A Model for Secure Data Access from Smartphones using Short Message Service

Afreen Khan¹ Ashwini Chougule² Mahesh Mookanagoudar³ Vidya R. Kulkarni⁴
¹²MCA Dept. ³Android Developer ⁴HOD (MCA Dept.)
¹²⁴Gogte Institute of Technology College, Belgaum, Karnataka

Abstract—The Smartphone usage among people is increasing rapidly. To perform remote accessing of mobile contents there are many ways which make use of internet. This paper presents an Android based Application which is a secure data access from android Smartphone using any normal phone without the internet connection. This Model provides the short message service where we can access the data from Smartphone such as contacts, call log, messages, files by sending a message as well as secure deletion of the same data from android mobiles using any mobile through SMS using different commands for performing different actions. This model doesn’t need internet for accessing the data.

Keywords: Commands: Android, Short Message Service, contacts, call log, files.

I. INTRODUCTION

Now a day, usage of mobile has become a vital part of day-to-day activities of people. We can refer the current time as the era of Smartphone’s. Recently with the introduction of android in Smartphone, there exist many applications for different purposes which meet wide variety of user requirements. Amongst those is the remote accessing of mobile, when the mobile is left in somewhere like house or office or the mobile which is geographically away from the user’s place by just sending an SMS with appropriate commands to the android phone wherein this solution is implemented. It includes getting the incoming call numbers, incoming messages, accessing call logs, changing phone’s GPS, WIFI and profile settings and retrieving of contacts.

The major objectives of this paper have been listed below.
- Erase the critical data that has been stored in the mobile.
- Access the call log, contacts and SMS.
- Access the files that has been stored in the mobile
- Access and change GPS, WIFI and profile settings through SMS.

II. PROPOSED MODEL

The proposed model gives an innovative approach to access one Smartphone through another Smartphone or through a normal mobile. This model uses SMS as the communication channel.

The proposed model performs based on the SMS received. The SMS contains the commands; the commands include password and the action that the user wants to perform on their mobile. The concept involved is like this, the app(solution) implemented will be triggered for every SMS that the mobile receives. On receiving the SMS, the app will fetch the SMS and verify for the password. On successful authentication, it checks for the remaining commands. If the command match with the predefined commands then based on the type of command that is categorized, whether they are meant for accessing the data like SMS/Call/Contacts/Files or they are meant for deletion of SMS/Call log/Contacts or changing the GPS/WIFI/Profile settings, the actions will be performed and for access operations on successful fetching of the requested data, the data will be sent to the user mobile which had made the request via SMS whereas for deletion operations no result is sent. On unsuccessful match or if no data exists then nothing is performed. The major advantages of this concept are that it need not be running continuously in the background and it doesn’t need internet and the client can have any mobile to perform operations on the Android mobile wherein the solution is implemented. The system makes use of Shared Preferences for setting the password which is used to authenticate the user when a request is made to access specific data through SMS. The technique of performing actions on an android mobile needs a pattern which will make the mobile to understand and process the action/request.

Considering that the password is set, commands need to be defined to perform various requests. Let the commands be, GET_CONTACT for accessing the desired contact number, GET_SMS for accessing the SMS, GET_CALL_LOG for accessing the call log, GET_FILE to access the contents of a file, WIFI_ON/WIFI_OFF which will switch on/off the WIFI, GPS_ON/GPS_OFF which will Switch on the location finder and inform the user about the current location. It will work in background to track and inform the user often or Mobile tracking will be disabled. Similarly let the commands be, DEL_SMS to delete the contents of the SMS, DEL_CONTACTS to delete the contents of Phonebook, DEL_CALLOG to delete the call log.

Once the command is defined, the user can make a request to access specific data by sending an SMS like the following.

A. Password, command

The above pattern remains same for all actions apart from accessing contacts. The following is the pattern for accessing contacts

B. Password, desired_contact_name

Considering the user has made a request for accessing a contact number of a particular contact. Then the system behaves in the following manner, the first stage of operation deals with monitoring for the new SMS. Once the SMS enters the mobile, the BroadcastReceiver class informs our system about the arrival and the system fetches the newly
arrived SMS to verify whether or not the application is meant for the system. Once the system fetches the SMS, it retrieves the content of it to verify that the first word in the SMS is the password. If the password matches then it tries to verify what action is being requested by the user by verifying the second word of the SMS with the predefined command. On successful match, it tries to perform requested action. Difference over here would be if the second command is GET_CONTACT then the system needs to fetch the third word from the SMS which reveals the name of the desired contact number. Based on the second command the actions will be performed. For accessing the contact, the third word from the SMS will be considered. The so extracted word will be used with a query to check whether the requested contact exists in the phonebook, if it exists then it will be extracted and an SMS will be composed with the body part containing the extracted contact number and the two part containing the number from which the request was made and will be sent.

