

LAB

7

Static Analysis on Android Malware

By the end of this section of the practical, you should be able to:

- Explain the behavior of android malware
- Using suitable static analysis tool to investigate android malware
- Identify malware trace from static analysis

7.1 Introduction

Malware analysis is a process which is used on malware in order to understand its operation, code structure and its functionality. Android Market is where the application developers can market applications to the users. As android is an open mobile operating system it allows flexibility for sharing applications compared with Apple iOS. There are advantages as well as disadvantages with this method; the advantage is that any application developer can list his application in the Android Market thus providing variety of options to the common users. The main drawback in this approach is it allows lot of attackers to develop malicious software which can compromise systems and can gain user information.

The Android operating system allows user to decide whether an application is malicious or not. [10] That means users must analyse and confirm its identity and if found malicious need to report and thus will ensure its removal from the Android Market. For

example, a malware application was released in the name droid09 which allowed users to carry out banking activities. User need to provide the account information details and it would tunnel communication to the bank. But actually that malware used to provide a browser login to the bank homepage the credentials were sent to the attacker. So, an Android application needs to show the permissions during the installation so that user can judge whether to install or not. But attackers have become more sophisticated they find different ways to enter the system by exploiting the vulnerability or by just tricking the common user.

Among The android malware behavior are:-

- Stealing device and user credential information.
 - Most of the android malware are collecting information regarding International Mobile Equipment Identity (IMEI) number, International Mobile Subscriber Identity (IMSI) number, GPS Location, Phone number, SDK version and Installed package.
 - Android malware also captured user activities such as SMS send or received and call out or in duration.
- Communicate with Command and Control (C&C) server.
 - Android malware have the intention to communicate to a C&C server similar to Botnet. Communication can be made through HTTP or SMS.
 - This connection are made for sending captured information from the device and also for updating the malware package or receiving any other malicious information for instance SMS premium number or any other malicious URL.

- Sending premium rate SMS and spam.
 - Android malware has the ability to automatically send SMS without the user knowledge.
 - This can be used as a spam or gathering illegal income by charging user when the SMS is send to the premium number
- Search engine Optimization.
 - Android malware can have an automatic script to visit ad, for increasing the number of visitor to a website or for the purpose of generating a charge per click without having actual interest in the target of the ad's link. Even though, it seem like it does not cause any harm, this activity can increase mobile data consumption especially to the user who are subscribing for the mobile broadband it will cause them extra charges.
- Updating and download package.
 - Android malware can used the communication made to the C&C server to update the existing package into a more malicious intention or even download a new package and installed it automatically in the user devices.
- Draining resources.
 - Android malware application can executed a malicious payload that can utilize all the disk storage or memory (RAM) quotas and hogging the CPU.

Android application package (APK) is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware. To make an APK file, a program for Android is first compiled, and then all of its parts are packaged into one file. An APK file contains all of that program's code (such as .dex files), resources, assets, certificates, and manifest file.

An APK file is an archive that usually contains the following files and directories:

- MANIFEST.MF: the Manifest file
- lib: the directory containing the compiled code that is specific to a software layer of a processor
- res: the directory containing resources not compiled into resources.arsc (see below).
- assets: a directory containing applications assets, which can be retrieved by AssetManager.
- AndroidManifest.xml: An additional Android manifest file, describing the name, version, access rights, referenced library files for the application. This file may be in Android binary XML that can be converted into human-readable plaintext XML with tools such as AXMLPrinter2, android-apktool, or Androguard.
- classes.dex: The classes compiled in the dex file format understandable by the Dalvik virtual machine
- resources.arsc: a file containing precompiled resources, such as binary XML for example.

