspondiente. Cuando es necesario acceder a un fichero determinado éste se descifra y se vuelve a cifrar una vez finalizada su modificación. El proceso se realiza a nivel de bloque del sistema de ficheros.

Actualmente, el sistema también permite el cifrado de dispositivos de almacenamiento externo. Una vez realizado el proceso, los ficheros cifrados sólo serán accesibles desde el mismo dispositivo.

III-B. Firma de aplicaciones y Keystore

Cualquier aplicación debe estar firmada para poder ser instalada en el entorno *Android*. La firma del paquete (*APK*) se realiza a través de un certificado digital auto firmado, éste es generado a partir de la *keystore* creada por el desarrollador. Toda modificación y/o actualización del *APK* deberá firmarse con el mismo certificado. La utilización de autoridades de certificación se limita en el uso de la navegación segura y durante el uso de las *VPN* configuradas en el dispositivo.

A través de la *KeyStore* el sistema operativo realiza la gestión de claves criptográficas, de esta manera se facilita el almacenamiento de claves de forma segura por parte de las aplicaciones sin tener que aplicar medidas de seguridad adicionales durante el desarrollo.

En la versión 4.3 se han aplicado mejoras de seguridad debido a la capacidad de gestión multiusuario de las claves y la mejora en el respaldo de claves basado en sistemas *hardware* [5]. Así pues en los dispositivos multiusuario, una misma aplicación puede almacenar y gestionar diferentes claves dependiendo del usuario que utiliza la aplicación. Por otra parte, el respaldo de claves basado en sistemas *hardware* ofrece mayor seguridad ya que las claves no pueden ser exportadas ni manipuladas por ningún otro elemento que no sea el *hardware* usado para tal finalidad.

III-C. Mecanismos de control de acceso

Los dispositivos siempre han dispuesto de mecanismos de control de acceso. Ya en la versión *Gingerbread 2.2*, la segunda versión más utilizada después de *Jelly Bean* [6], la protección se realizaba a partir de un patrón, un *PIN* o una contraseña elegida por el usuario. Actualmente las opciones se han incrementado añadiendo la posibilidad de proteger el dispositivo mediante desbloqueo facial-voz o bien deslizando el dedo por la pantalla, aunque las últimas opciones contemplan un nivel muy bajo de seguridad.

Respecto al control de procesos y ficheros en el sistema, la ejecución de las aplicaciones se realiza de manera aislada debido al sistema *POSIX*, así pues cada paquete (*APK*) tiene asignado un *UserID (UID)* y el código de la aplicación se ejecuta en un proceso de manera aislada a la de cualquier otra. El *UID* también restringe el acceso a los archivos de la propia aplicación frente a otras siempre que éstos se almacenen en el directorio de la aplicación. El acceso al sistema de ficheros sigue las directrices establecidas en *Linux* (*rwx*) de la misma forma que ocurre con el sistema *POSIX*.

III-D. Sistema de permisos

En el caso de *Android*, durante la instalación de la aplicación se informa al usuario de los permisos que ésta va a requerir para su correcto funcionamiento. El usuario tiene la opción de aceptar el procedimiento o bien rechazarlo si cree que alguno de los permisos puede ser perjudicial para el dispositivo o los datos que éste contiene. La desventaja principal se encuentra en el hecho que no se puede aceptar o rechazar un subgrupo determinado de permisos.

III-E. Repositorio de aplicaciones (Play Store)

La tienda oficial de aplicaciones de *Android*, antiguamente conocida como *Google Play*, aún tiene fallos de seguridad importantes debidos a la falta de revisión de las *Apps* subidas por los desarrolladores. Recientemente Google ha publicado una patente que podría ser utilizada para evitar que aplicaciones *malware*, copias de otras ya existentes, sean introducidas en la tienda oficial [7]. De esta manera, cuando un desarrollador suba una aplicación, el sistema hará una comparación de los recursos utilizados con otras aplicaciones disponibles en la *Play Store*. En caso que un número razonable de recursos coincidan (ficheros multimedia, de datos o ejecutables), la aplicación pasará a ser revisada manualmente para comprobar si se trata de una aplicación pirata o copia de otra. Esta novedad permitirá ofrecer más seguridad tanto a los desarrolladores de las aplicaciones que pueden ser víctimas de copias o suplantación, como a los usuarios que podrán adquirir más confianza en la tienda oficial de *Android*.

