A Survey of Security and Privacy Protection in Mobile Devices

I

Brian Krupp
Cleveland State University, USA

Wenbing Zhao
Department of Electrical and Computer Engineering, Cleveland State University, USA

Nigamanth Sridhar
Cleveland State University, USA

INTRODUCTION

Mobile systems have experienced rapid adoption since their inception. Starting from a device that was designed to allow a consumer to make a phone call from anywhere to today having a device that is able to download data at speeds much faster than a home Internet service (Mossberg, 2012). With the introduction of tablets, the growth of mobile computing devices is growing at an even faster rate where the number of mobile-connected tablets tripled in 2011 (Perez, 2012). This growth is challenging us with a new paradigm as consumers shift away from more traditional platforms such as desktops and laptops, where in 2011 sales of smartphones overtook shipments of PCs (Symantec, 2012, p. 13). With this new paradigm, there is a growing need to ensure that the security of these platforms and associated services are sound. Securing these devices however has different challenges than more traditional platforms: computing capabilities, limited power, mobility, and desired user experience constrain methods that can be implemented to achieve this goal. This article will look at the inherent security of the two most popular mobile platforms, Android and iOS, the current issues that mobile platforms face in ensuring the security of the platform and privacy of the consumer's data, and methods that have been proposed and implemented in achieving the security of these devices.

BACKGROUND

Intrusion Detection and Prevention

Several methods have been proposed to protect mobile devices that have been used on traditional computing platforms. A common system to implement is an Intrusion Detection System (IDS). Two common methods to detect intrusions are signature based and anomaly based. A *signature based IDS* checks for a sequence of bytes within executable code that has been identified as malware from a signature provider. An *anomaly based IDS* checks for events and activity on the system to see if there is an anomaly from how a system is expected to operate. An IDS does not prevent intrusions but an Intrusion Prevention System (IPS) can. Preventing an intrusion from occurring is a challenge because a decision needs to be made real time if the operation to occur is valid or if it is part of an intrusion. This decision however should not affect the system from operating normally as the verification of the operation is typically blocking, meaning that future operations will not continue until verification is completed. This verification process prevents both a valid or intrusion based operation from occurring until the current operation being analyzed is verified to not be a threat. If this verification process is not completed fast enough, it may affect user experience or the system from operating to expected levels. This makes intrusion prevention systems less ideal for mobile systems as they are limited in computing power and need to make a decision immediately to prevent

DOI: 10.4018/978-1-4666-5888-2.ch414

Table 1. Comparison of several inherent security attributes of iOS and Android

		iOS	Android
Encryption	Key Size	256	128
	Protection Classes	Device first unlocked, Device unlocked, File open, None	Device first unlocked
Code Signing	Required	Yes	Yes
	Certificate Authority Verification	Yes (developer and Apple)	None
	Permissions	Ability to select (entitlements)	All or nothing
	Inter Process Communication	Custom URL schemes	Several methods supported
	Application Stores	One	Many

a poor user experience which has contributed to their fast adoption.

Inherent Security Controls for iOS and Android

The two most popular mobile operating systems iOS and Android (StatCounter, 2012) have security controls today to help prevent malware and intrusions on the device, as well as protect the privacy of the consumers data. We will briefly review those controls here (Table 1).

In Apple's iOS operating system, applications are signed using certificates so that only applications signed by Apple can run on iOS. Apple signs and publishes third party applications developed by registered developers once an application has been reviewed by Apple (Apple, 2012, p. 5). Code signing, application review, and a single repository for applications helps prevent malware from entering iOS devices. When an application is executed on iOS, iOS sandboxes the application to ensure it can only access files that reside in its home directory. It also restricts the application to the entitlements that were granted when the application was created and approved by the user (Apple, 2012, p. 6). Entitlements allow applications to access data and services outside its sandbox such as a user's contacts and location. Applications are prevented from sharing data with other applications unless they use Apple's Custom URL Schemes (Apple, 2012, p. 6). Data stored on the device can utilize Apple's Data Protection API to ensure data is encrypted using the advanced encryption standard (AES). There are three ways a file can be protected, one is it remains encrypted when a device is locked, another when a file is not open, and lastly when the device has not been unlocked from its last boot. To ensure each file is protected, a 256 bit class key is derived from the devices unique identifier and the user's passcode, which is also used to protect an additional 256 bit key generated for each file that exist on the filesystem. Additionally, a random key protects the metadata of the filesystem which is first generated when iOS is first installed or when the device is wiped by the user (Apple, 2012, pp. 7-8).

Apple's file encryption (Figure 1) has proven to be difficult for forensic teams to crack (Garfinkel, 2012). Apple also supports address space layout randomization (ASLR) which randomizes the location of executable code each time an application is run (Apple, 2012). This has helped protect against techniques such as return oriented programming (ROP) that can exploit the operating system (Miller, Blazakis, Zovi, Esser, Iozzo, & Weinmann, 2012, p. 212). Additionally with data execution prevention (DEP) there is no generic way to write an exploit for the operating system, which means that there are typically two vulnerabilities needed, one to obtain code execution, and another to leak a memory address (Miller, Blazakis, Zovi, Esser, Iozzo, & Weinmann, 2012, p. 8).

On Android, all applications need to be code signed by the developer, otherwise they will be rejected by Google Play or the installer on the phone. However, certificate authority (CA) verification is not done and applications can be signed by anyone that generates a self-signed certificate (Android, 2012). Also Google supports any third-party application as long as it is code-signed. They do offer their market for applications, Google Play which offers security services such