JARAMOGI OGINGA UNIVERSITY OF SCIENCE AND TECHNOLOGY

SMSENCRYPT ANDROID APPLICATION

(ALGORITHMIC APPROACH)

NAOMI MARIGA

I132/0883/2013

A PROJECT SUBMITTED TO THE SCHOOL OF INFORMATICS AND INNOVATIVE SYSTEMS IN PARTIAL FULFILLMENT FOR THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SECURITY AND FORENSICS AT JARAMOGI OGINGA ODINGA UNIVERSITY OF SCIENCE AND TECHNOLOGY (JOOUST)

DECEMBER 2016

DECLARATION

I **Naomi Kabura Mariga** declare to the best of my ability that this project is as a result of my own efforts and has never been submitted for any academic award to this university and any other university or institution.

No part shall be reproduced without my consent.

Name: **Naomi Mariga**

Signature……

Date ……..:……

Supervisor

Name: **Dr. Solomon Ogara**

Signature……

Date ……..:……

ACKNOWLEDGEMENT

I acknowledge God for giving me strength and the will to undertake this study and friends for their support. I also acknowledge contributions of all my lecturers with their intellectual and tutorial advice, more so I acknowledge Dr.Ogara for being the best supervisor on guiding me accordingly and Billcountry Mwaniki for being my technical adviser in the encryption algorithm testing and implementation, Jaramogi Oginga Odinga University of Science and Technology for provision of a well-equipped and modern library for access to information.

DEDICATION

I dedicate this to my supervisor for his material and academic guidance and constant support as well as all friends and lecturers who gave me the support. Lastly to my parents and my siblings for support and love they gave me in fulfillment of the achievement of this study.

ABSTRACT

Due to the risk involved in stored messages on phones being accessed by persons who are not supposed to access it was proposed that an application that stores messages in encrypted form be developed. SMSEncrypt has been developed to take care of this problem. SMSEncrypt is an android based application which sends and stores messages in a cipher form.

This paper focuses on showing how SMSEncrypt has been developed: The methodology used, the conceptual design, testing and implementation. It also shows how this application has addressed the problem. Ahmet et al introduces BabelCrypt, a system that addresses the problem of retrofitting arbitrary mobile chat applications with end-to-end encryption. This system protects messages against access by the messaging service providers. Protecting against service providers is not enough since they are not the only potential threats to messages privacy. Therefore SMSEncrypt takes care of encrypting the messages in storage. The methodology used is software prototyping since it was necessary to consider feedback from users and improving on the prototype. The implementation and testing is done using two android real devices.

SMSEncrypt sends and stores messages in encrypted form using a base 64 encoding, 8set encryption algorithm as well as sender and receiver phone numbers encoding. SMSEncrypt has successfully solved the problem and has as well ensured maintenance of confidentiality and integrity as goals of security. This study has recommended that further work be done on biometrics and threads management in this application.

TABLE OF CONTENTS

DECLARATION ii

ACKNOWLEDGEMENT iii

DEDICATION iv

ABSTRACT v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

LIST OF ABBREVIATIONS ix

CHAPTER 1: INTRODUCTION 1

1.1 What is SMSEncrypt 1

1.2 Problem formulation 1

CHAPTER 2: LITERATURE REVIEW 3

CHAPTER 3: METHODOLOGY 8

3.1 SDLC- Software Prototyping 8

3.2 Rapid Prototyping 8

3.3 Basic Requirements Identification 8

3.3.1 Product Requirements 8

3.3.2 Development Requirements 8

3.1.3 Application Requirements Analysis 9

User registration form 9

User login form 9

Messaging interface 9

Encryption technique 9

3.4 Developing the initial Prototype 9

CHAPTER 4: DESIGN 11

4.1 Application Design 11

4.2 Application Design Analysis 12

Login and registration interface 12

Messaging interface 12

4.3 Encryption and Decryption Logical Design 13

4.4 Encryption and Decryption Logical Design Analysis 13

CHAPTER 5: IMPLEMENTATION AND TESTING 15
5.1 Application Implementation 15
Message inbox 17
Compose message 18
Message outbox 19
5. 2 Application testing 21
5.2.1 Testing definition 21
5.2 .2 Testing types 21
CHAPTER 6: DISCUSSION, CONCLUSION AND RECOMMENDATIONS 26
6.1 Discussion 26
6.1.1 How SMSEncrypt works 26
6.1.2 Why SMSEncrypt 27
6.2 Conclusion 27
6.3 Recommendations 28
REFERENCES 29
APPENDICES 31
Work plan 31
Budget 31
Encryption algorithm documentation 32

LIST OF FIGURES

1. Figure 4.1 : Application interface 11
2. Figure 4.2: encryption/decryption logical design 13
3. Figure 5.2: User sign in screen 16
4. Figure 5.4: Encrypted inbox screen 17
5. Figure 5.5: Decryption screen 18
6. Figure 5.6 Write message screen 18
7. Figure 5.8 : Decryption logic 20
8. Figure 5.9: Decryption Logic (changed phone number) 21
9. Figure 5.10 : local and instrumented unit testing 22
10. Figure 5.11 : Changed number testing 23
11. Figure 5.12: Contacts integration testing 24
12. **Table 1: Work plan** 31
13. **Table 2 : budget** 31

LIST OF ABBREVIATIONS

GSM ---------- Global System for Mobile communications

SMS ---------- Short Message Service

MLE ---------- Message Locked Encryption

AES ---------- Advanced Encryption Standard

3D- AES----------Three Dimensional Advanced Encryption Standard

DES ---------- Data Encryption Standard

SMSS ---------- Short Message Service Secure

RSA ---------- Rivest- Shamir-Adleman

MMS ---------- Multimedia Messaging Service

EFTPOS---------- Electronic Funds Transfer at Point of Sale

API ---------- Application Programming Interface

ECC ---------- Elliptic Curve Cryptography

AKO ---------- Army Online Knowledge

CACs ---------- Common Access Cards

PKI ---------- Public Key Infrastructure

SDLC ---------- Software development Life Cycle

SDK ---------- Software Development Kit

JRE ---------- Java Runtime Environment

JDK ---------- Java Development Kit

SQL ---------- Structured Query Language

smse:// ---------- Short Message Service Encryption

UI ---------- User Interface

CHAPTER 1: INTRODUCTION

Due to the risk involved in stored messages on phones being accessed by persons who are not supposed to access, there was a need to develop an application that will help users store their messages in an encrypted form to prevent this event from happening or in other words make it less likely from happening.

Therefore in this paper we will be looking at such an application known as SMSEncrypt. We will look at how it has been developed, beginning with conceptual design to actual implementation and testing in a real android environment.

1.1 What is SMSEncrypt

SMSEncrypt is an android based application that sends and store messages in an encrypted form. SMSEncrypt uses an 8set based encryption algorithm to encrypt and decrypt sent, received as well as stored messages. This algorithm is used in conjunction with base64 encoding /decoding and also sender and receiver phone numbers encoding/decoding to enhance security.

1.2 Problem formulation

Text messages while in storage they are usually in plain form. This poses a risk in the privacy of user messages since another party could access the phone and intentionally or accidentally find their way to the phone inbox where they can read the messages in the plain form.

This plain messages in the inbox tells a lot of private information to these unauthorized party which they could actually use as a weapon against the phone's owner. This could involve threatening them to do things that will benefit them else they could expose their secrets to people whom it may concern.

Again when transmitting messages through GSM networks we assume that service providers do the encryption of the messages while in transit. This is a wrong assumption, we need to be sure that our conversations are safe from eavesdroppers and that our messages reach as we sent them without being interfered with or changed while in transit.

There is therefore the need to store messages in an encrypted form to prevent such events from occurring or slow down such events. For these messages to be stored in an encrypted form then

they have to be sent and received in an encrypted form so that they will continue to remain in that state even after reading the information.

