Ciencias Matemáticas, Vol. 29, No. 1, Pag. 31-39, 2015
Recibido 05-2015

Utilizar un teléfono Android como dispositivo de cifrado versátil
Using an Android phone as versatile encryption device

Anna Fernández Gironés[1], Alejandro Tamayo Castillo[1], Miguel Katrib Mora[1]*

Resumen Los teléfonos inteligentes son ya omnipresentes en las comunicaciones modernas. Al ser los dispositivos móviles más utilizados, resulta interesante considerar cómo aprovechar la capacidad de los mismos, en particular aquellos con Sistema Operativo Android, para aplicar mecanismos de cifrado de forma versátil. Este artículo tiene como objetivos comprobar la factibilidad, analizar la seguridad y versatilidad de usar un teléfono Android como opción a los llamados TPD[*], así como ilustrar cómo pudiera ser su utilización en la práctica. Para ello se hace uso de la tecnología *TrustZone* incorporada en el *hardware* de los últimos teléfonos Android, la cual proporciona confianza directamente desde el *hardware* al igual que los TPM[**]

Abstract Smartphones are now ubiquitous in modern communications. As most used mobile devices, it is interesting to consider the potential thereof, particularly those with Android Operating System, to implement encryption mechanisms in a versatile way. This article aims to test the feasibility, analyze security and versatility of an Android phone as option to called TPD and illustrate how it could be use in practice. For this purpose it is used The TrustZone technology built into the hardware of the latest Android phones, which provides confidence directly from the hardware like the TPM.

Palabras Clave
Android, Cifrado, Codificación, Certificados Digitales, Dispositivo Móvil, Llave Segura, Plataforma de Confianza, Privacidad, Seguridad, Teléfono Inteligente, Zona Segura

[1] *Facultad de Matemática y Computación, Universidad de La Habana, Cuba, annafgirones@gmail.com, tamayo@matcom.uh.cu, mkm@matcom.uh.cu*
***Autor para Correspondencia**

1. Introducción

Hoy en día, gracias a los dispositivos móviles se ha hecho popular acceder a datos e información en cualquier momento y lugar. Un factor importante a considerar entonces, es la seguridad y confiabilidad de los datos consultados, generados y almacenados (ya sea en el propio dispositivo o en lo que se conoce como **Nube Móvil** (MCC)[1] [1]). La Nube Móvil es un nuevo modelo computacional que combina la nube, la infraestructura de comunicación inalámbrica, los dispositivos de cómputo portables, los servicios de geolocalización y otras características. La diferencia con el paradigma de Computación en la Nube (*Cloud Computing*) es sutil, en la Nube Móvil los actores protagónicos son los dispositivos móviles, y las aplicaciones y servicios se ajustan a sus características específicas como la resolución de pantalla, el ancho de banda, la capacidad de almacenamiento y procesamiento, los sensores, etc. [2].

Existen muchas amenazas a la confiabilidad e integridad de la información almacenada en los dispositivos del usuario final. Algunas de esas amenazas están basadas en las propias Aplicaciones, la Web y la Red utilizada. También están las amenazas físicas como el robo o pérdida del dispositivo permitiendo a ajenos disponer de la información que contiene [3], [4].

Los Proveedores de Servicios en la Nube (CSP)[2] insisten en que sus servidores y los datos almacenados en ellos están lo suficientemente protegidos contra cualquier tipo de robo, argumentando que los datos contenidos en sus servidores están más seguros que los datos residentes en los dispositivos personales del usuario. Sin embargo, en la realidad los CSP también son víctimas de ataques como el incidente ocurrido con *iCloud* en el 2014 [5], sin contar que se parte de la confiabilidad y honestidad de dichos CSP ya que información sensible queda a merced de estos. Por otra parte las nubes no son infalibles y los datos almacenados en ellas no están exentos de extraviarse o modificarse, lo mismo a consecuencia de un fallo en la seguridad que por algún error humano [6].

[1] *Mobile Cloud Computing*

[2] *Cloud Service Providers*

Además de estos problemas, las Redes de Comunicación (dígase por ejemplo Internet), pueden llegar a ser hostiles en cuanto a la seguridad e integridad de los datos.

Por estas y otras razones puede ser conveniente aplicar medidas adicionales de seguridad para proteger la información. Los primeros pasos para alcanzar mayor seguridad son la **autentificación** (que el que accede sea verdaderamente quien dice ser) y el **cifrado de la información** (que la información solo la pueda ver quien tenga el secreto de cifrado) [4].

Esto no significa que se deban rechazar las bondades del almacenamiento en la nube que brindan los CSP como el bajo costo de un servicio que garantiza alta calidad y disponibilidad. Por tanto, hay que encontrar una variante que a la vez permita seguir utilizando estas facilidades y garantice la seguridad de los datos almacenados. Una solución puede ser codificar los datos, utilizando un algoritmo de cifrado fuerte, antes de subirlos a la nube. De esta forma, ni el CSP ni alguien que logre obtener el dato almacenado podrían conocer su contenido.

El problema entonces estaría en dónde almacenar el secreto para descifrar los datos y cómo realizar este proceso para que sea versátil y seguro. Si se almacena el secreto (una llave de cifrado) en la nube, se tendría el mismo problema. Existen soluciones empresariales para esto que consisten en que la empresa instala un servidor de llaves propio, y los datos viajan constantemente desde el CSP hacia este servidor a través de la red para realizar el descifrado [7], [8]. De esta manera el secreto (la llave privada en una PKI[3]) necesaria para descifrar la información, nunca tiene que salir de la empresa ni se transmite por Internet.

