Micro Utilities Secure Message Utilities on low-cost mobiles

Jayalakshmi Srinivasan

Assistant Professor, Department of B. Sc IT, V.E.S. College of Arts, Science & Commerce,
Chembur– 400 071. Mumbai, India.

Abstract

With abrupt advancement in mobile technologies more people are connecting to each other in different ways.. There was a time when the applications were only available in the ordinary desktop computers. But now, it is available also in wireless devices taking the current state of the technology available for accessing 3G networks into consideration. Mobile networks evolve into a more powerful medium for information sharing and the devices that are connected to them grow in their ability to process the data at faster rates. Chat applications are built-in with only some brand of mobile phones and they are compatible with them alone. They are costly and difficult for the naive ones to procure. As per the recent statistics, out of 1 billion mobile users in India only 137 million users are using internet. So to serve the rest of the users, designing mobile application with necessary utilities is mandatory. This Paper "Micro Utilities" is an attempt to bring the overall effectiveness of mobile applications with Encryption and decryption of the received messages and facility to send private SMS as an add-on utility so as to ensure the security. In spite of the existing in-built security mechanisms such as password and pattern for the folders, the confidentiality of the message is violated. So to ensure the security services, encryption and decryption of the messages of Inbox folder is introduced. This phase of the research deals with the encryption and decryption of the existing messages in the Inbox folder. It also ensures secure SMS communication between two users. During this phase, the author proposes the symmetric crypto system algorithm to encrypt/decrypt the sent/received messages to ensure the confidentiality and privacy of the communication.

This paper in overall suggests a model for mobile users in terms of data security there by providing a less economical facility for low cost mobiles.

Keywords*:* Encryption, security in mobile, private SMS, Decryption

1. INTRODUCTION

Cell phones have become a part of our daily lives. They have evolved to such a point now, that people cannot think of a life without mobiles. Now days, The cell phone users can take pictures, play games and most importantly send, receive and store data. These are possible because of the internet connections in the mobiles which ensure the better connectivity and competencies of the users. As per the census report of 2014[1], out of 1.27 billion population, 137 million users are using internet in mobiles. They contribute only 10.84% to the total population. This research is to concentrate the left out 89.16% of Indian populations by providing cost-effective and the most useful utilities by using J2ME technologies. Improving the security of the messages of low-cost mobile users by introducing the new utilities such as encryption, decryption and private SMS facility, which are economical and in par with the utilities which are available for the Smartphone users is the main objective of this paper. The research is limited to only those two utilities as specified in objectives and it targets the audience group having low cost mobiles and wanting to have basic Smartphone like utilities.

2. TECHNOLOGY SELECTION

The researcher selects J2ME as the technology to build the utilities. The J2ME is a specification of a subset of the Java platform aimed at providing a certified collection of Java APIs for the development of software for small, resource-constrained devices. The CLDC (Connected Limited Device Configuration) contains a strict subset of the Java class libraries, and is the minimal needed for a Java virtual machine to operate. CLDC is basically used to classify countless devices into a fixed configuration. When coupled with one or more profiles, the Connected Limited Device Configuration gives developers a solid Java platform for creating applications for consumer and embedded devices. MIDP (Mobile Information Device Profile) boasts GUI API, and MIDP 2.0 includes a basic 2D gaming API. Applications written for this profile are called MIDlets. Almost all new cell phones come with a MIDP implementation.

3. ENCRYPTION/DECRYPTION OF MESSAGES

While providing connectivity to the users, we must also ensure the privacy of the information as sensitive data also flows through the network. So, Data encryption becomes a necessity. Hence, the proposed work is designed to transmit the text messages of the users in encrypted format to the communicating parties. It ensures user-friendly, cost effective and platform independent secure communication. The author proposes a private key cryptography and one of its algorithms for the mobile applications in order to ensure privacy and security in the communication. Private key cryptography works on an agreement between the two users by having a shared key. The key is private in nature. In order to ensure the confidentiality of the message, The original text is converted into ciphertext by using keys known as encryption. To achieve integrity and consistency, The receiver converts cipher text to original text by using the same shared key, and this process is known as decryption.

