El Costo de la Seguridad en Dispositivos Móviles

Enrique V. Carrera[1]

Resumen— Los dispositivos móviles han evolucionado vertiginosamente convirtiéndose en la plataforma de comunicación y computación preferida por las personas. Sin embargo, la creciente popularidad de estos dispositivos va acompañada de amenazas cada vez más preocupantes a la seguridad de sus usuarios. Si consideramos las limitaciones tecnológicas aún existentes en los dispositivos móviles y el hecho de que la mayoría de los mecanismos de seguridad basan su accionar en técnicas criptográficas, podemos esperar ciertas restricciones al intentar proteger las aplicaciones y datos manejados por estos dispositivos.

Es así que este artículo estudia los compromisos existentes entre una adecuada seguridad y factores como el desempeño y consumo de energía de aplicaciones ejecutándose sobre teléfonos celulares y asistentes digitales. Los resultados obtenidos muestran que la seguridad en las aplicaciones móviles no es gratuita y que su correcta implementación requiere una selección cuidadosa de cada uno de los parámetros utilizados por los mecanismos criptográficos. Basados en este estudio, hemos también propuesto un 'framework' de seguridad fácil de usar para el desarrollo de aplicaciones móviles en J2ME.

Abstract—The mobile devices have rapidly evolved, they have become into the platform of communication and computation preferred by people. Nevertheless, the growing popularity of these devices is not always positive because they threaten the safety of their users. If we consider that the technological limitations are still present in these mobile devices and that most of them have safety mechanism which based their operation on cryptographic techniques, so we can expect some restrictions at the moment of attempting to protect the applications and data handled by these devices.

For this reason, this article investigates the commitment between: safety, performance factors and the energy that these devices consume at the moment of being applied on mobiles and digital assistances. The obtained results show that safety in the mobile applications is not free and that a good implementation requires a careful selection of each parameters used by cryptographic mechanisms. Based on this study, we have also proposed a "framework" of safety easy to use for the development of mobile applications in J2ME.

Palabras Claves— Seguridad informática, desempeño de aplicaciones móviles, consumo de energía, telefonía celular.

1Enrique Carrera, Docente de Posgrados, Universidad Tecnológica Equinoccial

Introducción

Los dispositivos móviles se han convertido en una herramienta esencial para cada persona alrededor del mundo. Estos dispositivos combinan una creciente capacidad de cómputo con rápidas velocidades de comunicación, proveyendo transmisiones de voz y datos sin importar la ubicación del usuario. De esta forma, los dispositivos móviles han evolucionado de ser simples agendas electrónicas o PDAs (Personal Digital Assistants) para convertirse en sofisticados y compactos computadores personales. No obstante, debido a que los dispositivos actuales se conectan a una amplia variedad de redes, incluyendo el Internet, para intercambiar un sin número de informacion entre ellos, se han abierto nuevos canales que permiten ataques contra la seguridad de todos sus usuarios.

En otras palabras, mientras los dispositivos móviles van ganando popularidad, van también creciendo los riegos a los que se enfrentan sus usuarios. Aún peor, debido a la creciente penetración de la telefonía celular que en países como EEUU ya supera el 94 % (Ecuador bordea el 85 % [1]), estos dispositivos se han convertido en un objetivo atractivo para los escritores de virus y terroristas que buscan causar daño u obtener ganancias ilegales [2]. Complementando esta situación de mercado, están también las vulnerabilidades existentes en los protocolos y estándares como GSM (Global System for Mobile Communications), SMS (Short Message Service), MMS (Multimedia Messaging Service), Bluetooth, WiFi, entre otros [3], [4], [5].

Afortunadamente, la mayoría de plataformas para el desarrollo de aplicaciones en dispositivos móviles incluye algún tipo de soporte para implementar mecanismos de seguridad. Un ejemplo de dichas plataformas es J2ME (Java 2 Micro Edition) [6], [7] y su biblioteca criptográfica "Bouncy Castle" [8].