Existen numerosos dispositivos personales, en particular, aquellos que cumplen con fuertes demandas de seguridad, privacidad y confiabilidad los cuales son denominados Dispositivos Personales de Confianza (TPD de su nombre en inglés). Para garantizar estos aspectos los TPD tienen *hardware* dedicado a técnicas de cifrado [9].

Algunos de los teléfonos inteligentes más recientes con sistema Android incorporan ya como parte de su *hardware* la posibilidad de almacenar este secreto de cifrado de forma segura, así como la posibilidad de realizar las operaciones de cifrado en lo que se denomina un entorno protegido que está separado del entorno de ejecución normal del dispositivo. Esta característica permitiría convertir a estos dispositivos móviles en dispositivos de cifrado personal, realizando la codificación de los datos entre el CSP y el usuario.

Note que los datos, una vez cifrados, pueden almacenarse lo mismo remotamente en el CSP que localmente en la memoria del dispositivo móvil; lo importante es cómo realizar el cifrado de forma tal que el secreto se proteja y que el proceso de cifrado no se realice fuera del entorno de ejecución seguro. En el presente artículo se explorará la versatilidad y factibilidad de utilizar estos dispositivos con tal propósito.

En la sección II, se brinda una breve explicación sobre el cifrado de datos y en particular sobre el cifrado en los dispositivos móviles así como las peculiaridades de los mismos en este entorno. En la sección 3 se presenta la justificación de este trabajo y en la sección 4 la propuesta de solución. La sección 5 presenta algunas pruebas experimentales y se discuten estos datos obtenidos experimentalmente para dar las conclusiones en la sección 6. Posteriormente en la sección VII se presenta un posible trabajo futuro.

[3] *Public Key Infrastructure.*

2. Estado del Arte

2.1 Medios de Cifrado Existentes.

Actualmente existe en el mercado una gran variedad de software de cifrado (*Dekart Keeper*[4]*, CryptoForge*[5]*,* etc.). Estos funcionan hospedados en un dispositivo de almacenamiento, como una memoria USB, y cifran una parte de él. La forma de acceder a los datos almacenados en dicho dispositivo es a través de una contraseña que provee el usuario. Su debilidad es que se requiere de un software y un dispositivo hospedero para acceder a dichos datos por lo que está expuesto a *tampering*, es decir, un atacante puede modificar el software o el dispositivo hospedero para obtener la contraseña y por consiguiente hacerse de los datos descifrados.

Existen también dispositivos USB más sofisticados, llamados dispositivos USB de confianza o *SPD*, *Secure Pocket Drive* como los de la compañía *SPYRUS*[6] que cargan un sistema operativo propio para ejecutar directamente en la computadora anfitriona de tal modo que se pueden hacer las operaciones necesarias con la seguridad de no dejar rastros al concluir.

La desventaja es que se necesita siempre de un dispositivo anfitrión para procesar la información por lo que entonces se corre el riesgo de que se *hackee* el *driver* o *software* que ejecuta en el dispositivo hospedero permitiendo acceder a la información segura.

2.2 Cifrando la Información en los dispositivos móviles. Seguridad de la Llave.

Hoy en día tenemos numerosas técnicas de cifrado para proteger la información y los datos como por ejemplo: Cifrado de disco duro (FDE)[7], utilizada por *iOS* y *BlackBerry*, Cifrado de disco virtual (VDE)[8] y Cifrado de ficheros o carpeta (FE)[9] usada por *Windows Phone* y Android [4].

El cifrado de disco (FDE) tradicionalmente trabaja bajo el sistema de ficheros para proveer codificación y descodificación instantánea para todas las tareas de lectura/escritura en el dispositivo. Por esta razón, la llave de cifrado debe de estar accesible para el sistema de ficheros y en consecuencia si el dispositivo está en el estado desbloqueado, entonces la llave y los datos son vulnerables. Además, como FDE cifra tanto los ficheros del usuario como los ficheros del sistema operativo, existe una mayor probabilidad de *hackear* la llave,

[4] *http://www.dekart.com.*
[5] *http://www.cryptoforge.com.*
[6] *http://www.spyrus.com.*
[7] *Full Disk Encryption.*
[8] *Virtual Disk Encryption*
[9] *File / Folder Encryption*

ya que el *hacker* conoce tanto el texto plano (bloque donde se encuentran datos siempre iguales del Sistema Operativo los cuales no varían por dispositivo) como el texto cifrado (de estos bloques del Sistema Operativo) y puede utilizar técnicas conocidas y documentadas en la literatura para obtener la llave.

Si se trabaja en un ambiente con computadoras estas se pueden poner en suspensión o apagarse para proteger la llave. Con los teléfonos inteligentes sin embargo, esto es imposible pues es necesario mantener las funcionalidades básicas de comunicación.

En [10] proponen una solución para proteger la llave en FDE. El cifrado de fichero o carpeta (FE) disminuye las vulnerabilidades de FDE hasta que el usuario se autentifica satisfactoriamente y los datos son descodificados. Pero una vez que esto ocurre cualquier proceso que se esté ejecutando en el dispositivo (como puede ser un virus) con acceso a los datos del usuario puede también tener entonces acceso a la información. Esto sin contar la pérdida o robo de la contraseña de autentificación [4] con un *keylogger* o un *malware* que adquiera una copia de la llave de la memoria del dispositivo (en soluciones en que el almacenamiento de la llave es por software).

