Mobile Security Testing Approaches and Challenges

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

Yazan Alshboul
College of Business and Information System
Dakota State University
Madison, SD 57042
yaalshboul@pluto.dsu.edu

***Abstract*—Mobile devices such as smartphones and tablets are widely used for personal and business purposes. A mobile device may carry sensitive data and becomes an easy target for cyber criminals. Mobile security is thus important. Mobile security testing targets to detect vulnerabilities and malicious apps on a mobile device. In this paper, we present four testing approaches for mobile security: mobile forensic, penetration test, static analysis, and dynamic analysis. A mobile security testing network is further demonstrated in the paper to evaluate the effectiveness of the four testing approaches. Our testing results indicate that mobile security testing tools are still in their early development stages and efforts are desired to improve these tools. We conclude the paper with a summary of mobile security testing challenges and future directions.**

Keywords—Mobile security, testing approaches, challenges

I. INTRODUCTION

Mobile devices, such as smartphones and tablets, have been widely used for personal and business purposes. According to a recent report from KPBC, the number of smartphone users worldwide has risen above 1.6 billion in 2013 [1]. A mobile device is a data centric device and it may carry sensitive data, such as user name, password, contact list, credit card account number, etc. [2]. Thus, mobile devices are easy targets for cyber criminals.

Many threats and attacks have been reported on mobile devices. These threats and attacks include, but are not limited to, virus, sniffing, spamming, spoofing, phishing, etc. [2]. The first mobile phone virus emerged as early as in 2004. Among all the threats and attacks, malware is one of the greatest threats to mobile security [3]. Many efforts have been conducted to prevent malware on mobile devices. Security tools are developed to prevent malicious apps. However, the testing results from the existing mobile security tools are not encouraging [4]. Unlike a desktop computer and a laptop computer, a mobile device has many uniqueness features. A mobile device is a multiple-entrance open system. It is platform-oriented, uses central data management, and is vulnerable to theft and lost. Due to this uniqueness, challenges are also encountered when developing security solutions for mobile devices. These challenges include, but are not limited to, inefficient security solutions, limitations of signature-based mobile malware detection, lax control of third party app stores, and uneducated or careless users, etc. [5].

Mobile security testing targets to detect vulnerabilities and malicious apps on a mobile device. In this paper, we focus on mobile security testing approaches and challenges. We summarize threats and attacks on mobile devices and present a threat model for mobile security. We present four testing approaches for mobile security: mobile forensic, penetration test, static analysis, and dynamic analysis. We further demonstrate a mobile security testing network to evaluate the effectiveness of the four testing approaches. Our testing results indicate that mobile security testing tools are still in their early development stages and many efforts are desired to improve these tools.

The paper is organized as follows: Section II discusses the related work. Section III summarizes mobile threats and attacks. Section IV presents four mobile security testing approaches. Section V demonstrates a mobile security testing network to evaluate the effectiveness of the testing approaches, followed by a summary of mobile security challenges and future directions in Section VI. Section VII summarizes the paper and future works.

II. RELATED WORK

Malware is one of the greatest threats to mobile security [2]. Mobile malware falls in three main categories: virus, Trojan, and spyware [3]. Trojan and spyware are the dominant malware on mobile devices. 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, i.e., 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. TrendLabs estimated that there were 718,000 malicious and high risk Android apps in the second quarter of 2013 [6].

Mobile security is essential for mobile subscribers. In [4], the authors examined four representative mobile anti-virus software: AVG Antivirus Free v2.9, Lookout Security & Antivirus v6.9, Norton Mobile Security Lite v2.5.0.379, and TrendMicro Mobile Security Personal Edition v2.0.0.1294. These security software products use different approaches in their design and implementations. However, the testing results are not encouraging. Out of all 1260 malware samples from 49 malware families, Lookout detected 1003 malware samples in 39 families; TrendMicro detected 966 in 42 families; AVG detected 689 samples in 32 families; and Norton detected the least samples 254 in 36 families [4]. It is apparently that more efforts need to be conducted on mobile security.

Mobile security solutions can be divided into two categories: client-side solutions and server-side solutions. Signature-based malware detection is one of the current main malware detection techniques used on client-side. By analyzing known malware results, this approach prevents installation of known malicious apps on a mobile device. 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 anti-malware application to detect. This approach catches known malware, but fails to stop new or unknown variants in the wild.

