International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 05 | May-2016 www.irjet.net p-ISSN: 2395-0072

Android Device and User Data Security via Web Services and SMS Command

Khushal Khairnar, Pratik Parkhe, Piyush Kharade, Chaitanya Kakade, Krishna Kokate

[1]*Khushal Khairnar*, Dept of Information Technology, *MMIT, Pune*
[2]*Krishna Kokate, Dept of Information Technology, MMIT, Pune*
[3]*Pratik Parkhe, Dept of Information Technology, MMIT, Pune*
[4]*Piyush Kharade,* Dept of Information Technology, *MMIT, Pune*
[5]*Chaitanya Kakade, Dept of Information Technology, MMIT, Pune*

Abstract - *As per the increase use of android device nowadays its security is main concern today. Now protection for android has many flaws and shortages. In our system we propose an efficient and secure EasySMS, which provides end to end secure communication through SMS between end users. We provide browser free multilevel smart phone privacy protection system, which is based on the android sensor platform. User can send SMS to his device as operating instruction; after then the sensors on remote phone executes the instruction and perform desired operation such as locking the phone. Secondly our system runs this process to work as a daemon process which will always run at the background of the device and prevented from being forcefully closed or uninstalled. This system provides backup functionality so that the data redundancy will be maintained and we can access our data from database. One can get the last location of victim phone by providing GPS function this will help to protect device of victim which been stolen by thief. Another functionality is to wipe the data, if the victim user feel the threats of the data i.e. phone data is been use by the malcious user or misuse data. These points ensure full protection and privacy of the system.*

Key Words: **EasySMS, GPS, SIM, Remote Lock, Daemon Process, Android Sensor Framework**

1.INTRODUCTION

Android is an open-source, programmable software application framework. Users with help of their smart devices use this to download Android applications, which bring much ease and convenience to their life. Convenience results in a way such that many people always use their smartphone to store their many work related information. User gets benefit so that it can access the data on the go. However, information means a lot to the users, as it consists of various private and various work concern data. If information is used by malicious attackers, such as contact information, SMS, and data on storage card, the data can be used for any misleading purpose. Till today protection methods for Android have some limitations and a escape way: primary security mechanisms can (screensaver lock, electronic password lock) sometimes prevent users from using phones at a rapid access. Users often close these security applications for their personal convenience in local mode, this in case affects the data protection which can be altered when phones are lost or stolen. Various phone protection systems are based on client-server (C/S) interaction model, which also brings much convenience and ease. Taking in consideration of all problems mentioned above, this paper proposes a browser-free multilevel smart phone protection system and data, which is based on the Android sensor framework module. In our system , After a valid installation on user phones, useful information is autosync to servers users database. When the phone is lost, the user can use trusted phone numbers to send the the particular designed command to lock the lost or stolen phone. Second, the whole system work on the basis of daemon process mechanism which is used to prevent the system from being maliciously handled and uninstalled externally from other users, which increases the flow and security of our system. The remote locking and wipe system consists of a remote control module on a server and a command handling module on a smartphone. For example, when the users send a lock command to the users smartphone via the remote control module access, the remote handling module function enables the password locking functional module to lock the users smartphone which is stolen or lost. Similarly, by sending a wipe command with a command number, all personal data such as wallet data, smart keys, important any sort of documents, contacts, SMS, E-mail, photos, messages, images and movie clips are erased with the help of remote function. The remote locking and wipe service will be very useful when users smartphone is lost or stolen. However, there might be the case that the malicious user misuses this function by sending such commands to the normal users in order to disrupt the service. Hence, it is very important to check if the command is originated from the trusted person source which is integrated in users phone.

Introduction for Android Sensor Framework. Android is an openware operating system platform for mobile devices collaborated for framework, and core applications for the

users. We can access or use this sensors available on the device and acquire sensor data by using the Android sensor framework. The sensor framework can be used for gaining access several classes and interfaces that help one to perform a wide variety of sensor-related tasks and work. The Android software application stack is built with the help of Linux kernel, which is used for its device drivers access, memory management access, process management access, and networking related constrains. In this paper, we propose software-based sensors and using the functionality of Android system to protect the mobile privacy and data.

2. LITERATURE SURVEY

Various papers for providing the security for android devices as illustrated and discussed here.