The encryption and decryption process maybe executed either bit by bit basis or on the blocks, which are of 64 bits

in size. Java Platform supports different modes of encryption specification and it supports a range of private key algorithms. Out of the available algorithms, the author uses DES (Data Encryption Standard) Algorithm for encryption and decryption which is a 56 bit block cipher.
It involves the following functionalities:

- User Input
- Encode the messages/ Decode the messages
- Delete the messages

The proposed work accepts Valid password for secure login. Upon validation, it allows you either to write a new message or to select the existing messages from the folders. If the user selects the option 'Write a new message', the user shall be given private message option. Using this option, the communication between the user and the recipient can be made secure. If the user selects existing messages, the user will be given an option to encrypt the existing message which produces a new copy of the message enabled with encryption. If user wishes, he can delete the non encrypted message to protect his private data. The UI is provided by the emulator which is provided by Netbeans IDE. The entered details are stored using RMS of J2ME. The second functionality involves the vital task of encoding SMS using symmentric cryptosystem. As we are using symmetric cryptosystem where the same key is required for encoding as well as decoding the messages, this functionality includes also the facility of decoding the messages. The researcher proposes the usage of javax.crypto and javax.crypto.spec for cryptography and key specifications respectively. For the private message option, The researcher proposes the usage of javax.microedition and javax.wireless.messaging packages along with the selected encryption algorithm.
The final functionality will provide the option of deleting the messages. This is achieved with the help of javax.wireless.messaging package.