2.3 Herramientas que utilizan a Android para codificar y decodificar datos.

Existen ya algunas herramientas para cifrado en teléfonos con sistema Android. Estas pueden dividirse en dos grupos:

- **Herramientas de cifrado local**, dónde la aplicación de cifrado ejecuta en el teléfono y los datos se almacenan en el mismo (ejemplo *Boxcryptor*[10], *Cryptonite*[11]).
- **Herramientas de cifrado remoto**, dónde la aplicación de cifrado ejecuta en el teléfono pero los datos se almacenan en la nube. Este grupo puede subdividirse a su vez en dos: aquellas herramientas dónde el almacenamiento remoto es un servicio nativo especializado (ejemplo *SpiderOak*[12] y *Wuala*[13]) o aquellas que reutilizan servicios de almacenamiento existentes (ejemplo *Boxcryptor*, *Cryptonite*) como *Google Drive, Dropbox, SkyDrive*, etc [11].

La mayoría de estas herramientas utilizan el sistema de archivos *EncFS*[14], diseñado para crear una capa de abstracción entre un sistema de archivos virtual, dónde el usuario almacena sus datos y el sistema de archivos real dónde se almacenan los ficheros ya encriptados [12]. *EncFS* permite utilizar como sistema de archivos real, tanto el disco duro local del dispositivo como los servicios en la nube (*Google Drive, Dropbox, SkyDrive*, etc). Para el usuario final, el sistema de archivos virtual es transparente, ya que se visualiza y accede como un "disco duro" más conectado al dispositivo [13].

Estas herramientas utilizan AES[15] con diferentes modalidades (CBC[16], CFB[17], XTS[18]) para cifrar los datos, y algunas de ellas, añaden un nivel de seguridad adicional, generando una llave AES aleatoria para cada fichero y esta se cifra a su vez con RSA[19]. Utilizando RSA+AES se mejora la seguridad, ya que si un *hacker* se hace con la llave AES que se utilizó para cifrar el fichero, solo habrá obtenido acceso a dicho fichero y no a los restantes. Pero esta mejora conlleva proteger fuertemente el bloque cifrado con RSA, utilizando una llave de 4096 bits mínimo (antes de Android 4.3 solo se podía utilizar hasta 2048) y un algoritmo de relleno criptográficamente fuerte, ya que éste entonces sería el objetivo fundamental del *hacker* (si se obtiene de alguna forma la llave privada, se tendría acceso a todos los datos cifrados). Pero como se puede lograr que el bloque RSA siempre contenga información aleatoria entonces le sería muy difícil a un *hacker* realizar ataques conocidos. Utilizar el algoritmo RSA para cifrar todo un fichero no sería conveniente ya que el proceso de descifrado de RSA es lento [14] y se expondría la llave privada a mayores ataques.

Todas estas herramientas tienen una característica en común: la aplicación encargada de cifrar o descifrar ejecuta donde mismo el usuario va a acceder a los datos. Si se quisiera que un dispositivo Android y una PC de escritorio compartan los datos cifrados almacenados en la nube, habría que compartir la llave privada entre ambos dispositivos e instalar la herramienta también en la PC de escritorio. Pero independientemente de que la llave privada se pueda proteger por contraseña, al tener que estar almacenada en todos los dispositivos que acceden a los datos cifrados (hablamos de diferentes sistemas operativos, diferente *software* y distintas características de seguridad y protección) se amplían entonces las posibilidades de *hacking*.

La solución podría ser entonces que cada uno de estos dispositivos (incluyendo la PC de escritorio) brindase por *hardware* un mecanismo para el almacenamiento seguro de la llave privada y la ejecución segura de las operaciones de cifrado, similar al *TrustZone* de Android. Pero en la práctica, actualmente sólo un conjunto pequeño de dispositivos brindan esta característica, por lo que las herramientas anteriores están expuestas por diseño a múltiples tipos de ataques documentados en la literatura.

Una herramienta interesante que utilizan los dispositivos Android para cifrar y descifrar los datos es *DroidVault* [15]. *DroidVault* garantiza a los dueños de datos sensibles la protección de los mismos en dispositivos *Android* potencialmente inseguros. Esta herramienta enfoca a servidores de datos remotos (empresas) como los dueños de los datos sensibles. Estos datos sensibles son compartidos en los dispositivos móviles de los usuarios finales (empleados de la empresa) que se definen como los usuarios de los datos. Esta herramienta hace uso

[10] *https://www.boxcryptor.com*
[11] *http://code.google.com/p/cryptonite.*
[12] *https://spideroak.com*
[13] *https://www.wuala.com.*
[14] *FUSE-based cryptographic filesystem.*
[15] *Advanced Encryption Standard.*
[16] *Cipher Block Chaining.*
[17] *Cipher Feedback.*
[18] *Xor encrypt xor based tweaked codebook mode with ciphertext stealing.*
[19] *Rivest Shamir Adleman.*

de la nueva tecnología incorporada en los teléfonos inteligentes que se presenta en la Sección IV A al igual que nuestra propuesta que será presentada en la misma sección.

