A CONTEXT BASED APPLICATION DEPLOYMENT IN ANDROID FOR DATA SECURITY

Pragya Gajbhiye
Soit, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India

Varsha Sharma
Soit, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India

Vivek Sharma
Soit, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India

Abstract

Android has covered the complete market all over the world. Every second person is using Android operating system over their phones. All the personal data resides inside this phone. So, security of this data is of utmost importance. But applications access whole data stored in phone irrespective of their context. So in this paper a context dependent technique for data security in Android has been proposed. It is compared with previous context aware techniques and it is found that proposed technique outperforms previous one.

Key words: Android, Context Aware Application, RSA.


http://www.iaeme.com/IJARET/issues.asp?JType=IJARET&VType=8&IType=4

1. INTRODUCTION

The versatile market has extended at an extremely huge rate. The year on year development was logged at 35% toward the finish of 2010 as indicated by a report. Among cell phones, cell phones have gotten unbelievable reception. The reason of the expanded interest for cell phones is the tremendous accessibility of uses that can be downloaded and introduced effortlessly on cell phones. For instance, Google Play [1] of Google contains numerous Android applications to download. Alongside the merchant gave programs, outsider applications are likewise present on these commercial centers. The aggregate application downloads have crossed the 25 billion check.

The expansion in the levels of comfort and components of cell phones causes a noteworthy development in the quantity of Smartphone clients. For instance, telephone comprises of call log with data about the calls, the client has made and gotten, an address book that contains the
clients contact, perusing history, photographs and so on. As these are on the whole private data, it must be guaranteed that they don't fall in the wrong hands. To shield individual data of client from applications of cell phones which are malevolent, another method of security is required in cell phones. This new mode can alter an applications access to client's close to home data. The client will be able to control the entrance of individual data to the application in a way that, which data can be gotten to and which can't. Further, the client ought to have run-time control to alter the beforehand given consent.

Despite the fact that numerous forms of Android have been discharged up until now, of which the most recent is 4.0 named as Ice Cream Sandwich, the center engineering continues as before. It is a layered engineering with the layers comprising of the base linux part, libraries, a virtual machine to run applications and an application system to deal with the applications.

Platform
The Android Operating System depends on the Linux Kernel. It utilizes the form v2.6.x of the Kernel tree [14]. The Kernel is a mediator of the product stack and the equipment. Like ordinary linux frameworks, it is utilized for process, memory administration and so forth.

![Android Linux kernel](image)

**Figure 1** Android Linux kernel

Center Libraries
The different parts inside the Android OS utilize libraries written in C/C++. These libraries are additionally utilized by the application system. These are not the standard libraries display in ordinary linux frameworks, yet a particular rendition of a subset of them. For instance, there is no glibc display [9].

Android Runtime
An arrangement of Java based libraries for use at runtime are additionally accommodated supporting usefulness much the same as the center libraries introduce in the Java programming dialect [16]. A Java Virtual Machine called Dalvik, altered for use in Android is additionally given.

Virtual Machine
The Java class documents are been changed over into .dex organize by the Dalvik Virtual Machine. Each example of Dalvik VM runs one Android application. A different Linux process is assigned to each occasion. Various occasions can keep running on one gadget. Procedures
detachment, memory administration execution and threading are rely upon the Linux Kernel [2]. The Dalvik VM is planned, in order to run ideally on a machine with moderate CPU, less RAM, and without swap space OS, while having the extra power requirement of running on the battery [2]. It is upgraded to weed out excess that a normal .bump_file has, while coordinating each current Java class document. It does as such, by effectively connecting consistent data as indicated [2]. Accordingly, memory spared by means of insignificant reiteration, understood writing and marking.

Dalvik VM is enlist based. This aides in dodging superfluous memory get to and getting additional significance per direction. Subsequently, a similar work is being finished with 30% less directions and 35% fewer code units [2].

Application Framework
Application structure layer de_ne s how applications are assembled and how they act. The Android application structure incorporates [1]:

- UI for application which is worked by sees including records, networks, content boxes, catches, and implanted web program.
- An arrangement of Content Providers which empower applications to share their information and to get to information from different applications.
- Access to assets is given by a Resource Manager.
- A Notification Manager empowers the show of custom cautions.
- An Activity Manager gives the regular route stack to all application and furthermore oversee application life cycle.

Java programming dialect is utilized to build up the applications in Android. Assemblage of code is finished by different Android devices with the assistance of other data in a document with an .apk addition. All the code in a solitary .apk _le is thought to be one application and is the document that gadgets running on Android, use to introduce the application [1]. One can distribute the application over the Android Market so it can be circulated to different clients also. An application can just utilize the assets, that it has authorizations to do as such.

Application Components
Android application is comprised of various Application parts. These segments are the distinctive passage focuses for the framework. There is no strict or set arrangement characterized overseeing where a part can really enter the framework. There isn't even a principle () work characterized for application forms. For client, every single segment doesn't fill in as passage point. Segments can be between subordinate, yet every segment has a particular part and has its own particular element. The application’s general conduct is relying upon these building squares i.e segments.

