ISSN No: - 2456- 2165

Smali Code Based Protection for Android Package

D. Suganthi[1], Jerin Paulose[2], N. Girikumar[3], M. Sundaramoorthi[4], Tenzin Tsultrim[5]
[1]Professor, Department of CSE, RVS College of Engineering and Technology, Coimbatore
[2][3][4][5]Final Year, Department of CSE, RVS College of Engineering and Technology, Coimbatore

***Abstract*— The Android Device is used widely throughout the world. The android is having vulnerabilities that can be exploited by a hacker. Getting the root privileges in android is common of all. Additionally, a hacker can easily recover source code from android package file using reverse engineering technique. This leads to disclosure of sensitive information in android package. Hackers will further recompile the original source code along with the bug to generate fake malicious android application. However, the reverse engineering cannot be prevented. The only way is to protect the data in android package. So, in order to protect the sensitive information in android package a Smali code based protection algorithm for android package is proposed over here.**

Keywords*—*Smali Code; Android Security; Reverse Engineering; Android Package; Apktool;

I. INTRODUCTION

Android is a Linux-based mobile operating system designed for touch screen based smartphones and tablets. The first official version of android is developed by Open Handset Alliance (OHA), a consortium of technology manufactures along with google in November 2007. Later the development was taken over by Google and the source code for Android itself is released mostly under the open source Apache License. Google have also developed an online store known as google playstore to publish and access 3rd party android applications for free by any android user.

The android applications are written mostly in java though other languages can be used and the compiled output is known as android package file with file extension Apk. The user can install android applications using google playstore or manually running the installer. The applications will run inside Dalvik's virtual machine (DVM), a simplified version of java virtual machine (JVM) embedded with android[1].

In the past years, android came through a large number of security attacks such as system rooting, adware's, web-based threats, SMS Trojan virus, malwares etc. Hackers have succeeded in implementing malwares such as Zeus botnet, a username and password stealer for banking applications[1]. DIY Android Trojan is another one used to steal contacts, messages, GPS locations etc. that breached privacy of android users.

The android package is vulnerable to reverse engineering technique. The hackers can use the above technique to recover the original but almost the same java source code of compiled applications. Furthermore, the bytecode in android package can be converted to another language known as Smali, providing with 100% accuracy. This process is known as backsmaling. The hackers will further modify the Smali code and inject the malware along with it. This modified code is recompiled back to bytecode and repacked as original android application. The Victim will install the application in his device unaware of the bug along with it. This gives the hacker exclusive control of the device.

Since the original source codes can be viewed by hacker transparently, the sensitive information's embedded in android application such as cryptographic keys, network locations, passwords, application license data etc. is disclosed. So, in order to protect the sensitive information in android package a Smali code based protection technique can be used.

II. BACKGROUND AND RELATED WORK

A. Android Device Platform

The Android uses modified Linux kernel, which is highly optimized to run on portable devices. The kernel itself can handle device drivers, power management, memory management, device management and resource access in android.

The Android Software stack is mainly categorized into five parts. They are Linux kernel, native libraries (middleware), android Runtime, application framework and applications[5]. The Android operating system consists of inbuilt system applications and Java core libraries running under the Java-based object oriented application framework and the Dalvik's Virtual Machine (DVM).

The fundamental components in android are activities, views, services, content providers, intents, fragments and AndroidManifest.xml. The AndroidManifest.xml file contains the information about the application access permissions. These access permissions are verified at the time of installation and initialization of each application.

The Android Runtime provides the most important part of android system, i.e. Dalvik's virtual machine (DVM). Dalvik's virtual machine uses core functions of Linux such as multithreading and memory management that enables each android application to run its own process. The DVM will run the java bytecode in its virtualized environment.

ISSN No: - 2456- 2165

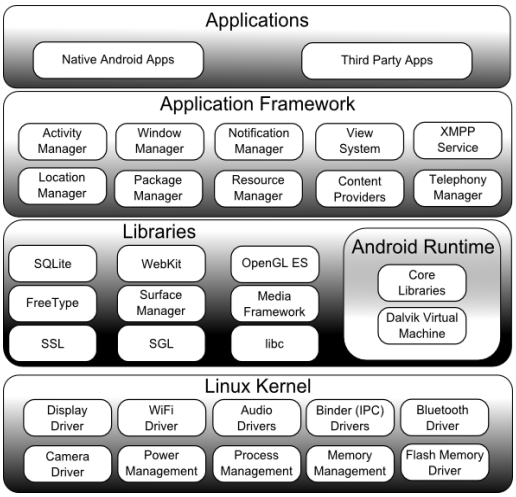

Fig.1. Android Device Stack

For multimedia Support, the Android Operating system provides 2D and 3D graphics, common audio, video codecs. It also supports multi-touch input, motion sensors, GPS receiver and Wi-Fi connectivity.