CHAPTER 2: LITERATURE REVIEW

According to the article Message-Locked Encryption and Secure Deduplication, Message-Locked Encryption (MLE), where the key under which encryption and decryption are performed is itself derived from the message. According to the authors MLE provides a way to achieve secure deduplication (space-efficient secure outsourced storage), a goal currently targeted by numerous cloud-storage providers. We provide definitions both for privacy and for a form of integrity that we call tag consistency. Based on this foundation, we make both practical and theoretical contributions (Bellare et al, 2013).

The key generation algorithm of an MLE scheme K maps a message M to a key K. The encryption algorithm E takes input the key K and a message M and produces a cipher text C. The decryption algorithm D allows recovery of M from C given the key K. The tagging algorithm T maps the cipher text C to a tag T used by the server to detect duplicates. (Tag correctness requires that tags corresponding to messages M1, M2 are likely to be the same iff M1, M2 are the same.) All algorithms may depend on a parameter P but the latter is public and common to all parties including the adversary, and thus is not a key. Any MLE scheme enables deduplication of cipher texts. CE is captured by our syntax as the MLE scheme that lets K = H (M), C = E(K, M) and tag T = H(C) (Bellare et al, 2013).

In the article SMS Encryption using 3D-AES Block Cipher on Android Message Application. SMS messages is one of the popular ways of communication. Sending any message is very easy. We can send and receive our confidential data at the time of transmission. During transmission of message through SMS is very difficult to protect it and also it is widely used in mobile banking. Security is the important thing in this but SMS does not provide a secure medium. SMS transmission through GSM network is also not secure, so there is need to secure SMS by providing encryption process. Encryption is important during transmission of SMS. There are so many types of encryption algorithms like AES, DES, and RC4 available. In these algorithms AES is most widely suitable algorithm. The authors develop an application which is based on Android platform which allows the user to encrypt the messages before it is transmitted over the network. The 3D-AES block cipher symmetric cryptography algorithm is used for secure transmission of message in this particular system. 3D AES block cipher symmetric algorithm is

used for providing a secured medium by providing encryption. If message size is more than 256 bits then it requires more time and size for sending that message (Ariffi et.al, 2013).

According to the article SMS -A secure SMS messaging protocol for the m-payment systems the author introduces GSM as the network with the greatest worldwide number of user that provide security. The short message service (SMS) is one of its superior and well-tried services with a global availability in the GSM networks. The main contribution of this paper is to introduce a new secure application layer protocol, called SSMS, to efficiently embedded the desired security attributes in the SMS messages to be used as a secure bearer in the m-payment systems. SSMS efficiently embeds the confidentiality, integrity, authentication, and non -repudiation in the SMS messages. It also provides an elliptic curve-based public key solution that uses public keys for the secret key establishment of a symmetric encryption and the attributes of public verification and forward secrecy. It efficiently makes the SMS messaging suitable for the m-payment applications where the security is the great concern (Toorani, 2008)

The article SMS encryption for mobile communication, the authors introduces an application which deals with SMS encryption for mobile communication. The SMS transmission in GSM network is not secure, therefore it is desirable to secure SMS by additional encryption. In SMS, there are compared differences in the use of symmetric and asymmetric cryptography for SMS transfer securing. In the next part, there is the description of design and implementation of the application for mobile phones, which encrypts and signs SMS using an asymmetric RSA cipher. At the end, there are described attacks on secured SMS and future extension of the application (Lisonek, 2008).

According to the article Trusted SMS communication on mobile devices, the author has introduced the higher growth of the Short Message Service (SMS) use has transformed this service in a widespread tool for social and commerce messaging. However, security concerns have been raised as applications become more critical and complex. Thus, this paper introduces an SMS security framework, which allows programmers and users to exchange confidential, non -reputable and digitally signed text messages. This framework can fit in many development scenarios, such as commercial transactions or bureaucratic delegations. In addition, the proposed framework is highly flexible and e efficient, since programmers can choose among several

encryption algorithms according to the computational power and battery usage of each mobile device (Albuja & Carrera, 2009)

The article Building secure user-to user messaging in mobile telecommunication networks, the author explained that Short Message Service (SMS) and Multimedia Message Service (MMS) are popularly used and will be more popular in the future. However, the security of SMS (Short message service) and MMS (Multimedia message service) messages is still a problem. There is no end -to -end security (including confidentiality, integrity, authentication and non -repudiation) in these services. This hinders service providers to provide some services that require communication of high -level security. There have been some solutions proposed for this issue in literature, but these are not suitable for user -to -user communication. In the paper, the author reviews existing solutions and analyze their weaknesses (Zhao, Aggarwal, & Liu, 2008)

Harb et al (2009) in the article Secure SMSPay: secure SMS mobile payment model introduce a secure mobile payment model suitable for transactions that consist of cost, simplicity, security, and performance of transaction, which contain minimum number of cryptography key usages, and less encryption/decryption operations as compared to other models. This model can use for symmetric and asymmetric cryptography. And there is no need of trusted 3rd parties or even PKI complexity. Now a days it is based on SMS as a transport channel which provides the capability to send transactions to payer not to payee; as usually done in most current payment transaction models. The payer receives a secured SMS message waiting his/her confirmation that is yes or no. Each things in the payment system payer/payee trusts only his/her bank respectively, so the transaction will always go through trusted nodes. The payer/payee can also use any bank payment instrument like credit card, debit card, or even current account without revealing confidential data during the payment. This model can be used for any payment application e.g. e -check, money transfer, e-commerce, and even normal EFTPOS transactions with leverage infrastructure supporting the above mentioned payment applications.

Soram (2008) in the article Mobile SMS banking security using elliptic curve Cryptosystem, mobile devices have many differences in their capabilities, computational powers and security requirements. Mobile devices can be used as the enabling technology for accessing Internet based services, as well as for personal communication needs in networking environments. Mobile services are spread throughout the wireless network and are one of the crucial components

needed for various applications and services. However, the security of mobile communication has topped the list of concerns for mobile phone users. Confidentiality, Integrity, Authentication and Non-repudiation are required security services for mobile communication. Currently available network security mechanisms are inadequate; hence there is a greater demand to provide a more flexible, reconfigurable, and scalable security mechanism. This project provides e effective security solution using Public key cryptography. The implementation of this project is divided into two parts first, design of API for ECC (Elliptic Curve Cryptography) which generates shared secret key required for secure communication and secondly, a web service is created which distributes this key to validate mobile user.

In the article, A proposal for enhancing the security system of short message services in GSM, introduced that Short message service will play a very important role in the future business areas which is popularly known as m-commerce, mobile banking etc. In future commerce, SMS could make a mobile device in a business tool as it has the availability and the effectiveness. SMS is not free from the eavesdropping, but security is the main thing for any business company such as banks who will provide these mobile banking. Now a days there is no such scheme which can give the complete SMS security. The author therefore proposed a security scheme for improving the SMS security. At first plaintext of SMS would be made as cipher text with the help of GSM encryption technology, then this cipher text would be digitally signed .It can be signed with the help of public key signature. These have to be made compatible to existing infrastructure of GSM security. The proposed system will give total authenticity, data integrity, confidentiality, authorization and non -repudiation which are the most essential and common issues in m -commerce or mobile banking and in securing any messaging (Hossain et al., 2008).

In the article, Secure asynchronous communication for mobile devices the author states that Short Message Service is now widely use as business tool, security of SMS has become a major thing for any business organizations and customers. There is therefore the strong need for an end to end SMS Encryption to provide a secure medium for communication. This paper evaluates RSA, ELGamal and Elliptic curve encryption techniques using random SMS messages of various sizes for measure their encryption and decryption time. The results are presented to show the effectiveness of each algorithm and to choose the most suitable and good algorithm for SMS encryption (Kuaté et al, 2009).

The U.S army uses a secure email exchange known as Army Knowledge Online (AKO). These system allows secure messages and documents to be passed between the military contractors and federal officials using credentialed common access cards (CACs) (smallbusiness website, 2016).

Ozcan, A. T. et al in the article Babel Crypt: The Universal Encryption Layer for Mobile Messaging Applications introduces a system that addresses the problem of retrofitting arbitrary mobile chat applications with end-to-end encryption. This system protects messages against access by the messaging service providers (Ozcan et al, 2015).

