Technische Universität München

Preventing Repackaging of Android Apps Using Integrity Checking

Master Thesis

Supervisors: Prof. Dr. Alexander Pretschner, Aleieldin Salem
Email: pretschn, salem @ in.tum.de
Phone: +49 89 289 – 17, 314
Starting date: Immediately

Fakultät für Informatik
Lehrstuhl 4
Software and Systems Engineering
Prof. Dr. Alexander Pretschner

Boltzmannstraße 3 85748
Garching bei München

Tel: +49 89 289 17885,
+49 89 289 17314

Web: https://www4.in.tum.de

Context

Repackaging is a technique that has been adopted by attackers–leveraging the ease of disassembling and reverse engineering of Android apps–to graft arbitrary (malicious) payloads into legitimate trusted apps (e.g., Whatsapp) [1, 5]. To counter this phenomenon, researchers have been implementing preventive measures to hinder the repackaging of Android apps [2, 4]. Nevertheless, such approaches hinge on networks of homogeneous integrity checkers that are decoupled from the original app functionality. This design facilitates statically locating such integrity checkers and enables attackers to remove them without jeopardizing the original app's functionalities. In this context, there is a need for more advanced anti-repackaging techniques that are intertwined with the protected app's code and renders the static patching of the protected app infeasible for the attackers.

Goal

The primary objective of this thesis is to design, implement, and evaluate an Android anti-repackaging technique using integrity checking techniques. The technique, named *Praetorian*, should take into consideration the shortcomings of the existing solutions, and attempt to overcome them. To do so, the proposed technique is based on replacing the destinations of inter-procedural calls in the app, with dynamic values calculated during runtime. The replacement of the aforementioned calls is carried out prior to compiling the code (e.g., by the marketplace on which the app resides), and depends on a value unique to the target device (e.g., Android device ID, IMEI, serial number, etc.); we envision this unique value to be forwarded to the marketplace by the device requesting to download the app.

In order to circumvent this design, attackers that alter the app's code need to predict or assume the unique values corresponding to the devices they target–from among billions of possibilities–with their repackaged apps, and re-calculate all the call destinations. Furthermore, the process of requesting their own versions of the protected apps (i.e., with their own unique values), finding the call destinations during runtime, and patching them is a time-consuming process that is hindered by the need to convince the user to install the patched version of the app, instead of the marketplace version.

To evaluate the implemented technique, we plan to utilize benign apps (e.g., downloaded from the Google Play marketplace), protect them using *Praetorian*, repackage them using Repackman [3], and evaluate (a) whether the technique manages to capture the repackaging attempt, and (b) whether the technique affects the functionality and/or performance of the original app.

Technische Universität München

Fakultät für Informatik
Lehrstuhl 4
Software and Systems Engineering
Prof. Dr. Alexander Pretschner

Boltzmannstraße 3 85748
Garching bei München

Tel: +49 89 289 17885,
+49 89 289 17314

Web: https://www4.in.tum.de

Work-plan

1. Enumerate the types of repackaging attacks usually launched against anti-repackaging techniques.
2. Design *Praetorian*'s anti-repackaging technique.
   a. Assess the feasibility of the proposed approach.
   b. Identify the required technologies and tools.
   c. Revise the enumerated repackaging attacks.
3. Implement *Praetorian*.
4. Evaluate the implemented technique.
   (a) Identify evaluation criteria and design experiments.
   (b) Prepare the evaluation dataset.
5. Document the design, implementation, and evaluation of *Praetorian*.

Required Skills

We are looking for a motivated student with the following expertise:

- Very good Java programming skills.
- Very good understanding and familiarity with the Android platform.
- Good C/C++ programming skills.
- Basic understanding of security fundamentals.
- Self motivation and ability to work independently.

Deliverables

- The source-code and design of the implemented technique.
- The evaluation dataset used to evaluate *Praetorian*.
- A thesis document in accordance with TUM's guidelines.

References

[1] Li Li, Daoyuan Li, Tegawendé F Bissyandé, Jacques Klein, Yves Le Traon, David Lo, and Lorenzo Cavallaro. Understanding android app piggybacking: A systematic study of malicious code grafting. *IEEE Transactions on Information Forensics and Security*, 12(6):1269–1284, 2017.

[2] Lannan Luo, Yu Fu, Dinghao Wu, Sencun Zhu, and Peng Liu. Repackage-proofing android apps. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 550–561. IEEE, 2016.

[3] Aleieldin Salem, F. Franziska Paulus, and Alexander Pretschner. Repackman: A tool for automatic repackaging of android apps. In *Proceedings of the 1st International Workshop on Advances in Mobile App Analysis*, pages 25–28. ACM, 2018.

[4] Lina Song, Zhanyong Tang, Zhen Li, Xiaoqing Gong, Xiaojiang Chen, Dingyi Fang, and Zheng Wang. Appis: Protect android apps against runtime repackaging attacks. In *Parallel and Distributed Systems (ICPADS), 2017 IEEE 23rd International Conference on*, pages 25–32. IEEE, 2017.

[5] Yajin Zhou and Xuxian Jiang. Dissecting android malware: Characterization and evolution. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 95–109. IEEE, 2012.