2015

An Automated Virtual Security Testing Platform for Android Mobile Apps

Yong Wang

An Automated Virtual Security Testing Platform for Android Mobile Apps

Yong Wang
College of Business and Information Systems
Dakota State University
Madison, SD 57042
yong.wang@dsu.edu

***Abstract*—This paper proposes an automated virtual security testing platform for Android mobile apps. The testing platform includes three key components: customizing Android OS to include mobile app trace information, creating a virtual testing platform using the customized OS, and developing static and dynamic analyzing techniques for mobile malware detection. The proposed testing platform is a server-side malware detection solution. It can utilize both static and dynamic analysis and is a good compensation to the client-side mobile security software.**

Keywords—Android, mobile app, security, malware, detection

I. Introduction

Mobile devices such as smartphones and tablets have been widely adopted for personal and business purposes. However, the popularity of the mobile devices also raises many security issues and challenges. A mobile device is an easy target for cyber criminals due to its central data management. Many threats and attacks have been reported on mobile devices. Among all the threats and attacks, malware (virus, Trojan, and spyware) is one of the greatest threats to mobile security [1]. According to TrendLabs, there were 718,000 malicious Android apps in the second quarter of 2013 [2].

Signature-based detection is the most often used approach to detect malware. However, the testing results from the existing mobile security software are not encouraging. Using a data set including 1260 malware samples, a research team conducted tests using four representative mobile security software: AVG, Lookout, Norton, and Trend Micro. The experiments show that the best case detects 79.6% of malware samples and the worst case detects only 20.2% [3]. The issue with signature-based detection is that apps could change through updated code or modified just enough to throw off the signature for the mobile security software to detect. The approach catches known malware, but fails to stop new or unknown variants in the wild.

In this paper, we propose an automated virtual security testing platform for Android mobile apps. The proposed approach is a server-side malware detection solution. It can utilize both static and dynamic analysis and is a good compensation to the client-side mobile security software.

II. Related Work

Mobile malware malicious infections arise through various techniques such as installing repackaged legitimate apps with malware, updating current apps that piggy back malicious variants, or even driven by a download from an app store. The infections themselves will perform at least one or multiple of the following techniques, privilege escalation, remote control, financial charge, and information collection, etc. The previous stated techniques provide a malicious attacker with a variety of options to utilize a compromised mobile device.

Most client-side malware detection tools are based on signatures. However, the signature-based approach can only be used to detect known malware. Google has introduced a server-side approach, Bouncer, to detect malicious apps before they hit the Google Play Store [4]. This technique is great for apps that are downloaded through the Google Play Store, but is disadvantageous for the users who use third party app stores. A cloud-based mobile malware detection framework is introduced in [5]. The proposed testing platform in this paper can be fully integrated into the framework in [5].

III. An Automated Virtual Security Testing Platform for Android Mobile Apps

In this paper, we propose an automated virtual security testing platform for Android mobile apps. The approach includes three key components: customizing Android OS to include mobile app trace information, creating a virtual testing platform using the customized OS, and developing static and dynamic analyzing techniques for mobile malware detection.

A. Customizing Android OS

A mobile malware usually disguises as a normal application through an app store or a website. Users may unintentionally download the malware to a mobile device. After infiltrating a mobile device, the malware attempts to control its resources, collect data, or redirect the mobile device to a premium account or malicious website.

Mobile malware targets the resources on a mobile device and intents to control the resources and manipulate the data. In order to do these, a mobile app needs to invoke libraries in the application framework, libraries, and even Linux Kernel in the Android OS (Figure 1). The road map of the function calls left by a mobile app contains critical information to detect if a mobile app is malicious or benign. The road map information is usually not available to a client-side malware detection software. However, our approach is novel in the way that we move the client-side malware detection to server-side and utilize a customized Android OS to track the function calls from a mobile app. Thus, we are able to collect these valuable information and use it for malware detection.

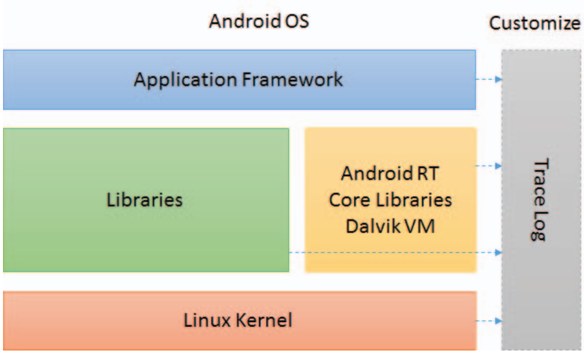

Figure 1.Customize Android OS with Trace Information

A customized Android OS can be built based on the latest release of Android to include trace information in Android application framework, libraries, and Linux Kernel (Figure 1). The trace information will be written in system log files and will be used later for malware detection. The focus of the customization is to include trace information in function calls accessing critical resources on a mobile device, such as, device ID, contact list, call information, etc. The modification occurs only on the Android platform and does not require any changes to a mobile app.