In their article User-Friendly Chat (6.857 final project) designed and implemented a small end-to-end encrypted chat app that builds on a PKI, Keytree, for security. The app's design lets users securely chat with other users. Our app supports one-on-one conversations between pairs of users, but not group chat. In the app, messages are end-to-end encrypted and only readable on the devices the users use to access the application (Bangert)

Applications that have already been developed for messaging encryption have emphasized on end to end encryption with the purpose of enhancing the main security goals that is confidentiality, integrity, availability and non-repudiation. Others talk about encryption is service provider storage to ensure that service providers cannot get the message meaning without the decryption. Most of these algorithms used in the cryptosystems involve use of the secret keys in the encryption and decryption.

There is therefore the need to ensure these security goals are maintained in the respective user phone storages by encrypting messages in storage use a keyless algorithm.

CHAPTER 3: METHODOLOGY

3.1 SDLC- Software Prototyping

Software prototyping is suitable for developing SMSEncrypt since it will involve building software application prototypes which will display limited functionality of the product while still under development.

Prototyping will also enable to understand user requirements at an early stage of development. It will help get valuable feedback from the user and help software design and development capture exactly what is expected from the product under development.

3.2 Rapid Prototyping

This type of prototyping is preferable in developing SMSEncrypt since it uses very little efforts with minimum requirement analysis to build a prototype. Once the actual requirements are understood, the prototype will discarded and the actual application will be developed with a much clear understanding of user requirements.

3.3 Basic Requirements Identification

3.3.1 Product Requirements

- User Registration form
- User login form
- Messaging interface
- Encryption technique (algorithm approach)

3.3.2 Development Requirements

Tools and Techniques

- Java programming language (developing encryption technique)
- Computer (windows operating system)
- Android studio (application programming interface)
- Emulator (virtual device for testing)
- Two android phone (real device for testing)

- Net beans (java platform)
- Java runtime environment (JRE)
- Java development kit (JDK)

3.1.3 Application Requirements Analysis

The requirements mentioned above are explained in details:

User registration form

This is a form that allow the user to sign up in order to be able to use this application these credentials will be used login in to the system.

User login form

This form will be used to authenticate the user accessing the application. This will work by comparing the credentials provided to the ones stored in the databases. If the credentials match then the user is allowed access, otherwise access denial.

Messaging interface

After the user access has been authenticated then they are redirected to this interface where they can send and open messages that are in encrypted form. This interface will also incorporate telephony services for sending messages and receiving messages.

Encryption technique

These layer will be executed in the background where the user does not actually know what is happening behind the scenes.

3.4 Developing the initial Prototype

This stage the very basic requirements will be showcased and user interfaces will be provided. These features may not exactly work in the same manner internally in the actual software

developed and the workarounds are used to give the same look and feel to the user in the prototype developed.

3.5 Review of the Prototype

The prototype developed will then be presented to the user and the other important stakeholders in this project. The collected feedback will be organized in a manner and used for further enhancements in the product under development.

3.5 Revise and enhance the Prototype

The feedback and the review comments will be discussed during this stage and some negotiations happen with the user based on factors like, time and budget constraints and technical feasibility of actual implementation. The changes accepted will again be incorporated in the new Prototype developed and the cycle repeats until user expectations are met.

Why software prototyping

- Missing functionality is identified easily
- Quicker user feedback is available leading to better solutions.
- Since a working model of the system is displayed, the users get a better understanding of the system being developed.
- Reduces time and cost as the defects can be detected much earlier.
- Increased user involvement in the product even before implementation

CHAPTER 4: DESIGN

4.1 Application Design

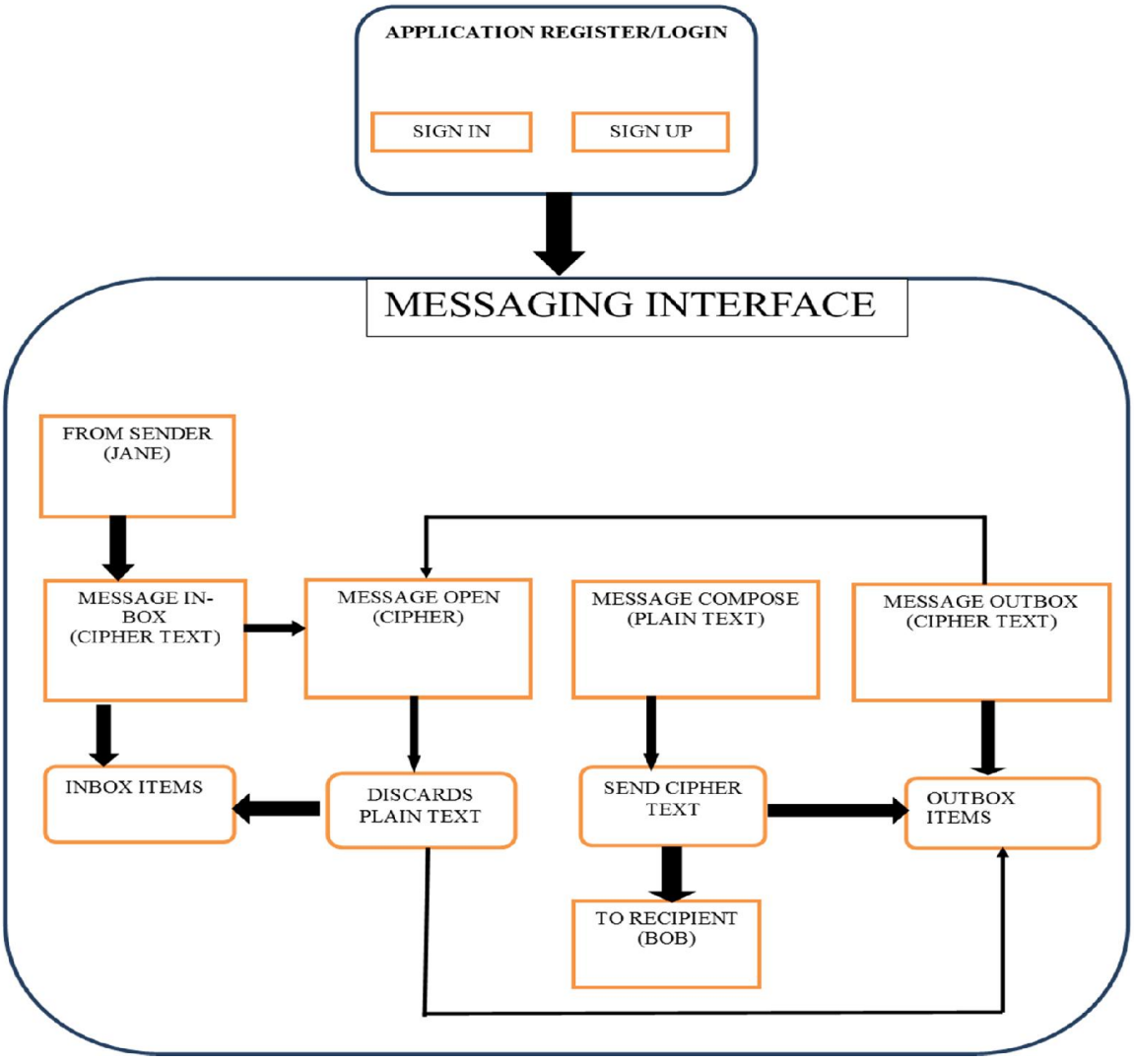

Figure 4.1 : Application interface

4.2 Application Design Analysis

Login and registration interface

Sign in this is a button view which calls the login function when the user clicks on it. It matches the user provided credentials with the ones stored in the android SQLite database.

Sign up on clicking this button the details provided by the user for registration are stored in the android SQLite database. User provides details such as **phone number** and **password.** Upon providing this details a user account is successfully created**.**

Messaging interface

Message inbox stores the received items in a cipher form. Items in the inbox are stored in list. If the user wishes to read the text they click on it calls the open message function which displays the plain text in a dialog. On clicking outside the dialog the plain text is discarded.