B. Reverse Engineering

Reverse Engineering is the method of recuperating the data or design from a product published by third person. The hackers will use this method to decompile the android package file. The decompiled file contains java source codes, manifest file, res folder (contains UI design xml files) and library files[2].

Apktool is a free open source tool, widely used to decompile and recompile android package files[3]. The dex file in android package holds the bytecode that executes in Dalvik's virtual machine (DVM). The bytecode can be converted to a different programming language known as Smali, with hundred percentage accuracy[10]. Hackers will add bug to the Smali code and recompile it to make new malicious application.

III. PROBLEM AND MOTIVATION

The Android package can easily be bugged by a hacker using reverse engineering tools, mainly Apktool. This tool helps the hacker to decompile the android package file and also reveals the sensitive information inside the file.

However, the decompilation of android package cannot be prevented. So, the java class names, strings, integer constants can be viewed in clear text by hacker[4]. This provides the essential information for the hacker, to create a modified version of application or to bug it.

So, it is necessary to protect this information in android package from unauthorized access. This will make the hacker harder to decrypt the data inside the package[6].

IV. PROPOSED APPROACH

Decompilation of the android package is done by using Apktool. This gives the source codes in Smali language[7].

To protect information in Smali code, we proposed a new procedure denoted below. This scramble's the class names and strings in Smali code which further recompiled into new protected android package file.

1) To Protect Default classnames;

a) Store all the file names (without extension) in folder to a table i.e. table_class(classname, hash_classname).

b) hash_classname for each row is calculated by selecting first 6 characters of md5(classname) excluding numbers.

c) update all the filenames with corresponding hash_classname.

d) Open each file in folder and replace every occurance of classname in table_class to hash_classname

2) To Protect Strings in smali code;

a) Open each file one by one.

b) Read each string to variable temp_str.

c) Apply temp_str = string_enc(temp_str).

d) String_enc(temp_str)

- *Return SHA64_enc(temp_str)*

e) Replace original string with below code

- *String_dec(temp_str)*

f) Add below function to each smali file

- *String_dec(temp_str)*

Return SHA64_dec(temp_str)

Classnames are not needed to be decrypted since original classnames are not used during runtime. Strings are encrypted using SHA64 bit algorithm and for decryption a function is added to each Smali file.

V. EXPERIMENTAL SETUP

To demonstrate the experiment, the android device is configured with allowing installation of applications from modified sources in settings[7]. A new android application with name as 'test' is also developed. Further for decompiling and recompiling of android package, a tool known as Apktool is used[13]. For adding protection to Smali code, a python based tool ProtectSmali.py is also developed, using the above-mentioned algorithm.

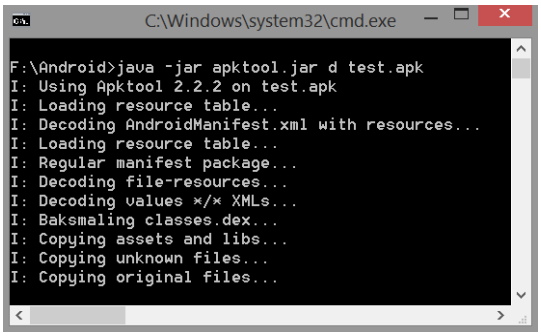

Fig.2. Android Package Decompilation

ISSN No: - 2456- 2165

A. *Decompiling Android Package*

The Android Package is decompiled using an open source Apktool. The Decompilation of test.apk is shown in figure 2. The output is automatically saved in a folder with same application name.

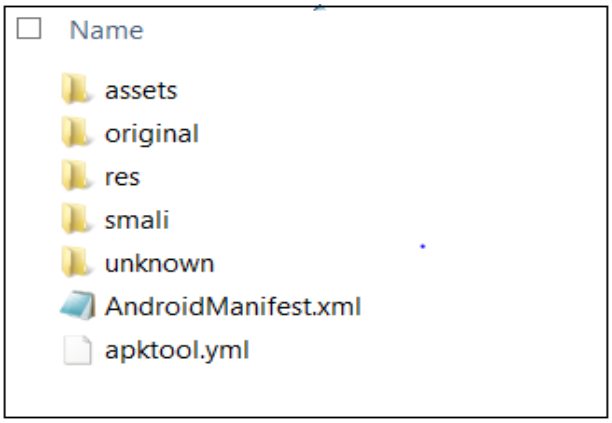

Fig.3. Apktool Output Folder Structure

The Decompiled output folder Structure of 'test.apk' is shown in figure 3.

B. *Smali Code Based Protection*

To provide security to Smali code, a tool ProtectSmali.py is developed. It is developed using python. This tool will modify the Smali source file, by adding the protection features specified in algorithm. The arguments to ProtectSmali.py script is the folder where the decompiled Smali code exists. The execution of ProtectSmali.py is shown in figure 4.