Sin embargo, la seguridad no es gratis, especialmente cuando es implementada en un software, pues la mayoría de los mecanismos de seguridad basan su accionar en técnicas criptográficas que consumen abundantes recursos computacionales que contrastan con los escasos recursos de procesador, memoria y batería disponibles en los dispositivos móviles de uso común.

Basado en lo anterior, este artículo busca determinar los costos existentes para la implementación de seguridad en las aplicaciones móviles, especialmente en relación al desempeño de las mismas y al consumo de energía de los dispositivos que las ejecutan. Para ello, hemos desarrollado el prototipo de una aplicación con elevados requisitos de seguridad, donde se han analizado las diferentes vulnerabilidades y amenazas posibles. En base a este análisis inicial se ha propuesto un conjunto de mecanismos de seguridad que ayudarán a reducir los riesgos establecidos. Estos mecanismos son basados en técnicas estandarizadas que posteriormente se evalúan de forma individual para determinar sus características de desempeño y consumo de energía. En términos de desempeño también realiza una evaluación general de la aplicación. Los resultados obtenidos en dichas evaluaciones mencionadas indican que la seguridad es un servicio esencial para cualquier aplicación móvil, pero presenta ligeros inconvenientes que afectan el desempeño de las aplicaciones y que pueden drenar fácilmente la batería del dispositivo. Adicionalmente, este artículo establece que el desempeño y consumo de energía pueden ajustarse mediante una cuidadosa selección de parámetros para cada uno de los mecanismos de seguridad involucrados. Está última condición, nos llevó a proponer un framework de seguridad simple y flexible que permite el desarrollo de aplicaciones para dispositivos móviles. Al momento, este framework está disponible en la forma de una biblioteca J2ME.

Fundamentos Teóricos

Esta sección describe brevemente los principales conceptos involucrados en el estudio realizado: teléfonos celulares, seguridad informática y J2ME.

A. Teléfonos Celulares

Con cerca de 4 mil millones de unidades operando en más de 230 países alrededor del mundo, los teléfonos celulares son los dispositivos móviles con mayor penetración en la sociedad actual [9], [10]. Estos dispositivos de comunicación se están convirtiendo de a poco en los elementos motores de la economía mundial. Muestra de ello es que los mensajes móviles han generado rentas por US$ 130 mil millones a nivel global en 2008, y estos números crecerán a un mercado de US$224 mil millones para el 2013 [11].

La mayoría de estos dispositivos basan su accionar en el uso de procesadores simples que normalmente están alimentados por baterías Li-Ion. En teléfonos celulares básicos, los procesadores utilizados son variaciones de la serie ARM9 (procesador RISC de 32 bits) operando entre 100 y 250 MHz, y sus baterías por lo general bordean los 900 mAh. Por otro lado, los teléfonos tope de línea pueden usar procesadores ARM o XScale operando sobre los 600 MHz, pero estos todavía son una minoría en el contexto mundial actual. Adicionalmente, estos dispositivos tienen al menos conectividad GSM (que incluye SMS/MMS), GPRS (General Packet Radio Service) y Bluetooth/IR. Los últimos modelos inclusive ofrecen 3G/3.5G, WiFi, GPS (Global Positioning System), entre otros servicios de comunicación.

Finalmente, si bien cada fabricante puede proveer una plataforma diferente para el desarrollo de sus aplicaciones, más del 90% de los dispositivos móviles pueden ser programados mediante J2ME gracias al soporte incluido en los procesadores ARM para el uso del lenguaje Java [2].

B. Servicios de Seguridad y Criptografía

Un servicio de seguridad es cualquier servicio de procesamiento o comunicación que es provisto por un sistema para dar una clase específica de protección a los recursos del sistema. X.800 y RFC 2828 definen 6 servicios de seguridad [12]:

• ***Autenticación:*** Servicio relacionado con asegurar que una comunicación es auténtica. Para ello, la fuente y el destino de la comunicación deben identificarse mutuamente.

