INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

Adopting eLockBox to achieve Mobile Data Security

Akshay Prakash Pawar[1], Rohan Manoj Thakkar[2], Saurabh Sahebrao Salunkhe[3], Sarvesh Suhas Kapre[4]

[1]K. J. Somaiya College of Engineering, Student of Computer Engineering Department, Vidya Vihar (E), Mumbai 400077, India
akshay070993@gmail.com
[2]K. J. Somaiya College of Engineering, Student of Computer Engineering Department, Vidya Vihar (E), Mumbai 400077, India
rohan.m.thakkar@gmail.com
[3]K. J. Somaiya College of Engineering, Student of Computer Engineering Department, Vidya Vihar (E), Mumbai 400077, India
saurabh.salunkhe81@gmail.com
[4]K. J. Somaiya College of Engineering, Student of Computer Engineering Department, Vidya Vihar (E), Mumbai 400077, India
sarveshk1808@gmail.com

***Abstract**: Smart phones like android are becoming common. Phones are very small in size and hence easy to misplace. Even a small inattention can lead to phone theft and hence information theft. Abusers can use your address lists, account information and various passwords that are stored in your phone. There are many applications that are available to lock your data stored on your phone but since this data is stored on the phone abuser can somehow manage to breach the lock because the password and the data is stored on your phone itself. Elockbox can provide a new layer of protection for sensitive data in mobile devices. It can provide meaningful protection without making any major changes in cell phones or the Operating Systems. Here we store the password and the data on the server instead of storing it on the phone. So even if abuser gets access to one's phone he can't get access to the server where actually the files are stored. eLockBox also provides a feature in which user or an organisation can specify the location from where the eLockBox/Client user can access the server. ELockBox will restrict the access to the server from other location co-ordinates. For further security eLockBox can also restrict anyone from accessing the files on server on unclassified date and time. If access is restricted on Sunday from 8 am to 5 pm no person will be able to access the server at the specified time.*

***Keywords:** Security Systems, LockBox, Mobile Data, Data Security, eLockBox, Mobile Data Security.*

1. INTRODUCTION

An individual has many personal email accounts and hence he needs to remember their passwords. Remembering all the credit card numbers is a heavy duty [6]. ELockBox can encrypt one's file containing all the ID's, passwords etc. and store it on the server instead of storing it on the phone itself. For Encrypting and Decrypting the file AES algorithm is used. AES algorithm can be implemented with 128, 192 and 256 bits [1]. AES algorithm is efficient to protect classified information which is on SECRET level. For TOP SECRET 192 and 256 bit key AES is recommended. AES algorithm is the main part in the eLockBox. The passwords required for authentication of a person and file is stored on the server rather than in one's phone. In the authentication process, a person is required to prove his identity using Login ID's or draw patter (CAPTCHA) or both for better security. After successfully entering into eLockBox a person can encrypt his file and then upload it on the server. A person is required to remember the secret key that is generated automatically while the process of encryption. This key will be further used while downloading the file from the server and decrypting it. In case if the user forgets his key he can remotely login into his account through his Desktop and retrieve his key. In case of phone theft the person can remotely login and delete all of his content that is stored on the server if he wishes [3].

More level of security can be added to this eLockBox by restricting the access to server only from the certain geographical location by using longitudinal and latitudinal co-ordinates. These co-ordinates are hardwired in the application code. E.g. If an organisation wants its employees to access the server only in the 10 meter radius it can set the co-ordinates of the Organisation location and limit to 10 meter [5]. Further the security level can be extended by restricting the access to the server at the specific time. The best example of this can be a military base station allowing the access to its server to the employees from base itself and at office hours.

2. LITERATURE REVIEW

Security mechanisms are based on two major determining variables – location and time.

2.1 Location based mechanism

Location based mechanism can be added to provide more security to our data. If the Organisation wants its employees to access the server from the particular location it can set its locational co-ordinates into the code. The employees will be able to access the server from those inbuilt locational co-ordinates only. The operational range of the application will be 10 meter radius from the seed co-ordinate. This range can be defined by the user [8].

