A CASE STUDY OF MALWARE DETECTION AND REMOVAL IN ANDROID APPS

Wichien Choosilp and Yujian Fu

Department of Electrical Engineering & Computer Science, Alabama A&M University,
Normal AL, USA

ABSTRACT

With the proliferation of smart phone users, android malware variants is increasing in terms of numbers and amount of new victim android apps. The traditional malware detection focuses on repackage, obfuscate and/or other transformable executable code from malicious apps. This paper presented a case study on existing android malware detection through a sequence of steps and well developed encoding SMS message. Our result has demonstrated a solid testify of our approach in the effectiveness of malware detection and removal.

KEYWORDS

Malware detection, Wireless Network, Mobile Network, Virus, Worms & Trojan horse

1. INTRODUCTION

Few people would disagree that the cell phone device has become an important part of everybody's life today. Statistics data from LAObserver[1] shows that 91% of U.S. adults own cell phones and 56% own smart phones. Android OS is one of the most popular mobile platforms. In October 2012, there were approximately 1,000,000 apps available for Android, and the estimated number of applications downloaded from Google Play, Android's primary app store, was 50 billion as of September 2013. With the explosive growth of smart mobile devices market and usage, there are an increasing number of malicious mobile applications that are developed to target these devices and platforms. These malicious applications are called mobile malware. Nowadays, mobile malware have reached a new level of maturity. Threats targeting smart phones and tablets are beginning to pose meaningful challenges to users, enterprises, and service providers alike. The number of instances of just one family of malware can be in the thousands. The largest proportion of malware is targeting on the Android mainly due to the dominant market share of the Android platform and its open market policy.

However, many of the smart phone and tablet users have, for the most part, not been aware of the risks of mobile malware, of which there are many. This module aims at introducing the working mechanisms of mobile malware and some defense methods that may be employed. To protect mobile users from the severe threats of Android malware, many different solutions have been proposed.

DOI : 10.5121/ijmnct.2014.4205

2. BACKGROUND

Created by Google, Android is a most popular cell phone operating system for mobile devices including smartphones and tablets. It is available to all kinds of developers with various expertise levels, ranging from rookie to professional. Based on the Linux kernel Android can provide a middleware implementing subsystems such as telephony, window management, management of communication with and between applications, managing application lifecycle, and so on. On top of the kernel, Android provides all types of apps for users.

Android applications are programmed primarily in Java though the programmers. Native programming are allowed via Java native interface. Different from Java compilation process[2], instead of Java bytecode, Android outputs and runs Dalvik bytecode. In comparison to Java, all the compiled classes are generated and packed together into a single .dex file in Dalvik.

An Android application is composed of four types of components, namely activities, services, broadcast receivers, and content providers. All these four components are defined as classes in the library. They are further declared in the AndroidManifest (a .xml file used for the web browser). The Android end user apps will directly or indirectly use these components and interact with the kernel through them. AndroidManifest is a manifest file defined as binary XML file, which declares the application package name. A package id is defined as a string and needs to be unique to an application. It also declares other things (such as application permissions) which are not so relevant to the present work.

Android SDK (Software Development Kit)[3] includes a virtual mobile device emulator that allows android apps to run on the computer. The Android emulator mimics all of the hardware and software features except it cannot place actual phone calls. The most recent version of Android is known as "Jelly Bean", which was released on July 2012. The new features of android is provided using Xamarin.Android.

3. SYSTEM ARCHITECTURE AND DESIGN

We expect the proposed system consisting of following features:

- An Android mobile device can subscript to any service provider and the provider will send a list of nearby registered devices. The mobile device is responsible of maintaining the list and decides how to communicate or interact with nearby devices.
- An Android mobile device checks its environment such as operating system version, network type and IP address periodically and reports back to service provider. The service provider will perform necessary analysis on the data and accordingly make recommendations to the mobile device.
- An Android mobile device updates its current location and sends information to service provider with user's permission. The service provider keeps location history of each device for analysis purpose.
- A rooted android device should be able to take more proactive actions than nonrooted devices. Some possible actions include but not limited to: snapshot network states; capture network packets; intrusion detection; malware removal; etc.

Out project aims at to implement a system that carries the above features. Our design model will include three classes – WC_SMS_Activity, WC_SMS_Display_Activity, and WC_SMS_BroadCast_Receiver. The first two classes are inherited from Activity class. Each class is defined in Figure 1. The system architecture developed in UML class diagram is shown in Figure 2.

WC_SMS_Activity
~recNum : EditText ~secretKey : EditText ~msgContent : EditText ~send : Button ~cancel : Button
+onCreate(savedInstanceState : Bundle) : void +sendMessage(recNumString : String, encryptedMsg : String) : void + convertByeToHex(b : byte []) : String +encryptMessage(secretKeyString : String, msgContentString : String) : byte [] +generateSecretKey(secretKeyString : String) : Key

WC_SMS_Display_Activity
~secretKey : EditText ~encryptedMsg : TextView ~decryptedMsg : TextView ~submit : Button ~cancel : Button ~originNum : String = "" ~msgContent : String = ""
+onCreate(savedInstanceState : Bundle) : void +convertHexToByte(b : byte []) : byte [] +decryptMessage(secretKeyString : String, encryptedMsg : byte []) : byte [] +generateSecretkey(secretkeyString : String) : Key

WC_SMS_BroadCast_Receiver
+onReceive(context.Context, intent.Intent) : void

Figure 1. Class definition in UML model.

