BUILD A DYNAMIC MALWARE ANALYSIS SYSTEM FOR ANALYZING ANDROID APPS

MSc. In information Technology - SLIIT
By Rukmal Hewawasam
02-08-2016

Abstract

The mobile phone security industry analysis reports in year 2012, that they have found 26850 of malware infections on Android. Stealing confidential information (i.e IMEI numbers, mobile numbers or Contact details) and send SMS messages without user permission are the most damaging attacks done by the malware in the past. Now it is not like that, the smart terminals perform the same functionality like computers. Data transmission, online shopping using credit cards are few examples. Therefore the cost of the vulnerability is very high than the past couple of years.

In such situation identifying malware is a challenge because thousands of apps are developed daily and there are millions of apps in the app stores. Antivirus companies are failed to identify threats in new viruses because of this. And currently, detection on mobile malware is based on traditional computer virus detection method based on signature or behaviors.

Abstract...

DroidBox is tool developed to offer dynamic analysis of Android applications. It is well known malware analysis sandbox and it will dynamically analyze the Android apps. DroidBox, authored by Patrick Lantz, is a sandbox for the Android platform. It focuses on detecting information leaks that were derived from performing taint analysis for information-flow tracking on Android trojan applications. DroidBox can be used to get an overview of malicious activities triggered by the app. Unfortunately this tool is built in Dalvik Runtime environment and currently it become obsolete since there is a new Android ART is in the market.

Currently, ART is available on a number of Android 4.4 devices, such as the Nexus 4, Nexus 5, Nexus 7, and Google Play edition devices. But droidbox not works with these devices. In this project I'm going to analysis **how to use Droidbox for analysing the malware** in ART and what are the possible **solutions to compile Droidbox** to fix ART issues.

Literature review

There are mainly two kinds of Android malware detection techniques used:

1. signature-based (static) 2. behavior-based(dynamic) detection

Static Analysis: it is a reverse technology, extract Android package and decompile **DEX** file to get smali file to analysis the malware and the potential threats by detecting the sensitive API in the source code, and determines whether there is a sensitive data leakage.

Literature review ...

Static Analysis Tools:

1. **N-gram** - is a software program that uses machine learning based algorithm to detect the that given application is malware.

2. **Android-apktool** is an reverse engineering tool it can decode the resources.

Disadvantages: Code obfuscation by tools like **ProGurd**

Obfuscation tool detects and deletes useless class, field, methods and attributes, and deletes useless annotation to make the byte code optimized. The source code has poor readability and greatly increases the difficulty of code analysis.

Literature review ...

Dynamic analysis: Analysing the logs, accessing details and identify malware.

I.e. Use smali files generated in the static analysis to add the corresponding log generated code after sensitive API function, and then repackage application and signature, which will install repackaged application to the modified simulator and the API will analyze all generated sensitive log.

Literature review ...

Android Operation system Architecture.