If the employee try to access the server from some random location which doesn't satisfy the locational co-ordinates the application won't open.

For gathering the co-ordinates Android device will use GPS. Whenever user will try to login into the application a method will be invoked which will check the current locational co-ordinates of the place through GPS [10]. These co-ordinates

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

will be cross verified with the co-ordinates that are embedded into the code. If the co-ordinates satisfies the co-ordinates and range, application will proceed else application will give error message.

Location restriction Source code at main_activity

```
private void CheckMyLocation()
 {
                    // TODO Auto-generated method stub

if(gps.canGetLocation())
{

        double lat = gps.getLatitude();
        double long = gps.getLongitude();
        if (lat<=19.072675 && lat>=19.072800)
          {
          if(long >=72.900615 && long <= 72.900750)
          {
                    Toast.makeText(getApplicationContext(),
" You are in ABC Organisation",
Toast.LENGTH_LONG).show();
                    Intent ic=new
Intent(MainPage.this, LockPatternDemo.class);
                                        startActivity(ic);
          }
          else
          {
                    Toast.makeText(getApplicationContext(),
" You are not in ABC Organisation",
Toast.LENGTH_LONG).show();

          }
```

2.2 Time based mechanism

If the organization wants to allow employees to access the server on particular days (e.g Monday-Friday) and on particular time (e.g 10.00am to 5.00pm/ Office hours) it can change the code in the application and set the time and day dimensions to the relevant ones. Hence the employee can only be allowed to access the server at specified time and day. When user attempts to login into the application the method will be invoked which will check the current time and date of the phone. If the time and date matches the restricted time and date user won't be allowed further access. The code can also be written in such a way that the invoked method will retrieve the time and date from internet so that the user changing the device time and date and accessing the server is avoided [4].

Time based restriction code at main_activity

```
public void onCreate(Bundle savedInstanceState)
{

// TODO Auto-generated method stub

super.onCreate(savedInstanceState);
setContentView(R.layout.main_page);
gps = new GPSTracker(MainPage.this);
c = Calendar.getInstance();
// create object of calendar class
s = c.get(Calendar.SECOND);
//get system current Second
h = c.get(Calendar.HOUR);
// get System current Hour
m = c.get(Calendar.MINUTE);
ve = c.get(Calendar.AM_PM);
if (ve == 1)
{
          dz = "PM";
}
else
{
          dz = "AM";
}
showMe = String.format("%d:%02d:%02d-%s", h, m, s, dz);
//%02daninteger two digits

if (((h >= 9 && dz == "AM") || (h <= 5 && dz == "PM")))
{
/*if(c.get(Calendar.DAY_OF_WEEK) ==
Calendar.SUNDAY)
{
          Toast.makeText(getApplicationContext(),"Sunday
NoT works",      Toast.LENGTH_LONG).show();
          MainPage.this.finish();

}
else if
 {
          Toast.makeText(getApplicationContext(),
"Loading....", Toast.LENGTH_LONG).show();
}
else if
 {
Toast.makeText(getBaseContext(),"Oops Your time is: "+
showMe+ "\n Your application using time is after 9.00 AM
and before 6.00 PM but Not
Sunday",Toast.LENGTH_LONG).show();
finish();
}
}
```

3. NEW PROPOSED SCHEME

The scheme proposed by us in this research paper functions in the following steps-

3.1 Login

Initially the user is required to login into the application Lockbox. The Login can either be done by entering the Username and password or by drawing a CAPTCHA. The Login process is same as standard Login into any application. If CAPTCHA is chosen instead of standard Login, user is required to draw exactly the same pattern twice on the screen. If user enters his credentials right or draw the CAPTCHA right he can gain access to the Lockbox. For implementation of the CAPTCHA, SHA algorithm is used [2].