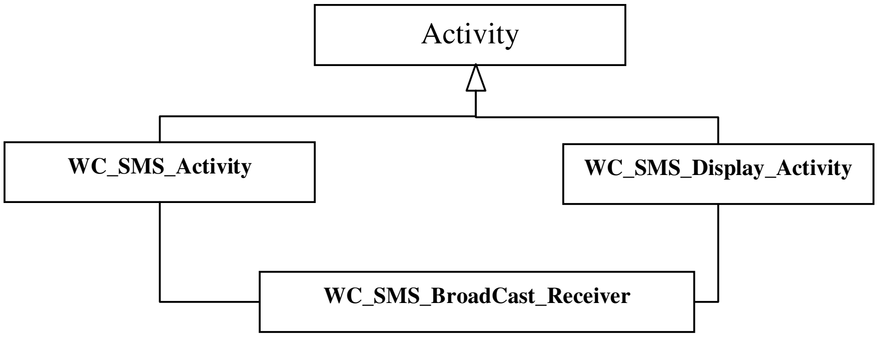


Figure 2. System architecture denoted in class diagram.

4. SYSTEM IMPLEMENTATION

To implement android system, either emulator on the computer or android device can be used as platform. The system environment is set up by several kits including JDK, Eclipse, SDK, and ADT bundle. They are simply explained in the following.

- JDK - Java Development Kit[4] is a program development environment for writing Java applets and applications. It consists of a runtime environment that "sits on top" of the operating system layer as well as the tools and programming that developers need to compile, debug, and run applets and applications written in the Java language.
- Eclipse SDK[5] - In computer programming, Eclipse is multi-language Integrated development environment (IDE) comprising a base workspace and an extensible plug-in system for customizing the environment. It is written mostly in Java. It can be used to

develop applications in Java and, by means of various plug-in, other programming languages including Ada, C, C++, COBOL, Fortran, Haskell, JavaScript, Lasso, Perl, PHP, Python, R, Ruby (including Ruby on Rails framework), Scala, Clojure, Groovy, Schema, and Erlang. It can also be used to develop packages for the software Mathematica. Development environments include the Eclipse Java development tool (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

- Android SDK - A software development kit[3] that enables developers to create applications for the Android Platform. The Android SDK includes sample projects with source code, development tools, and emulator, and required libraries to build Android applications. Applications are written using the Java programming language and run on Dalvik, a customized virtual machine designed for embedded use which runs on top of a Linux kernel.
- Android Development Tools (ADT) - A plug-in for the Eclipse IDE that is designed to give a powerful, integrated environment in which to build Android applications. ADT extends the capabilities of Eclipse to allow for quick set up of new Android projects, to create an application UI, to add packages based on the Android Framework API, to debug your applications using the Android SDK tools, and even to export signed (or unsigned) .apk files in order to distribute your application.

Unlike other programming paradigms in which apps are launched with a main() method, the Android system initiates code in an Activity instance by invoking specific callback methods that correspond to specific stages of its lifecycle. There is a sequence of callback methods that start up an activity and a sequence of callback methods that tear down an activity.

WC_SMS_Display_Activity.java file contains four important methods:

- onCreate(Bundle savedInstanceState) - to set up the UI variables.
- convertByteToHex(**byte**[] b) - get the length of characters and convert to the string.
- decryptMessage(String secretKeyString, **byte**[] encryptedMsg) - to decrypt the text message from the user input.
- generateSecretKey(String secretKeyString) - to generate the 16-Character Secret Key.

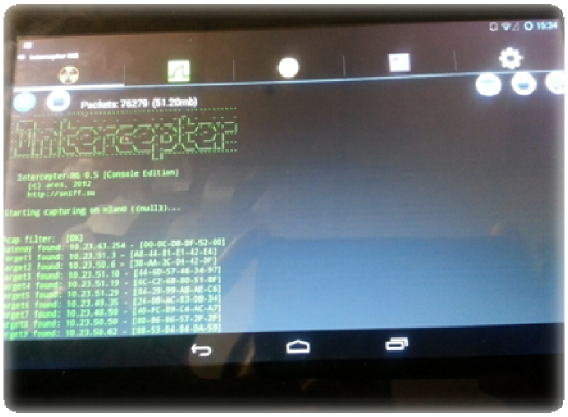
WC_SMS_Activity.java file contains four important methods:

- `onCreate(Bundle savedInstanceState)` - to set up an UI variables.
- `sendMessage(String recNumString, String encryptedMsg)` - to send the text message.
- `convertByteToHex(byte[] b)` – to get the length of characters and convert to the string.
- `encryptMessage(String secretKeyString, String msgContentString)` - to encrypt the text message.
- `generateSecretKey(String secretKeyString)` - to generate the 16-Character Secret Key.