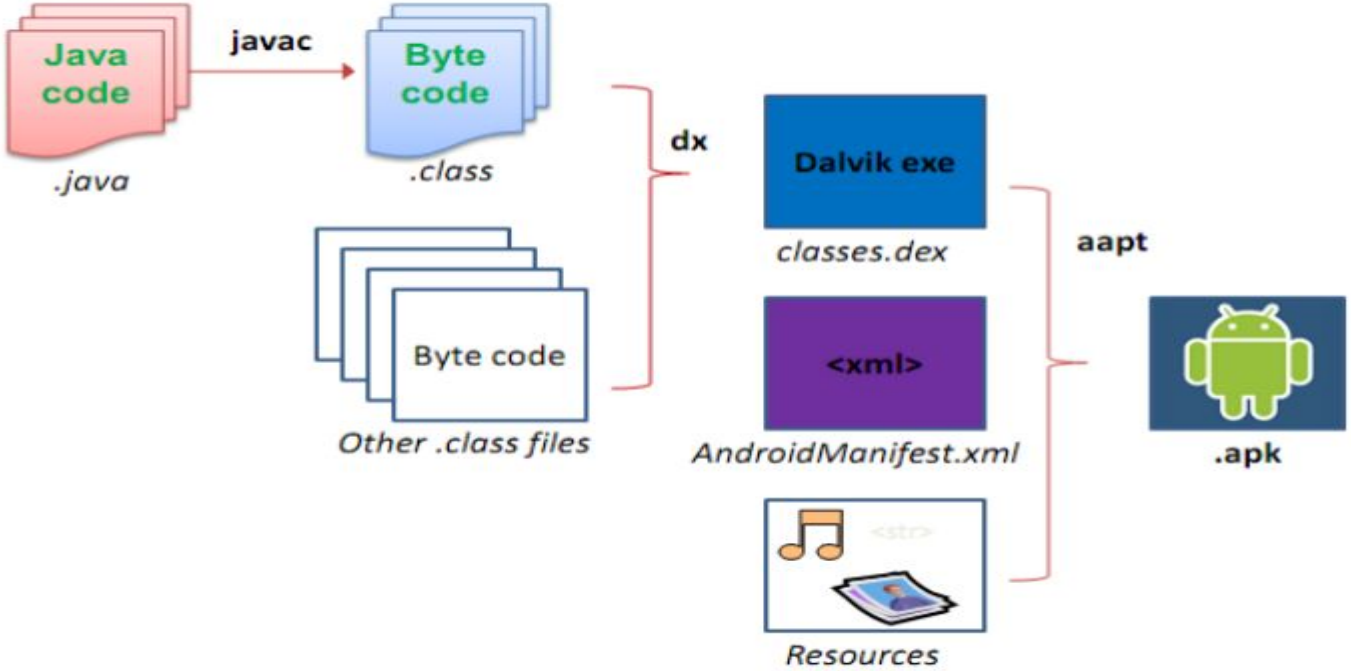
Android is based on **linux kernel**, and it is highly optimized for mobile operating systems, and it made small as possible. On top of the kernel there is an **Android runtime** etc. It up to the Original equipment manufacturer (**OEM**) to customize these packages according to the their device. Android Runtime include **Core Libraries**, and there are **two separation for Android 4.4 and Android 5**. That is **Dalvik** is in older versions (4.4) and **ART** is in Android 5 upwards. ART is introduced in **KIT-KAT** and c**ompletely replaced in Lollipop**. Android SDK.

Literature review ...

ART uses **a head of time compilation**, that easy and faster to run apps, because they are compiled to machine code upon installation rather than application is initiated until feature is being called. To compile an app, you need to use compiler which include in Android SDK.

Literature review ...

The Compilation Process



Literature review ...

The application source code is normal **java code**, it is a *.java, and it will be compiled from **Oracle JDK**. (javac). This will produced a class file, (byte code). There will be an optional third party tool called **ProGurd**. This will be done for **obfuscate the code**, it will reduce the readability of the code, and so on. So **decompilers cannot read the code correctly**. Then regardless of the **ProGurd** , then class file will be converted to ***.DEX bytecode** file. The dex file will be distributed via application package.

Literature review ...

The Manifest File: File which is including **all the components** details of the application. The manifest does a number of things in addition to declaring the application.

