TrubiZone

Securing Critical Mobile Applications for Android Using ARM TrustZone

Tiago Brito, tiago.de.olveira.brito@tecnico.ulisboa.pt

Instituto Superior Técnico

Abstract. With the ever-growing number of connected devices worldwide, and with a more conscious and sharing society, mobile devices are becoming interesting data banks. With such an impact in our lives, mobile developers must focus on developing secure and privacy aware applications to protect users and services. Today's mobile operating systems do not offer fully secure methods to support critical applications, such as e-Health, e-voting and e-money, by not taking advantage of secure hardware technology like TrustZone. This paper proposes a new model for the development of critical mobile applications and a system based on TrustZone implementing it.

Keywords:
TrubiZone, Security, Privacy, e-Health, Mobile Devices, Android, TrustZone

1 Introduction (2/3pgs)

According to a Comscore's whitepaper [3] from early 2015, with key statistics for the U.S. market, mobile devices, such as smart-phones and tablets, dominate digital media time over the Personal Computer (PC), with the trend being to continue raising. This shows a future where mobile devices might be the predominant device.

Present in our everyday personal and professional lives, mobile applications (apps) start to handle privacy and security-sensitive data, notably photos, health and banking information, location and general documentation, thus becoming an attractive target for attacks.

This paper will focus, for the remaining of this project, on the e-Health market, also known as Mobile Health (mHealth). In 2013, Research2Guidance [2] reported the existence of more than 97.000 mHealth apps across 62 app stores, with the top 10 mHealth apps generating up to 4 million free and 300.000 paid downloads per day. This market is expected to grow even further, from $6.21 billion in revenue in 2013 to $23.49 billion by 2018, according to a report from MarketsAndMarkets [1].

The health sector is an interesting market for attackers due to its information value. According to Reuters [1], "medical identity theft is often not immediately

[1] http://www.reuters.com/article/2014/09/24/us-cybersecurity-hospitals-idUSKCN0HJ21I20140924

identified by a patient or their provider, giving criminals years to milk such credentials. That makes medical data more valuable than credit cards".

Such valuable information as lead to critical attacks on hospital networks around the world, consequently leaking or stealing health records of millions of patients. In September 2014, a group of hackers attacked the network of the University of California, Los Angeles (UCLA)'s Hospital accessing computers with sensitive records of 4.5 million people. According to Cable News Network (CNN) [2], among the stolen records were the names, medical information, Social Security numbers, Medicare numbers, health plan IDs, birthdays and physical addresses of UCLA's patients.

In the mobile context, data is even more exposed and vulnerable because of the portability of such devices, unregulated management of sensitive medical information, the sharing of information to third-party advertisers by device manufacturers and mobile app developers and security flaws on medical or consumer device software.

To protect sensitive data, apps rely on ad-hoc Operating System (OS) and application-level methodologies, which in most cases depend upon a very complex Trusted Computing Base (TCB) code. Unlike most popular mobile platforms based on Android [3], iOS [4] or Windows [5], which have a TCB comprising of a full featured OS and system libraries with millions of lines of code (LOC), in a small, dedicated runtime, consisting of only the necessary code to run the intended apps, is easy to ensure the absence of exploitable code vulnerabilities.

This project aims to fill the security gap in the mobile application market by proposing TrubiZone, a development system with a small, dedicated TCB which, by using ARM TrustZone technology, allows app developers to execute parts of the application logic in a trusted environment isolated from the OS, thus supporting the development of secure mobile applications independent from the full blown platforms and its inherent vulnerabilities.

TrubiZone will be based on Genode [6] and implemented upon a Freescale i.MX53 START development board, which supports ARM's TrustZone.

To illustrate TrubiZone's functionality an mHealth will be implemented.

Two worlds etc...

In summary this work expects to contribute with:

- The design of a novel security system, based on TrustZone, for development of secure mobile applications.
- Implementation, on a development board, of a prototype of this new security framework.
- Implementation of a mobile health application using the prototyped framework.
- Assessment of the prototype and mHealth app.

[2] http://money.cnn.com/2015/07/17/technology/ucla-health-hack/
[3] https://www.android.com/
[4] http://www.apple.com/ios/
[5] http://www.microsoft.com/en-us/windows
[6] http://genode.org/index

The remainder of this document proceeds as follows. Section 2 highlights the related work on mobile security, mobile health and TrustZone technology.

2 Related Work (17pgs)

Fig. 1. caption

References

1. Mobile health apps & solutions market by connected devices (cardiac monitoring, diabetes management devices), health apps (exercise, weight loss, women's health, sleep and meditation), medical apps (medical reference) – global trends & forecast to 2018. Technical report, MarketsAndMarkets, 09 2013. http://marketsandmarkets.com/.
2. Mobile Health Market Report 2013-2017. Technical report, Research2Guidance, 03 2013. http://research2guidance.com/.
3. Digital Future in Focus. Technical report, Comscore Inc., 03 2015. http://www.comscore.com/Insights/Presentations-and-Whitepapers/2015/2015-US-Digital-Future-in-Focus.

A Appendix

Appendix files and refs will go here. Such as your thesis work scheduling.

A.1 Work Scheduling Example

Simple work schedule is presented in Table 1. You can do something more fancy link a Gantt chart or whatever.

Table 1. Work Scheduling

Month	Work
February	Do Stuff
February	Do Stuff
March	Do Stuff
April	Do Stuff
May	Do Stuff
May	Do Stuff
June	Do Stuff
July	Do Stuff

Table of Contents