B. *Creating a Virtual Testing Platform using the Customized Android OS*

A virtual Android device could be created using the customized OS. There are a few options which could be used to create virtual devices using Android OS. One option is to utilize Android-x86 in a VirtualBox [6]. Another option is to run Android apps using Genymotion (www.genymotion.com). An ARM emulator is usually required to run a mobile app on a virtual device. Our testing results based on Genymotion show that many mobile apps could be launched on the Android virtual device. The virtual device with the customized Android OS becomes a testing platform for mobile apps.

Using the customized Android OS, a virtual testing platform for mobile apps could be created. The testing platform created not only provides a controlled environment to run a mobile app, but also keeps track of the behavior of a mobile app by monitoring the road map of the library function calls. A thorough analysis of the road map is able to detect if a mobile app is malicious or benign. Testing tools, such as Robotium [7], can be used to simulate user's interactions with the mobile apps and can automate the testing process. Thus, an automated testing platform could be built.

C. *Developing Static and Dynamaic Analyzing Techniques for Mobile Malware Detection*

As discussed, mobile malware targets resources on a mobile device and utilizes Android OS libraries to access the data and manipulate data. A malicious mobile app will certainly leave evidences in the trace information.

A mobile app can be downloaded and tested on the virtual testing platform. Signature-based approach could still be used to detect malware. Further, since the behavior of the mobile app is monitored by the trace information, we are also able to develop dynamic analyzing techniques to detect malware using the trace information collected. In addition, the proposed virtual testing platform is a server-side solution and there are no limitations on CPU, RAM, and battery. More effective approaches for static analysis could be developed and used too.

IV. Summary and Future works

Android is the largest installed base of mobile platform for smartphones and tablets. It also has the largest malicious and high risk applications. In this paper, we propose an automated virtual testing platform for Android mobile apps. The proposed approach can be used to test mobile apps and requires nothing to be changed to a mobile app. Unlike a client-based mobile detection approach which must consider the constraints of a mobile device, the proposed approach is a server-side solution which can utilize more powerful and effective techniques to analyze data and detect malware. The proposed approach is also scalable. It is easy to clone the virtual testing platform in a cloud, launch multiple instances of the testing platform, and use them to test multiple mobile apps simultaneously.

We have demonstrated that many mobile apps can be launched in the virtual testing platform using our Genymotion setup. However, restrictions may also exist due to using the virtual device and the ARM emulator. Emulators usually do not provide 100% of device functionality and it may block some apps to be launched in the virtual platform. In addition, a malicious app may also detect that it runs on a virtual device and reacts to this, e.g., by not acting malicious at all. Our future work includes developing and implementation of the proposed testing platform and testing mobile apps using the testing platform.

ACKNOWLEDGEMENTS

This work is partially supported by NSF Grant No. CNS-1337529. The authors thank the anonymous reviewers for their valuable comments on this manuscript.

References

[1] Y. Wang, K. Streff, and S. Raman, "Smartphone Security Challenges," *Computer (Long. Beach. Calif).*, vol. 45, no. 12, pp. 52–58, Dec. 2012.

[2] Trend Micro, "TrendLabs 2Q 2013 Security Roundup," 2013.

[3] Y. Zhou and X. Jiang, "Dissecting android malware: Characterization and evolution," *IEEE Secur. Priv.*, no. 4, pp. 95–109, 2012.

[4] H. Lockheimer, "Android and Security," *Google Mobile Blog*, 2012.

[5] N. Penning, M. Hoffman, J. Nikolai, and Y. Wang, "Mobile Malware Security Challenges and Cloud-Based Detection," in *the 2014 International Conference on Collaboration Technolgies and Systems*, 2014.

[6] Android-x86, "Porting Android to x86." [Online]. Available: http://www.android-x86.org/.

[7] Robotium, "robotium, The world's leading Android™ test automation framework," 2014. [Online]. Available: https://code.google.com/p/robotium/.