Bartosz PAWŁOWICZ, Mateusz TYBURA
DEPARTMENT OF ELECTRONIC AND COMMUNICATION SYSTEMS
RZESZOW UNIVERSITY OF TECHNOLOGY,
12 Powstańców Warszawy St., 35-959, Rzeszow, Poland

Privacy and security of data on mobile devices with Android and Windows Phone operating systems

Abstract

The paper focuses on three significant aspects which can improve safety of data and also increasing privacy of users using tablets and smartphones. The chosen platforms are Android and Windows Phone. The aim of the work is to develop library independent from operating system API that enables encrypting and hiding the data stored in mobile device. Also applications that demonstrate functions of the library was developed. The library will be used in securing data from NFC modules used as RFID readers.

Keywords: security, mobile, steganography, cryptography.

1. Introduction

Security and privacy are very complex issues. Their violation allows to gain unauthorized access to the device, remove money from the account or for example to seriously damage the IT infrastructure as a result of actions leading to the so-called cyberwar.

Today mobile devices are integrated with many communication systems such as mobile network, WiFi, NFC, Bluetooth, USB ports. Different kind of sensors such as the ones responsible for detecting proximity to the ear during call, accelerometers, gyroscopes etc. were also placed in these devices. These devices has been already embedded in structure of mobile devices to gain many information about the user and the environment. Potentially harmless phones, became a tool allowing constant tracking their users location. One simple data read became sufficient to determine by the sensor if the phone is applied to the ear. Thanks to this kind of phone's motion detection to obtaining accurate information about owner's movement became easy.

Today mobile devices, are also equipped with an integrated NFC module by default. NFC technique gives a number of new possibilities, e.g. allows for fast establishing of a direct connection between devices, proximity payments or reading out RFID labels of marked products. Replacement of classic barcodes by their electronic counterparts brings new possibilities. For example, with a dedicated application installed on smartphone, a customer can read basic and enhanced information from memory of RFID transponder placed as label on product. Thanks to this fact, a connection with the local shop's database is unnecessary to show price and other information of a product. Manual products scanning allows for generation of a list of a virtual shopping cart which informs a customer online about total cost of products put into the cart. Information can be transmitted to computer during checkout. These functions ale also a challenge from the viewpoint of security and privacy. Reading of product labels gives the information about food, clothes, and many other consumer goods bought by device user and such information is stored in memory of mobile device. Examples of mobile applications which enables reading of RFID labels are already known. So far security and privacy in those applications weren't taken seriously. When such applications will become popular sensitive data which they deliver should be hidden or encrypted. The aim of this work was to develop universal library independent from system API which can be used to secure data and might be adapted to various applications. Examples of three test applications based on developed library were also presented.

2. Application structure

Many applications have been designed for purpose of increasing privacy level and data security of the users. During current project software library that enables encrypting and hiding data on mobile devices has been developed along with three example applications which functions utilize these library.

First application implements encryption functions. It aims to deliver to user a data encryption environment with ability to encrypt data in-flight with chosen algorithm. Hiding data function has been implemented in another application. User can take a photo which then records the text data in a particular key manner. The finished picture has been saved in standard location of graphic files to find it easier by user. Already at the design stage there were noticed few potential problems of usage of this application. First if the image will be saved both before and after hiding the data it would be entitled to easy analysis of changes at the level of specific pixels thereby it would be easy to expose the data [1]. Next problem was to find a graphic format that would be supported by both tablets and smartphones and also if data record is not associated with operations of compression loss in which the saved data would be lost [1, 2, 3, 4]. The final problem was the eventual size of the created file. Because of using compression less format there might have been problem with big size files and lack of ability to send this files to different devices. Last application has been designed only for smartphones and it the functionality of extremely simple contact manager was implemented in it. Because of extended privacy the written data were only the name of the contact and it is phone number. Additionally the access to contact list was protected by using a pin code.