• ***Confidencialidad:*** Servicio que protege los datos transmitidos de ataques pasivos. Varios niveles de protección pueden ser identificados, aunque el servicio más general protege todos los datos transmitidos entre 2 usuarios por un período de tiempo establecido.

• ***Integridad:*** Servicio que depende si el mismo es o no orientado a conexión. Un servicio de integridad de datos sin conexión trata con mensajes individuales proveyendo protección contra modificación de mensajes únicamente. El servicio orientado a conexión también provee protección contra pérdida y cambio de orden de los mensajes.

• Innegabilidad: ***Servicio para prevenir que la fuente o destino*** nieguen la existencia de un mensaje transmitido.

• ***Control de Acceso:*** Habilidad para limitar y controlar el acceso a sistemas y aplicaciones mediante enlaces de comunicación. Para esto, cada entidad que trata de ganar acceso es primero autenticada, de tal forma que los derechos de acceso sean asignados correctamente. Disponibilidad: Propiedad de un sistema o recurso de estar accesible y disponible bajo la demanda de una entidad autorizada del sistema y de acuerdo a especificaciones de desempeño preestablecidas.

Los servicios de seguridad son implementados a través de mecanismos de seguridad. La criptografía es uno de los mecanismos de seguridad más usados para proveer autenticación, confidencialidad, integridad e innegabilidad. Control de acceso y disponibilidad pueden ser garantizados por la red en el caso de la telefonía celular [5].

La criptografía, en particular, no es más que el uso de algoritmos matemáticos para transformar datos a una forma que no es inmediatamente legible. La transformación y subsecuente recuperación de los datos dependen de un algoritmo y cero o más llaves criptográficas. Los algoritmos de criptografía más conocidos usan esquemas de llave compartida (e.g., DES, IDEA, Blowfish, AES) y llave pública (e.g., RDA, DSA, El Gamal, ECC) [12].

Una aplicación interesante de la criptografía es la firma digital. La firma es un conjunto de datos anexado a una unidad de comunicación que permite al recipiente de esa unidad probar la fuente e integridad de los datos protegiéndola contra suplantaciones.

J2ME

J2ME es un esfuerzo de Sun Microsystems para portar el lenguaje Java a los dispositivos con limitaciones de recursos, manteniendo disponible un subconjunto de la funcionalidad base. De hecho, nuestro trabajo se enfoca en las clases CLDC (Connected Limited Device Configuration) y MIDP (*Mobile Information Device Profile*) [7], [8]. Ambos conjuntos de clases definen el perfil MIDP en la terminología J2ME.

El perfil MIDP ha sido desarrollado para soportar el nicho de los teléfonos celulares o dispositivos similares limitados por restricciones de pantalla y teclado, además de las restricciones obvias de batería, procesador, memoria y ancho de banda. Este perfil contiene una serie de APIs (*Application Programming Interfaces*) que permiten crear cualquier aplicación yendo desde juegos con gráficos personalizados hasta aplicaciones de negocios a gran escala basadas en fuentes de datos internas y externas.

El desarrollo de una aplicación MIDP (*o midlet*) ha sido simplificada mediante un ambiente introducido por Sun Microsystems: J2ME WTK (*Wireless ToolKit*). Adicionalmente al WTK, utilizamos 2 paquetes opcionales descritos brevemente a continuación:

Bouncy Castle. Este paquete es una implementación Java de algoritmos criptográficos. El paquete está organizado de forma que contiene un API ligero que se puede utilizar en cualquier ambiente, incluyendo J2ME. Este API también incluye un proveedor para JCE (*Java Cryptography Extension*). La ventaja de escribir código de aplicaciones que usan interfaces de proveedores es que el proveedor actual puede ser escogido en tiempo de ejecución. Esto es valioso para aplicaciones que desean usar proveedores con soporte en hardware para los cálculos criptográficos, o donde una aplicación puede haber sido desarrollada en ambientes con controles de criptografía para exportación.