One of the most powerful spying tools is Intercepter-NG[6]. It is a free application with unrestricted functionality and is virtually universal: works on Windows, Linux, Mac OSX, iPhone and Android. It is a multifunctional network toolkit for various types of IT specialists[7]:. It has functionality of several famous separate tools and more over offers a good unique alternative of ireshark for Android. The main features are:

- Sniffing passwords\hashes of the types: ICQ\IRC\AIM\FTP\IMAP\POP3\SMTP\LDAP\BNC\SOCKS\HTTP\WWW\NNTP\CVS \TELNET\MRA\DC++\VNC\MYSQL\ORACLE\NTLM\KRB5\RADIUS
- Sniffing chat messages of: ICQ\AIM\JABBER\YAHOO\MSN\IRC\MRA
- Reconstructing files from: HTTP\FTP\IMAP\POP3\SMTP\SMB
- Promiscuous-mode\ARP\DHCP\Gateway\Port\Smart Scanning\
- Capturing packets and post-capture (offline) analyzing\RAW Mode

After connected to the AAMU WiFi using Intercepter-NG, we can run the scan command to see all the devices with IP addresses that are connecting to AAMU WIFI. Result is shown in Figure 3, where a list of AAMU WiFi address was recognized and displayed to the screen.



```
Packets: 76279 (51.20mb)
Intercepter

Intercepter-NG 0.5 [Console Edition]
  (c) ares. 2012
  http://sniff.su

Starting capturing on wlan0 ((null))...

cap filter:  [OK]
ateway found: 10.23.43.254 - [00-0C-DB-8F-52-00]
arget1 found: 10.23.51.3 - [A8-44-81-E1-42-E4]
arget2 found: 10.23.50.6 - [38-AA-3C-D4-42-DF]
arget3 found: 10.23.51.10 - [44-6D-57-46-34-97]
arget4 found: 10.23.51.19 - [6C-C2-6B-80-51-8F]
arget5 found: 10.23.51.29 - [84-38-99-AB-AE-C6]
arget6 found: 10.23.49.35 - [24-DB-AC-E3-D8-34]
arget7 found: 10.23.48.50 - [40-FC-89-C4-AC-A7]
rget8 found: 10.23.50.50 - [80-86-86-57-2F-2F]
rget9 found: 10.23.50.82 - [88-53-84-84-BA-59]
```

Figure 3. List of IP address found on AAMU WiFi.

After that, the application is starting to collecting packets that is sending and receiving through this WIFI. In most case, this application can collect the information like user name and password.

5. CONCLUSIONS AND FUTURE WORKS

This case study aims at implementing the android malware detection and removal in Android apps. This is the first time android security application at Alabama A&M University. It is successfully implemented for all of the preliminary steps to this project and the project came together in phases. First, we demonstrated the detecting and removing of malware using Advanced Mobile Care Application. In addition, the process of sending and receiving text messages between two devices with encryption and decryption was completed. Finally, we successfully connected to AAMU Wi-Fi and collected information from the devices that connect to that network, through using the Wi-Fi packet snooping application Intercepter-NG. The future work will be the successful implementation of all the various components of the project and the observation of their operation.

ACKNOWLEDGEMENTS

We would like to thank for Lockheed Martin and the Dean's office for the support of this project. This project is also partially supported by NSF subaward DUE-1202690 and NSF grant DUE-1225654. We would like to thanks for all valuable reviews.

REFERENCES

[1] Lacter, M. (June 2013). Stunning stat: 91% of U.S. adults own cell phones. LA Observed. Available from: http://www.laobserved.com/biz/2013/06/stunning_stat_91_of.php.

[2] Shama, R. Developing for Android – Introduction. Available from: http://www.cprogramming.com/android/android_getting_started.html. CPprogramming.

[3] Android SDK Developer. Available from: https://developer.android.com/sdk/index.html.

[4] Java SE downloads. Oracle. Available from: http://www.oracle.com/technetwork/java/javase/downloads/index.html

[5] Eclipse project. Available from: http://www.eclipse.org/downloads/

[6] Intercepter-NG. Available from: http://intercepter.nerf.ru/

[7] XDA Developes Forum. Available from: http://forum.xda-developers.com/showthread.php?p=35159281

[8] InfoSec Institute. Available from: http://resources.infosecinstitute.com/backtrack-5-part-1/

[9] Broida, R. (2013) How to easily root an Android device. Mobile World Congress. C.Net. Available from: http://howto.cnet.com/8301-11310_39-57608195-285/how-to-easily-root-an-android-device/

Authors

Short Biography

Wichien Choosilp is a graduate student at the department of Computer Science at Alabama A&M University. His research focuses on the Android malware detection, analysis and removal.

Yujian Fu is an associate professor at the department of Computer Science at Alabama A&M University. Her research focuses on the embedded and real time system design, cyber physical system, programming language, formal methods.