The primary assumption for developed library was to obtaining independence from API delivered by operating system. The aim was to achieve easy software development and increase independency from changes introduced in external modules.

In next stage of design in each part of library the two different modules have been separated. Implementation module contains classes, types of data and all other components being implementations of adequate action and data used in the entire system. In the module known as interface module the constructions known as interfaces were included. In that way it was defined that the public API is available for each module and the independence of own application from own APIs was gained. Moreover it allowed easy change in the case of the desire to add another possibilities or creating totally new variants of existing solutions.

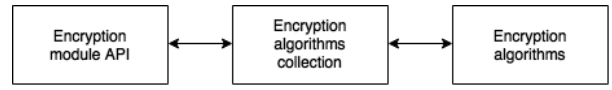

Fig. 1. The essential functional blocks of encryption module

After designing the structure of library two essential functions of application were implemented: encryption and hiding data. Because of security of information in a whole process the only place where information is sent explicitly is the input of encryption module. It's output was encrypted as fast as possible. Because of using many algorithms and possibility to add them in the future encryption has been divided to three essential functional blocks (Fig. 1).

In the project three software modules has been developed (Fig. 2). The inheriting has been used after *CryptoAlgorithm* class. It was set not only as base class for all classes implementing encryption algorithms but also it includes a primary implementation of the algorithm involving returning the same public text as acquired on input. It was handled in this way a situation in which there was not any implementation found. In class the field containing algorithm repository has been included. In case of relation between encryption engine and repository we have to deal with including one repository in one engine. In case of algorithms many of them are stored in one repository. Relation between encryption engine and algorithm have been stemmed from using concrete algorithms after accessing them from repository.

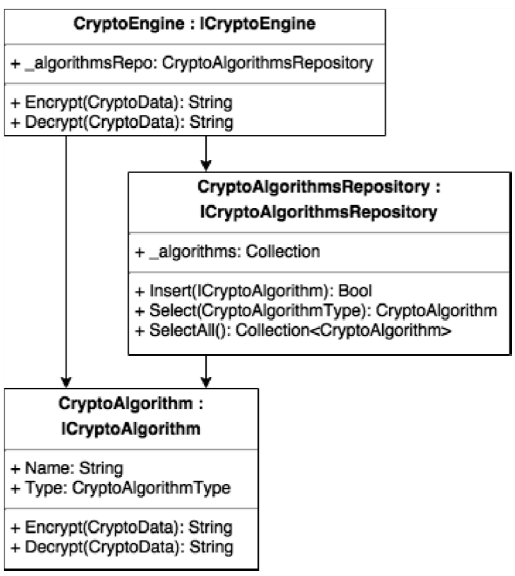

Fig. 2. The encryption module class project

A few class dealing with encryption has been designed. There are the easiest possible implementations of encrypting algorithms in them, delivered by both API of operating systems as the ones designed by the authors of this work. All these classes have been established as a child classes of primary encryption algorithm class. There is an implementation in it, which action was only to return at the output a series supplied on the input. In turn in the field signifying type of algorithm and in its description there is a clear set of its uselessness.

Steganography [5] module was built by using the same way of division as in case of module dealing with encryption. Only one implementation of steganography algorithm have been used. Also a different way of writing input data has been designed. Because of hiding data in a files a string as only data to hide, arrays as files data has been used in which it was necessary to hide the data and calculation to determine the selected algorithm.

3. Application development

After designing and performing unit test of all created library modules it was necessary to move on to designing application for use. Initially assumed was using MVC standard. Views were built in declarative way, using domain specific XML language [6]. As first software and modules were performed in which all cryptographic operations were included (Fig. 3). Yet at the design stage it was assumed that none operations of transmitting through network or eventually storing any files with keys or they fragments were implemented. Moreover it was decided to implement programs so that operations were executed maximally fast, better directly during a moment of inputting data by user. In this way during conception phase the complex algorithms like DES, AES [7] or RSA [8] have been rejected and more classic ones like Ceasar, Vigenere and Playfair have been chosen [9].