WMA. Este paquete provee una interface común para permitir a los midlets enviar y recibir mensajes de texto o binarios, e incluso mensajes multimedia. Típicamente, estos mensajes son parte de sistemas de mensajería store and forward tal como SMS y MMS que garantizan la entrega de los mensajes. WMA (*Wireless Messaging API*) no coloca límites al tamaño de los mensajes ni ninguna otra restricción, pero el programador debe estar consciente que el mecanismo de transporte base podría hacerlo.

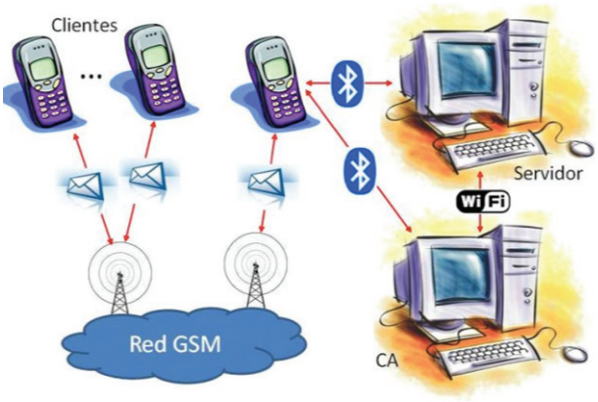

Figura 1. Estructura general de la aplicación de voto electrónico.

Metodología

Con la finalidad de realizar una evaluación pormenorizada del efecto de los mecanismos de seguridad en el desempeño y consumo de energía de las aplicaciones móviles se implementó el prototipo de un sistema de voto electrónico. Sobre este sistema se adaptaron mecanismos de seguridad estandarizados para los diferentes procesos de comunicación.

A continuación se presenta una descripción más detallada.

Voto Electrónico Móvil

La figura 1 muestra la estructura general del prototipo de voto electrónico mediante dispositivos móviles. Cada votante puede acceder a la lista de candidatos y registrar su voto por el candidato de su preferencia mediante un teléfono celular con capacidad SMS. Adicionalmente, la comunicación entre los servidores y el teléfono principal es mediante enlaces Bluetooth y WiFi. Del esquema propuesto se percibe que existen varios riegos a la seguridad del sistema relacionados con la autenticación, confidencialidad, integridad e innegabilidad de cada uno de los votos.

De esta manera, con la finalidad de autenticar los diferentes entes operando en el sistema, se ha adicionado una autoridad de certificación (CA) que entrega certificados digitales (DC) a cada uno de los dispositivos, incluyendo al servidor de la aplicación. Así, el sistema de voto electrónico cuenta con 3 fases bien definidas:

1. Inicialización: Entrega de los DC a cada uno de los dispositivos previa solicitud y entrega de su llave pública. Es importante notar que el par público/privado debe ser generado en cada dispositivo por seguridad.

2. Votación: Los participantes de la elección se autentican con el servidor intercambiando sus DC y posteriormente registran su voto mediante mensajes que son incorpora dos a la base de datos del servidor.

3. Revocatoria: Si existe algún problema con un DC, este debe ser revocado. Esta funcionalidad no está implementada al momento.

Seguridad SMS

Con la finalidad de garantizar la seguridad de cada voto enviado mediante SMS se ha implementado un esquema similar al propuesto por PGP para intercambio de e-mails [12], [13].

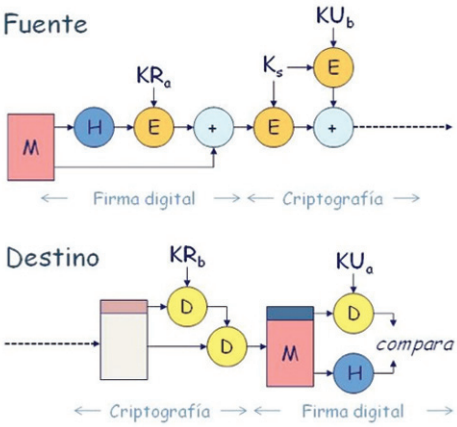

Figura 2. Esquema de seguridad para la comunicación SMS.