Samsung released a security system known as Samsung KNOX for Samsung Android mobile devices [7]. KNOX addresses platform security with a comprehensive three-pronged strategy to secure the system: Customizable Secure Boot, ARM TrustZone-based Integrity Measurement Architecture, and a kernel with built-in security enhancements for Android. In addition, KNOX also includes an application known as Samsung KNOX container. These security enhancements help to protect data on a mobile device. However, KNOX does not provide malware detection. The approach is also device specific and does not help to protect mobile devices from other vendors.

In [8], the authors propose TaintDroid to track the flow of sensitive data on a mobile device. TaintDroid provides real time analysis by introducing tracking information in Android's virtualized execution environment. Using TaintDroid, it is able to identify malicious apps which misuse sensitive data on a mobile device. D2Taint also uses information flow tracking to identify apps which may cause data leakage on a mobile device [9]. Both TaintDroid and D2Taint are client-side solutions. They require customized images to be loaded on a mobile device.

Google has introduced a method of detecting malicious apps before they hit the Google Play Store. Bouncer is a server-side solution which is able to scan mobile apps and detect new mobile malware before they hit the app market [9]. Bouncer has the approach to take newly developed applications and determine if they attempt to send SMS out to malicious sites. This technique is great for the 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 proposed in [10]. The benefit of having a cloud-based detection approach will place all of the work outside of the mobile device. It will prevent mobile device from scanning the apps on the client side and instead push the scanning onto more powerful and efficient systems. The framework utilizes both static analysis and dynamic analysis to detect malware.

III. Mobile Threats and Attacks

A mobile device usually includes the following elements:

- It comes with a pre-installed modern mobile operating system such as iOS, Android, or Windows Mobile.
- It supports a carrier's networks (2G/3G/4G), Wi-Fi networks and Bluetooth networks.
- It may be also a NFC device and can talk to another NFC device.
- It is able to access the Internet. Internet accessibility is provided through either a carrier's networks or a local Wi-Fi network.
- It is capable of running third party applications downloaded from mobile app stores through the Internet.
- It supports MMS messages and has embedded sensors inside.

A. Threats and Attacks

Mobile devices are vulnerable to various threats and attacks. These threats and attacks are summarized in Table 1. Mobile threats and attacks are usually carried out by malware that disguises itself as normal mobile apps such as games, a security patch, or other desirable applications and is then downloaded to a mobile device.

B. Threat Model

A mobile device threat model can be divided into three

Threats and Attacks		Description
Sniffing		Tapping or eavesdropping, e.g., GSM A5/1 cracked
Spamming		Email spam and MMS message spam, e.g., unsolicited MMS
Spoofing		Spoof *"Caller ID"* or MMS *"Sender ID"*, e.g., spoofed MMS messages from 611
Phishing		Steal personal information using a spoofed target mobile application
Pharming		Redirect web traffic to a malicious website and followed by more specific attacks
Vishing		Voice phishing by utilizing VoIP technique
Data leakage		Unauthorized transmission of data, e.g., mobile virus ZitMo
Vulnerabilities of Webkit engine		Vulnerability allowing attackers to crash user applications and execute code, e.g., the Webkit vulnerability revealed by CrowdStrike
DoS	Jamming	Jamming radio channel
	Flooding	MMS message flooding attacks and incoming phone call flooding attacks
	Exhausting	Battery exhaustion attack
	Blocking	Use smartphone blocking functions to disable smartphone

Table 1. Threats and Attacks on Mobile Devices

layers, application layer, communication layer, and resource layer as indicted in Figure 1 [2].

Application layer includes all the applications in a smartphone or a tablet. Malware is usually disguised as a normal application and attracts smartphone subscribers to download. Communication layer includes communication channels to a mobile device, such as, carrier networks (3G/4G), Wi-Fi connectivity, Bluetooth network, NFC, Micro USB port, and MicroSD slot. Malware can spread through any of these communication channels. It will also plan an exit to escape from the mobile device through one of the communication channels. Resource layer includes the flash memory, camera, microphone, and sensors within a smartphone. Since smartphone resources contain sensitive data, malware targets to control these resources and manipulate data from them.

An attack on a smartphone forms a loop from malicious users, through malware, smartphone, premium accounts/malicious websites, back to malicious users. Malware prevention aims to break the attack loop by detecting the malware, isolating or deleting the malware.