4.LOGIC BEHIND/SNIPPET OF IMPLEMENTATION

```
//<editor-fold defaultstate="collapsed" desc=" Generated Getter: Encrypt ">//GEN-BEGIN:|148-getter|0|148-preInit
   public Command getEncrypt() {
      if (Encrypt == null) {//GEN-END:|148-getter|0|148-preInit
         Encrypt = new Command("Item", Command.ITEM, 0);//GEN-LINE:|148-getter|1|148-postInit
      }//GEN-BEGIN:|148-getter|2|
      return Encrypt;
   }
//</editor-fold>//GEN-END:|148-getter|2|
//<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand ">//GEN-BEGIN:|150-getter|0|150-preInit
try{
            String orgmsg=tfMessage.getString();
            byte[] msg =orgmsg.getBytes();
            byte[] enMsg = new byte[10000];
            Cipher c = Cipher.getInstance("DES");
            byte[] b = "SECRET!!".getBytes();
            c.init(Cipher.ENCRYPT_MODE, new SecretKeySpec(b,0,b.length,"DES"));
            int numBytes = c.doFinal(msg, 0, msg.length, enMsg, 0);
            String s = new String(enMsg,0,numBytes);
           // System.out.println("Hello");

SendMessage.execute(tfDestination.getString(),"5000",s);
       }        catch(Exception e)        {}
//<editor-fold defaultstate="collapsed" desc=" Generated Getter: decrypt ">//GEN-BEGIN:|309-getter|0|309-preInit
        else if (command == decrypt) {//GEN-LINE:|7-commandAction|47|310-preAction
             String domsg=list3.getString(list3.getSelectedIndex());
             try {
            byte[] dmsg=domsg.getBytes();
            byte[] demsg=new byte[10000];
            Cipher c = Cipher.getInstance("DES");
            byte[] b1 = "SECRET!!".getBytes();
            c.init(Cipher.DECRYPT_MODE, new SecretKeySpec(b1,0,b1.length,"DES"));
            int numBytes=dmsg.length;
            int numByte = c.doFinal(dmsg, 0, numBytes, demsg, 0);//System.out.println("After doFinal Decode ");
            String s1 = new String(demsg,0,numByte);
            alert_decrypt = new Alert("Decoded Message");
            alert_decrypt.setString(s1);
            alert_decrypt.setTimeout(4000);
            display.setCurrent(alert_decrypt);
            }            catch(Exception e){}

  public Form getForm3() {
      if (form3 == null) {//GEN-END:|233-getter|0|233-preInit
         form3 = new Form("Write message", new Item[]{});//GEN-BEGIN:|233-getter|1|233-postInit
         form3.addCommand(getPvt_back());
         form3.addCommand(getSave_as_draft());
         form3.setCommandListener(this);//GEN-END:|233-getter|1|233-postInit
         tfDestination = new TextField("Destination", "", 13, TextField.PHONENUMBER);
          tfMessage = new TextField("Message", "", 160, TextField.ANY);
          cmdSend = new Command("Send", Command.OK, 1);
          form3.append(tfDestination);
       form3.append(tfMessage);
       form3.addCommand(cmdSend);
       form3.setCommandListener(this);
       display = Display.getDisplay(this);
       }//GEN-BEGIN:|233-getter|2|
      return form3;
   }
   public Command getInbox_delete() {
if (inbox_delete == null) {//GEN-END:|275-getter|0|275-preInit
         inbox_delete = new Command("Delete", Command.ITEM, 0);//GEN-LINE:|275-getter|1|275-postInit
       }//GEN-BEGIN:|275-getter|2|
       return inbox_delete;
   }
public Command getNew_draft() {
if (new_draft == null) {//GEN-END:|291-getter|0|291-preInit
new_draft = new Command("New", Command.ITEM, 0);//GEN-LINE:|291-getter|1|291-postInit
       }//GEN-BEGIN:|291-getter|2|
       return new_draft;
   }
public Form getForm4() {
if (form4 == null) {//GEN-END:|293-getter|0|293-preInit
form4 = new Form("Message", new Item[]{getTextField4()});//GEN-BEGIN:|293-getter|1|293-postInit
          form4.addCommand(getBackCommand10());
          form4.addCommand(getOkCommand6());
          form4.setCommandListener(this);//GEN-END:|293-getter|1|293-postInit
       }//GEN-BEGIN:|293-getter|2|
       return form4;
   }
public Form getForm5() {
if (form5 == null) {//GEN-END:|299-getter|0|299-preInit
form5 = new Form("Encrypt Message", new Item[]{});//GEN-BEGIN:|299-getter|1|299-postInit
          form5.addCommand(getEn_Back());
          form5.setCommandListener(this);//GEN-END:|299-getter|1|299-postInit
           tfDestination = new TextField("Destination", "", 13, TextField.PHONENUMBER);
           tfMessage = new TextField("Message", "", 160, TextField.ANY);
           ecmdSend = new Command("Send", Command.OK, 1);
           form5.append(tfDestination);
          form5.append(tfMessage);
       form5.addCommand(ecmdSend);
       form5.setCommandListener(this);
       display = Display.getDisplay(this);
       }//GEN-BEGIN:|299-getter|2|
       return form5;
   }
public Command getDecrypt() {
if (decrypt == null) {//GEN-END:|309-getter|0|309-preInit
decrypt = new Command("Decode", Command.ITEM, 0);//GEN-LINE:|309-getter|1|309-postInit
       }//GEN-BEGIN:|309-getter|2|
       return decrypt;
   }
public void inboxRead()
   {
       try{

             byte[] recData = new byte[1000];                    int len;
                for(int i = 1; i <= inbox.getNumRecords(); i++){
                if(inbox.getRecordSize(i) > recData.length){
                recData = new byte[inbox.getRecordSize(i)];
                }
                len = inbox.getRecord(i, recData, 0);
                list3.append(new String(recData, 0, len), null);
              }          }              catch(Exception e){}    }
public  void draftRead()
{  try{

              byte[] recData = new byte[1000];                   int len;

              for(int i = 1; i <= draft.getNumRecords(); i++){
                if(draft.getRecordSize(i) > recData.length){
                recData = new byte[draft.getRecordSize(i)];
                }
                len = draft.getRecord(i, recData, 0);
                list4.append(new String(recData, 0, len), null);
              }              }        catch(Exception e){}
public  void sentRead()
{       try{

              byte[] recData = new byte[1000];                   int len;

                for(int i = 1; i <= sent.getNumRecords(); i++){
                if(sent.getRecordSize(i) > recData.length){
                recData = new byte[sent.getRecordSize(i)];
                }
                len = sent.getRecord(i, recData, 0);
                list5.append(new String(recData, 0, len), null);
               }              }              catch(Exception e){}}
public void notifyIncomingMessage(MessageConnection conn) {
       Message message;
       try {
           message = conn.receive();
           if (message instanceof TextMessage) {

TextMessage tMessage = (TextMessage)message;
String inboxmsg=tMessage.getPayloadText()+"\n"+tMessage.getAddress()+"\n"+tMessage.getTimestamp();
list3.append(inboxmsg,null);
                         try{
inbox=RecordStore.openRecordStore("Inbox",true);
byte[]                imsg              =inboxmsg.getBytes();
inbox.addRecord(imsg, 0,imsg.length);
                         }
                         catch(Exception e){}
           } else {
              list3.append("Unknown Message received\n",null);
           }
       } catch (InterruptedIOException e) {
       } catch (IOException e) {           }}
class SendMessage {
public final String port="5000";
public static void execute(final String destination, final String port,final String message){
Thread thread = new Thread(new Runnable() {
public void run() {
MessageConnection msgConnection;
try {
msgConnection = (MessageConnection)Connector.open("sms://"+destination+":"+port);
TextMessage textMessage = (TextMessage)msgConnection.newMessage(
MessageConnection.TEXT_MESSAGE);
textMessage.setPayloadText(message);
msgConnection.send(textMessage);
msgConnection.close();
              } catch (IOException e) {            }} });
```

