Implementation of A Lesser Used Android Malware Detection Technique

Kevin Terusaki
kterusa1
Hugh Troeger
htroege1
April Bowen Wang
bwang1

Abstract

Due to the increase in Android targeted malware attacks, the need for software techniques to capture and log malware activity to better counter attacks is on the rise. Since there are already plenty of malware detection tools in the computing world, we restricted our attention to developing a malware detection tool that successfully deals with a specific spyware that is underly dealt with, rather than a comprehensive generic one. We plan to build a logger application for Android that detects and records system calls pertaining to the behavior of a specific malware using Eclipse and Android Virtual Device.

1 Motivation

The popularity of Android smartphones is steadily surpassing that of other smartphone suppliers–namely iOS, Symbian, Blackberry and Windows. Along with Androids large market share and the nature of its open source architecture, the number of Android targeted malware has seen a large increase. In 2013, the U.S. government reported that 79% [**?**] of mobile operating system malware attacks are targeted toward the Android OS. With Android users representing 70% [**?**] of the smartphone market worldwide, the need for software techniques to capture and log malware activity to better counter attacks is on the rise. Currently, there are several security apps provided by well-established computer security firms (Symantec, AVG, etc.) [**?**] that contain various loopholes that attackers can manipulate in order to escape detection.

2 Background

Malware is the type of software that exhibits malicious behavior, such as viruses, botnets, worms and Trojan horses. In addition to malware, the two other major categories of threats to mobile devices are personal spyware and grayware. Spyware collects information such as user location, SMS messages and call history without the victims knowledge, while grayware is far less serious– instead of disrupting computer operations or gathering sensitive information, it might change the users font colors or install pop-ups.

There are many malware detection techniques [**?**], such as static analysis, dynamic analysis, application permission analysis, cloud-based detection and battery life monitoring. While each has its strength and weakness, we are mainly interested in dynamic analysis technique, as the ability to analyze actions performed by a program during execution clearly makes it a more powerful technique than the other types of analysis.

We currently know the techniques employed by two dynamic analysis tools, TaintDroid and Sandbox. TaintDroid provides dynamic taint tracking for Android on four levels (variable, method, message, file). However, it might suffer from false negative and false positive results. In addition, it focuses solely on dataflow and doesnt consider other vulnerabilities. Sandbox performs both static and dynamic analyses in offline mode. Static analysis disassembles the application image binary and uses the disassembled code for search for any suspicious patterns. Dynamic analysis component executes the binary in an Android emulator and logs the system calls. Its limitation is that its Android-Monkey-generated inputs are not as effective as testing with real users.

An article in ACM Computing Surveys, Vol. 44 [**?**] provided an overview of 18 dynamic analysis based tools and the techniques they each employ. Since there are already plenty of malware detection tools in the computing world, we restricted our attention to developing a malware detection tool that successfully deals with a specific malware that is underly dealt with, rather than a comprehensive generic one.

3 Idea

We plan to build a logger application for Android that detects and records system calls pertaining to the behavior of a specific malware. After more research we are going to select a documented subclass of non-destructive malware (spyware), for which there does not yet exist a specialized detection application. We will draw upon others research to build a logger which records detailed information from the calls and areas that the spyware typically uses.

We chose spyware so that we dont need to worry about damage to the system or interference with our application, and general safety. There exist phone tracking and other spyware applications easily available for download.

We are going to use Java and the Android SDK to build an application that monitors and logs system calls that can be used to detect our chosen type of spyware. We will likely develop in Eclipse, and use an Android Virtual Device as our testing platform. We will make use of Android APIs, including the Logger class. Depending on our progress we may try to write detection algorithms that search for patterns indicative of spyware activity.

4 Goals

4.1 Milestone 1

Have a specific piece of spyware chosen. Have done extensive research on the spyware and its behaviors and system calls; ways that it accesses information in the Android OS. Have done lots of reading on the Android OS, be familiar with its components, architecture, channels of communication. Be familiar with existing logger applications for Android, and how they function / what APIs they make use of.

4.2 Milestone 2

Have working logger application. Be able to record relevant information in a log file. Be able to display statistics about the information in the log file.

4.3 Milestone 3

Be able to detect behavior recorded in the log that is indicative of an application acting as spyware, and log or display warnings. Possibly, do analysis of performance overhead.

References

[1] CNBC. Us security agencies say android mobile main target for malware@ONLINE, October 2013.

[2] Manuel Egele, Theodoor Scholte, Engin Kirda, and Christopher Kruegel. A survey on automated dynamic malware-analysis techniques and tools. *ACM Comput. Surv.*, 44(2):6:1–6:42, March 2008.

[3] Adrienne Porter Felt, Matthew Finifter, Erika Chin, Steve Hanna, and David Wagner. A survey of mobile malware in the wild. In *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, SPSM '11, pages 3–14, New York, NY, USA, 2011. ACM.

[4] H Mccracken. Who's winning, ios or android? all the numbers, all in one place @ONLINE, October 2013.

[5] Vaibhav Rastogi, Yan Chen, and Xuxian Jiang. Droidchameleon: evaluating android anti-malware against transformation attacks. In *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, ASIA CCS '13, pages 329–334, New York, NY, USA, 2013. ACM.