DroidVault es un sistema diseñado por partes que está separado del sistema operativo y presenta tres componentes fundamentales: El módulo de procesamiento de la información, un módulo puente y un módulo el de entrada y salida de los datos hacia la pantalla del dispositivo. El módulo de procesamiento de la información es el encargado de mantener un canal de comunicación seguro con el servidor de datos con el fin de intercambiar información y de procesarla. El módulo puente facilita las comunicaciones entre el sistema operativo y el módulo de procesamiento y el último módulo es el encargado de mostrar los datos en la pantalla al usuario y de recoger cualquier dato producido por el mismo.

3. Objetivos de este trabajo

Integrar funcionalidades bajo un mismo dispositivo ha sido el gran éxito de los teléfonos inteligentes. Los teléfonos móviles de hoy, además de su funcionalidad básica (las comunicaciones por voz) juegan también los papeles de agendas personales, cámaras digitales, dispositivos de juego, dispositivos de almacenamiento y GPS entre otras muchas aplicaciones. Los teléfonos móviles son dispositivos de uso permanente que se han convertido prácticamente en una "extensión" de su propietario humano. Si pudiéramos utilizarlos además como TPD se les agregaría una nueva funcionalidad, cumpliendo con el deseo de utilizar un solo dispositivo para varias funciones, para que nuestra información en el teléfono se mantenga segura.

Utilizar un teléfono como TPD tiene muchos usos y posibles escenarios. En un escenario empresarial por ejemplo, un ejecutivo puede utilizar su móvil como TPD para codificar la información antes de subirla a la nube, o bien guardar directamente dicha información (ya codificada) en el teléfono. Otro posible escenario es el personal, donde no queremos almacenar en la nube información privada o íntima sin cifrar antes la misma para que no esté expuesta a fallos técnicos o ataques en los CSP.

El objetivo del presente trabajo es demostrar la factibilidad de utilizar un teléfono Android como Dispositivo Personal de Confianza comprobando su viabilidad y versatilidad en el cifrado de la información.

Este trabajo está enfocado a los teléfonos inteligentes con *Android* por ser el sistema operativo predominante en el mercado y por presentar las APIs necesarias para la implementación de aplicaciones que utilicen el cifrado por *hardware* y por consiguiente la Zona Segura.

4. Propuesta de cómo usar un teléfono android como TPD.

4.1 Cifrar los datos y guardar la llave de forma segura.

Algunos de los teléfonos inteligentes Android más recientes tienen en su *hardware* una característica llamada Zona Segura (ARM TrustZone®), la cual tiene la tarea de crear un entorno seguro separado del Sistema Operativo al cual los atacantes no tienen acceso. Se llama Entorno de Ejecución Confiable (TEE)[20] y esta tecnología se utiliza para proveer confianza en el almacenamiento de la llave así como operaciones de cómputo seguras [16], [17].

La zona segura está basada en dos ambientes completamente distintos llamados *worlds*, cuyo aspecto fundamental es que usan recursos de *software* y *hardware* separados. El primero *normal world* es el conocido hasta ahora y es en el que funcionan el sistema operativo y las aplicaciones. El segundo mundo es nuevo y se ha denominado *secure world*. En él, corre un sistema operativo llamado *secure world OS* y las aplicaciones deben cumplir ciertos requisitos con el fin de proveer servicios de seguridad especiales tales como el almacenamiento seguro de la llave. Un ejemplo de estos requisitos es que el TEE debe permitir que las aplicaciones ejecuten separadas lo que asegura que aplicaciones maliciosas no puedan acceder o modificar el código o los datos de otra aplicación. Otro requerimiento fundamental es el almacenamiento seguro de los datos para proteger el secreto y la integridad de los datos binarios que representan las aplicaciones así como los datos que éstas usan mientras no están en ejecución [16]. El sistema operativo Android ya implementa el almacenamiento por *hardware* de la llave de cifrado en los dispositivos móviles que ya tienen Zona Segura (por ejemplo, la línea de teléfonos Nexus de Google) y brinda una API para que pueda ser utilizado por las aplicaciones.

En los teléfonos actuales que disponen de Zona Segura, toda la memoria del sistema está separada, incluyendo la RAM y los registros de los CPUs. Una parte dedicada al "mundo normal" y otra para el "mundo seguro". Lo cual significa que el mundo normal no puede acceder a la memoria del mundo seguro. La Zona Segura también tiene un procesador dedicado al cifrado y almacenamiento de las llaves que solo puede ser accedida por el mundo seguro [16].

Estas características serán utilizadas en nuestro trabajo para el almacenamiento seguro de la llave privada que se utilizará para codificar la información.

Un limitante que presenta la utilización de la Zona Segura es que la llave nunca podrá extraerse de la misma. Por lo que si la llave se generó también en la zona segura entonces en caso de pérdida o daño físico del dispositivo móvil, los datos cifrados no podrán volverse a descifrar. Una solución para esto pudiera ser generar la llave fuera del entorno seguro para poder resguardarla también por otra vía (por ejemplo, un dispositivo USB guardado en una caja tan fuerte, blindada y ultra secreta como se quiera) y luego introducirla en el mundo seguro, de dónde no se podrá extraer.