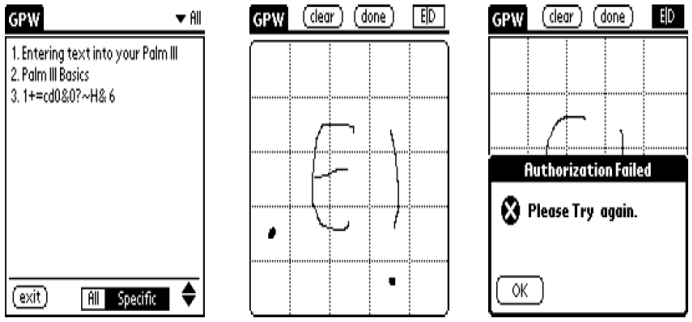
When user Login's for the first time he should register for the username and password. This Username and password will be used by the user to Login for the next time and

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

onwards. This application can be used by a single user on one phone.

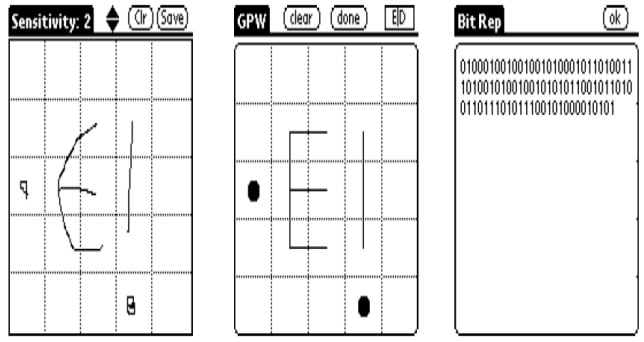
Draw-A-Secret

A technique DAS (Draw-a-secret) allows user to draw a unique pattern. A user should draw a simple pattern on a 2D grid. The coordinates of the grid are stored in the memory in the order it is drawn. During authentication, the user should re-draw the pattern. If the pattern touches the same grids co-ordinates in the similar sequence, then the user/client is authenticated [9].



(d) Interface to database (e) Re-entry of (incorrect) secret (f) Authorization failed

Figure 1: Failure of authorization



(a) User inputs desired secret (b) Internal representation (c) Raw bit string

Figure 2: Authenticated

When the user draws the pattern for the first time his finger movements along X-axis and Y-axis are stored in the array and when the user re-draws it is cross verified with the stored co-ordinates.

Code for storing X-axis and Y-axis co-ordinates

```
public boolean onTouchEvent(MotionEvent event)
{
                eventX = (int) event.getX();
                eventY = (int) event.getY();
                invalidate();
 //Toast.makeText(getContext(), "x is "+eventX+" y is
"+eventY, Toast.LENGTH_SHORT).show();
                list = new ArrayList<Integer>();
                list.add(eventX);
                list.add(eventY);
   //Toast.makeText(getContext(), ""+list,
Toast.LENGTH_SHORT).show();
              set = new HashSet<Integer>(list);
 // Toast.makeText(getContext(), ""+set,
Toast.LENGTH_SHORT).show();
                sb= new StringBuffer();
                sb.append(set);
```

We will save the values in the variable and compare those values while re-entering the pattern. If the values matches then the user will be allowed the further access else access will be denied.

Select path of the file

Once we download this application in our mobile we need to choose what data or application we want to keep in lock box. Once we choose a file we have to enter a path where that file is stored. Then press on encrypt button, hence data is encrypted and store in lock box. Simultaneously a random generated key will be displayed in the textbox next to path selection textbox. User will have to remember this key because the same key will be required in decryption process. Then press on confirm button once we press on it make sure that we are connected on Internet and then that key is stored on Server.