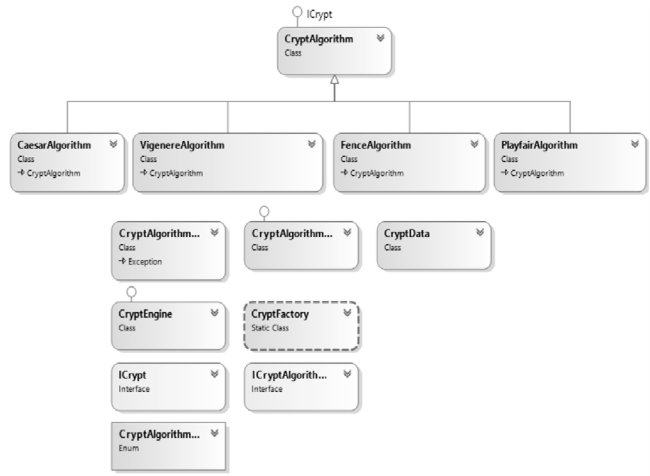

Fig. 3. Encryption module class diagram

In case of applications for steganography the app have been developed in uniform way. It resulted from problems noticed during developing universal application in case of Windows platform and designing one application for phones and tablets on Android platform. In the project code was also separated for model, view and control (Fig. 4). Additionally a few helping class have been developed.

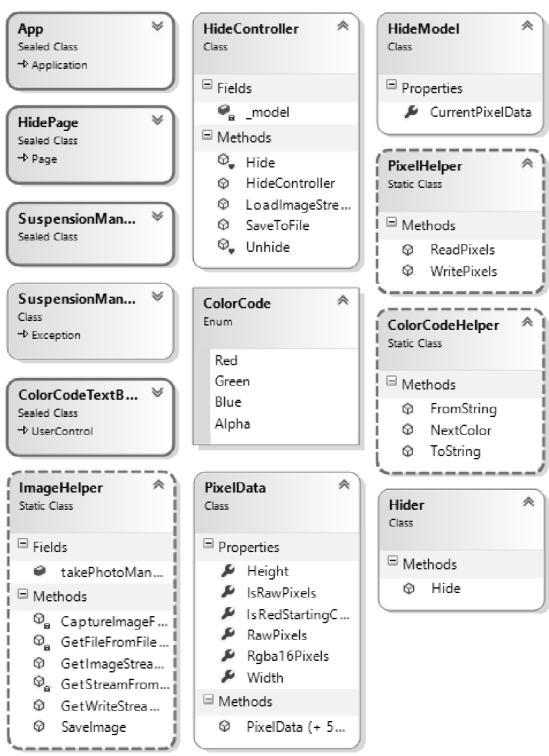

Fig. 4. Stenography program class diagram

In *HideController* class the whole application logic was adopted. Logic of application contains appropriately processes of hiding data, initiation of controller, image loading, its saving and pulling data. In *HideModel* class which contains model of application only data about pixels were included. In class *PixelHelper* the methods have been implemented in which the writing and reading of pixels have been conducted.

In *ImageHelper* class the methods responsible for operation on the graphic files were implemented. Among them it was listed that operations of acquiring new photo from camera, reading existing BMP file with help of system file chooser screen and downloading appropriate streams and finally saving graphic file [4]. In *ColorCodeHelper* the appropriate methods switching from *ColorCode* type to string and conversely and the one responsible for generating successive colors in RGB pallet has been developed. The *ColorCode* type contained only symbolic marks of another color components which contains red, green, blue and alpha. In turn *PixelData* type contained all information about remembered pixels. It have been dimensions of original photo, information about bit value – 8 or 16 on each color component and information whether components were arranged in order started with red color. It should also be noted that in case of version for Android system a few meaningful changes have been made. One of them was significant rebuild of *ImageHelper* class.