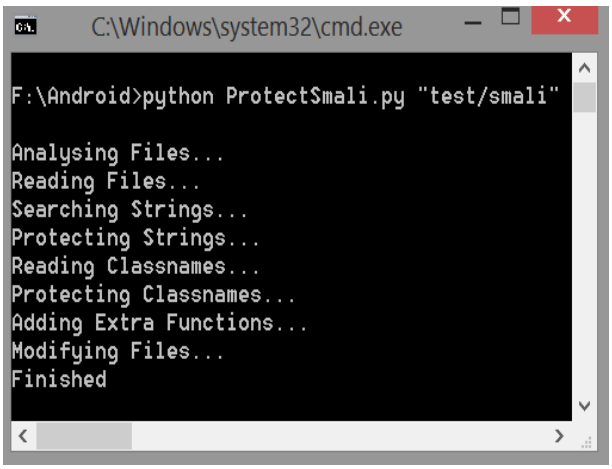

Fig.4. Running ProtectSmali.py on 'test/smali'

C. *Recompiling Android Package*

To recompile the folder to Android package, the Apktool is used. The arguments for Apktool are, for recompilation 'b' and input folder 'test'. The output file 'test.apk' is generated by Apktool in same folder. The figure 5 shows the recompilation process.

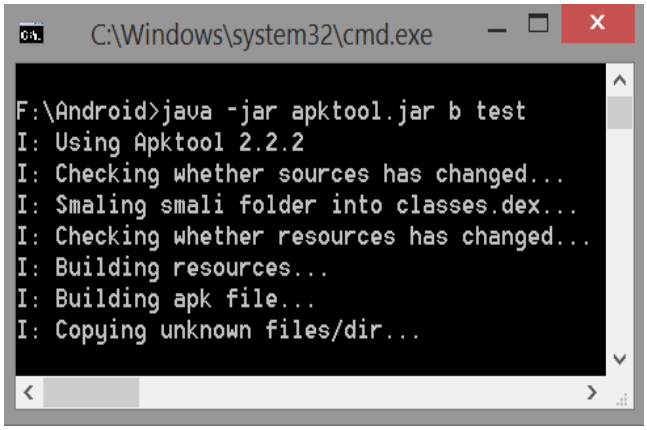

Fig.5. Android Package Recompilation

VI. EXPERIMENTAL RESULTS

The figure 6 shows the Smali code before applying protection and figure 7 after applying protection. it is clearly notified that in figure 6, the value of string variable key is 'abcd@1234'. But in fig. 7 value of key variable is modified to 'hjkhqw4yqk7jh'.

```
.line 3070
const-string v1, "abcd@1234"
```

Fig.6. Before Applying Protection To Smali Code

Whenever, a hacker tries to decompile the protected android package, he will be only able to get the modified strings[9]. This makes the hacker harder to understand the application.

```
.line 3070
const-string v1, "hjkhqw4yqk7jh"
```

Fig.7. After Applying Protection to Smali Code

VII. CONCLUSION

The Android Package can be decompiled easily by a hacker. This process reveals the sensitive information's such as cryptographic keys, URL's etc. embedded in package. So, it is essential to protect the sensitive information.

The Smali code based protection is applied to protect original information, such that strings are scrambled and cannot be understandable by hacker. The original strings are reconstructed at the time of execution only.

REFERENCES

[1] Khulood Al Zaabi, "Android Device Hacking Tricks and Countermeasures" IEEE International Conference on

Cybercrime and Computer Forensic (ICCCF), 2016, pp.1-10.
[2] Yang Cuixia, Zuo Chaoshun, Guo Shanqing, Hu Chengyu, Cui Lizhen, UI Ripping in Android: Reverse Engineering of Graphical User Interfaces and its application.
[3] Android-Apktool, http://code.google.com/p/android-apktool/
[4] Chell, D., Erasmus, T., Colley, S., & Whitehouse, O. (2015). The Mobile Application Hacker's Handbook.
[5] Android device platform, https://developer.android.com /guide /platform/index.html.
[6] Wu, X., & Li, X. (2013, October). Hack android application and defense. In Computer Science and Network Technology (ICCSNT), 2013 3rd International Conference on (pp. 676-680). IEEE.
[7] Gupta, A. (2014, March). Learning Pen testing for Android Devices (1st ed.).
[8] Packtpub. (2015). Practical Mobile Forensics. Retrieved March 10, 2017, from https://www.packtpub.com/packtlib/book/ApplicationDevelopment/9781783288311/pref05
[9] Casey, E., 2011, Digital evidence and computer crime: Forensic science, computers, and the internet, Academic press
[10] A. Memon, I. Banerjee and A. Nagarajan. GUI Ripping: Reverse Engineering of Graphical User Interfaces for Testing. Working Conference on Reverse Engineering, 2003: 260-269.
[11] Wu, X., & Li, X. (2013, October). Hack android application and defense. In Computer Science and Network Technology (ICCSNT), 2013 3rd International Conference on (pp. 676-680). IEEE.