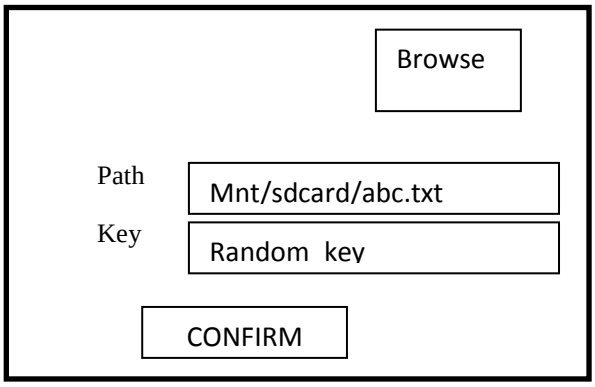


Figure 3: GUI of Select Path

3.2 Encryption Decryption Algorithm

A crucial problem for encryption is key storage. If keys are stored in an insecure part of the system, the encryption will not provide any protection. [7]. So key is stored in PC (Server) in encrypted form.Crypto class in the Lockbox is used to encrypt and decrypt strings type data using AES128. A SEED value is treated as a secret key. The stored content will be decrypted only with the same SEED. In general AES algorithm works on 4 basic steps:

1. Key Expansion – keys for each round are derived from Rijndael's key schedule.
2. Initial round- Bitwise XOR operation is performed between each state and round key.
3. Rounds

SubBytes: Each byte in the state is replaced by some other value in the lookup table.
ShiftRows: In this stage each row is shifted by some bits depending upon the number of round.
MixColumn: It takes 4 byte input and provides 4 byte output. Basically it provides diffusion.
AddRoundKey: In this stage each key is combined with each byte of state.

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

4. Final Round
   SubBytes
   ShiftRows
   AddRoundKey

- We use AES Algorithm here.
- For generation of AES key we use KeyGenerator .
- To store and use the generated key SecretKey is used.
- For encryption and decryption Cipher is used.

Encryption:

- A key(Seed) i.e. entered by the user during encryption.
- That key is passed to method called getRawKey(byte[] seed) where secret key is produced for the text.
- A Keygenerator is defined and as we are using AES encryption we pass an instance of the AES to the keygenerator.
- SecureRandom function is used to generate random numbers associated with the generated AES key.
- We want 128 bit key. We generate this key from the key generator and store it in a SecretKey.

Standard method for AES Encryption

```
private static byte[] encrypt(byte[] raw, byte[] ) throws
Exception
 {
      SecretKeySpec secretkey = new SecretKeySpec(raw,
"AES");
      Cipher c = Cipher.getInstance("AES");
      c.init(Cipher.ENCRYPT_MODE, secretkey);
      byte[] encrypted = c.doFinal(clear);
      return encrypted;
   }
```

3.3 Save on server

Initially a connection with the server is necessary before storing the contents on the server. This connection can be done using Client Socket.

```
Socket clientSocket = new Socket("192.168.18.116", 8080);
DataOutputStream toServer = new
DataOutputStream(clientSocket.getOutputStream());
BufferedReader  fromServer = new BufferedReader(new
printScr("TCP Connection established.");
```

For saving the contents on the server we will use JSON parser. We will convert the data into JSON format and send it to the server. JSON stores the information in the easy to access and organised manner.

Code

```
public JSONParser()
{
// function get json from url
// by making HTTP POST or GET mehtod
public JSONObject makeHttpRequest(String url, String
method,List<NameValuePair> params)
 {
          // Making HTTP request
          Try
           {
                    // check for request method
                    if(method == "POST"){
                    // request method is POST
                    DefaultHttpClient clienthttp = new
DefaultHttpClient();
                    HttpPost posthttp = new HttpPost(url);
                    posthttp.setEntity(new
UrlEncodedFormEntity(params));
                    HttpResponse responsehttp =
clienthttp.execute(posthttp);
                    HttpEntity entityHttp = httpResponse.getEntity();
                    }
                    else if(method == "GET")
                    {
                              // request method is GET
                    DefaultHttpClient httpClient = new
DefaultHttpClient();
                    String paramString =
URLEncodedUtils.format(params, "utf-8");
                    url += "?" + paramString;
                    HttpGet httpGet = new HttpGet(url);
                    HttpResponse httpResponse =
httpClient.execute(httpGet);
                    HttpEntity httpEntity = httpResponse.getEntity();
                    is = httpEntity.getContent();
                    }
try
                    {
                    BufferedReader read = new BufferedReader(new
InputStreamReader(
                                                  iread, "iso-
8859-1"), 8);
                    StringBuilder build = new StringBuilder();
                    String line = null;
                    while ((line = read.readLine()) != null)
                    {
                    build.append(line + "\n");
                    }
                    iread.close();
                    json = build.toString();
                    }
```

The necessary information about the user will also be saved on the server which will include name of the file, secret key. While downloading from the server the opposite procedure will be carried out. The file will be downloaded from the server and file will be converted from JSON format to text format. The method used to store data on the server completely depends on the programmer.