Smartphone Remote Lock and Wipe System with Integrity Checking of SMS Notification by Kyungwhan Park1, Gun Il Ma1, Jeong Hyun Yi1[1]. In this proposed paper, their system have works on a MAC-based remote lock module and wipe module system through the SMS push notification to protect the users data disclosure against stolen user when smart phone is lost or stolen. This proposed system provides the total integrity checking mechanism so that the malicious users are not able to launch denial of service attacks which can send the lock or wipe commands to the victim user on purpose. Also it satisfies the SMS length limit of 80 bytes long without altering the security parameter.

Utilizing GPS and SMS for Tracking and Security Lock Application on Android Based Phone By Yosi Kristiana, Hendrawan Armantoba and Michael Fransb[2]. Seeing many of missing case of smartphones they have try to develop a system for tracking and securing a lost or stolen phone. Android platform is taken into consideration because it is one of the demanded operating system for mobile phones right now and the user growth is increasing remarkably. The system flow is very sophisticated, these application will be on standby and look after the command which is received via SMS and it will execute in the background so user won't be get any pop-ups from this application. Using these commands user can lock his phone, sound the phone alarm, erase his phone data, access his phone data, and fetch his phone last known location. This application also consists of feature like locking the phone when change in SIM card occurs in users phone and send alert through web services. When the victim user tries to fetch the location of his lost or stolen through web services the desired commands will be send to the lost phone through which the GPS module execute the command and give back the required result to the user via a dedicated web service.

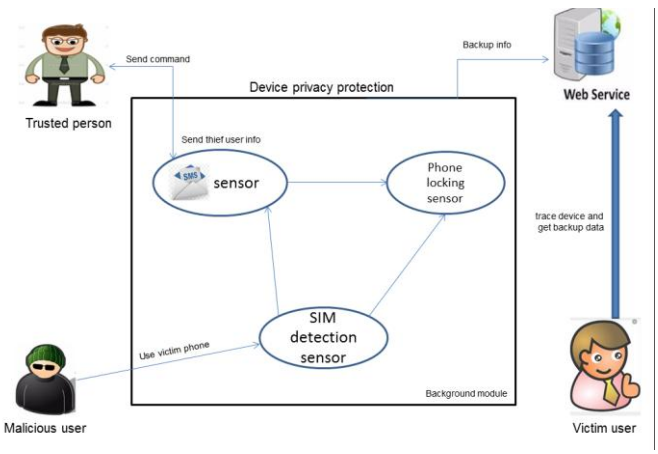
Phone Protector: Protecting User Privacy on the Android-Based Mobile Platform by Weizhe Zhang,Hui He,Qizhen Zhang,and Tai-hoon Kim[3]. Privacy protection of Android platform becomes a prime concern in this paper. Now securing an Android based smartphone has many limitations and setbacks. In this paper, they have proposed a browser-free multilevel smart phone privacy protection system, which is dependent on Android sensor modules. In this, protection is ensured by means of SMS, which results out to be easy, quick, and convenient. Users can send SMS to phones remotely as operating instructions; then the sensors on remote phones execute the instructions and return useful information. Second, the sensors based on the daemon process mechanism are used to prevent the sensors from being maliciously closed and uninstalled. Third, system adopts SIM detecting mechanism to judge whether the SIM card is removed or changed. If exception is detected, the phone will be locked automatically by its inside sensors. The three points ensure full protection of phone data privacy.

Authentication System for Android Smartphones by Swapnil Waghmare ,Madhumita Chatterjee , Satish Varma [4].In this paper , their system has proposed the security application which provide various features like unlocking the phone using random number , taking system backups via email, receive notification of SIM change via message, track incoming calls and messages when smartphone of user is lost or stolen. The system also gives SIM details to registered number of user by sending message when new SIM card is inserted in user's smartphone.

A Study of Android Application Security by William Enck, Damien Octeau, Patrick McDaniel, and Swarat Chaudhuri [5]. In this paper, the system introduce the ded decompiler, which recovers Android application source code directly from its installation image. It includes design and execute a horizontal study of smartphone applications based on static analysis of 21 million lines of recovered code. At the end it is concluded by considering the implications of these preliminary findings and offer directions for future analysis.