C. Broadcast Receiver android Content. Broadcast Receiver

When a matching event is generated in the system, Android delivers the event to that broadcast receiver. Applications with Broadcast Receivers registered in the manifest don’t have to be running when the Intent is broadcast for the receivers to execute. They will be started automatically when a matching. This is excellent for resource management as it lets you create event-driven applications that will still respond to broadcast events even after they’ve been closed or killed.

III. ARCHITECTURE

The Architecture diagram for the proposed model is shown in Figure 1: We can relate this architecture to client-server architecture where the normal phone is the client who sends the request to the server through the SMS where in the SMS contains the password and a particular command for the specific operation to be performed or the request to be handled. As soon as the server which is the android Smartphone where the application is installed receives the request that is the SMS then the Broadcast Receiver class will notify about the arrival of the SMS then the application handles the request where it authenticate the password which was already set with the password in the SMS.

If it matches it checks for the command and then according to the command specified it performs the action that is accessing the data which is stored in the database. Then, the data that is fetched according to the request that is the result is formed into an SMS and that result SMS is sent back to the normal phone in the form of response. Commands are the predefined words that will instruct the request handler to do a task. For example, SILENT-ON is a command that will instruct the request handler to change the mobile sound to silent and SILENT-OFF will change the phone profile to normal.

IV. SAMPLE CODE

A. Accessing contact Code

```java
ContentResolver contentResolver = context.getContentResolver(); Cursor cursor = cr.query(ContactsContract.Contacts.CONTENT_URI, null, "DISPLAY_NAME = '" + command3 + "'", null, null);
if (cursor.moveToFirst()) {
    String contactId = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts._ID));
    Cursor phones = cr.query(Phone.CONTENT_URI, null, Phone.CONTACT_ID + " = " + contactId, null, null);
    while (phones.moveToNext()) {
        number = phones.getString(phones.getColumnIndex(Phone.NUMBER));
    }
}
```

B. In the same way when the user requests to delete the contacts from phonebook then

```java
if(command2.equals(delContactCommand)) {
    Cursor cursor = contentResolver.query(ContactsContract.Contacts.CONTENT_URI, null, null, null, null);
    while (cursor.moveToNext()) {
        String lookupKey = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.LOOKUP_KEY));
        Uri uri = CONTENT_LOOKUP_URI, lookupKey);
        Uri.withAppendedPath(ContactsContract.Contacts.CONTENT_URI, null, null, null, null); 
        Uri.withAppendedPath(ContactsContract.Contacts.CONTENT_URI, null, null, null, null); 
        while (managedCursor.moveToNext()) {
            if(command2.equals(delCallLog)) {
                contentResolver.delete(CallLog.Calls.CONTENT_URI, null, null)
            }
        }
    }
}
```

C. To delete the call log

```java
if(command2.equals(delCallLog)) {
    contentResolver.delete(CallLog.Calls.CONTENT_URI, null, null)
}
```

D. To delete SMS

```java
for (int i = 0; i < vector_id.size(); i++) {
    str_id = vector_id.get(i);
    where = str_column_name + "=" + str_id;
    delRow = cr.delete(uri_sms, where, null);
}
```

E. To access call log

```java
while (managedCursor.moveToNext()) {
    
```
String cntName = managedCursor.getString(callName1);
String phNumber = managedCursor.getString(number1);
String callType1 = managedCursor.getString(type);
String callDate1 = managedCursor.getString(date);
Date callDayTime = new Date(Long.valueOf(callDate1));
String callDuration = managedCursor.getString(duration1);
String dir = null;
int dircode = Integer.parseInt(callType1);
switch (dircode) {
    case CallLog.Calls.OUTGOING_TYPE: dir = "OUTGOING"; break;
    case CallLog.Calls.INCOMING_TYPE: dir = "INCOMING"; break;
    case CallLog.Calls.MISSED_TYPE: dir = "MISSED"; break;
}
result = "Name= " + cntName + " Phone Number "+phNumber+" Date "+callDayTime+" Call Type "+dir+" Duration in sec "+callDuration; }
managedCursor.close();
}

