CRYPTOMAThIC

Overview of App & Code Hardening for Mobile Banking Apps

by Martin Rupp (guest) on 13. March 2020

Mobile App Security Banking MASC RASP

Application hardening usually consists in processing an already developed application, and transforming it so to make it difficult / impossible to reverse engineer and tamper.

This is usually combined with secure coding (code hardening) so that the attack surface is reduced as much as possible. There are techniques to measure the attack surface of an application. The **Relative Attack Surface Quotient** (RSQ) is one of them.

Passive App Hardening

Passive hardening is targeting the attacks based on static analysis, especially decompilation. These techniques mainly use obfuscation. The goal is to make it extremely hard to convert the application to an understandable code, even in pseudo-code. This doesn't change the logic and behavior of the application itself.

Active App Hardening

In real-life attacks, hackers are using much more than static analysis. They use dynamical analysis; running the mobile application in a simulated environment with debuggers and emulators. Active application hardening creates runtime protection and modifies the application's behavior so to respond accordingly to dynamical analysis tools if they are detected.

Such protections usually include:

- Anti-debug
- Anti-rooting/jailbreak
- Anti-repackaging of the mobile application

More on mobile application hardening here...

Code Hardening

Code hardening consists of preventing security holes in the code of an application so that even if the application could be reversed, no flaws could be exploited.

Some of the basic techniques used in code hardening are the following:

- Preventing all sorts of overflows by checking inputs, and if possible, using a defensive programming methodology, such as code constraints.
- Preventing buffer overflow
- Avoiding using interpreters and passing data to them
- Securing all calls to system variables
- Using formal methods

Code Signing

Code signing is an efficient way of hardening code. This allows the operating system to check that an application was not modified and is the original application. This also prevents fetching "rogue" updates by checking the authenticity of such application patches and upgrades.

Hardening is not optional

Application hardening and code hardening is a valuable technique to prevent source code from being stolen and applications from being reversed. In the context of mobile banking and mobile payment applications, this is actually not something that is optional.

For instance, regarding the protection of intellectual property (IP), the **U.S Defend Trade Secrets Act** (DTSA) and **EU Directive 943** both voted in 2016 to explicitly deny treating data obtained through reverse engineering as "secret data." As a consequence, there is no legal protection for the owners of a mobile banking application against third parties reversing their applications and publishing parts or totality of the source code. Therefore, these laws create new needs for very strong source code obfuscation. Additionally, the **PCI Council** now requires risk control systems to actively prevent the debugging and rooting/jailbreak of mobile phones where banking applications are installed:

"Mobile payment-acceptance applications should be hardened to prevent unintended logical access or tampering with the app such as code injection or reverse engineering. Numerous techniques can be used for hardening of the mobile payment-acceptance application that will reduce the attack surface" [1].

In regards to code hardening, the PCI Council specifically recommends that mobile banking applications are developed using secure coding techniques and guidelines, such as the **Payment Application Data Security Standard** (PA-DSS), the **CERT** Secure Coding Standards, and **ISECOM**, **OSSTMM**, or **OWASP** secure coding documentation.

Download White-Paper
Securing Mobile Banking Apps with MASC
An introduction to Cryptomathic Mobile App Security Core (MASC)
Download

References and Further Reading

- *Read more articles about application security for mobile banking applications (2018 - today), by Martin Rupp, Stefan Hansen and more*
- *Internet Security Threat Report, Volume 24 (February 2019), by Symantec Corporation*
- *MASC Mobile App Security Core (2019), Web page by Cryptomathic*
- *[1] PCI Mobile Payment Acceptance Security Guidelines for Developers version 2 (September 2017), by the Emerging Technologies, PCI Security Standards Council*