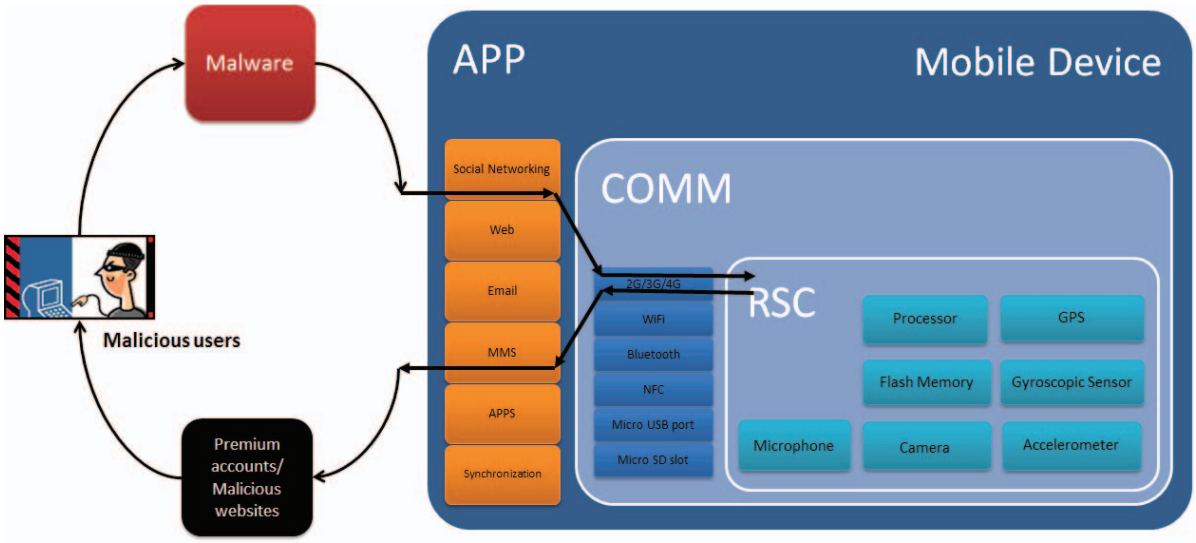

Figure 1. Mobile Security Threat Model

IV. Mobile Security Testing Approaches

Mobile security testing targets to detect vulnerabilities and malicious apps on a mobile device. It focuses on security perspective of a mobile app, e.g., malicious activities, vulnerabilities, security risks, etc. Approaches which can be considered for mobile security testing include, but are not limited to, mobile forensic, penetration test, static analysis, and dynamic analysis.

A. Mobile Forensic

Mobile forensic provides legitimate ways for businesses and users to retrieve and examine data on a mobile device. Mobile forensic tools such as Micro Systemation's XRY and Cellebrite UFED Touch Ultimate can perform logical and physical extraction of data from a wide range of mobile devices even if the data was deleted. When a device is completely opened for scanning, the amount and the type of data shown in the extracted data log is extensive. Data collected consists of copies of SMS/MMS messages, emails, call logs, calendar events, web traffic, bookmarks, pictures, voice mail messages, location information, app data, etc.

B. Penetration Test

Penetration tests can also be used to test security of a mobile app. A mobile app penetration test involves a careful setup of testing environment, launching testing procedures on the mobile device, comparison and analysis of testing results, etc. Many penetration-testing tools can be used. For example, Wireshark can be used to monitor the traffic through the Wi-Fi connectivity. Other tools which can be considered for penetration tests on mobile devices include nmap, Nessus, Metasploit, etc. Kali Linux is a Debian-driven penetration testing platform. It has many preinstalled penetration-testing tools. It works well as a mobile penetration testing platform.

C. Static Analysis

Static analysis is the process of analyzing an application without actual executing the app. Static analysis will review code of an app to find known or suspicious function calls or permissions that deem malicious. There are a few client-side applications which could be used to perform static analysis on an APK file for Android device, e.g., Android Reverse Tools (ART) and Static Android Analysis Framework (SAAf). Static analysis depends on decompiling tools to analyze the code. Static analysis also depends on individuals who inspecting the code. The outcomes of static analysis include signatures which could be used by signature-based malware detection software.

D. Dynamic Analysis

Dynamic analysis is the process of analyzing an application while executing the app in a controlled environment. Dynamic analysis will monitor network traffic and other communications to catch malicious activity. With a powerful dynamic analyzer, apps that attempt to connect out to unknown or malicious sites, or send SMS messages without authorization will be flagged as malicious and consequently be reported as threats. Dynamic analysis is based on the program's behavior and thus no decompiling tools are required. It also has the advantages than the static analysis since malware malicious activities could not be hid.

V. A Mobile Security Testing Network

To further evaluate the effectiveness of these four testing approaches, a testing network is created in Figure 2 to test security of a mobile device.