Open Message function triggers the decryption of the stored cipher texts both in the **inbox items** and **outbox items.**

Message outbox stores the sent items in a cipher form which are displayed in the inbox as a list. Again if the user wishes to read the text they click on it calls the open message function which displays the plain text in a dialog

Compose message composes the message in plain text form for better editing of the message content before sending it in cipher form. On sending the message a copy of the cipher text is stored in the inbox while discarding the plain text.

Send button triggers the encryption of the composed message before sending it to the recipient.

4.3 Encryption and Decryption Logical Design

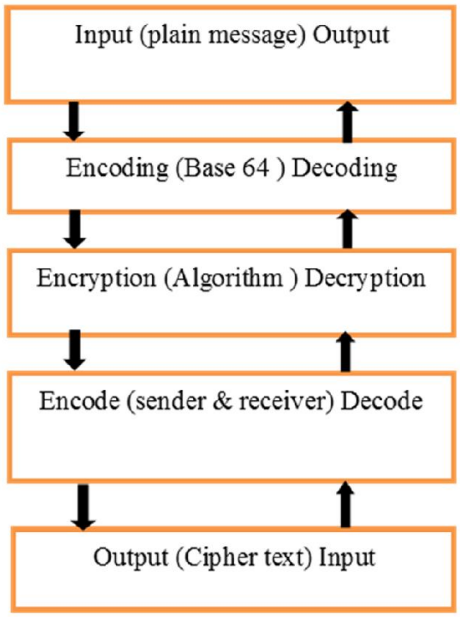

Figure 4.2: encryption/decryption logical design

4.4 Encryption and Decryption Logical Design Analysis

Input plain message is the user composed message. Which is in plain text.

Encode base 64 the input plain message is encoded by base 64 to ensure that the characters that are not specified in the character scope of the algorithm are transformed into a form understood by the base 64 characters.

Set algorithm encryption, the encoded output is then encrypted by the 8 set encryption algorithm.

Encode by number, the cipher generated from set encryption is the coded by the recipient and senders phone numbers. These is done to ensure only the two intended sender and receiver can read this message using the application in their phone. The cipher message cannot be decrypted by another SMSEncrypt application installed on another android phone.

The **output cipher** is the final cipher that is sent to the recipient.

The **input cipher** will be the received cipher by the recipient which will be used as the input for decryption process.

The reverse of this procedure is the decryption process. Which is shown by the upward arrows whereby the cipher text will be the input. The cipher will be first decoded by the sender and receiver phone number, the result is then decrypted by the 8set algorithm for decryption and finally decoded by base 64 and the result is a plain text.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Application Implementation

The development of this application is completed in android studio which has all the requirements needed in developing android applications such as android SDK, development platform as well as compiler, debugger and a virtual device which simulates a real android device for testing and running the application.

The encryption algorithm is implemented and tested on a java platform. Where by net beans for java developers has been used to demonstrate the operation of the set algorithm.

In this section captured screen shot of the emulator will be shown to demonstrate how the implementation and testing was done.

Testing of the application is done in a real android device since it involves sending of the cipher messages through the GSM networks.

User sign up screen

Below is the user registration screen where the user is supposed to feed the phone number and password then hit the sign up button and account is successfully created. It is the first screen for first time user.

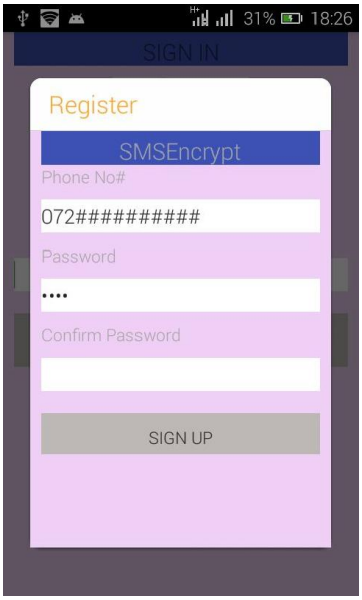

Figure 5.1: user registration screen

Upon hitting the sign up button the user personal details are stored in the android SQLite database. The password will be used for login. The phone number will be used for encrypting and decrypting messages (used in conjunction with recipient phone number).

Login screen

Below is a screen shot of the home screen where the user is required to feed the pass code they used in registration . upon matching credentials stored in the database access is granted otherwise denied . This is always the first screen for a regular user.

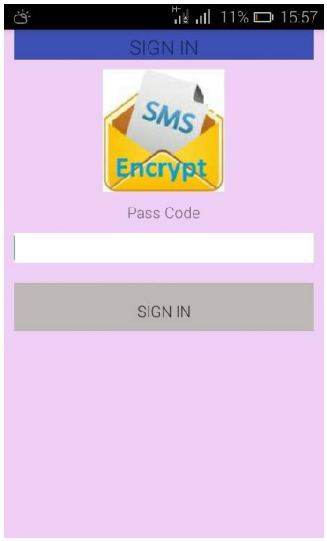

Figure 5.2: User sign in screen

Messaging interface

Below is the messaging interface where the user is redirected on login into the application.

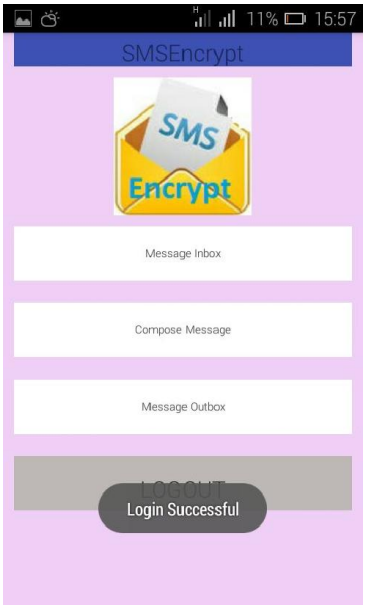

Figure 5.3: messaging interface

From here the user can select whether to read received encrypted messages using the message inbox, compose a message to send and read sent encrypted messages.

The log out button on click destroys the session and redirects to the login screen.

Message inbox

In this view all the encrypted received messages will be displayed. The messages encrypted and sent using this application contain a protocol "**smse://**" attached at the beginning of each message. This protocol enables the application to differentiate between its messages from other messages.

Figure 5.4: Encrypted inbox screen

Upon clicking on an encrypted message the decryption function is called and the screen window below pops out to display the decrypted message.

Figure 5.5: Decryption screen

Compose message

In this view the user will select from the contact list the address they wish to send a message. Upon hitting the send button the encryption function is called and the text is sent in cipher form and the "smse://" protocol is attached to the string.

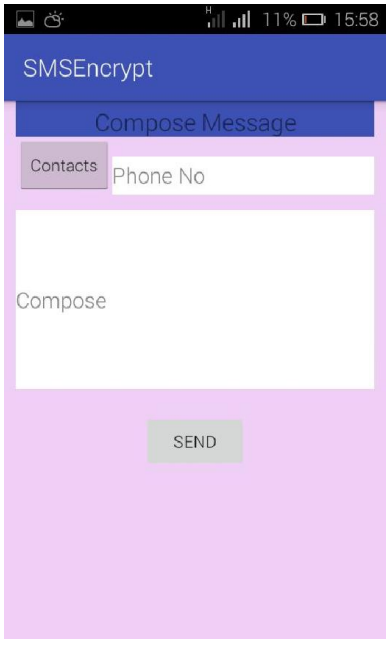

Figure 5.6 Write message screen

Message outbox

In this view all the encrypted sent messages will be displayed. The messages encrypted and sent using this application contain a protocol "**smse://**" attached at the beginning of each message. This protocol enables the application to differentiate between its messages from other messages

If a user trys to decrypt the message using a phone number that was not used to send the particular message , that is, if the receiver or sender phone number changes then the decryption will not be successful.

Stepwise encryption logic

The screen below shows a stepwise encryption logic whereby on inputting a plain message you will first be required to input the sender and receiver phone numbers. The plain message is converted via base 64 encoding.

On conversion to base 64 then the result is encrypted using the 8set encryption algorithm.

The cipher resulted is again encoded by the receiver and sender phone number. From this point the cipher is ready for transmission.