[20] *Trusted Execution Environment.*

4.2 Trasmisión de los datos de forma segura. Seguridad del Canal.

Los teléfonos son utilizados para almacenar y generar datos. Estos datos pueden transmitirse hacia afuera y hacia adentro de los mismos a través de la Wifi, Bluetooth, de la Red de Datos o un cable USB. Nuestra propuesta se basa en que la transmisión entonces se efectuará a través de la Wifi y utilizar protocolos de comunicación estándares donde al extremo conectado al dispositivo utilizado para cifrar se le añada solamente el requisito de tener un navegador web estándar para realizar la comunicación. Es decir, descartamos el uso del cable USB o el Bluetooth ya que para su utilización, se requiere instalar software adicional en el dispositivo y éste no es nuestro objetivo. Se descarta también la Red de Datos pues es inexistente en nuestro país, por tanto es imposible hacer pruebas sobre ella.

Como es conocido, existen varias técnicas de cifrado para el intercambio de información. En un **cifrado simétrico**[21], se utiliza una misma llave para codificar y decodificar la información. Esto implica que antes de enviar los datos, la llave debe ser compartida entre el emisor y el receptor. El emisor luego de haber cifrado la información usando la llave previamente adquirida, la envía y el receptor descifra la información haciendo uso de la misma llave que comparte con el emisor. Esto implica poner de acuerdo a cada par emisor-receptor lo que no es muy viable precisamente en el escenario móvil. Y no siempre el que codifica para enviar tiene que tener el poder de decodificar para ver.

En un **cifrado asimétrico**[22] por otra parte, existen dos llaves conocidas como llave pública y llave privada. La llave pública se utiliza para codificar la información y la llave privada, que es secreta, para decodificarla [6]. La ventaja de la Infraestructura de la llave Pública PKI sobre el cifrado simétrico es que la llave pública puede viajar por redes inseguras permitiendo que diferentes emisores dispongan de dicha llave para comunicarse con el receptor de forma segura porque para descifrar la información que envían éste también necesita de la llave privada. En el caso del cifrado simétrico, la llave tiene que conocerse por ambas partes y no puede transmitirse por redes públicas, por lo que es poco útil su uso en la nube.

Para proteger las comunicaciones del ataque conocido como ataque de hombre intermedio (MITM)[23] existen protocolos como TLS[24] y SSL[25] que utilizan una PKI para cifrar las comunicaciones. Pero para garantizar la seguridad adecuadamente, no basta con enviar la llave pública y esperar por la información cifrada, ya que un *hacker* pudiera interceptar la comunicación con el potencial emisor, enviarle a este una llave pública falsa para así leer el contenido cifrado que este envíe y para que el emisor y el receptor no se enteren de la interferencia, se recodificaría dicha información ya leída con la llave pública real antes de reenviarla al receptor. Esto se resuelve mediante un mecanismo de verificación de llave pública dónde el emisor solamente aceptará llaves públicas confiables. Para ello, se han establecido Autoridades Certificadoras como *VeriSing*[26], por ejemplo, que son las encargadas de generar pares públicos y privados, firmados digitalmente, de forma tal que posteriormente se pueda verificar la validez de la llave [18].

El problema que nos ocupa en nuestro caso es que en un mundo cada vez más dinámico, como es el caso de los escenarios en que participan los teléfonos inteligentes, dónde las direcciones IP cambian constantemente, es mucho más difícil utilizar un certificado digital válido generado por una Autoridad Certificadora de confianza, ya que habría que asociar también dinámicamente dicho IP con un nombre de dominio válido. Existen varios servicios como *DynDNS*[27], *FreeDNS*[28] y *NoIP*[29] que pretenden resolver este problema, pero requieren que el dispositivo esté conectado a Internet y ciertos mecanismos de autenticación. Estos servicios necesitan la instalación de una aplicación extra en nuestro dispositivo móvil como *DynDNS client*[30]. Una vez instalada, el IP del móvil se relacionaría, por ejemplo, con el nombre galaxys4.dyndns.org, por lo que se podría acceder a él vía http://galaxys4.dundns.org. En caso de que la IP cambie, *DynDNS client* se encargaría de actualizar el registro DNS, lo que permitiría el acceso a nuestro dispositivo de forma permanente. Esto resuelve sólo poder descubrir el dispositivo en la red pero no la seguridad del canal con SSL. Para asegurar el canal, utilizando HTTPS, se requiere un certificado digital válido (*Certificate Authority-signed certificates*) que ponga "en verde" la barra de nuestro navegador. No basta con tener un certificado digital (por ejemplo, uno auto-firmado), ya que si el navegador no es capaz de verificar la autenticidad de dicho certificado mediante una entidad certificadora, un hacker podría cambiar el certificado y simular HTTPS. El problema es que las autoridades certificadoras, sólo generan certificados para dominios registrados. Por tanto, si queremos HTTPS con una "barra verde", habría que comprar un dominio galaxys4.com y sacar un certificado para él. *DynDNS* también brinda este servicio[31] pero requiere costo adicional.

Si quisiéramos utilizar un certificado digital auto-firmado (*Self-signed*) para asegurar el canal con nuestro dispositivo móvil (y así no incurrir en gastos adicionales) tendemos que considerar que nos saldrá una advertencia de seguridad en el navegador y tendremos que validar o no el certificado digital manualmente y proseguir con la navegación.