Task 1



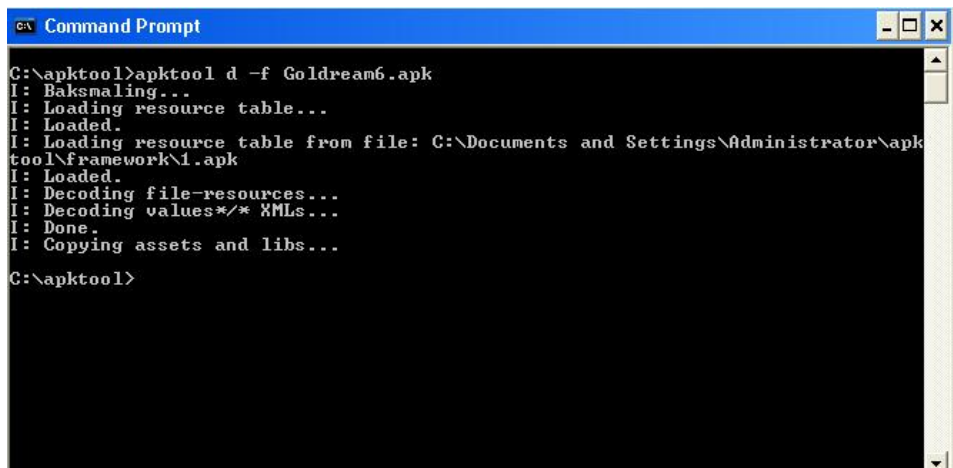
Static analysis on Android Malware

1. Analyse a sample of android Malware (example1.apk) with static analysis approach.
2. The tool needed in this task is given in a win7 virtual machine. Among the tool use are :-

Apktool

<https://ibotpeaches.github.io/Apktool/>

Apktool is the disassembler used in this project for analysing Android Malware. It can decode the malicious code to its original code also we can modify the code and recompile using Apktool . It uses Baksmali for disassembling and Smali for assembling the code. This dex format is used by Dalvik VM. Baksmali and Smali are the Icelandic names for “Disassembler” and “Assembler”.



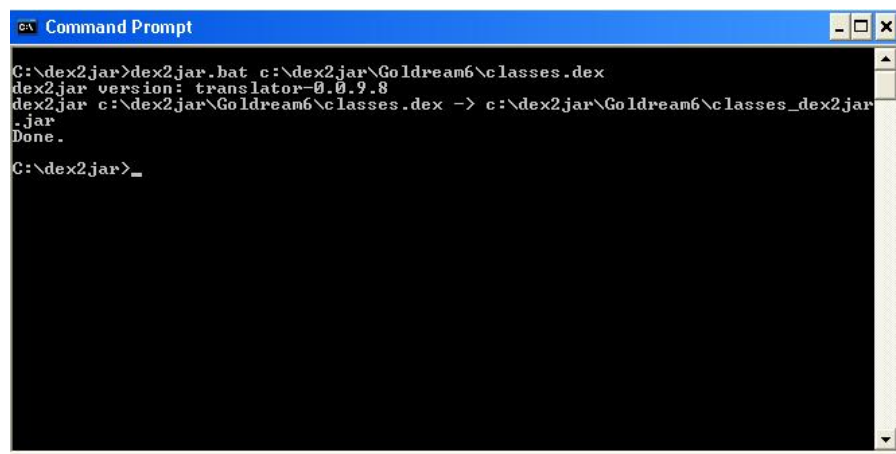
```
C:\> Command Prompt
C:\> C:\apktool>apktool d -f Goldream6.apk
I: Baksmaling...
I: Loading resource table...
I: Loaded.
I: Loading resource table from file: C:\Documents and Settings\Administrator\apktool\framework\1.apk
I: Loaded.
I: Decoding file-resources...
I: Decoding values*/*.xmls...
I: Done.
I: Copying assets and libs...
C:\> C:\apktool>
```

This tool requires a Java runtime Environment to run.

Dex2Jar

<https://github.com/pxb1988/dex2jar>

By using 7zip to unpack the .apk file and of the file obtain is the classes.dex. Applying the dex2jar to the classes.dex will give a .jar file. Generally dex2jartool, is used to convert the dex code into *.jar Java file.



```
C:\dex2jar>dex2jar.bat c:\dex2jar\Goldream6\classes.dex
dex2jar version: translator-0.0.9.8
dex2jar c:\dex2jar\Goldream6\classes.dex -> c:\dex2jar\Goldream6\classes_dex2jar.jar
Done.
C:\dex2jar>_
```

3. From APKTOOL you can acquired the package name of the android app and also information such as list of Permission required by the app. This entire information can be obtained from the manifest files
4. Dex2jar and JD-Gui Provide you with the .jar that contain the java class. This can provide you with the API and Class used by the sample.
5. Refer <http://docsdrive.com/pdfs/academicjournals/rjit/2014/325-341.pdf> for detail information on Android Malware Detection.

JD-GUI

JD-GUI is another tool, which is used to view those *.jar files. It provides a GUI which can load all the packages embedded in the jar file and lists the *.java code.



Task 2



Static analysis on Android Malware

1. Choose three sample of android malware from the sample folder. By using the above task do a report on the analysis you done on the sample.
2. Your report should consist
 - The list and screenshot of each step taken during the analysis.
 - The finding which include the the permission list, package name and possible malicious API.