La figura 2 muestra las acciones llevadas a cabo en base a las llaves públicas (KUi) y privadas (KRi) de las partes, así como una llave compartida (Ks) generada de forma aleatoria. Las llaves privadas nunca salen del dispositivo y las llaves públicas son obtenidas de los DC obtenidos durante el proceso de autenticación. Como se puede observar, funciones hash (H) y operaciones criptográficas (E y D) son la base de este proceso.

Seguridad Bluetooth

De forma semejante, para garantizar la seguridad en los enlaces Bluetooth se ha implementado un esquema similar al usado por TLS [14], [15]. Este esquema también se fundamenta en el uso de las llaves públicas y privadas de cada parte, así como en la aplicación de criptografía de llave compartida. Al igual que en el caso de SMS, las llaves privadas nunca salen del dispositivo y las llaves públicas son obtenidas de los DC intercambiados al momento de crear la conexión segura.

Evaluación Y Resultados

Basados en el prototipo descrito en la sección anterior, se llevaron a cabo una serie de evaluaciones en términos de desempeño y consumo de energía usando teléfonos móviles de

rango medio, un Nokia3500 Classic y un Sony Ericcson K550i. Estos teléfonos incluyen un procesador ARM9 corriendo a 104 y 218 MHz, respectivamente. La batería de teléfono Nokia3500 es la BL4C (860 mAh Li-Ion) y tiene una expectativa de vida de 12 días en standby y cerca de 3 horas hablando por teléfono. Por otro lado, la batería de teléfono Sony Ericcson es la BST-33 (950 mAh Li-Ion) con una expectativa de vida de 14 días en standby y cerca de 7 horas hablando por teléfono. Ambos teléfonos soportan CLDC 1.1 y MIDP 2.0, además de WMA 2.0.

Todo el código fue compilado en un computador Linux ejecutando el IDE NetBeans 6.8 con WTK 2.5.2. La versión de la máquina virtual Java es la 1.6.0 (IcedTea6 1.8). Las medidas de tiempo fueron hechas a través de temporizadores Java incluidos en las aplicaciones, mientras que las mediciones de energía fueron tomadas usando multímetros Fluke de alta precisión.

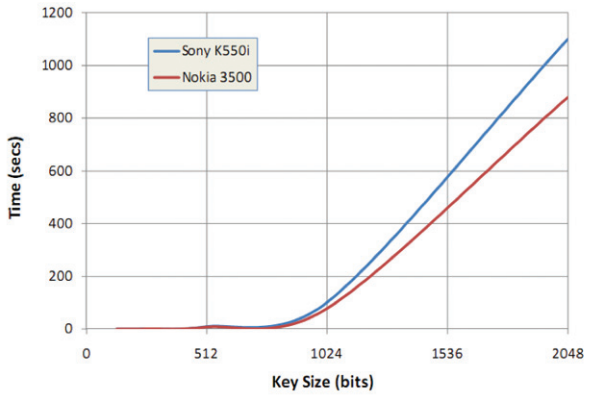

Figura 3. Tiempos de generación de un par público/privado RSA en función del tamaño de las llaves.

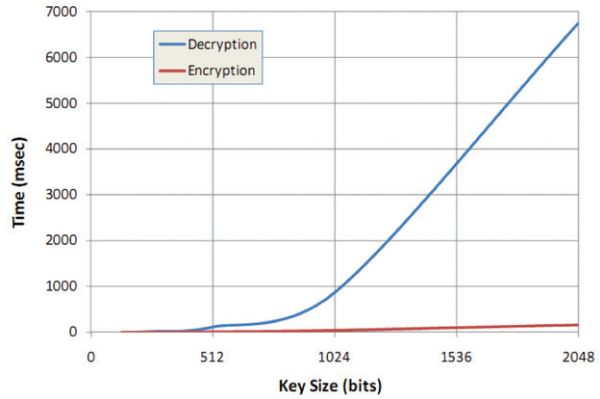

Figura 4. Tiempos de la criptografía RSA sobre 64 bytes en el Nokia-3500.