3.4 Decryption

When user wants to get the file which he has stored on the server he should login every time whenever he needs the file. The process is same as login in for the process of encryption. Here user will select the file that he wants and enter the Random key that was generated at the time of encryption. After entering the key user should press confirm button.

The file will be downloaded from the server and decrypted on the user's phone.

The Decryption process for the AES algorithm is Inverse of the Encryption process.

In this process the stages that will be used are:

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

Inverse SubBytes: Each byte in the state is inversely replaced by some value in the lookup table.

Inverse ShiftRows: In this stage each row is inversely shifted by some bits depending upon the

Inverse MixColumn

Standard method for Decryption

```
private static byte[] decrypt(byte[] raw, byte[] encrypted)
throws Exception
 {
      SecretKeySpec secretkey = new SecretKeySpec(raw,
"AES");
      Cipher c = Cipher.getInstance("AES");
      c.init(Cipher.DECRYPT_MODE, secretkey);
      byte[] decrypted = c.doFinal(encrypted);
      return decrypted;
   }
```

4. UML DIAGRAMS

Unified modeling diagrams are used to portray the flow of entire software model. Some UML diagrams used for our project are-

4.1 Application flow diagram

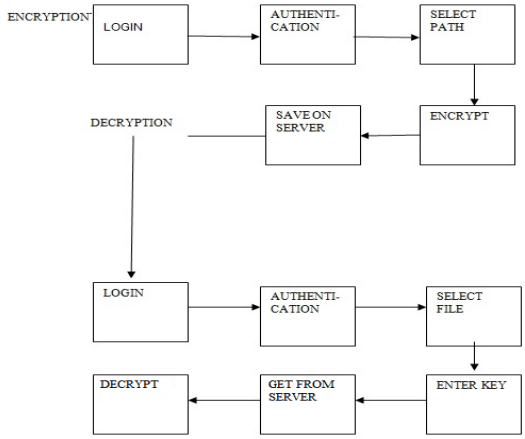


Figure 4: Application Flow diagram

4.2 Use Case diagram

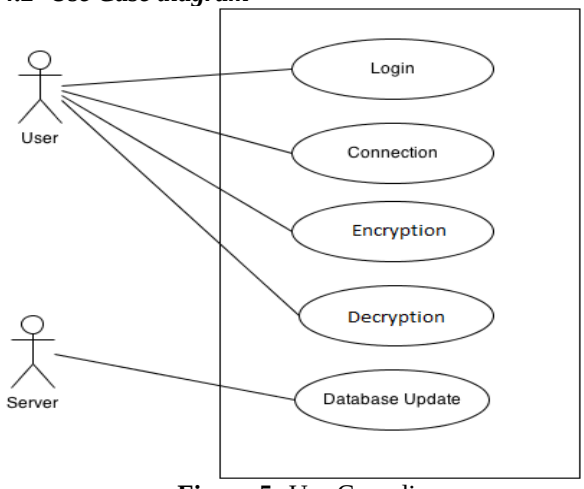


Figure 5: Use Case diagram

4.3 Class diagram

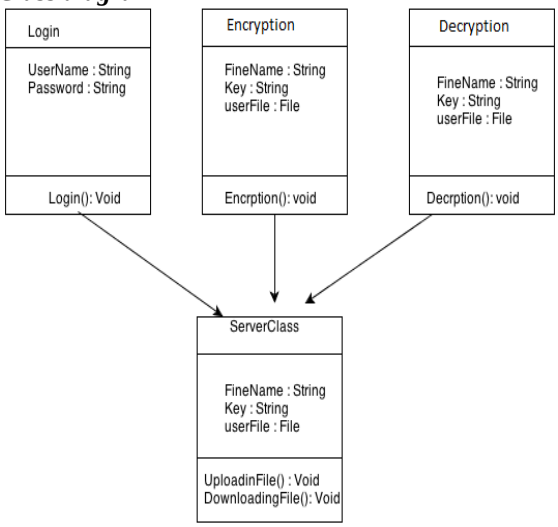


Figure 6: Class diagram

4.4 Sequence diagram

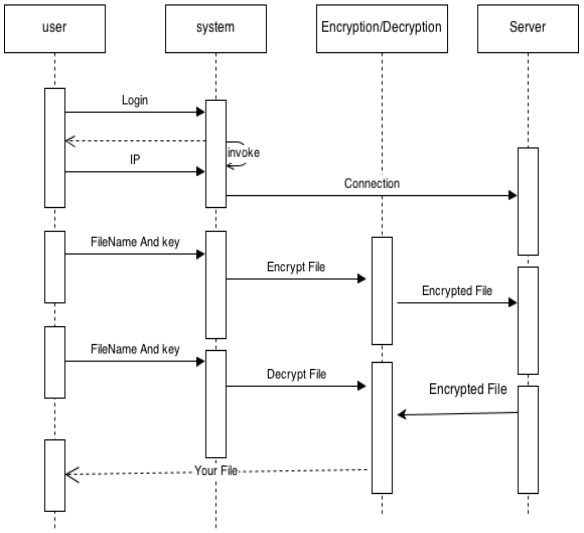


Figure 7: Sequence diagram

5. CONCLUSION

The proposed security system is more advanced than the ones used previously due to various reasons. To spot a few, they are -

1. The data that we encrypt is secured because we store it on the server rather than on device. So misplacing of device will not exploit our data.