Para determinar que el certificado digital es el correcto, habría que mostrar el *thumbprint* (lista de números o *hash* que identifican al certificado de forma única) en la pantalla del dispositivo Android para que el usuario pueda comparar

[21] *DES, Data Encryption Standard, Triple DES, AES, RC2, RC4 Rivest Cipher, etc.*
[22] *RSA, DSA Digital Signature Algorithm, PGP Pretty Good Privacy, etc.*
[23] *Man In The Middle Attacks.*
[24] *Transport Layer Security*
[25] *Secure Sockets Layer*
[26] *https://www.verisign.es*
[27] *http://es.dyn.com/dns/*
[28] *http://freedns.afraid.org/*
[29] *http://www.noip.com/managed-dns*
[30] *https://play.google.com/store/apps/details?id=com.dyndns&hl=es...This*
[31] *http://es.dyn.com/standard-dns/*

éste con el que muestra el navegador y en caso de coincidir saber que es el certificado correcto. Una vez que se valide el certificado, la navegación será tan segura como con "barra verde".

No obstante, es un paso manual que atenta contra la versatilidad del uso de esta propuesta. Claro, el usuario siempre puede instalar el certificado digital (sin la llave privada) manualmente en el dispositivo final, y con esto se obtendría la "barra verde", pero instalar un certificado digital, es una operación que usualmente requiere permisos de administración y no podrá realizarse fácilmente en cualquier ordenador.

4.3 Garantizar la autentificación del Usuario.

Como se sabe la autentificación por contraseña no es del todo segura debido a que a veces estas pueden ser fáciles de adivinar, los usuarios las suelen escribir en algún papel, las envían por email o las comparten de alguna forma con un tercero. Un mecanismo de autentificación de dos pasos disminuye estos problemas.

El primero de estos pasos sería la generación de un secreto volátil (código autogenerado y aleatorio) que se confirmaría a través de un mecanismo alternativo a la conexión de red, para así autenticar el canal de comunicación, pudiendo ser este un *QRCode* que pueda validarse a través de la cámara del teléfono u otro sensor. El segundo paso sería un secreto estático (PIN, contraseña) que solo el usuario conoce, que le permita solo a él acceder a su información.

Con el primer paso se garantiza que la persona que esté sentada en la PC y que intenta acceder a la información privada sea realmente el portador del teléfono (y no otra persona conectada a la Red) y con el segundo paso se autentica al dueño de la información privada (por si alguien se robó o se encontró el teléfono perdido). Está claro que esto no ayuda si la contraseña y el móvil fueron robados a la vez (por ejemplo si el usuario está bajo una amenaza física), pero en cualquier caso que eso suceda es menos probable y ya no depende de la tecnología.

Una vez superada satisfactoriamente la autentificación viene la fase de la navegación donde se trabaja con los ficheros guardados en el teléfono. Para ello se utiliza el protocolo (WebDAV)[32] el cual consiste en una extensión del Protocolo HTTP con el objetivo de administrar recursos remotamente. En el caso de una PC *Windows*, con WebDAV se podrán visualizar los datos del teléfono como si fuese una carpeta más del sistema operativo y el usuario final (el dueño del teléfono Android) se abstraerá de los mecanismos de transmisión y codificación que existen por detrás. (Otros dispositivos con otros sistemas operativos accederían también a los datos, pero a través del navegador web utilizando la interfaz de usuario web). Es entonces cuando se realiza el intercambio entre el navegador y la aplicación de cifrado y el teléfono cumple con su función de TPD.

[32] *Web Distributed Authoring and Versioning*

4.4 Arquitectura Propuesta y su Funcionamiento.

Nuestra propuesta funcionaría de la siguiente manera. El dispositivo Android estará cumpliendo la función de servidor e intercambiando información con un cliente, es decir, se conectaría el móvil a la PC usando la Wifi como muestra la 1.

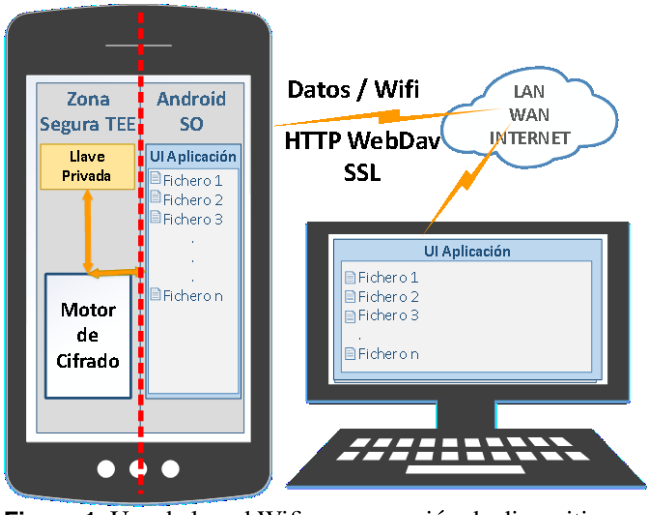

Figura 1. Uso de la red Wifi para conexión de dispositivo Android.

Existen un conjunto de fases que se deben satisfacer para establecer una comunicación exitosa y se cumpla el objetivo perseguido. En la primera fase se levanta un Servicio Web en el móvil con la Aplicación de Cifrado y se despliega un mecanismo de descubrimiento. El servidor web brinda una IP y un puerto por los cuales se establece la comunicación, siempre utilizando SSL. El cliente del servicio web podría ser entonces una aplicación personalizada o simplemente uno de los navegadores web más utilizados hoy en día. Un navegador es una propuesta más versátil ya que estos están previamente instalados en la generalidad de los dispositivos posiblemente utilizados y no necesitaríamos de una preinstalación personaliza.