A. Desempeño

La figura 3 muestra los tiempos requeridos por los 2 teléfonos evaluados para generar un par de llaves RSA en función del tamaño de las llaves solicitadas. Puede observarse que llaves de tamaño inferior a 1024 bits presentan tiempos aceptables, pero que en el caso de llaves de 2048 bits pueden requerir hasta 10 minutos. Si bien este proceso se realiza una sola vez durante la fase de inicialización, tampoco es recomendable que se desafíe a la paciencia de los usuarios.

Por otro lado, la figura 4 presenta los tiempos requeridos para una criptografía RSA en el teléfono Nokia-3500 en función del tamaño de las llaves. Los tiempos para el teléfono Sony Ericcson son bastante parecidos. Podemos observar que el tiempo de criptografía con la llave pública (Encryption) es bastante aceptable en todos los casos. No obstante, el tiempo de criptografía con la llave privada (Decryption) puede llegar a casi 7 segundos con llaves de 2048 bits, lo que podría limitar la escalabilidad de las aplicaciones.

Con relación a la criptografía de llave compartida, la figura 5 muestra los tiempos requeridos por los algoritmos más conocidos para codificar 64 bytes. Note que todos los tiempos son inferiores a 6.2 ms lo que permite latencias de usuario imperceptibles en la mayoría de casos. Finalmente, los tiempos utilizados por funciones hash como MD5 y SHA1 son también bastante bajos. En ambos teléfonos y usando mensajes de 64 bytes, MD5 toma aproximadamente 8 ms mientras que SHA1 demora en torno de 6 ms. Estos tiempos corresponden a funciones hash aplicadas a 64 bytes de datos.

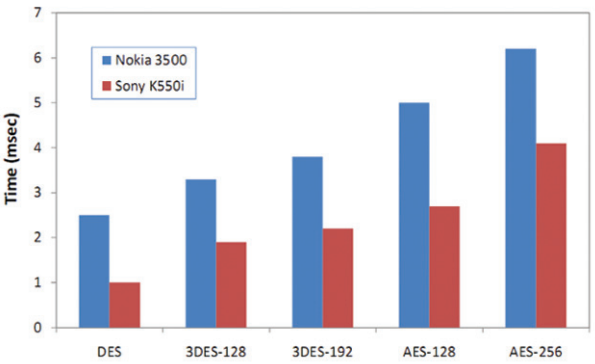

Figura 5. Tiempos de la criptografía de llave compartida sobre 64 bytes.

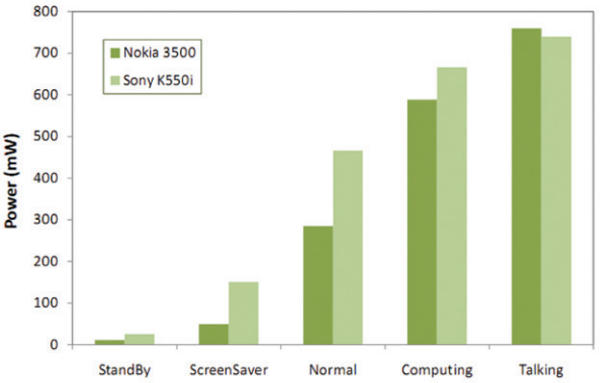

Figura 6. Potencia consumida por los teléfonos móviles durante ciertas actividades comunes.

B. Consumo de Energía

Si bien el desempeño de las aplicaciones móviles es determinante en el nivel de satisfacción de los usuarios finales, el consumo de energía de estas aplicaciones también afecta en forma significativa la experiencia de uso del dispositivo. Es por ello que esta evaluación no estaría completa sin una medición de los niveles de energía requeridos por los mecanismos criptográficos.