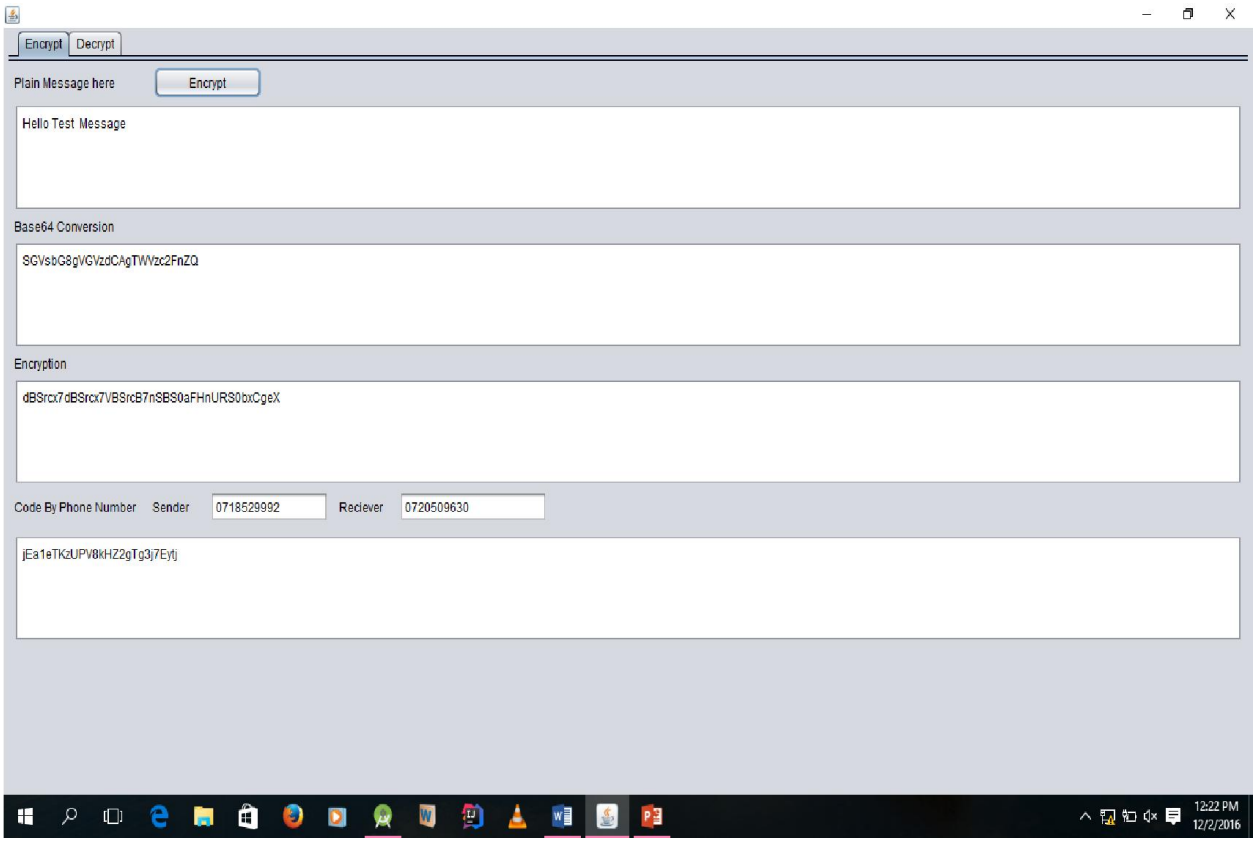

Figure 5.7 : Stepwise encryption logic

Stepwise decryption logic

The screen below shows a stepwise decryption logic whereby on inputting a cipher message you will first be required to input the sender and receiver phone numbers. The cipher message is first decoded by the original sender and receiver phone number.

On decoding then the result is decrypted using the 8set encryption algorithm.

The result again decoded by the base64 decoder. From this point the plain message is ready.

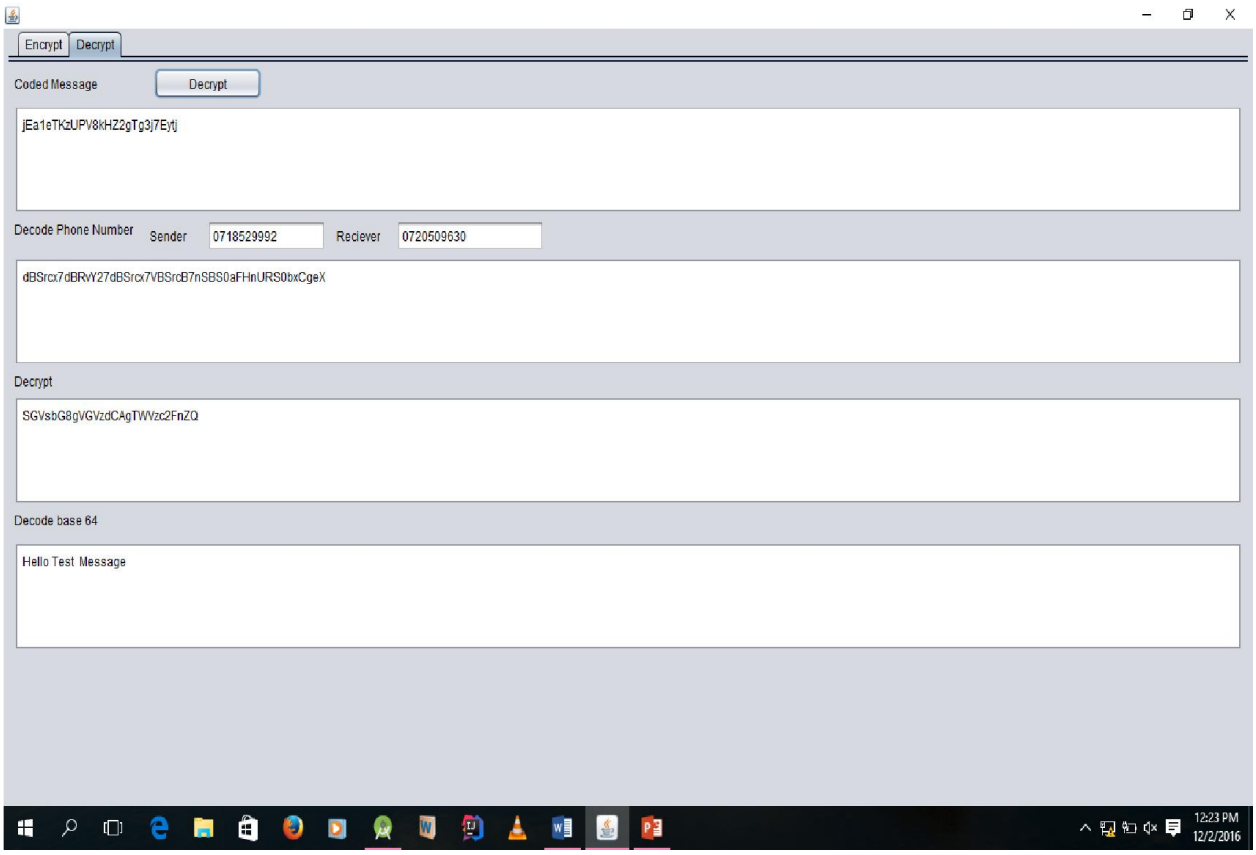

Figure 5.8 : Decryption logic

Stepwise decryption logic (changed receiver phone number)

At this point when the receiver or sender phone number changes from the original the decrypted message will lose its meaning since the receiver number that was used to decode it is different from the one being used.

The result of the meaningless plain message is as shown in the screen below.