3. SYSTEM ARCHITECTURE AND ALGORITHM

3.1 System Architecture



3.2 System Algorithm

1. Start
2. Install App with all the details.
3. (If user's device is stolen) Trusted User can lock user's device (User must have to tell Trusted user about stolen device)
4. Trusted User can also wipe the data from user's device
5. User can track GPS position of the device through Web Service
6. If SIM is removed by malicious user) Device is locked by Phone locking Sensors of the Device.
7. (If SIM is removed GPRS/WIFI is disabled then) It is not possible to track the device
8. END

4. SYSTEM IMPLEMENTATION

4.1 Modules

4.1.1 Getting Last location

Using the Google Play services location APIs, your app can request the last known location of the user's device. In most cases, We are interested in the user's current location when device get lost, which is usually equivalent to the last known location of the device. Specifically, use the fused location provider to retrieve the device's last known location. The fused location provider is one of the location APIs in Google Play services. It manages the underlying location technology and provides a simple API so that user can specify requirements at a high level, like high accuracy. Android API show how to make a single request for the location of a device using the getLastLocation() method in the fused location provider. Following are step to get last location.

Set Up Google Play Services

To access the fused location provider, your app's development project must include Google Play services. Download and install the Google Play services component via the SDK Manager and add the library to your project.

Specify App Permissions

Apps that use location services must request location permissions. Android offers two location permissions: ACCESS COARSE LOCATION and ACCESS FINE LOCATION. The permission you choose determines the accuracy of the location returned by the API. If you specify ACCESS COARSE LOCATION, the API returns a location with an accuracy approximately equivalent to a city block. This application requires only coarse location. Request this permission with the uses-permission element in your app manifest, such as:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.google.android.gms.location.sample.basiclocationsample" >

  <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
</manifest>
```

Connect to Google Play Services

To connect to the API, you need to create an instance of the Google Play services API client. In your activity's onCreate() method, create an instance of Google API Client using GoogleApiClient.Builder. Use the builder to add the LocationServices API. The sample app defines a buildGoogleApiClient() method, called from the activity's onCreate() method, which includes the following code.

```
protected synchronized void buildGoogleApiClient() {
    mGoogleApiClient = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
}
```

Get Last Location

Once you have connected to Google Play services and the location services API, we can get the last known location of a user's device. When user app is connected to these we can use the fused location provider's getLastLocation() method to retrieve the device location. The precision of the location returned by this call is determined by the permission setting you put in user app manifest.
To request the last known location, call the getLastLocation() method, passing it your instance of the GoogleApiClient object. Do this in the onConnected() callback provided by Google API Client, which is called when the client is ready. The following code sample illustrates the request and a simple handling of the response:

```
public class MainActivity extends ActionBarActivity implements
        ConnectionCallbacks, OnConnectionFailedListener {
    ...
    @Override
    public void onConnected(Bundle connectionHint) {
        mLastLocation = LocationServices.FusedLocationApi.getLastLocation(
                mGoogleApiClient);
        if (mLastLocation != null) {
            mLatitudeText.setText(String.valueOf(mLastLocation.getLatitude()));
            mLongitudeText.setText(String.valueOf(mLastLocation.getLongitude()));
        }
    }
}
```

4.1.2 Synchronization

Many mobile applications require to sync data with a server if they operate in the client server model to exchange data with the central repository. If the server serves up resources

through a REST API, then all sync logic can be handled on the client side. Each table (both server and client) participating in the sync process should include two additional fields: Ts as timestamp and Deleted as Boolean. The Ts (timestamp) field is maintained by the server side. For example SQL Server automatically generates a unique incremental value on each insert or update. The Deleted field is used to mark records for deletion so the clients would remove such records from local repositories, whereas clients with initial sync would simply skip them during download. Additionally the client data storage should include the 'Dirty' field as Boolean. The Dirty field designates the records changed locally. Once those records sent successfully to the server the Dirty flag reset.

Synchronization algorithm

1. Initialize the list of tables required to sync
   /******* Upload client changes **********/
   For each sync table:
2. Get the list of local storage changes for inserts on the server
3. Iterate through each record in the local changes for inserts
   - Insert the record on the server (using POST)
   - Get the timestamp for the inserted record from the response
   - Update the record in the local storage with new timestamp
   - Reset the dirty flag
4. Get the list of local storage changes for updates on the server
5. Iterate through each record in the local changes for updates
   - Update the record on the server (using PUT)
   - Get the updated timestamp for the updated record from the response
   - Update the record in the local storage with new timestamp
   - Reset the dirty flag

   /******* Post sync operation ***********/
   For each sync table:
6. Delete records where Deleted=true and Dirty=false

4.1.3 Device Lock

You can set the maximum period of user inactivity that can occur before the device locks.
For example:

```
DevicePolicyManager mDPM;
ComponentName mDeviceAdminSample;
...
long timeMs = 1000L*Long.parseLong(mTimeout.getText().toString());
mDPM.setMaximumTimeToLock(mDeviceAdminSample, timeMs);
```

You can also programmatically tell the device to lock immediately:

```
DevicePolicyManager mDPM;
mDPM.lockNow();
```

4.1.4 Wipe Data

You can use the DevicePolicyManager method wipeData() to format data in the device. This is useful if the device is lost or stolen. Often the decision to wipe the device is the result of certain conditions being met. For example, you can use setMaximumFailedPasswordsForWipe() to state that a device should be wiped after a specific number of failed password attempts.
You wipe data as follows:

```
DevicePolicyManager mDPM;
mDPM.wipeData(0);
```

5. EXPERIMENTS AND ANALYSIS

The system is tested for 7 types of smart phones. They are HTC A610, HTC A616, MOTO E, YUREKA PLUS, Samsung Galaxy y, Karbon A2 and MOTO G. The test results are as follows.

5.1. Compatibility Testing: We check the compatibility of our system by installing it on 7 different kinds of smart phones. The compatibility passing rate is 100%.

5.2. Performance Testing: We test the starting time consumption, CPU utilization, and memory utilization. The average start time is 0.63 s; the average CPU utilization is 2.9%; the average memory utilization is 8.16MB. Compared with other phone applications, this resource occupation is relatively much lower.

6. CONCLUSIONS

In this paper, we propose a smart phone privacy protection system, which is based on the Android sensor framework. In this system, operation instructions are sent through SMS to control the remote phone. When the phone is lost, the user can use the trusted contact to send SMS to lock it. Also, the sensors with daemon process mechanism prevent the system from being maliciously closed or uninstalled. This increases the robustness of the system. Finally, the system adopts SIM detection sensor to prevent malicious replacement or removal of SIM card. Test results show that our system has good robustness and low resource consumption. However, something more can be done based on our works. We use process communication mechanism to prevent the system.

REFERENCES

[1] Kyungwhan Park, Gun Il Ma, Jeong Hyun Yi, Member, IEEE, Youngseob Cho, Sangrae Cho, Sungeun Park Soongsil University, Electronics and Telecommunications Research Institute, KSIGN

Corp., "Smartphone Remote Lock an d Wipe System with Integrity Checking of SMS Notification," IEEE International Conference on Consumer Electronics (ICCE) 2011

[2] Yosi Kristiana, Hendrawan Armantob, and Michael Fransb, "Utilizing GPS and SMS for Tracking and Security Lock Application on Android Based Phone ," Available online at www.sciencedirect.com 2012

[3] Weizhe Zhang, Hui He, Qizhen Zhang, and Tai-hoon Kim, " PhoneProtector: Protecting User Privacy on the Android-Based Mobile Platform ," Volume 2014,Article ID 282417, Hindawi Publishing Corporation, 2014.

[4] Swapnil Waghmare, Madhumita Chatterjee, and Satish L. Varma, "Authentication System for Android Smartphones ",Volume 87 – No.5, International Journal of Computer Applications (0975 – 8887), February 2014

[5] William Enck, Damien Octeau, Patrick McDaniel, and Swarat Chaudhuri," A Study of Android Application Security " , Systems and Internet Infrastructure Security Laboratory Department of Computer Science and Engineering.

BIOGRAPHIES

[1] Prof. Khushal Khairnar, Dept of Information Technology, MMIT, Pune
[2] Mr. Pratik Parkhe, Dept of Information Technology, MMIT, Pune
[3] Mr. Piyush Kharade, Dept of Information Technology, MMIT, Pune
[4] Mr. Chaitanya Kakade, Dept of Information Technology, MMIT, Pune
[5] Mr. Krishna Kokate, Dept of Information Technology, MMIT, Pune