III-F. Conectividad USB

Los móviles y las tabletas pueden conectarse mediante un *USB*, ya sea para copiar datos de usuario no protegidos o bien para realizar operaciones a través del *Android Debug Brigde (ADB)*. A partir de la versión 4.2, la opción para habilitar la conexión *USB* solo es visible en caso de activar el modo desarrollador en el dispositivo. Más segura es la conexión con la versión 4.2.2 ya que además de habilitar la conexión *USB*, se debe confirmar la conexión mediante la aceptación de la firma RSA del ordenador al cual conectamos nuestro dispositivo. A primera instancia la conexión *USB* puede parecer *inofensiva* y

las medidas de seguridad adoptadas en la versión 4.2.2 pueden parecer poco resolutivas pero más adelante comprobaremos que no es así.

IV. Análisis de seguridad

En esta sección se presentan tres vulnerabilidades que pueden o podrían haber afectado a una cantidad considerable de usuarios de *smartphones* o tabletas con consecuencias graves para la privacidad y seguridad de los datos de dichos usuarios.

IV-A. Vulnerabilidad en la privacidad de WhatsApp

En marzo de 2014 se conoció una importante vulnerabilidad que afecta a la privacidad de los usuarios del servicio *WhatsApp* [8]. La brecha de seguridad fue detectada al observar que el histórico de las conversaciones era guardado en una parte de la memoria interna (*/sdcard*) o en la *SD Card* externa, donde cualquier aplicación con permisos para acceder al almacenamiento externo y a Internet podía subir el histórico a un servidor remoto. Aunque el archivo de *backup* estaba cifrado, el método de cifrado era muy simple ya que utilizaba la misma clave de 192 bits en todos los dispositivos que utilizaban la aplicación. Esa clave fue descubierta y distribuida en 2011.

Posteriormente a la publicación de esta vulnerabilidad, *WhatsApp* cambió el método de cifrado y utilizó una clave única por dispositivo generada a partir de la cuenta *WhatsApp* asociada a éste. Aunque se produjo una mejora considerable, sólo con acceder a las cuentas del dispositivo (permiso *GETACCOUNTS*) y utilizar un nuevo método de descifrado, se pudo obtener nuevamente la clave [9].

Esta vulnerabilidad fue aprovechada en 2013 por una aplicación llamada *Balloon Pop 2* que se distribuyó a través de la *Play Store*, una vez detectada fue retirada por parte de *Google*. La aplicación, a grandes rasgos perseguía los mismos objetivos, las copias de las conversaciones eran almacenadas en un sitio web en el cual pagando una cantidad determinada y introduciendo el móvil de la víctima, cualquier usuario podía espiar el histórico de conversaciones.

IV-B. Conectividad USB

En octubre de 2013 fue reportado un fallo de seguridad relativo al mecanismo de control de acceso al dispositivo. Según el informe [10] la vulnerabilidad afectaba las versiones 4.0, 4.1, 4.2 y 4.3, posteriormente se depuró en la versión 4.4. En el informe se demuestra como cualquier dispositivo que tenga activada la depuración *USB* está expuesto a la amenaza. El fallo se encuentra en la implementación de la clase *ChooseLockGeneric*, ésta se ocupa de seleccionar el método de acceso al dispositivo por parte del usuario, en caso de existir uno y querer cambiarlo se debe introducir correctamente el anterior. Este error ha sido verificado en un dispositivo *Android 4.1.2* a través del comando *ADB* y la aplicación de test que proporciona el creador del informe y que permite inhabilitar el sistema de *login*. A su vez se ha comprobado que la vulnerabilidad no afecta a un dispositivo con *Android 2.3.7*, así pues suponemos que en alguna de las actualizaciones posteriores se debió introducir el error.

Otra amenaza a la cual podríamos estar sometidos los usuarios son los cargadores de batería de uso público. El hecho de disponer de un cargador público, ya sea con un coste para el usuario o bien de uso gratuito, es cada vez más frecuente debido a la poca duración de la batería de los dispositivos. En el campus universitario *Campus Nord (UPC)* disponemos de un equipamiento que suministra energía mediante la conexión de un cable *USB* de datos, como se muestra en la figura 1. En este equipamiento, la energía se obtiene a través de la radiación solar y se almacena en una batería conectada a la celda solar. Además, dispone de un conjunto de conectores dependiendo del dispositivo que se quiera conectar, el tiempo de conexión recomendado es de 30 minutos. Esta solución puede convertirse a su vez en una amenaza en caso que hubiera algún tipo de mecanismo , por ejemplo una *Raspberry* o cualquier elemento similar, que acceda a los datos del dispositivo e incluso pueda copiarlos. En este caso, la extracción de la información se podría llevar a cabo a través de un script que ejecute comandos en *ADB* y realizar así una copia de determinada información. Al utilizar esta infraestructura, el usuario debe confiar en el buen uso que se hace de ella por parte de la empresa o institución responsable, aunque cabe recordar que si este elemento se encuentra en el espacio público, podría ser incluso manipulado.