2. LITERATURE REVIEW
The developing of setting mindful administrations has quickened scholastic enthusiasm toward setting mindful strategies particularly for those situations which put cell phones under the spotlight. Some imperative commitments in this field have been coordinated through the meaning of strategy models reasonable for contextware situations [3]. Get to control frameworks ought to have the capacity to help and see any new setting data keeping in mind the end goal to address get to control prerequisites. To make this conceivable, Cheaito et al.
displayed an extensible get to control arrangement in light of XACML making it ready to see new characteristics information sorts including the capacities that are utilized as a part of the approach to assess clients' solicitations [4]. Another fascinating work is [5] where Li et al. proposed a get to control strategy show based on setting and part that can be proper for web administrations. The model takes setting as the inside to characterize and perform get to control strategies. It utilizes the settings of client, condition and asset to execute dynamic parts task and oblige the approval choice. Another intriguing work, which tends to strife issues in setting mindful arrangements, is [6] where creators propose a structure where approval for a specific get to ask for is chosen progressively in light of setting data. They additionally bolster dynamic clash determination where current arrangement is picked at run time based on setting data. At long last, the developing of setting mindful administrations and relative portable applications is making it important to update genuine versatile OS's security systems. Current cell phone frameworks barely enable clients to determine the conduct of the applications through the nearness of logical data, Conti et al. propose CR’ePE [7], an Android security augmentation which enables setting related arrangements to be set (even at runtime) by both the client and approved outsiders, locally or remotely. Approaches which can be characterized in CR’ePE depend on the status of factors detected by physical (low level) sensors, similar to time and area. Peak [8] is an augmentation of the Android consent structure which enables clients to determine itemized runtime requirements to confine the utilization of touchy assets by applications. The client can indicate his requirements through a basic interface of the broadened Android installer presented by creators called Poly. Inside Android, Google give an alleged "Gadget Policy for Android" condition that permits to set strategies to implement utilization of PIN or secret key and screen bolt on the gadget and enable a head to wipe the gadget remotely. This structure has the main point of anticipating physical access to data on a gadget which is not under direct client control. It has no connection with the conduct's control of a specific application. In the field of changes to the security of standard working frameworks, it is surely understand the instance of Security Enhanced Linux [9] where over Linux an altered security structure was acquainted with improve the capacities of the working framework. SELinux turned out to be a piece of the standard Linux dispersion once embraced by the group. This article goes a similar way, presenting a more highlighted arrangement framework on an officially accessible stage. Correspondingly get to control for web applications is talked about in [10] and particular arrangements are proposed. In [11] security answers for Short message administrations are examined alongside security conventions for Android working framework. In [12, 13] a security structure for Android and USB investigating for security is proposed. A comparable cloud based security answer for Android is proposed in [14]. PassDroid [15] and SecureDroid [16] are two secure Android working frameworks which are proposed and contrasted and existing design.
<table>
<thead>
<tr>
<th>Technique</th>
<th>Description</th>
<th>Limitations</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short Message Based Security</td>
<td>This work tried to reduce the window time for stealing confidential data from mobile phone. The idea is to develop a Short Messaging Service (SMS) based system and a web based application that interacts with database and remotely wipe out the access rights to enterprise application of a mobile phone and minimize the data theft window.</td>
<td>Short message based security incurs extra expense which has to be borne by the user.</td>
<td>[11]</td>
</tr>
<tr>
<td>Distributed Android Security Framework (DASF)</td>
<td>This paper introduces a Distributed Android Security Framework (DASF). DASF allows servers to dynamically impose security policies by using an application-layer message protocol implemented in the system.</td>
<td>This technique provides a distributed solution to provide security in which lots of servers communicate with each other. This incurs a communication overhead.</td>
<td>[12]</td>
</tr>
<tr>
<td>Security for USB Debugging</td>
<td>This research analyzes its protection effects on ADB based attacks and found that the new feature cannot provide sufficient protection when the host used to connect with Android devices has been compromised.</td>
<td>The majority of security issues arise when android is connected with computer through USB. So this scheme has major advantages but it also slows down the USB transfer.</td>
<td>[13]</td>
</tr>
<tr>
<td>Cloud Based Android Security</td>
<td>This paper propose a cloud based security system for collection, visualization, analysis and correlation of application logs, statistics and determining abnormal application and network behavior on the device.</td>
<td>Cloud computing provides a pay as per use service so it is costly.</td>
<td>[14]</td>
</tr>
<tr>
<td>PasDroid</td>
<td>Authors created a real-time security scheme on Android, called PasDroid, to trace dubious data flow and to alert the users with information for judging if a transmission should be allowed or not.</td>
<td>It tracks all the incoming and outgoing data out of the android cell phone. So it slows down the cell phone and results in enhanced battery consumption also.</td>
<td>[15]</td>
</tr>
<tr>
<td>SecureDroid</td>
<td>Authors propose SecureDroid: an extension of the Android security framework able to enforce flexible and declarative security policies at run-time, providing a fine-grained access control system.</td>
<td>This is an extension of Android operating system. And it calls for a huge challenge to replace the existing OS by this new OS.</td>
<td>[16]</td>
</tr>
</tbody>
</table>
3. PROPOSED ALGORITHM

