# Runtime-based Behavior Dynamic Analysis System for Android Malware Detection

Luoxu Min
School of Computer Science and Engineering
Beihang University
Beijing, China
minluoxu@163.com

Qinghua Cao
School of Computer Science and Engineering
Beihang University
Beijing, China
caoqinghua@buaa.edu.cn

***Abstract*—The most serious threats for Android users is come from application, However, the market lack a mechanism to validate whether these applications are malware or not. So, malware maybe leak user's private information, malicious deductions for send premium SMS, get root privilege of the Android system and so on. In the traditional method of malware detection, signature is the only basis. It is far enough. In this paper, we propose a runtime-based behavior dynamic analysis for Android malware detection. The new scheme can be implemented as a system. We analyze 350 applications come from third-party Android market, the result show that our system can effectively detect unknown malware and the malicious behavior of malware.**

***Keywords-Android Security; Dynamic Detection; Android; malwares***

## I. Introduction

Smartphone is becoming more and more important in our lives, according to the Morgan Stanley, Smartphone will outstrip PCs in 2012 and tablet shipments will reach 100mm in 2012 [1]. The first quarter of 2011, the market share of Android in the global over Symbian for the first time, Ranking first in the world. Until July 2012, China's market share was 76.7%.Android is an open source Linux-based operating system led by Google. Equipment Android devices can run a variety of mobile application software on an application runtime environment .so as the malware. Unfortunately, the increasing adoption of Smartphone comes with the growing prevalence of mobile malware. The most popular type of malware is Trojan, it may cause threats such as information leakage, get root privileges. The Geinimi[2]and DroidDream [3] appeared in china Android market is the representative.

Android strengthens the security in several aspects, the most important is privilege separation and the sandbox model [4]. In normal, the application cannot harm to the other application and system. Application to apply for permission to the system, the system will Grant permission to it, And to establish the appropriate data structure. However, once the permission is given, Applications have the ability to do any of the things that can be done, An application with access rights to confidential data and eligible to exposure data to public can easily steal phone owner's information. Is the application using or abusing its access rights?

In this paper, we propose a runtime-based behavior dynamic analysis system for Android malware detection. The system consists of event creator and Log monitor and Parser. The event creator simulates user's action, through the result of static Analysis. Android platform has system debug monitor called Logcat. When the application runs on the Customized Emulator, The log monitor module collects the useful log. And the Parser extracts the malicious behavior. Until now, we can analysis the personal information leakage and the behavior of sending premium SMS. We have implemented prototype system and analyze 350 application collected from Amazon Android market. The system detected 82 applications that leak some kind of private date and 8 applications that send Premium SMS.

The rest of the paper is organized as follows: Section 2 will describe background information on the Android platform, Section 3 design and implementation of our proposal, Section 4 presents results from our study, and section 5 summarizes our conclusions.

## II. Background Information

Android is a Linux-based operating system designed primarily for touch-screen mobile devices such as Smartphone and tablets[5].as depicted in figure 1,It is divided into four layers from top to bottom, the Linux kernel is responsible for the management of hardware resources. The middleware and Libraries run on the top of Linux. The application developer can call the application API that provided in the third layer, and is executed within a Virtual Machine called Dalvik.

Android defines four component types. Activity is used to define the interface. Service used to define the program running in the background. Content provider store and share data using a relational database interface. Broadcast receiver act as mailboxes for message from other application.

For security, Android mediates IPC based on permission labels. A permission label is simply a unique text string that can be defines by developer. A list of permission labels required in its package manifest. Android assigns permission to the application at the install time and the permissions can't be changed at runtime. However, requested permissions are not allowed .for example, an application request the SEND SMS and READ SMS permissions may be leak personal private information.

The potential risk for abuse of private information stored on the phone is very serious, when the application gets the sensitive permission, it maybe leak your private data in the background, but the user can hardly prevent the app from making illegal use that information so far. The sensitive personal data mainly include telephone number/E-mail address, contact list, IMSI(International Mobile Subscriber Identifier), IMEI(International Mobile Equipment Identifier),location and so on.

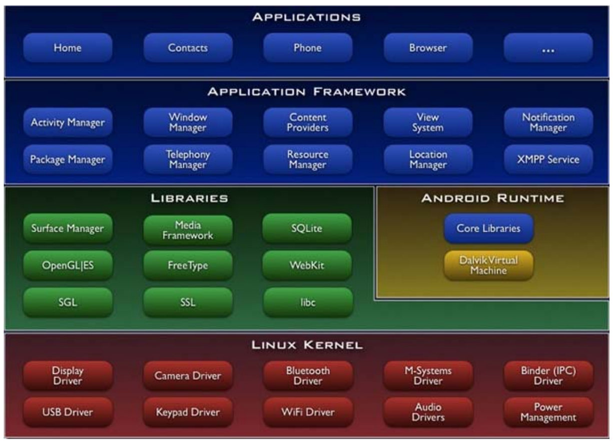


Figure 1. Architecture of Android

The next most common threat of the Smartphone security is send premium SMS, we use FakePlayer[6] example, an SMS Trojan that sends short messages to premium, it disguise as legitimate application, and send SMS without user consensus. The application maybe send a specific content to a premium numbers, including "10621918", "10621900", "1066185829 " and etc[7].