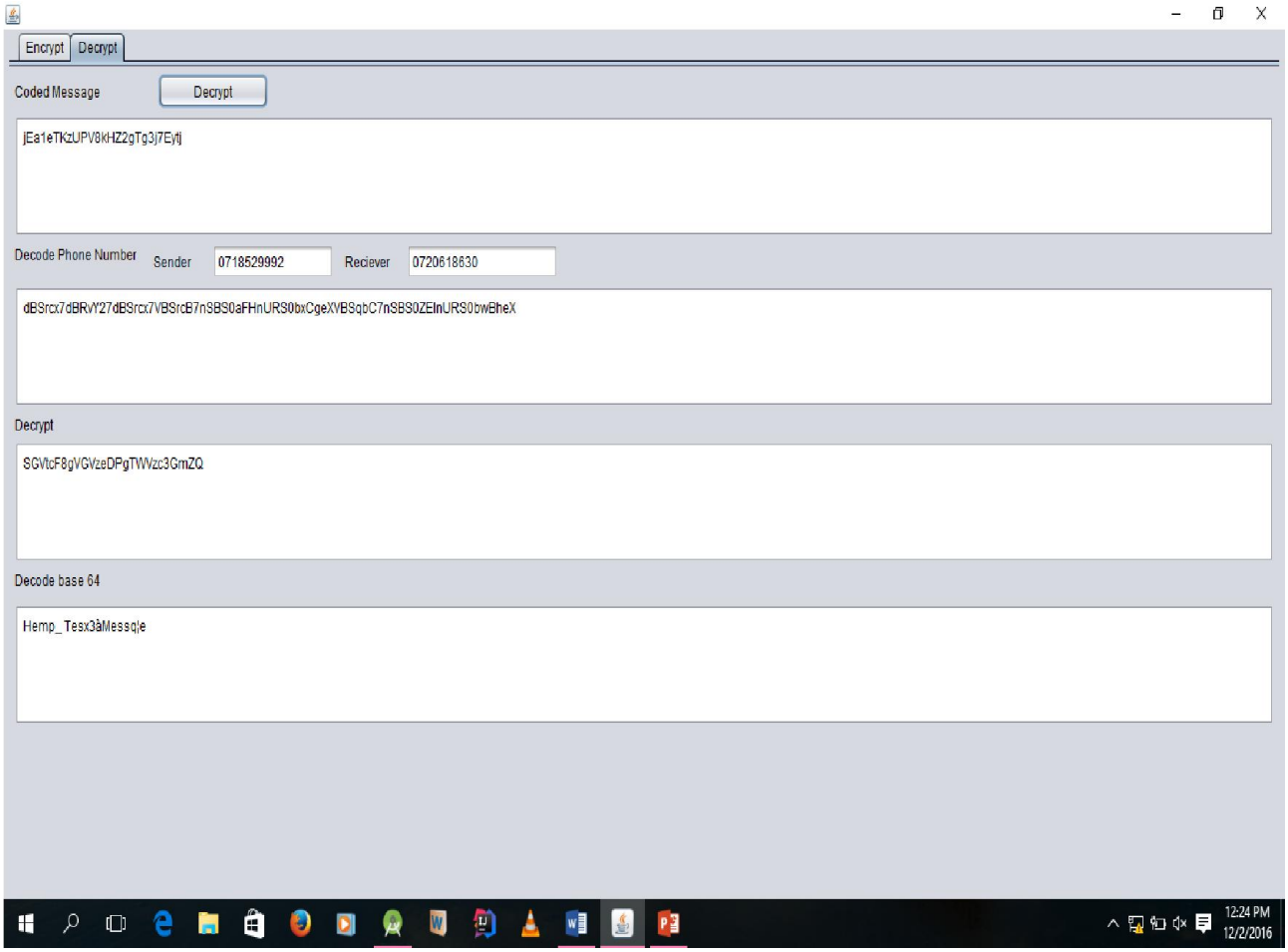

Figure 5.9: Decryption Logic (changed phone number)

5. 2 Application testing

Android tests are based on Junit whereby they can be run as local tests on the JVM or implemented tests on Android Device.

5.2.1 Testing definition

Testing the application is an integral part of the app development process. Testing is defined as the process in which defects are identified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction. Testing allows you to verify the correctness, functional behavior, and usability of your app before it is released publicly (Android developer website, 2016).

5.2 .2 Testing types

Android has two types of testing which include unit test and integral tests.

Unit testing

Unit testing is can be performed by either local unit test, instrumented unit test or by both of them

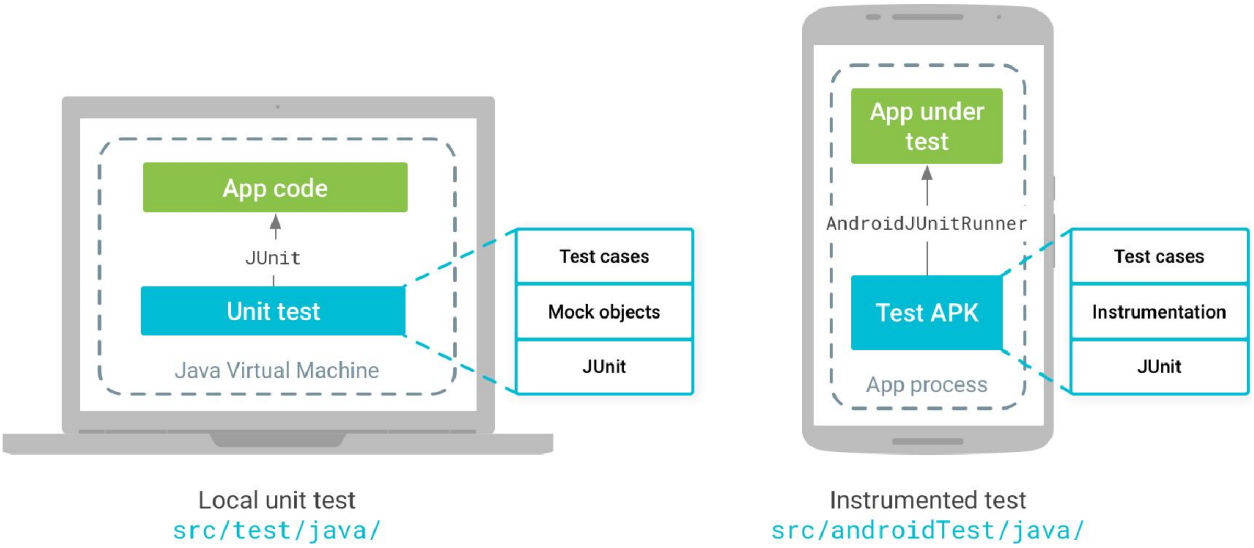

Figure 5.10 : local and instrumented unit testing

Local unit tests

Located at *module-name*/src/test/java/.

These tests run on the local JVM and do not have access to functional Android framework APIs (Android developer website, 2016).

Instrumented tests

Located at *module-name*/src/androidTest/java/.

These are all tests that must run on an Android hardware device or an Android emulator.

Instrumented tests are built into an APK that runs on the device alongside your app under test. The system runs your test APK and your app under tests in the same process, so your tests can invoke methods and modify fields in the app, and automate user interaction with your app (Android developer website, 2016).

In testing SMSEncrypt the instrumented unit test methodology was used where by an APK was built and installed on a real android device.

Integral testing

Integral testing is subdivided into two types which involve components within your application only and cross application components. SMSEncrypt android application was tested using both the cross application components testing and the components within the application only.

Components within application only

This type of test verifies that the target app behaves as expected when a user performs a specific action or enters a specific input in its activities. For example, it allows you to check that the target app returns the correct UI output in response to user interactions in the app's activities. UI testing frameworks like Espresso allow you to programmatically simulate user actions and test complex intra-app user interaction (Android developer website, 2016).

The components tested within the application were as follows:

Encryption and decryption function

The encryption with both the sender and receiver phone numbers functionality was test by opening the encrypted message with a different phone number other than the one used in the encryption.