CONTEXT BASED APPLICATION CONTROL IN ANDROID

STEP 1: ANDROID SIMULATION AND KEY GENERATION USING RSA

Algorithm Android Simulation & Key Generation

Step 1: Creation of Android environment
A. Android script creates a directory named ‘Android’ in ~/Music directory.
B. Further subdirectories are created for contacts, messages and gallery.

Step 2: Creation of secret keys using RSA using openssl
A. Android executes the key generation command
B. A pair of keys are generated for every subdirectory.

Step 3: End of key distribution
A. Both the keys are displayed on the user’s terminal.

Step 4: Data encryption using keys
A. All the data is encrypted in each subdirectory using respective keys.

In the scheme Android environment is simulated in ~/Music directory. In this directory a directory for Android named Android is created and inside it various subdirectories for contacts, messages and gallery are created. Using RSA algorithm of openssl package we have generated keys for all the subdirectories so that data of individual subdirectory can be encrypted using that key.

```
echo -n "Android content is stored at ~/home/hadoop/Music/Android "
echo -n ""
ssh-keygen -t rsa -f ~/Music/Android/contacts/id_rsa1
cat ~/Music/Android/contacts/id_rsa1
openssl rsa -in ~/Music/Android/contacts/id_rsa1.pub.pem
```

```
cat ~/Music/Android/messages/id_rsa2
```

```
cat ~/Music/Android/gallery/id_rsa3
```

STEP 2: APPLICATION INSTALLTION IN SIMULATED ANDROID

Algorithm for Application Installation

Step 1: Enter name of application to be installed
A. User sends the name to Android play store
B. Application is downloaded from play store which is simulated here using simple directory creation.
C. A subdirectory is created inside Android directory for application name specified.

Step 2: Context creation for Application
A. Android asks for context to be limited for application.
B. User enters the context name.
Step 3 : Key distribution for specified context  
A. After successful installation of application and sub directory creation.  
B. Keys are copied from the context sub directory according to context specified.

After successful creation of Android environment and key generation using RSA; in this step context is fixed for applications. During application installation Android asks for the context to be set for the application. User enters the context options and after that keys are distributed for the specified context.

```
Figure 2 Key generation for each operation

echo -n "Applications will be installed in ~/home/hadoop/Music/Android/ "
echo -n ""
read name
mkdir ~/Music/Android/$name
read option
s='contacts'
s2='messages'
s3='gallery'
if [ $option=$s ];
then
    cp ~/Music/Android/contacts/id_rsa1 ~/Music/Android/$name/
fii
if [ $option=$s2 ];
then
    cp ~/Music/Android/messages/id_rsa2 ~/Music/Android/$name/
fi
if [ $option=$s3 ];
then
    cp ~/Music/Android/gallery/id_rsa3 ~/Music/Android/$name/
fi
```
STEP 3: EXECUTION OF INSTALLED APPLICATIONS ACCORDING TO CONTEXT

ALGORITHM FOR APPLICATION EXECUTION

Step 1: User enter the application name which is to be executed
A. User enters the name of application.
   B. Android checks if application is installed or not.
   C. If found Android asks for the context in which application should run.

Step 2: Context verification from user
A. User enters the context in which application was installed.
   B. If context is verified then private key is fetched.

Step 3: Decryption of data
   A. After successfully fetch of key.
   B. Data is decrypted for that context sub directory.

Step 4: Application is executed
   A. If data is successfully decrypted content is displayed on terminal.
In Application execution user is required to enter application name and context in which it was installed. If verified data for that sub directory will be decrypted and provided to the application.

```
if [ $option=$s ];
then
    echo -n "decrypting data using private key of application ">
    cat ~/Music/Android/contacts/cipher.txt | opensslrsautl -decrypt -inkey
~/Music/Android/$name/id_rsa1
fi
```

### 4. RESULTS

The graph shown below holds the comparison of execution time for sub processes in proposed approach for context based application access control in Android.

Following sub processes are involved:

- Time to install the application in simulated environment.
- Time to encrypt data in Android local memory.
- Time to decrypt to data according to key of application based on context.

The time is recorded in milliseconds in Linux environment.
A Context Based Application Deployment In Android For Data Security

5. CONCLUSION
In this paper a context based access control mechanism for Android application has been proposed. The proposed approach is found having less overhead in terms of delay. Key generation and management is done for providing context based access.
REFERENCES