First the code responsible for downloading image from camera has been separated. In Android system version the class has been made responsible for generating JPEG and PNG files, acquiring image from specific logical location and storing it. The *Hider* class has been rebuilt. It was made responsible for process of hiding and reading data. Because of specific way of operations on pixels the new additional private methods have been added to switch one of the color component and also second one which main task was returning specific color component. In case of Android operating system color was returned not as an array of division for specific color components but one 32bit number in which information about all components that have been encoded [3].

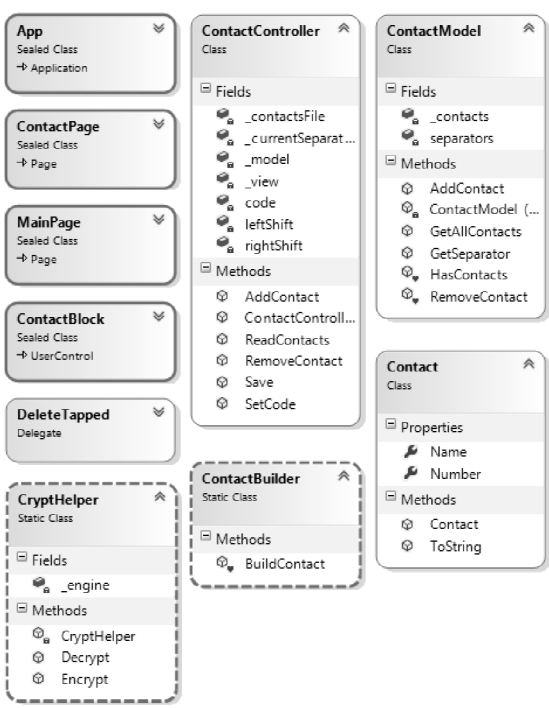

Fig. 5. Contact module class diagram

Application in which contacts have been hidden was made as last. Total simplicity and easy interface with low amount of data that bonded with concrete contact has been assumed to solve the problems related with too much amount of data that have been placed in standard phone contacts (Fig. 6). Code has been divided into model, view and controller, contact classes and two helping classes (Fig. 5). A methods have been adopted in which there is a possibility to add, delete and save contacts with setting security code. That code was used during encryption and decryption of a file with contacts and also during searching for a file. The name of the file has been encrypted by using the code and it has been putted in private for application location. The process of encrypting and decrypting file with contacts was based on two steps. In first one the name of the file has been generated. For each code combination the different name has been generated thereby different file was opening. Thus access to users' data has been prevented while in case of attack it was sought to make it look like the access has been granted.

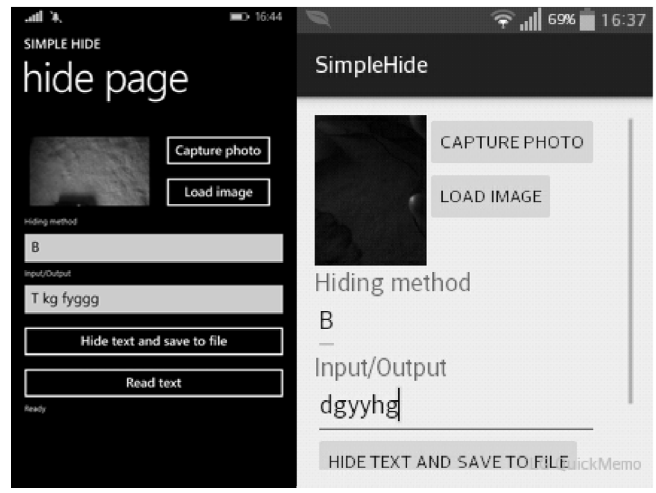

Fig. 6. Graphical interface of example developed applications

4. Tests

At first the eventual security has been tested thanks to which it would be possible to block acquiring the data from phones and tablets if they would be connected with computer through USB port. Test were first run by using laptop with Windows 8.1 operating system. Test was started with phone with Android system. After connecting it to computer Windows system has recognized new device correctly as LG-E410i phone. Then it was checked if the device has recognized both internal memory and installed in it microSD memory card. Access to both memories has been granted without any complications however the characteristic structure familiar in Linux operating system for it was not recognized.