La siguiente fase es el mecanismo de autentificación de dos pasos donde el primero de estos pasos sería una contraseña que se le enviaría a la aplicación de cifrado residente en el dispositivo móvil. El segundo paso es un *QRCode* generado en tiempo real en el momento de la validación y una vez leído se habilitará la navegación. Así se garantiza que el dueño del teléfono sea el que está tratando de acceder al mismo ya que tiene la contraseña de la primera fase y que el mismo no ha sido robado por tener la necesidad de leer el *QRCode* usando el teléfono.

Una vez superada satisfactoriamente la autentificación viene la fase de la navegación donde se trabaja con los ficheros guardados en el teléfono. Para ello se utiliza el protocolo WebDAV.

5. Resultados de las Pruebas Experimentales.

Algunos investigadores han hecho experimentos con dispositivos móviles comprobando la eficiencia de los mismos con diferentes algoritmos de cifrado. En [20], por ejemplo, se muestra el costo computacional y gasto de energía de dispositivos móviles, en particular, PDAs[33].

Nuestro objetivo es determinar la factibilidad del uso de los dispositivos Android como TPD y para ellos hemos realizado pruebas experimentales y se ha implementado un prototipo de aplicación que cifre ficheros de diferentes tamaños dónde se mide el tiempo de trasmisión de estos a través de la Wifi, así como los tiempos de cifrado y descifrado. Para enviar los datos a través de la Wifi se han dividido los ficheros en pedazos de 512 y 1024 Kb. En estas pruebas se ha utilizado como dispositivo de experimentación un LG Nexus 4.

En un primer experimento se han obtenidos los datos reflejados en la tabla 1 donde todos los procesos (trasmisión, cifrado y descifrado) se hacen secuencialmente.

En un segundo momento hacemos uso de la paralelización con el fin de explotar la capacidad multi-núcleo que brinda este teléfono. Estos resultados se muestran en la tabla 2.

De estas pruebas experimentales podemos deducir que el proceso de cifrado no afecta el rendimiento.

El cuello de botella está en la trasmisión de los datos ya que la velocidad promedio es de aproximadamente 2.5 Mb/s que depende de la velocidad de la Wifi que es de 56 Mbit/s (aunque ya existen dispositivos modernos con una mayor velocidad de 150 Mbit/s).

Dado que el proceso de cifrado tarda aproximadamente lo mismo que el proceso de trasmisión de la información en lo que llega un pedazo del fichero al dispositivo Android se puede ir cifrando otro de modo que el tiempo de cifrado no perjudique el rendimiento.

6. Conclusiones.

En nuestros días es una necesidad la protección de información sensible como medida de seguridad obligatoria debido a la gran cantidad de ataques existentes. Disponer entonces de un mecanismo fácil de utilizar, pero a la vez seguro y que no implique gastos adicionales, sería de gran utilidad. Esta propuesta podría ser una herramienta segura para la protección de los datos sensibles en los dispositivos Android. Un caso de uso, sería almacenar las contraseñas, los datos bancarios, los estados de cuentas o los registros médicos del usuario en el teléfono de forma segura y portable, ya que viajarían junto a él por estar en el teléfono y podrían accederse en todo momento. Al estar cifrados utilizando la última tecnología (zona segura), se protegerían contra robo u otros tipos de ataques.

Este trabajo muestra que el reemplazo de los TPD por teléfonos Android es posible siempre y cuando el usuario acepte las limitaciones que esto implica. La primera de ellas es la pérdida de la llave si se pierde o se le es sustraído el dispositivo, pero esta limitante también existe cuando se usa un TPD. Una ventaja en comparación con un TPD, es que se puede resguardar la llave privada mediante otro mecanismo y restaurarla en caso de pérdida o extravío, así como tener respaldo de la información cifrada en la nube. Otra limitante es la seguridad del canal de comunicación (con "barra verde") que no es tan fácil de lograr, ya que habría que pagar servicios adicionales o realizar la verificación del certificado digital manualmente. La última de las limitantes es la velocidad de trasmisión de los datos desde y hacia el dispositivo móvil, pues la velocidad de la Wifi que aunque en los nuevos dispositivos es cada vez mayor aún no se puede comparar con la de un dispositivo USB 3.0. Sin embargo, no usar el USB nos independiza de llevar el cable USB del teléfono junto con él y de necesitar el driver para que funcione en la PC que se esté utilizando para acceder al móvil.

[33] Personal Digital Assistants

7. Trabajo Futuro.

El presente prototipo aún no tiene implementado el mecanismo de salva y restaura de la llave privada.

En un escenario dónde el teléfono móvil se extravía, es robado o se rompe, se perdería junto con él la llave privada que se utiliza para descifrar los datos codificados y por ende toda la información sensible del usuario que esté cifrada. Con el fin de que se tenga alguna forma de evitar esta pérdida, en trabajos futuros hay que concebir un mecanismo de respaldo para la llave privada pero de tal forma tal que la exposición a ataques sea mínima.

Referencias

[1] S. a. Z. J. a. L. D. a. J. J. Nepal, «A mobile and portable trusted computing platform,» *EURASIP Journal on Wireless Communications and Networking,* vol. 2011, nº 1, p. 75, 2011.