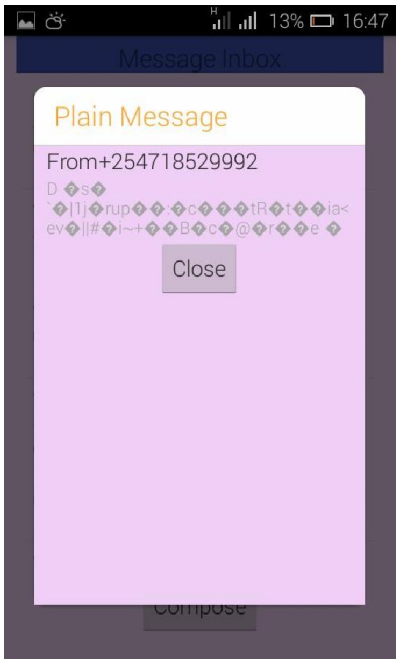

Figure 5.11 : Changed number testing

Send and receive and store functions

Testing was also done with two different android devices to ensure that the application was able to send and receive encrypted messages and ensure they stayed in encrypted form in the default android inbox.

Open message function

Opening of the encrypted messages in the application was done to ensure that the application was actually in a position to decipher the ciphers correctly.

Cross-app Components

This type of test verifies the correct behavior of interactions between different user apps or between user apps and system apps. For example, you might want to test that your app behaves correctly when the user performs an action in the Android Settings menu. UI testing frameworks that support cross-app interactions, such as UI Automator, allow you to create tests for such scenarios (Android developer website, 2016).

Components tested in the cross application components for SMSEncrypt were as follows:

Default Android messaging application.

This was to ensure that the message sent or received using the application were stored in the normal android in the encrypted format.

Android contacts

This test was done to ensure that a user can automatically select contacts from the android contacts database while composing the system.

Figure 5.12: Contacts integration testing

Input keyboard

The input keyboard test was done to ensure that whenever a user wants to compose a message or even in the registration they would easily do that with the default android input keyboard.

CHAPTER 6: DISCUSSION, CONCLUSION AND RECOMMENDATIONS

6.1 Discussion

The essence of this application is to ensure that messages in storage provides nothing meaningful to an unauthorized person who accesses private messages, in other words the unauthorized person cannot reconstruct a previous conversation since the messages will be in an encrypted form.

SMSEncrypt allows the sender of the message to first compose a message in the plain text form, thus, making it easy for the sender to easily correct typo's before sending the message unlike in other applications where the encryption layer is on top of the underlying keyboard and encryption happens automatically when the key values are pressed thus does not give the sender a chance to proofread their message before sending them.

SMSEncrypt also provide the user with a login platform for more security purposes. There is a registration form where the user first registers before actually getting started with using this application.

6.1.1 How SMSEncrypt works

The whole message is composed in plain text, once the send button is clicked the message is converted into a cipher text and the plain text message is discarded. A copy of the cipher text is sent to the target recipient while another copy is preserved in outbox.

Similarly on receiving the encrypted message the message is decrypted by opening it through the decryption function and on reading the plain text message is discarded while the cipher message is stored in the inbox.

The plain message is first converted to a base 64 encoding, then using the 8set algorithm the conversion is transformed to a cipher. The cipher is once again coded by both the sender and receivers phone numbers to ensure that only those two numbers can decipher to the original message.

The received cipher text on the other hand is first decoded by the original sender and receiver phone number. Again the result is deciphered by the 8set decryption algorithm, then decoded by base 64 decoder and this results to the plain message

SMSEncrypt attaches a protocol "smse://" to the sent messages to differentiate messages sent with this application. Only messages containing this protocol will be read and decrypted by this application.

6.1.2 Why SMSEncrypt

SMSEncrypt gives the sender of a message the opportunity to proofread the composed text message to avoid sending typos, unlike other message encryption applications that use encryption function on the underlying keyboard and transform the pressed letter into a cipher immediately thus depriving the sender a chance to proofread the composed message while it still is in plain text.

Given the fact that users use SMS to communicate on daily basis. It would be hectic to open each and every message using different keys that different senders of messages will use to encrypt their messages. SMSEncrypt simplifies the hustle by decrypting the SMS automatically as long as the SMS was sent using the application.

SMSEncrypt protects the privacy of user's conversation since it stores both the sent and received messages in a cipher form. If anyone accidentally or intentionally access a user's android phone then they would not be able to reconstruct the previous conversation stored on that phone. SMSEncrypt also gives the users an added advantage since the messages will be encrypted in transit.

SMSEncrypt provides security since anyone trying to access this application interface will be prompted for login credentials for permission to access the messages.

Again using the original sender and receiver phone numbers that were used to encrypt ensures that another third party with the same application cannot decrypt back to the original plain message.

Messages in the normal android inbox will be in encrypted. Until one uses these application to read the messages then it will be of no meaning.

6.2 Conclusion

SMSEncrypt has addressed the problem mention earlier in this paper successfully. It has successfully met the security goals of confidentiality and integrity. Confidentiality is that aspect that ensure private information is kept private to the authorized persons only while integrity ensures that information does not change its originality.

The aspect of encrypting messages in the inbox ensure confidentiality while the aspect of another third party using a different phone number other than the original sender and receiver phone number cannot successfully interfere with this messages.

6.3 Recommendations

Any further development of these application I recommend that they improve on the security of this application in the following areas:

Session management

When the user leaves the application unattended for some few minutes then the application should be able to automatically log them out.

Biometrics

The application should ensure a layered security in the login by integration of some biometrics techniques for user authentication in the login. That Is it could be finger prints or voice recognition biometrics (what the user is) in combination with password (what the user knows)

Contacts

The application should be able to read the contacts saved in the user's android phones in storage of these encrypted messages. Such that instead of displaying the phone number it should display the contact name

Chats

This application should combine the inbox and outbox and store these messages inform of threads /conversation specific to each sender

Keyboard

Any further work on this application should also involve application with keyboard to ensure the plain message is automatically correct where necessary before sending the message.

REFERENCES

Albuja, J. P., & Carrera, E. V. (2009, May). Trusted SMS communication on mobile devices. In *Proceedings of the 11th Brazilian Workshop on Real-Time and Embedded Systems, Recife-Brazil* (pp. 165-170).

Ariffi, S., Mahmod, R., Rahmat, R., & Idris, N. A. (2013, December). SMS Encryption Using 3D-AES Block Cipher on Android Message Application. In *Advanced Computer Science Applications and Technologies (ACSAT), 2013 International Conference on* (pp. 310-314). IEEE.

Bangert, J., van den Hooff, J., & Tabidze, T. User-Friendly Chat (6.857 final project).

Bellare, M., Keelveedhi, S., & Ristenpart, T. (2013, May). Message-locked encryption and secure deduplication. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 296-312). Springer Berlin Heidelberg.

Harb, H., Farahat, H., & Ezz, M. (2008, August). SecureSMSPay: secure SMS mobile payment model. In *2008 2nd International Conference on Anti-counterfeiting, Security and Identification* (pp. 11-17). IEEE.

Hossain, M. A., Jahan, S., Hussain, M. M., Amin, M. R., & Newaz, S. S. (2008, August). A proposal for enhancing the security system of short message service in GSM. In *2008 2nd International Conference on Anti-counterfeiting, Security and Identification* (pp. 235-240). IEEE.

Http://smallbusiness.chron.com/encrypt-email-using -ako-49930.html , October 2016.

Https://developer.android.com/training/testing/start/index.html, December,2016.

Kuaté, P. H., Lo, J. L. C., & Bishop, J. (2009, April). Secure asynchronous communication for mobile devices. In *Proceedings of the Warm Up Workshop for ACM/IEEE ICSE 2010* (pp. 5-8). ACM.

Lisonek, D., & Drahanský, M. (2008, December). Sms encryption for mobile communication. In *Security Technology, 2008. SECTECH'08. International Conference on* (pp. 198-201). IEEE.

Ozcan, A. T., Gemicioglu, C., Onarlioglu, K., Weissbacher, M., Mulliner, C., Robertson, W., & Kirda, E. (2015, January). BabelCrypt: The Universal Encryption Layer for Mobile Messaging Applications. In *International Conference on Financial Cryptography and Data Security* (pp. 355-369). Springer Berlin Heidelberg.

Soram, R. (2009). Mobile sms banking security using elliptic curve cryptosystem. *International Journal of Computer Science and Network Security*, *9*(6), 30-38.

Toorani, M., & Beheshti, A. (2008, July). SSMS-A secure SMS messaging protocol for the m-payment systems. In *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on* (pp. 700-705). IEEE.