## III. System Design and Implementation

This section described design and implementation of Runtime-based behavior dynamic system. First of all, we will describe the architecture of proposed system. Then we present how to create Trigger event and how to collect and analyze the malicious behavior.

### *A. System architecture*

How can we tell whether an application is using or abusing its access rights? When a game app requests the INTERT permission, it maybe only shares the score. To find the hidden threat in the app, we propose the runtime-based behavior dynamic analysis system. Figure 2 depicts architecture of the System. At first, we perform static analysis on the App, when the app run on the Customized Emulator, the event Creator triggers the sensitive event. Log monitor collect the log and analysis to get the malicious behavior.

### *B. Static analysis*

The goal of static analysis is to use static analysis to find ways to trigger potentially malicious or risk behavior when the app runs in the dynamic analysis. We will get the list of permissions on the app, the list of registered activities, services and broadcast receivers from the app also are extracted. In the module, the baksmali[8] decompiles the app, and extracted the information.

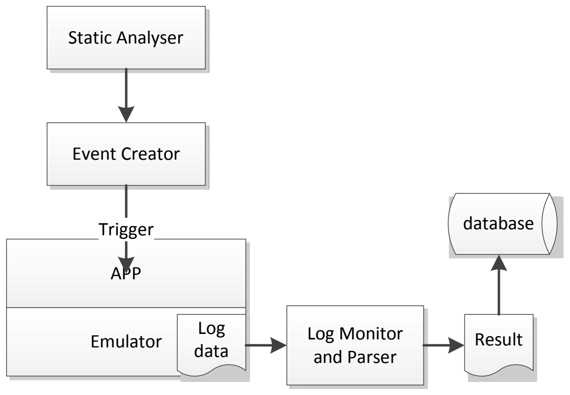


Figure 2. System architecture

The static analysis provides the approximate semantic analysis of application's run-time behaviors, the object of static analysis consists of collecting risk permission information, accessing APIs of private source and the APIs of activity. The flow of static analysis as follows

a) *Decompile, input an application, the analyzer engine decompiles the application, the output is the manifest.xml and java code.*

*b) Semantic analysis, the manifest.xml as input, the engine analysis manifest.xml, and pick up the risk permissions, activity list, Service list and other base information such as package name, hash number and so on.*

*c) Data flow analysis, build up all user-defined method with sensitive API into Syntax tree. The tree will generate CFG file for the application, it is used in the event creating.*

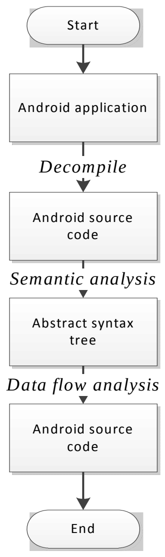


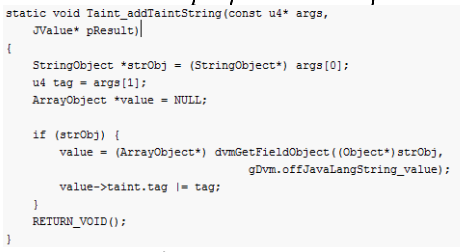
Figure 3. Flow of static analysis

## C. *Customized Emulator*

The customized Android is base on version 2.3.4,the private data we trace is contacts, calendar , SMS , call log, network and so on. We track the flow of sensitive information through app. The LKM(loadable kernel module)[9] API hook technology is Applied to the module。The steps of realize the hook as follows:

*a)* *Find the entry address of system calls, and call the function form sys_call_table to perform your function.*

*b)* *Write the hook function, we can screen suspicious system call with the entry address, it the system call is suspicious, and feedback. There are a example for taint data function.*



```
static void Taint_addTaintString(const u4* args,
    JValue* pResult)
{
    StringObject *strObj = (StringObject*) args[0];
    u4 tag = args[1];
    ArrayObject *value = NULL;

    if (strObj) {
        value = (ArrayObject*) dvmGetFieldObject((Object*)strObj,
                                    gDvm.offJavaLangString_value);
        value->taint.tag |= tag;
    }
    RETURN_VOID();
}
```

*c)* *Put the hook function in the Android kernel, when an application run on the Android version, the application action will be hook, and print log when the action is occurred.*

When the App run on the customized emulator, if the sensitive information is sent, the log will record this behavior .In order to simulate events, the events creator use the Robotium to certain types of views such as buttons, menus and etc. use the system debugging tool Logcat to send the log data to Log Parser to analysis the log.