IV-C. Control de acceso: reconocimiento facial

Por último, hemos realizado un *test* para comprobar la seguridad del mecanismo de acceso mediante reconocimiento facial, técnica implementada en la versión 4.0 de la plataforma. Para ello hemos habilitado el mecanismo en la *Samsung Galaxy Tab3* y posteriormente hemos realizado una fotografía con una tableta *Sony Xperia Z* a la misma persona que ha activado el reconocimiento facial. Se ha podido comprobar que se ofrece un nivel de seguridad bajo, tal como se indica en la subsección III-C, ya que el dispositivo protegido por reconocimiento facial ha sido desbloqueado utilizando la fotografía realizada en la tableta *Sony Xperia Z*. Cualquier persona que tenga acceso a nuestro dispositivo y a una foto nuestra, ya sea impresa en papel o mostrada por pantalla, podría conseguir introducirse en nuestro sistema. Para incrementar la seguridad del mecanismo, en la versión 4.1 se añadió la necesidad de parpadear durante el desbloqueo del dispositivo para evitar que se utilicen fotografías de la víctima durante el reconocimiento facial. Adicionalmente, en junio de 2012 Google presentó una patente [11] que finalmente se aceptó en junio de 2013. La patente propone una mejora en la seguridad del control de acceso facial añadiendo la posibilidad de usar gestos en la detección facial. El usuario deberá establecer un gesto determinado para poder desbloquear el dispositivo, de esta manera el atacante además de disponer de una fotografía de la víctima tendría que simular la mueca escogida por el usuario. Algunos ejemplos son sacar la lengua, sonreír o bien mover las cejas. Con esta medida resulta más complejo llevar a cabo el ataque ya que se deberían utilizar técnicas de edición

Figura 1. Cargador móvil solar

de imagen para simular la acción del usuario propietario del dispositivo.

V. Recomendaciones para la mejora de la seguridad

A raíz de los casos del apartado IV, se derivan algunas cuestiones relativas a la seguridad y la usabilidad de los dispositivos móviles en el entorno *Android* en las cuales queremos incidir particularmente. En esta sección propondremos algunas acciones que permitan mejorar la seguridad de los dispositivos.

El caso IV-A señala la importancia de proteger adecuadamente el sistema de almacenamiento del dispositivo, así como valorar correctamente cuáles son las acciones que puede realizar una aplicación.

V-A. Almacenamiento externo: La primera recomendación hace referencia al acceso al almacenamiento externo ya que este no se rige por los sistemas descritos en la subsección III-C. A menos que haya un cambio sustancial en el sistema de acceso a datos de la memoria externa, creemos que la información sensible no debería ser almacenada en medios compartidos. Eso implica necesariamente una revisión del sistema operativo en la gestión de archivos que a su vez, debería incidir en el diseño de las aplicaciones que utilicen dispositivos de almacenamiento externo. Teniendo en cuenta que los cambios en el *SO* no son decisión del usuario final, aunque la comunidad de desarrolladores puede incentivarlos, proponemos otra solución que sí está al alcance del usuario, el cifrado del dispositivo y la *SD Card*.

V-B. Cifrado del dispositivo y la SD Card: Si optáramos por cifrar el dispositivo obtendríamos una solución parcial a la vulnerabilidad descrita anteriormente. Durante la utilización del *smartphone* o la tableta cualquier aplicación maligna continuaría siendo una amenaza ya que los ficheros no se encontrarían protegidos. Por otra parte, en caso de pérdida o robo, nuestra información permanecería inaccesible. Sin embargo, si eligiéramos cifrar la *SD Card*, conseguiríamos una protección completa frente la vulnerabilidad presentada ya que durante la utilización del *smartphone* o la tableta, una aplicación maligna que intentara acceder a nuestros datos necesitaría la contraseña de descifrado. Aun siendo una propuesta válida para mitigar la amenaza, ésta conlleva un empeoramiento en el nivel de usabilidad debido a que se produce una ralentización en el tratamiento de ficheros, así como se requieren más autorizaciones por parte del usuario durante la realización de acciones.