F. To access SMS

Uri smsUri = Uri.parse("content://sms/inbox");
Cursor cur = context.getContentResolver().query(smsUri, null, null, null, null);
String smscontent = "";
while (cur.moveToNext()) {
    if(cur.getPosition() <= 2 && (null != cur.getString(2) || "" != cur.getString(2)) && (null != cur.getString(11) || "" != cur.getString(11))
        {
        System.out.println(cur.getPosition()); smscontent += "From :" + cur.getString(2) + " : " + cur.getString(11) + "\n";
        }
}

V. FLOW CHART

A. DELETE SMS/Call log/Contacts

Fig. 2: Delete Flow Diagram

B. DataAccessing

Fig. 3: Data accessing Flowchart

VI. IMPLEMENTATION

The proposed model for Smartphone remote access has been implemented in the Android 2.2 platform Operating System. The normal mobile users have to type the command from the messaging interface of their mobile. The following commands are implemented.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET_SMS</td>
<td>Access the recent 3 SMS from the remote Smartphone</td>
</tr>
<tr>
<td>GET_CONTACT</td>
<td>Access the contact no of a specified person from the phonebook</td>
</tr>
<tr>
<td>GET_CALL_LOG</td>
<td>Access all the call-logs</td>
</tr>
<tr>
<td>DEL_SMS</td>
<td>Deletes all the SMS from the remote Smartphone</td>
</tr>
<tr>
<td>DEL_CONTACT</td>
<td>Deletes all the contacts from the phonebook</td>
</tr>
<tr>
<td>DEL_CALLLOG</td>
<td>Deletes all the call logs</td>
</tr>
<tr>
<td>DEL_CONTACT_OF</td>
<td>Deletes the contact of a specified person</td>
</tr>
<tr>
<td>GET_FILE</td>
<td>Access the file which is specified</td>
</tr>
<tr>
<td>DEL_FILE</td>
<td>Deletes all the files</td>
</tr>
<tr>
<td>WIFI_ON</td>
<td>Wifi will be switched on by SMS command</td>
</tr>
<tr>
<td>WIFI_OFF</td>
<td>Wifi will be switched off to save the battery</td>
</tr>
<tr>
<td>GPS_ON</td>
<td>Switch on the location finder and inform the user about the current location. It will work in background to track and inform the user often.</td>
</tr>
<tr>
<td>GPS_OFF</td>
<td>Mobile tracking will be disabled.</td>
</tr>
<tr>
<td>SILENT_ON</td>
<td>Silent all the sound of the mobile.</td>
</tr>
<tr>
<td>SILENT_OFF</td>
<td>Silent option will be removed.</td>
</tr>
</tbody>
</table>

Table 1: List of commands in the prototype implementation
In Figure 4 contains the 2 screen shots of the implemented model where the left one is the homepage of the application where we can set the password, about the application, how to use an exit buttons are there which will guide us to use the application and the right one is the screen which we get when we click on the password button an can set our password there.

In Figure 5, 6 and 7 it is shown how we can retrieve the SMS, Contacts and Call-logs by sending the commands through an SMS. And in Figure 8, 9 and 10 it is shown how we can delete the same data by sending the SMS which contains the commands to perform the action.

VII. CONCLUSION

The proposed model has been implemented in android operating system. This paper is based on the concept of SMS and many Android telephony API’s. As Android is open source, there exists loads of resources to understand and it becomes easy to implement and deploy the solution. It was tested in Samsung galaxy pro Smartphone. This provides the encouraging result. The model can be implemented in other Smartphone platforms like windows, apple, etc. The conclusions drawn from the proposed system are listed below:

- The proposed model facilitates accessing of the device from a remote location using any other mobile terminal.
- The system has been designed in such a way that the mobile terminal used for accessing the remote android device, need not be an android device.

REFERENCES

[1] Secure Data access from Android Smartphones without using Internet by Mahesh Mookangouda , S P Sajjan,
Shashidhar Halligerimath published in International Journal of Computer Applications.