Sin embargo, para tener un punto de referencia es apropiado comenzar especificando el consumo normal de un teléfono celular bajo situaciones normales. Así, la figura 6 resume el consumo de potencia de ambos dispositivos usados en este estudio. Note que la utilización del procesador en actividades de cálculo extremas (i.e., 590–670 mW) es comparable al consumo del dispositivo en medio de una llamada de voz (i.e., 740–760 mW), lo cual es sumamente alto si consideramos que la última actividad involucra activar los circuitos de transmisión inalámbrica y amplificación. Aunque la potencia consumida es un indicativo interesante, la energía requerida por cada actividad es el factor determinando para establecer el tiempo de vida de una batería. De esta manera, la figura 7 resume el consumo de energía para enviar un mensaje SMS con únicamente firma digital, criptografía y con firma + criptografía en el teléfono Nokia-3500. Los mensajes evaluados incluyen 64 bytes de datos, criptografía RSA con llaves de 512 bits, criptografía AES con llave de 256 bits y funciones hash SHA1.Note que al usar firma digital o criptografía por separado se produce un desbalance en la energía consumida durante el envío y recepción debido al uso de la llave pública o privada, según sea el caso.

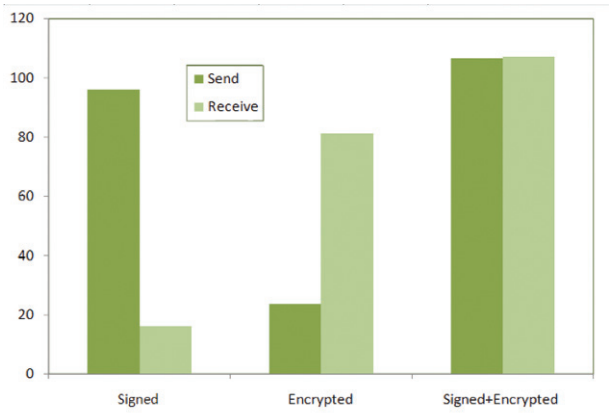

Figura 7. *Energía consumida por el Nokia-3500 para enviar y recibir un SMS*

Por otro lado, mensajes firmados y encriptados requieren alrededor de 107 mJ (excluyendo la energía gastada por el dispositivo conectándose a la red celular del proveedor) tanto para envío como recepción, lo que representa una fracción bastante insignificante de los aproximadamente 11 KJ disponibles en una batería BL-4C completamente cargada.

Adicionalmente, se ha establecido que durante la transmisión o recepción de mensajes a través de Bluetooth se con sumen aproximadamente 610 mW en el teléfono Nokia-3500. Este valor prácticamente equivale al de uso intensivo de su procesador, por lo que el número de mensajes intercambiados, así como la cantidad de bytes enviados, debe minimizarse.

C. Framework de Seguridad

Como se puede observar en los resultados presentados hasta el momento, el desempeño y consumo de energía de las aplicaciones móviles seguras dependen de cada uno de los parámetros utilizados por los mecanismos criptográficos. La selección de estos parámetros no es obvia y varía conforme los recursos computacionales disponibles en cada dispositivo, lo cual cubre un rango sustancialmente grande de opciones.

Para simplificar el desarrollo de estas aplicaciones móviles hemos sintetizado los diferentes mecanismos criptográficos en una biblioteca J2ME de fácil utilización y que permite adaptar dinámicamente los requisitos de seguridad y desempeño (incluyendo consumo de energía) conforme las características del hardware donde son ejecutadas las aplicaciones. De esta forma, los detalles de bajo nivel asociados al uso de diferentes esquemas criptográficos, así como la selección adecuada de parámetros, se enmascaran en llamadas de alto nivel que simplifican la tarea de los programadores que muchas veces no desean distraerse de la funcionalidad básica de su aplicación.

La mencionada biblioteca J2ME está disponible en http://latolita.usfq.edu.ec/voting.html e incluye una amplia variedad de clases para manejo de DC, criptografía de llave pública, criptografía de llave compartida y funciones hash.