V-C. Antivirus: La tercera solución que proponemos es proteger el dispositivo usando un antivirus. Actualmente en la plataforma *Android* hay una gran variedad de soluciones que realizan escaneo de *malware*, protección de navegación web, así como escaneo de aplicaciones y contenidos en la *SD Card*, entre otras cosas. Aun así, un estudio realizado por V. Rastogi et al. [12] demuestra que este tipo de *software* debe mejorar ya que es susceptible a ataques de transformación por parte del *malware* y virus residentes en los dispositivos. Por otra parte, la eficacia de la aplicación antivirus se ve muy reducida en caso de que esta no posea privilegios de usuario administrador. En estas circunstancias, la aplicación no conseguiría monitorizar las actividades de otras aplicaciones debido a la falta de privilegios, y a su vez, tampoco podría monitorizar las actividades consideradas como más peligrosas. Así pues, en caso de utilizar esta opción, se recomienda *rootear* el dispositivo previamente.

V-D. Sistema de permisos: Por último, la solución más simple y plausible sería adoptar cambios en la gestión del sistema de permisos tal y como se describe en la sección III-D, empezando por una mejor concienciación por parte del usuario sobre qué tipo de permisos requiere una aplicación y como ésta puede manipular sus datos personales. Seguido de una modificación del sistema que permita aceptar o rechazar un subgrupo de permisos de manera que el usuario no se deba aceptar todo el conjunto para así poder utilizar la aplicación deseada. Éste es un problema muy común y una de las principales fuentes de amenazas debido a la inexperiencia o poco conocimiento técnico de las personas que utilizan dispositivos móviles con plataforma *Android*.

El caso IV-B demuestra la importancia de los mecanismos de control de los dispositivos.

V-E. Depuración USB: Sólo deberían tener activada los desarrolladores, y utilizarla con suma cautela. A través de la

conexión *USB* con el *ADB* se pueden realizar muchas acciones que comprometen la seguridad y privacidad de los datos del dispositivo. Hacer una copia completa del dispositivo, eliminar datos o bien instalar aplicaciones *malware* son algunos de los ejemplos más comunes. Hasta la versión 4.2.2 la amenaza solo podría ser contrarrestada en caso de tener la depuración *USB* deshabilitada. A partir de la versión 4.2.2 hasta la 4.3 el acceso al dispositivo está sujeto a mayor control ya que además se debe confirmar la conexión mediante un diálogo basado en el algoritmo *RSA*, para ello se precisa desbloquear el control de acceso del dispositivo (III-C). En caso de que el usuario no haya establecido ningún mecanismo de protección de acceso, el dispositivo sufriría la misma vulnerabilidad que en las versiones anteriores de la plataforma ya que cualquier individuo con acceso físico al dispositivo podría aceptar el diálogo *RSA* y confirmar la conexión. Cabe destacar que no es conveniente utilizar la opción *"Permitir siempre en este ordenador"* en el diálogo *RSA* ya que eso inhabilitaría la protección adicional que nos proporcionan las versiones más recientes del sistema.

V-F. Conexión a otros dispositivos: Los dispositivos tienen la capacidad de interconectarse entre ellos usando tecnologías diversas como *Bluetooth*, *USB*, *Wi-Fi*, *NFC*... Así pues nuestra recomendación es tener activadas estas tecnologías únicamente cuando sea necesario, con ello conseguiremos tener nuestro dispositivo más protegido frente amenazas del entorno. De lo contrario, se podría dar el caso donde terceros accedieran a nuestros datos o podríamos sufrir infecciones de virus, *malware*, rootkits, etc. Además, también debemos ser conscientes sobre la importancia de conectar nuestros dispositivos a otros dispositivos de confianza, de lo contrario nos podríamos exponer a amenazas como las que hemos tratado en el caso IV-A o IV-B.