## D. *Log monitor and Parser*

A new log monitor is added in the existing monitor hierarchy to monitor the log output. The log monitor will collect the log data when the app runs, however, the log data contains lots of no-go data. The first job of log monitor is filtering the no-go data. According to the result of the static analysis, the parser analyzes the log data through Semantic analysis. The content contains two aspects. One is sensitive information leakage, other is the send some context to some premium numbers.

The monitor collected log and the parser analysis flow as below:

*a)* *First of all, when the Android application run on the emulator, the system Logcat will print the log that record the behavior of the application, the log collector gathers all log.*

*b)* *Secondly, as the log stream input, the log monitor filter the log data of uninterested information to reduce noise from collected log data.*

c) *Lastly, the parser analysis the log and pick up some useful information, the private data leakage and send SMS signatures, the private data consist of IMSI,SIM,IMEI,TEL,MAIL and so on. The technology of analysis is semantic analysis and regular expression.*

## IV. APPLICATION ANALYSIS RESULTS

We have implemented the prototype system on Android emulator and PC, First of all, we collect about the 350 apps form Amazon Android market, secondly, let the app run on the Customized Android on the emulator one by one. Lastly, the result will display.

The result for our experiment show that the current status of Android application' security is not very satisfactorily. There are some findings in our experiments and research.

- 56% applications use intent permission.
- 48% applications use third-party advertisement library such as Admob, Mobclix and so on which can collect some private information.
- 23% applications acquire user's private data and will leak it.
- 8 applications may send premium SMS which may be illegality.

## A. *Experiment Environment*

The experiment environment is based on Android 2.3 and Ubuntu 10.04, the communication between the PC and emulator via adb shell. Also we have implemented a customized ROM image on HTC G7.the host of system control the static analysis, collecting log data and detecting the spite behavior.

## B. *Static Analysis Result*

The result shows that our method of analysis is effectively for detecting the application that has potential risk for leaking private information. As showed in the Table 1, IMEI ,location information and some other private information of user will be leak if when the user use the application on their device. Take com.stylem.wallpapers for example, it totally has more than 30 million downloading for smart phone platform that means how popular it is, however, the IMEI as unique user id will be send to a server without encryption.

TABLE I. STATIC ANALYSIS RESULT

| App name | Location | IMEI | SMS |
|---|---|---|---|
| com.stylem.wallpapers | No | Yes | No |
| com.abarakat.news | Yes | Yes | No |
| com.saimu.soudan | No | Yes | No |
| org.moncter.runner | Yes | No | Yes |
| kr.yjs.GroupSMS | No | No | Yes |
| baltorogames.skijumping2012 | Yes | Yes | Yes |
| com.abarakat.news | Yes | Yes | No |

The static analysis also organizes all the activity and service to generate the CFG file. It is an xml file to describe the logic of the application. The event creator will create the event according to CFG file.

### *C. System Demo*

Take backgrounds.apk for example, when this application is tested in the system. First of all, as depicted in Figure 4, we start the customized emulator, the application run it automatic with trigger all the action. The log monitor will collect and analysis all the suspicious behavior described in Figure 5.

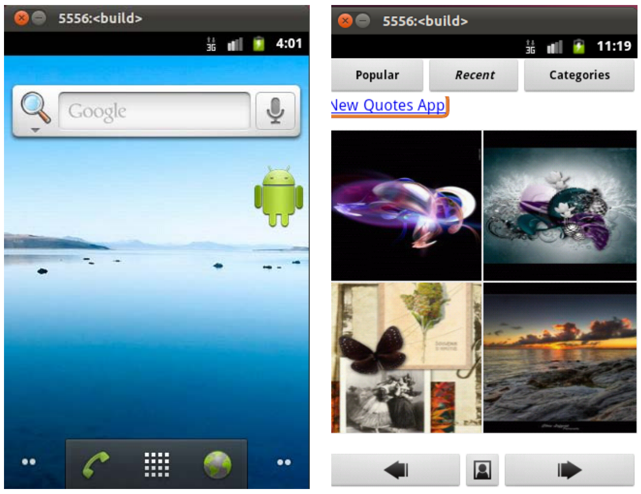
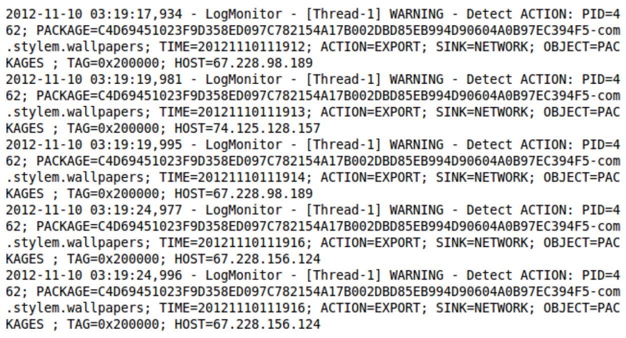