Conclusiones

El uso de mecanismos criptográficos dentro de las aplicaciones móviles es una actividad que puede consumir cantidades significativas de recursos que se ven reflejadas en tiempo y consumo de energía. Sin embargo, las nuevas aplicaciones de estos dispositivos *(e.g., Mcommerce, Mbanking, Mpayment, Mvoting)* requieren cada vez mayores garantías de seguridad para su adecuado funcionamiento. Es por ello que la selección adecuada de los mecanismos de seguridad y los parámetros criptográficos usados por las aplicaciones móviles son de extrema importancia para garantizar tanto una adecuada seguridad como un nivel óptimo de satisfacción por parte de sus usuarios.

Basados en este estudio, creemos que *frameworks* de desarrollo como el propuesto aquí tendrán una elevada relevancia y aceptación por parte de los programadores que no desean lidiar con los problemas de bajo nivel asociados a los

algoritmos criptográficos de uso común.

Bibliografía

[1] D. Salazar and D. Mora, "La brecha digital en los servicios de telefonía fija, telefonía móvil e Internet en el Ecuador," Master's thesis, Universidad San Francisco de Quito, Quito, Ecuador, December 2009.

[2] F. M. David, E. M. Chan, J. C. Carlyle, and R. H. Campbell, "Cloa ker: Hardware supported rootkit concealment," in IEEE Symposium on Security and Privacy, Oakland, CA, May 2008.

[3] E. V. Carrera and J. P. Albuja, "Secure and efficient SMS communica tion," Trends, vol. 3, no. 3, pp. 81–87, June 2009.

[4] E. Ferro and F. Potorti, "Bluetooth and wifi wireless protocols: A survey and a comparison," IEEE Wireless Communications, vol. 12, pp. 12–26,
2004.

[5] M. Toorani and A. A. Beheshti, "Solutions to the gsm security weak nesses," in II International Conference on Next Generation Mobile Applications, Services, and Technologies, Cardiff, UK, September 2008, pp. 576–581.

[6] J. Knudsen, Kicking Butt with MIDP and MSA: Creating Great Mobile
Applications. New York, NY: Prentice Hall, 2008.

[7] S. Li and J. Knudsen, Beginning J2ME – From Novice to Professional,
3rd ed. New York, NY: Apress, 2005.

[8]D. Hook, Beginning Cryptography with Java. Indianapolis, IN: Wrox Press, 2005.

[9] A. Browne, "Multiple SIMs: Quantifying the phenomenon taking mobile penetration beyond 100 %," Informa Telecoms and Media Report, Tech. Rep., May 2007.

[10] I. T. Union, "ITU Corporate Annual Report 2008," International Tele communication Union, Switzerland, Tech. Rep., 2009.

[11] Portio Research Ltd., "Mobile messaging futures 20092013," Portio Research Limited, LW1478.MMF0913, November 2008.

[12] W. Stallings, Cryptography and Network Security: Principles and Prac tice, 5th ed. New York, NY: Prentice Hall, 2010.

[13] J. P. Albuja, "Implementing security in mobile messaging," University San Francisco of Quito, Quito, Ecuador, BE's Thesis, May 2009.

[14] P. Ducos and F. Castillo, "Secure digital money exchange using mobile devices," University San Francisco of Quito, Quito, Ecuador, BE's Thesis, May 2008.

[15] R. Oppliger, SSL and TLS – Theory and Practice.
Norwood, MA: Artech House, 2009.

Enrique V. Carrera obtuvo su título de Ingeniero Electrónico en la ESPE (Ecuador) en 1992 Posteriormente, obtuvo su Maestría en Ingeniería Eléctrica en la PUCRJ (Brasil) en 1996. También obtuvo su título de Doctor en Ciencias especialidad Sistemas de Computación en la UFRJ (Brasil) en 1999. Su postdoctorado lo realizó en la Universidad de Rutgers (EEUU) entre los años 2000 y 2004. Actualmente, el Dr. Carrera es profesor en el Departamento de Ingeniería de Sistemas de la Universidad San Francisco de Quito, Ecuador. Entre sus publicaciones cuenta con aproximadamente 40 artículos técnicos publicados tanto en revistas como en conferencias internacionales. Correo electrónico para contacto: vcarrera@usfq.edu.ec.