V-G. Interfaz de login: Todos los usuarios deberían tenerla activada. Aunque las contraseñas o combinaciones de dígitos podrían verse expuestas a ataques de fuerza bruta o ataques de diccionario, de momento se consideran las opciones más robustas en el mecanismo de control de acceso. Los métodos usando parámetros biomédicos no son lo suficientemente eficaces como para adoptarlos como métodos de control de acceso. De hecho, el mismo dispositivo ya lo advierte en el momento de elegir el mecanismo y tal como se indica en el apartado IV-C se ha conseguido romper el control de acceso.

VI. Conclusiones y líneas futuras

Del análisis realizado y expuesto en este artículo se constata que la plataforma *Android* aún teniendo versiones comerciales estables y bastante seguras, debe mejorar e incrementar los métodos de seguridad utilizados en los dispositivos. De lo contrario, seguiremos encontrando vulnerabilidades (algunas de ellas graves) que afecten a un gran número de usuarios. Además, las vulnerabilidades detectadas, o bien la falta de implementación de algunas técnicas de seguridad, pueden provocar que la plataforma *Android* no sea considerada como una opción empresarial en beneficio de otras plataformas existentes como *Windows Phone*, *BlackBerry* o *iOS*.

Cabe destacar también la necesidad de compromiso entre seguridad y usabilidad, tan importante es un sistema seguro como suficientemente práctico y amigable para el usuario. En esta dirección el sistema *Android* debería asumir la necesidad de realizar algunos cambios estructurales que se han comentado anteriormente en la subsección V como por ejemplo el sistema de permisos de las aplicaciones.

Se han identificado algunas líneas futuras para profundizar el estudio realizado en este trabajo. Usualmente, las vulnerabilidades de seguridad se corrigen por parte de los desarrolladores cuando estas son detectadas. Aun así, es necesario un método general para poder identificarlas y evitar que los programadores de aplicaciones las introduzcan inadvertidamente. Por otro lado, la protección contra alguno de estos ataques (como por ejemplo, la identificación de la cara del usuario) depende directamente de la potencia de procesado del hardware.

Finalmente, por falta de espacio no se han incluido en el estudio otras vulnerabilidades identificadas, como la posibilidad de modificar el sistema operativo sin el consentimiento del usuario, aplicaciones capaces de eludir el sistema POSIX de permisos o aplicaciones que pueden ser modificadas por un atacante incluso después de su firmado digital.

Agradecimientos

Este trabajo se ha financiado en parte por los proyectos TAMESIS (TEC2011-22746) y ARES (CSD2007-00004).

Referencias

[1] J. Fingas, "Android climbed to 79 percent of smartphone market share in 2013, but its growth has slowed," January 2014. [Online]. Available: http://www.engadget.com/2014/01/29/strategy-analytics-2013-smartphone-share/

[2] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, and S. Dolev, "Google Android: A State-of-the-Art Review of Security Mechanisms," *ArXiv e-prints*, Dec. 2009.

[3] E. Chin, A. P. Felt, V. Sekar, and D. Wagner, "Measuring user confidence in smartphone security and privacy," *Proceedings of the Eighth Symposium on Usable Privacy and Security - SOUPS '12*, no. 1, p. 1, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2335356.2335358

[4] Google, "Notes on the implementation of encryption in android 3.0," 2014. [Online]. Available: https://source.android.com/devices/tech/encryption/android_crypto_implementation.html

[5] ——, "Android 4.3 apis," 2014. [Online]. Available: http://developer.android.com/about/versions/android-4.3.html

[6] ——, "Platform versions," 2014. [Online]. Available: http://developer.android.com/about/dashboards/index.html

[7] ——, "Detecting pirated applications," Febrero 2014. [Online]. Available: http://www.google.com/patents/EP2693356A2?cl=en

[8] B. Bosschert, "Steal whatsapp database (poc)," Marzo 2014. [Online]. Available: http://bas.bosschert.nl/steal-whatsapp-database/

[9] ——, "Steal whatsapp update," Marzo 2014. [Online]. Available: http://bas.bosschert.nl/steal-whatsapp-update/

[10] US-CERT/NIST, "Vulnerability summary for cve-2013-6271," Diciembre 2013. [Online]. Available: http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-6271

[11] Google, "Facial recognition," Junio 2013, pN 8457367. [Online]. Available: http://patft1.uspto.gov/

[12] V. Rastogi, Y. Chen, and X. Jiang, "Evaluating Android Anti-malware against Transformation Attacks," *NORTHWESTERN University*, no. March, 2013. [Online]. Available: https://www.eecs.northwestern.edu/docs/techreports/2013_TR/NU-EECS-13-01.pdf