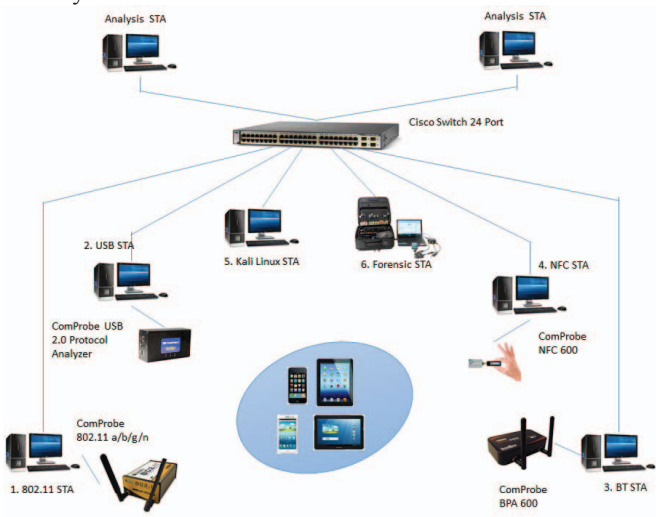

Figure 2. A Mobile Security Testing Network

The testing network includes six testing stations and two analysis stations. These six test stations are listed in Table 2.

Item	Description
1	FTE 802.11 Protocol Analyzer
2	FTE USB Protocol Analyzer
3	FTE Bluetooth Analyzer
4	FTE NFC Protocol Analyzer
5	Kali Linux Test Station
6	Micro Systemation XRY Complete Mobile Forensic System

Table 2 Mobile Security Test Stations

The testing network is able to support all the testing approaches discussed in Section IV. The remaining of the section discusses three testing cases.

A. Idenitfy Sensitive Data on a Mobile Device

Mobile forensic tools provide a great way to learn about what data is stored by mobile apps on a mobile device. For example, using Micro Systemation's XRY, we conducted forensic analysis on mobile devices such as iPhone 4s, DROID RAZAR, iPad 2, Galaxy Tab 3, etc. When a mobile device is accessible (i.e., having the passcode to unlock the device), the information collected by the forensic tools is extensive. We were able to recover WLAN passwords in one of the testing mobile devices. We were also able to retrieve all the instant messages sent through Skype on our testing iPhones. However, mobile forensic tools are not effective when a mobile device is locked. Interrupting bootloader and temporarily placing proprietary tools on a device may help to secure data acquisition. However, this approach only works for certain devices and does not work for the new versions of iOS and Android devices.

B. Conduct Penetration Tests on Mobile Devices

We are also able to conduct penetration tests on mobile devices using the Kali Linux station. A penetration test on a mobile device usually includes three steps, information gathering, vulnerability assessment, and exploitation. For example, using nmap in the Kali Linux station, we can start an information gathering process on mobile devices. nmap works well for desktop computers and laptop computers. However, the information collected from mobile devices are limited. For example, using nmap on an iPhone 6, we are able to collect the IP address, OS information, open ports, etc. nmap successfully detected one open tcp port, 62078/iPhone-sync, on the iPhone 6. However, nmap failed to detect OS version number.

C. Detect Data Leakage

The 802.11, Bluetooth, and NFC protocol analyzers provide sniffing tools to monitor the communication channels on a mobile device. By creating an isolated environment for a mobile app and monitoring these communication channels, we are able to track the behavior of a mobile app and look malicious activities in the traffic logs. This approach provides a great way to study the behavior of a mobile app on a mobile device. However, it also faces the challenges when the data is encrypted.

VI. Mobile Security Testing Challenges anf Future Directions

Mobile security testing is a challenging issue. There are no effective approaches to test mobile security so far.

A. Mobile Security Testing Challenges

1) Signature-based malware detection techniques can be spoofed easily.

Tools such as ART could be used to decompile and recompile an APK file. A legitimate app could be easily repackaged by a malicious user to change its signature and spoof malware detection software. According to [4], most existing Android malware (86.0%) were repackaged through other legitimate apps.

2) Mobile forensic tools are ineffective when security access is enabled on a mobile device.

Mobile forensic tools usually need to get access of a mobile device before they can scan it. If the device has access security enabled (passcode, biometric security, pattern code, etc.), it may not be possible for the mobile forensic tools to extract data from the device. In addition, flash memory is also encrypted. Physical scanning or cloning a mobile device flash memory without access code also does not produce much meaningful data.

3) Penetration testing tools need to be improved for mobile devices.