1. Identify any user **permissions** the application requires;

2. Declare the minimum application program interface **(API)** Level required by the application;

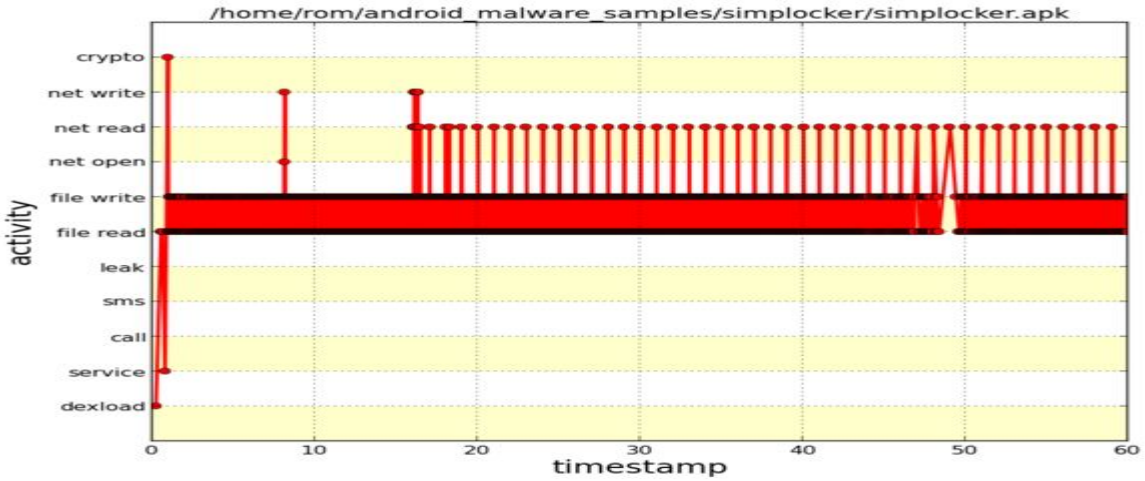
3. Declare **hardware and software features** used or required by the application;

4. API libraries the application needs to be **linked** against.

Literature review ...

DroidBox connects to target which is a emulator consist of malware. Then it will receives all the possible information and display in a text or graphical format. The diagram file shows overall distribution of operations. A chart showing the distribution of operations as a function of time.

Literature review ...Droidbox chart - simlocker



Simplocker. Chart of the operation distribution by time

Literature review...Droidbox Trace

The generated trace file which records access information as following format.

```
"Section name" {
   "Operation time (from start of launch)" {
      "Name of parameter being traced (e.g., for sendnet, this could be desthost, destport, etc.)": "Parameter value"
   }
}
```

i.e.

```
“cryptousage”: {
“0.9651350975036621”: {
“algorithm”: “AES”,
“key”: “95, -81, 109, &lt;…&gt;”,
“operation”: “keyalgo”,
“type”: “crypto”
}
&lt;…&gt;},
```

Literature review...

DroidBox v 4.1.1RC Build Environment...

Droid box is tested on Linux and Mac OS only.

It needs following libraries:

- Pylab
- Matplotlib

It uses Android 4.1.2 Nexus Emulators, This emulator is built in Android 4.1.1 and it has modified the implementation.

Java:- 1.6 jdk

Methodology

Fulfil the knowledge GAP - Learning Android Development

Analysing the Droidbox Functionalities - Download Droidbox, Compilation, And Building the Application.

Preparing Test Data - Preparing test data using Android SDK

Test Execution - Analyse Existing Features

Compiling Droidbox Emulator with new Android 5.0 and analysing the issues.

Discussion and Test Results

- Droidbox was built in Android 4.1.1. Which was compiled in ubuntu 12.04.01 64 bit, jdk 1.6. The new Android versions are compiled in ubuntu 14.04 64 bit OS, and uses jdk 1.7.
- Droid box application is compatible in ubuntu 14.04 and it can be run on jdk 1.7 environment without an issue.
- The Emulator we need to build in latest Android version, therefore the 1st challenge is to compile it in ubuntu 14.04.
- /usr/lib/x86_64-linux-gnu/libSDL.so: file not recognized

Goals of Project

- Android app development
- Understanding Malware Analysis Technique - Dynamic Analysis
- Understanding Malware analysis by reverse engineering. (Modifying parameters/return value before/after function's execution)
- Analysing the Droidbox functionalities.
- Analysing Droidbox issues when upgrading to ART.

Issues faced

1. Setting up environment in Virtual Machine is not feasible
    - High memory and cpu

    Solution : Install Ubuntu and setup the environment.

2. Limited information on DroidBox.

    Solution : submit the proposal to Google summer of codes program there I can get consultations from the honeynet.org

3. Finding Sample Malware apps

Issues faced

Solution : Building own application using SDK

4. Download Android source code: over 80 GB

5. Fixing Build Errors on Android Source Code:

Solution - research and identifying needed library files dependencies.

Questions?

Thank You!