Another attempt to acquire data was made by using Kali Linux distribution. Already at the start of testing environment a problems with both UEFI and SecureBoot which prevented from running it were encountered. Only thanks to changes in BIOS it was possible to run more tests. At first the LG device with Android system has been connected and in terminal by using *lsusb* command it was possible to acquire all information about devices connected with PC through USB port. It was established that device was connected and recognized without any problems. However the result was not fully satisfying. Phone has been recognized as digital player.

Next the content of the phone has been checked also stating that it was not noticed full structure of the files which was expected in Windows system. At the same time on the base of presence of the System Volume Information folder the thesis has been putted forward that inside the memory of the phone the NTFS file system has been used [10]. Also after using the mount command no information about mounted device has been acquired and from properties of one of the folders there was no acquired data basing on which access to memory would be granted. In case of tablets with Android system the same results have been achieved. Tablet with Windows was not installed, only loading has been started. However it is worth to mention that a whole access to the drive has been shared.

5. Conclusions

In the paper process of designing and development of software library supporting hiding and encrypting information stored in memory of mobile device was presented. Also examples of applications that utilize developed library were shown. Basic tests of applications were made. In the next step the more advanced encryption and steganography algorithms will be included in the library. Currently an application which main function is reading of ISO 15693 RFID transponders using NFC module is under development. In final step the presented library will be integrated with this application. It will enable security of data read from RFID labels and stored in device memory independent from system API.

6. References

[1] Windows Phone 8/8.1 specification https://dev.windowsphone.com/en-US/OEM/docs/Welcome/Windows_Phone_8.1
[2] Yaghmour K.: Embedded Android, O'Reilly Media Inc., 2013.
[3] Android specification: http://developer.android.com/develop/index.html
[4] Whitechapel A., McKenna Sean S.: Windows Phone 8 Development Internals, ISBN-13: 978-0735676237, Microsoft Press, 2013.
[5] Katzenbeisser S., Petitcolas F. A. P.: Information Hiding Techniques for Steganography and Digital Watermarking, ISBN 1-58053-035-4 Artech House, 2000.
[6] Boyce J., Shapiro J. R. and Tidrow R.: Windows 8.1 Bible, ISBN 13: 9781118835319, Wiley, 2014.
[7] FIPS 197 – AES specification http://csrc.nist.gov/ publications/fips/fips197/fips-197.pdf
[8] PKCS #1 v2.2 – RSA encryption http://www.emc.com/emc-plus/rsa-labs/pkcs/files/h11300-wp-pkcs-1v2-2-rsa-cryptography-standard.pdf
[9] Vaudenay S.: A Classical Introduction to Cryptography: Applications for Communications Security, ISBN 9780387258805, Springer, 2005.
[10] Russinovich M., Solomon D. A. and Ionescu A.: Windows Internals Part 1, 6th edition, ISBN: 978-0-7356-4873-9, Microsoft Press, 2012.

Received: 02.10.2016 Paper reviewed Accepted: 02.12.2016

Bartosz PAWŁOWICZ, PhD, eng.

Currently working at Rzeszow University of Technology in Department of Electronic and Communications Systems. Mainly involved in a research team focusing on the development of RFID contactless identification techniques and coordination of the work of the Students Scientific Circle of Electronics and Information Technology at the Faculty of Electrical and Computer Engineering.

e-mail: barpaw@prz.edu.pl

Mateusz TYBURA, Msc, eng.

Mateusz Tybura has graduated from Rzeszow University of Technology in Computer Science (Msc, eng). Currently he is both working as an software developer and studying for PhD degree on the same university. He is also a member of KNEiTI scientific circle. His main interests are security, mobile technologies and artificial intelligence.

e-mail: tyburam@hotmail.com