5. CONCLUSION

The main objective of this research is to provide security utilities to the low-cost mobile users in less economical way. By providing the above specified micro utility, the objective is fulfilled to the extent of its satisfaction. The implementation is done in the low-cost mobiles, the beta and UAT (User Acceptance Testing) has been carried out and the message security is ensured. As Java is freeware, the proposed software is free- of- cost. So its primary focus on user satisfaction is met. A good cause, in terms of micro utilities, by keeping user's perspectives in mind, this research gains competitive advantage. At the same time, this research keeps 90% (approx.) of the mobile users in India who are not having internet in their mobiles. Even if they are affordable to procure the connection for the first time, the maintenance cost is the major hurdle for them for not to be in the smart phone users' crowd. The mind-set and the mentality add to that hurdle. The research indirectly aims on "Social Responsibility" factor. If the utilities are used in their low-cost mobiles, the desire to go for the new phones just to enjoy these utilities will be reduced. M-Waste and the hazardous it creates to the society will also be reduced. In spite of some of the advantages, the research witnesses some of the limitations such as technological constraints. As J2ME has limitations with database, data storage and its persistent, the research also has its limitations when it has to implemented large scale. The future prospects of J2ME and its compatibility to support further versions, attaining large economies of scale is not far from reality.

References

[1]. www.r4r.co.in
[2]. http://ibnlive.in.com/news/mobile-internet-usage-in-india-outstrips-traffic-from-desktops-report/308772-26.html
[3]. www.dr-communications.com
[4]. www.leonsoftsolutions.com

AUTHOR

Jayalakshmi Srinivasan, M.Com., PGDCSA.,MCA.,MBA, working as Assistant professor for V.E.S College of Arts, Science and Commerce, Chembur, Mumbai since 2005.She was earlier working with Mulund College of Commerce for a year. She has authored and co-authored 9 books published by Vipul Prakashan. She has published 12 papers in National and International Conferences; An article in Developer IQ and 4 papers in online International Journal. Her paper was awarded as the best paper twice in two international conferences. She has been conferred with “Shiksha Rattan Puraskar Award” by IIFS, Delhi.