Penetration testing is an effective approach to exploit vulnerabilities on laptop and desktop computers. However, our tests show that current penetration testing tools are not effective on mobile devices. Some challenges we encountered include, but are not limited to, not many open ports available on a mobile device, not many useful information collected if using a secured Wi-Fi network, etc.

4) Static analysis requires code to be available and involves many manual processing.

Static analysis is less efficient and could be obfuscated by malware authors. Automatic static analysis is desirable. However, it also faces challenges as the signature-based malware detection software. It is likely that malware authors could use repackage techniques to spoof an automatic malware static analyzer too.

5) Dynamic analysis requires full examines on all mobile resources and is both computational and time extensive.

Dynamic analysis is not suitable as client-side software. Dynamic analysis requires a controlled testing environment to monitor app behaviors. The desired controlled environment is often not available on the client side. Further, monitoring app behavior is only the first step and detecting malware is the goal.

6) Lack of effective approaches to detect data leakage when data is encrypted

Four testing approaches are presented in Section IV. However, none of them is effective when data is encrypted. Many mobile apps use HTTPS to post data. This makes traffic analysis impossible. As discussed in the threat model, malicious apps target the sensitive data on a mobile device. Thus, sensitive data monitoring as in [8], [9] might be useful.

7) A mobile app may hide its malicious activitiy when mobile security software is present

A malicious app may detect the existence of mobile security software and react to this, e.g., by not acting malicious at all. This may limit the use of mobile security software on both client-side and server-side. Further, server-side mobile security solutions often use virtual environments to test mobile apps. Such a virtual environment often utilizes an emulator to emulate the real device. However, the functionality of the emulator is often limited and could not completely replace the real device. This may limit the number of mobile apps running on a virtual environment.

B. Future Directions

The challenges we face in the mobile security testing also point out a few future research directions.

1) Integrate both client-side and server-side security solutions to prevent malicious apps

Client-side mobile security software is certainly essential to mobile security. However, due to the constraints of CPU, memory, and battery on a mobile device, it is difficulty to launch computational and power intensive security software on a mobile device. Server-side security solutions do not have these limitations and it will be a good compensation for client-side mobile security software.

2) Compensate signature-based mobile security software with senstive data monitoring

Signature-based mobile security software is easy to be spoofed. Sensitive data monitoring could be used to compensate the signature-based mobile security software. However, any solutions on the client-side must also be power efficient and meet the constraints on the mobile devices.

VII. Conclusion and Future works

Mobile security testing targets to detect vulnerabilities and malicious apps on a mobile device. It is essential to all mobile subscribers. However, mobile security is also a challenging issue due to the uniqueness of mobile devices. This paper focuses on mobile security testing and we present four testing approaches for mobile security: mobile forensic, penetration test, static analysis, and dynamic analysis. We also demonstrate a mobile security testing network to evaluate the effectiveness of the four testing approaches. Our testing results indicate that mobile security testing tools are still in their early development stages and many efforts are desired to improve these tools. Our future work includes more testing and evaluation of mobile security using our testing network.

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] M. Meeker and L. Wu, “Internet Trends,” 2014.
[2] Y. Wang, K. Streff, and S. Raman, “Smartphone Security Challenges,” *Computer (Long. Beach. Calif).*, vol. 45, no. 12, pp. 52–58, Dec. 2012.
[3] Juniper Neworks, “2011 Mobile Threats Report,” 2012.
[4] Y. Zhou and X. Jiang, “Dissecting android malware: Characterization and evolution,” *IEEE Secur. Priv.*, no. 4, pp. 95–109, 2012.
[5] Y. Wang, J. Wei, and K. Vangury, “Bring Your Own Device Security Issues and Challenges,” in *The 11th Annual IEEE Consumer Communications & Networking Conference*, 2014.
[6] Trend Micro, “TrendLabs 2Q 2013 Security Roundup,” 2013.
[7] Samsung, “Samsung KNOX,” 2014. [Online]. Available: http://www.samsung.com/global/business/mobile/solution/security/samsung-knox. [Accessed: 01-Jan-2014].
[8] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, “TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones,” *Osdi ’10*, vol. 49, pp. 1–6, 2010.
[9] B. Gu, X. Li, G. Li, A. C. Champion, Z. Chen, F. Qin, and D. Xuan, “D2Taint: Differentiated and dynamic information flow tracking on smartphones for numerous data sources,” in *Proceedings - IEEE INFOCOM*, 2013, pp. 791–799.
[10] 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.