[2] Y. C. a. R. K. Dejan Kovachev, «Mobile Cloud Computing: A Comparison of Application Models,» Information Systems & Database Technologies RWTH Aachen University , Ahornstr. 55, 52056 Aachen Germany, 2011.

[3] P. G. Sujithra M, «Mobile Device Security : A Survey on Mobile Device Threats , Vulnerabilities and their Defensive Mechanism,» *International Journal of Computer Applications (0975-8887),* vol. 56, nº 14, pp. 24-29, 2012.

[4] M. S. M. S. Karen Scarfone, «Guide to Storage Encryption Technologies for End User Devices Recommendations of the National Institute of Standards and Technology,» 2007

[5] L. O´Connor, «Celebrity Nude Photo Leak: Just One More Reminder That Privacy Does Not Exist Online and Legally, There's Not Much We Can Do About It,» 2014. [En línea]. Available: digitalcommons.law.ggu.edu. [Último acceso: 28 01 2015].

Tabla 1. Procesos secuenciales (Tiempo en segundos).

Fichero Mb	Segmentación Kb	T. Transmisión	T. Cifrado	T. Descifrado
10	512	11,666721	21,0093156	21,9659112
	1024	11,9876396	18,9945899	19,0844895
32	512	19,0078923	38,3375016	38,9494236
	1024	16,7244139	31,7556396	31,8323425
64	512	62,2480659	116,7900762	117,3967133
	1024	60,5736679	96,5910032	99,1307446
128	512	165,3892591	286,2189747	297,6159404
	1024	154,122489	242,3879867	246,630425
256	512	126,4520881	295,3489466	305,2861638
	1024	117,4942528	227,6566316	229,1970667
512	512	351,3607036	750,1806037	849,919064
	1024	274,9837976	534,0583291	568,1994485
1024	512	604,638345	1444,949788	1906,783323
	1024	451,7867903	939,2094609	1021,723576

Tabla 2. Procesos paralelizados (Tiempo en segundos).

Fichero Mb	Segmentación Kb	T. Transmisión	T. Cifrado	T. Descifrado
10	512	4,0228048	3,841980	4,0693318
	1024	4,2270722	4,215069	4,3412256
32	512	12,286913	12,06821	12,272882
	1024	12,3267226	12,04925	13,0174481
64	512	23,5903803	23,58947	26,0900268
	1024	23,9045025	23,89781	25,0175025
128	512	47,3620253	47,27447	48,0567149
	1024	47,6968507	47,54943	48,7462562
256	512	94,7632723	95,11444	94,8198077
	1024	94,8572613	95,54894	95,9584601
512	512	188,857015	195,3625	190,0780195
	1024	189,7798195	188,6458	190,028967
1024	512	375,4802342	373,1416	373,433388
	1024	374,8283404	373,1119	374,4629174

[6] R. a. C. R. Bhadauria, «A Survey on Security Issues in Cloud Computing».

[7] C. W. K. R. a. W. L. Shucheng Yu, Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing, IEEE INFOCOM: Dept. of ECE, Worcester Polytechnic Institute, Dept. of ECE, Illinois Institute of Technology, 2010.

[8] Q. L. a. J. W. Guojun Wang, Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services, Changsha, Hunan Province, P. R. China: School of Information Science and Engineering, Central South University, 2010.

[9] L. M. A. L. J. C. P. J. v. D. Frank C. Bormann, *Concept for Trusted Personal Devices in a Mobile and Networked Environment.*

[10] A. a. O. P. C. V. Skillen, «Deadbolt : Locking Down Android Disk Encryption,» 2013.

[11] K. a. S. S. Raju, «Overview of Dropbox Encryption in Cloud Computing,» Department of IT, Mahendra Engineering College, Namakkal, India, 2014.

[12] V. Gough, «EncFs,» 2011.

[13] R. M. A. S. Zhaohui Wang, «ImplementingandOptimizinganEncryptionFilesystemonAndroid,» Department of Computer Science George Mason University Fairfax, USA, 2011.

[14] M. a. V. J. Shand, «Fast implementations of RSA cryptography,» Computer Arithmetic, 1993.

[15] H. H. G. B. Y. J. Z. L. P. S. Xiaolei Li, «DroidVault: A Trusted Data Vault for Android Devices,» Department of Computer Science and Graduate School for Integrative Sciences and Engineering, National University of Singapore , Singapore , 2014.

[16] T. Cooijmans, *Secure Key Storage and Secure Computation in Android,* 2014.

[17] J. d. R. a. E. P. Tim Cooijmans, «Analysis of Secure Key Storage Solutions on Android,» Radboud University Nijmegen , 2014.

[18] D. B. a. A. Lioy, «Towards Simplifying PKI Implementation: Client-Server based Validation of Public Key Certi[FB01?]cates,» Dip. Automatica e Informatica Politecnico di Torino, Torino, Italy, 2002.

[19] «Dyn.com,» Dynamic Network Services, [En línea]. Available: http://dyn.com/blog/never-lose-your-android-phone-with-dyns-new-update-client/. [Último acceso: 22 febrero 2015].

[20] H. a. H.-J. J. Rif a-Pous, «Computational and Energy Costs of Cryptographic Algorithms on Handheld Devices,» *Future Internet*, vol. 3, 2011.