Figure 4. Testing the application



```
2012-11-10 03:19:17,934 - LogMonitor - [Thread-1] WARNING - Detect ACTION: PID=4
62; PACKAGE=C4D69451023F9D358ED097C782154A17B002DBD85EB994D90604A0B97EC394F5-com
.stylem.wallpapers; TIME=20121110111912; ACTION=EXPORT; SINK=NETWORK; OBJECT=PAC
KAGES ; TAG=0x200000; HOST=67.228.98.189
2012-11-10 03:19:19,981 - LogMonitor - [Thread-1] WARNING - Detect ACTION: PID=4
62; PACKAGE=C4D69451023F9D358ED097C782154A17B002DBD85EB994D90604A0B97EC394F5-com
.stylem.wallpapers; TIME=20121110111913; ACTION=EXPORT; SINK=NETWORK; OBJECT=PAC
KAGES ; TAG=0x200000; HOST=74.125.128.157
2012-11-10 03:19:19,995 - LogMonitor - [Thread-1] WARNING - Detect ACTION: PID=4
62; PACKAGE=C4D69451023F9D358ED097C782154A17B002DBD85EB994D90604A0B97EC394F5-com
.stylem.wallpapers; TIME=20121110111914; ACTION=EXPORT; SINK=NETWORK; OBJECT=PAC
KAGES ; TAG=0x200000; HOST=67.228.98.189
2012-11-10 03:19:24,977 - LogMonitor - [Thread-1] WARNING - Detect ACTION: PID=4
62; PACKAGE=C4D69451023F9D358ED097C782154A17B002DBD85EB994D90604A0B97EC394F5-com
.stylem.wallpapers; TIME=20121110111916; ACTION=EXPORT; SINK=NETWORK; OBJECT=PAC
KAGES ; TAG=0x200000; HOST=67.228.156.124
2012-11-10 03:19:24,996 - LogMonitor - [Thread-1] WARNING - Detect ACTION: PID=4
62; PACKAGE=C4D69451023F9D358ED097C782154A17B002DBD85EB994D90604A0B97EC394F5-com
.stylem.wallpapers; TIME=20121110111916; ACTION=EXPORT; SINK=NETWORK; OBJECT=PAC
KAGES ; TAG=0x200000; HOST=67.228.156.124
```

Figure 5. Log Monitor

### *D. Dynamic Detecting Result*

We have found 82 applications that leak some kind of private information, including 50 applications leak IEMI, Location, 32 applications leak IMSI,8 applications leak SMS or E-mail information.

We found total 8 applications that send premium to service provider, these applications require SEND_SMS and RECRIVE_SMS permission, when trigger a button of the application, it will send a premium SMS, and it also delete the confirm SMS that is sent by service provider. The detail is described in Table 2

TABLE II. PREMIUM SMS

| App Name | Service Provider | Content |
|---|---|---|
| com.Green.SMS | 10626213 | aAHD |
| com.keji.send | 106691819 | 95pAHD |
| com.ksfs.danri | 10665123085 | 58#28AHD |

## V. CONCLUSIONS

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper.

We have proposed a runtime based behavior dynamic analysis for Android malware detection, which is not absolutely dependent on virus signatures. For unknown malware, this method is very effective. The system we implemented based on runtime behavior can be applied to security inspection.

[1] Gartner Inc. Android, Apple to spark surge in smart device sales http://www.gartner.com/it/page.jsp?id=1622614

[2] Symantec Inc. Android.Geinimi http://www.symantec.com/security_response/writeup.jsp?docid=2011-010111-5403-99

[3] Update: Android Malware DroidDream: How it Works https://blog.lookout.com/blog/2011/03/02/Android-malware-droiddream-how-it-works/

[4] ENCK,W.W.ONGTANG.,and mcdaniel,P.On lightweight Mobile Phone Application certification. In Procee-ings of the 16th ACM Conference on Computer and Communications Security.

[5] Android . https://www.Android.com

[6] AndroidOS.FakePlayer http://www.symantec.com/security_response/writeup.jsp?docid=2010-081100-1646-99

[7] Yajin Zhou ,Zhi Wang, Wu Zhou, Xuxian Jiang. hey,You Get Off of My Market:Detecting Malicious Apps in Offical and Alternative Android Market. the 19th Network and Distributed System Security Symposium (NDSS 2012), San Diego, CA, February 2012

[8] Smali https://code.google.com/p/smali/

[9] De Goyeneche, J.-M.; De Sousa, E.A.F.Loadable kernel modules Volume:16, Issue: 1 Digital Object Identifier:10.1109/52.744571 Publication Year: 1999 , Page(s): 65- 71