Varsha S. Bari, Nileema R. Ghuge, Chaitali C. Wagh, Sayali R. Sonawane , Mr.M.B. Gawali. Sms Encryption on Android Message Application. 2016

Zhao, S., Aggarwal, A., & Liu, S. (2008, April). Building secure user-to-user messaging in mobile telecommunication networks. In *Wireless Telecommunications Symposium, 2008. WTS 2008* (pp. 151-157). IEEE.

APPENDICES

Work plan

Table 1: Work plan

Work	Description	Time frame
Proposal presentation	Presenting the problem and solution in front of stakeholders (panel)	17th October,2016
Gathering information	Collecting all the necessary information and user views	18th-22nd October,2016
Hiring human resource	Hiring the necessary personnel for the labor	23rd – 29th October, 2016
Gathering requirements	Gathering all the necessary requirements for developing the application	30th -6th November, 2016
Initial user interface prototype	Developing an initial prototype	7th November, 2016
Prototype review	Review of prototype by stakeholders and users for feedback	8th- 21st November, 2016
Implementation and Testing	Developing the application and testing it in an android environment	22nd-3rd December, 2016
Product presentation	Present the final product to the stakeholders and users.(panel)	5th December, 2016

Budget

Table 2 : budget

Requirements	Cost
Android studio	Free
Android phone	Ksh.10000
Computer	Ksh.40000
Windows OS	Ksh.1000
Printing	Ksh.300
Human resource	Ksh.5000
Internet bandwidth	Ksh.2000
JRE/JDK	Free
Net beans	Free
Total	Ksh.58300

Encryption algorithm documentation

Below is a base 64 algorithm which divides the 64 characters into 8 sets where each set follows similar sets of step to achieve encryption and decryption of characters within the set. Since the encryption and decryption of characters requires no private or public key then it is advisable that it shouldn't be used alone without other encryption techniques put in place to enhance its strength.

I developed this algorithm with the intent to use it for encryption and decryption in my Android SMS application which store and sends messages in cipher form. I prefer this algorithm because it doesn't require a key, therefore it would be flexible in SMS environment where the user would feel bothered to input a key for each and every message from different senders now and every then

1	A	2	B	3	C	4	D	5	E	6	F	7	G	8	H
9	I	10	J	11	K	12	L	13	M	14	N	15	O	16	P
17	Q	18	R	19	S	20	T	21	U	22	V	23	W	24	X
25	Y	26	Z	27	A	28	B	29	c	30	D	31	e	32	f
33	G	34	H	35	I	36	J	37	k	38	L	39	m	40	n
41	O	42	P	43	Q	44	R	45	s	46	T	47	u	48	v
49	w	50	X	51	Y	52	Z	53	0	54	1	55	2	56	3
57	4	58	5	59	6	60	7	61	8	62	9	63	+	64	/

Table 3: Base 64 characters

SET 1

1 (A), 2 (B), 3 (C), 4 (D), 5 (E), 6 (F), 7 (G), 8 (H),

SUM SET 1 = 36

DIFF SET 1 = 16

QUOTIENT OF SUM SET & DIFF SET 1

36/16 =2.25

Product of quotient and number of pairs in the set.

2.25 * 4 = 9

Set 1 working value will be **9**

SET 2

k9 (I), 10 (J), 11 (K), 12 (L), 13 (M), 14 (N), 15 (O), 16 (P),

SUM SET 2 = 100

DIFF SET 2= 16

QUOTIENT OF SUM SET & DIFF SET 2

100/16=6.25

Product of quotient and number of pairs in the set.

6.25* 4 =25

Set 2 working value will be **25**

SET 3

17 (Q), 18 (R), 19 (S), 20 (T), 21 (U), 22 (V), 23 (W), 24 (X),

SUM SET 3 = 164

DIFF SET 3 = 16

QUOTIENT OF SUM SET & DIFF SET 3

164/16 =10.25

Product of quotient and number of pairs in the set.

10.25* 4 = 41

Set 3 working value will be **41**

SET 4

25 (Y), 26 (Z), 27 (a), 28 (b), 29 (c), 30 (d), 31(e), 32 (f),

SUM SET 4 = 228

SUM DIFF SET 4 = 16

QUOTIENT OF SUM SET & DIFF SET 4

228/16 =14.25

Product of quotient and number of pairs in the set.

14.25 * 4 = 9

Set 4 working value will be **57**

SET 5

33 (g), 34 (h), 35 (i), 36 (j), 37 (k), 38 (l), 39 (m), 40 (n),

SUM SET 5 = 292

DIFF SET 5 = 16

QUOTIENT OF SUM SET & DIFF SET 5

292/16 =18.25

Product of quotient and number of pairs in the set.

18.25 * 34= 73

Set 5 working value will be **73**

SET 6

41 (o), 42 (p), 43 (q), 44 (r), 45 (s), 46 (t), 47 (u), 48 (v),

SUM SET 6 = 356

DIFF SET 6 = 16

QUOTIENT OF SUM SET & DIFF SET 6

356/16 =22.25

Product of quotient and number of pairs in the set.

22.25 * 4 = 89

Set 6 working value will be 8**9**

SET 7

49 (w), 50 (x), 51 (y), 52 (z), 53 (0), 54 (1), 55 (2), 56 (3),

SUM SET 7 = 420

DIFF SET 7 = 16

QUOTIENT OF SUM SET & DIFF SET 7

420/16 =26.25

Product of quotient and number of pairs in the set.

26.25 * 4 = 105

Set 7 working value will be **105**

SET 8

57 (4), 58 (5), 59 (6), 60 (7), 61 (8), 62 (9), 63 (+), 64 (/),

SUM SET 8 = 484

DIFF SET 8 = 16

QUOTIENT OF SUM SET & DIFF SET 8

484/16 =30.25

Product of (quotient and number of pairs in the set).

30.25 * 4 = 121

Set 8 working value will be **121**

Obtaining the SUM SET and the DIFF SET

Sum set has been obtained by adding all the letters positions in that particular set.

For example:

SET 7

49 (w), 50 (x), 51 (y), 52 (z), 53 (0), 54 (1), 55 (2), 56 (3),

49+50+51+52+53+54+55+56=420

SUM SET 7= 420;

The **diff set** is the difference between sum of the 2 pairs from the upper position and sum of the other 2 pairs from the lower position of the set for example:

SET 5

33 (g), 34 (h), 35 (i), 36 (j), 37 (k), 38 (l), 39 (m), 40 (n),

g, h : i ,j are two pairs in upper set

33+34+35+36 = 138

k, l : m, n are the other two pairs in the lower set

37+38+39+40 = 154

DIFF SET = 154-138=16

These difference is constant in all the sets

Encryption and decryption

The working value will be used for the encryption and decryption by simply getting the difference between the working value and the plain/cipher letter provided.

For example in set 2:

9 (I), 10 (J), 11 (K), 12 (L), 13 (M), 14 (N), 15 (O), 16 (P),

Set 1 working value is 25

25 – J (10) =15

Whereby 15 is O

The cipher of J is O and inversion is the decryption.

plain	Cipher
J	O
O	J

Table 4 : Cipher result

Motivation of developing this algorithm

This algorithm is not the best neither does it beat any other encryption that has already been developed. It was developed for the purpose of learning and presenting the knowledge that has been gained throughout.

It can easily be broken but it can serve the purpose where the potential user does not want to use any encryption and decryption key in accomplishment of simple tasks like storing of messages.

Limitations of this algorithm

- ✓ Does not employ encryption and decryption key technique.

Assumptions made

- ✓ The difference should only be obtained from upper and lower positions in a set
- ✓ The use of key will be inconveniencing since no key exchange mechanism is specified.

Recommendations

- ✓ The use of this algorithm should only apply where the user have concerns in regards to exchange of the key issues or using the key would be quite bothersome since the encryption and decryption happens frequently on daily basis and involves lots of different entities sending messages.
- ✓ The user of this algorithm will use other techniques like base 64 encoding to take care of the characters not captured in the scope