2. The time based and location based mechanism can restrict the access to the server from unknown locations and unspecified time hence enhancing the security.
3. This idea can be of a great use to Military and other secret organization where data security and confidentiality is the main part.
4. Remote login to the server can allow you to wipe your data from the server or restore your password from the server in case you forget it.

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS.....

References

[1] **"Raytheon to Bring Powerful Mobile Apps to Military and Government Users," 10/12, 2010;http://www.raytheon.com/newsroom/feature/rtn09_iphnapps/**

[2] **Announcing the Advanced Encryption Standard (FIPS PUB 197)**

[3] **J. Indulska, P. Sutton, "Location management in Pervasive Systems," Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, Adelaide, Australia, February 2003.**

[4] **W. Jansen, V. Korolev, S. Gavrila, C. Séveillac, "A Unified Framework for Mobile Device Security," The International Conference on Security and Management (SAM'04), Las Vegas, USA, June 2004.**

[5] **Ian Jermyn, Alain Mayer, Fabian Monrose, Michael K. Reiter, and Aviel D. Rubin," THE DESIGN AND ANALYSIS OF GRAPHICAL PASS WORDS", Proceedings of the 8th USENIX Security Symposium Washington, D.C., USA, August 23–26, 1999.**

[6] **Wayne Jansen, Vlad Korolev, Booz-Allen Hamilton, "A Location-Based Mechanism for Mobile Device Security", 2009 World Congress on Computer Science and Information Engineering.**

[7] **Application Lockbox for Mobile Device Security, Eighth International Conference on Information Technology: New Generations, 2011.**

[8] **Mobile Apps for the Military, DARPA-SN-10-27, 2010.**

[9] **C. T. Lopez. "AKO 'Go Mobile' to give users virtual desktop in backpack," 10/12, 2010.**

[10] **M. Gruteser, G. Schelle, A. Jain, R. Han, D. Grunwald,"Privacy-Aware Location Sensor Networks," USENIX 9thWorkshop on Hot Topics in Operating Systems (HOTOS IX), Lihue Hawaii, May 2003, pp. 163-167**