Technische Universität München

Detection of Android Repackaged Malware with Active Learning

Bachelor 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 has been increasingly adopted by authors of Android malware to thwart detection techniques [1, 2, 6]. The primary threat that repackaged malware poses is undermining user trust in legitimate apps, their developers, and the app distribution infrastructure, which can potentially have devastat ing effects on the entire Android ecosystem.

Consequently, the research community has been working towards devising methods to detect Android repackaged malware. However, the main problem facing the research community working on devising techniques to detect this breed of malware is the lack of ground truth that pinpoints the malicious segments grafted within benign apps. In other words, researchers do not have access to samples of malicious and benign functionalities within Android apps. Without this crucial knowledge, it is difficult to train reliable classifiers able to effectively classify novel, out-of-sample repackaged malware, especially under adversarial settings in which malware authors design their instances to mimic benign apps relied on by detection methods as references for benign behaviors.

To circumvent this problem, we argue that reliable classifiers can be trained to detect repackaged malware, if they are allowed to request new, more accurate representations of an app's behavior. This learning technique is referred to as *active learning* [4].

Goal

In this thesis, we propose the usage of active learning to train classifiers able to cope with the ambiguous nature of repackaged malware under different detection settings viz., conventional and adversarial [3]. Our approach iterates over segments of static and dynamic representations of Android apps (e.g., decompiled source code and traces of API calls), extracts numerical features from such representations (e.g., TF-IDF), and trains a machine learning classifier. To verify whether the utilized segments, indeed, depict the malicious and benign natures of the apps, we validate the trained classifier using those segments. For misclassified apps (e.g., malicious app classified as benign), we retrieve different segments from their representations, re-train the classifier, and re-validate. This process is repeated until the best possible training accuracy is achieved.

The resulting classifier, we argue, should be trained using segments that reveal the true nature of the apps used during training. Given how common code reuse is, we argue that such classifier should capture (a subset of) malicious and benign behaviors in general. To evaluate our approach, we plan to utilize three malware datasets viz., Malgenome [6], Piggybacking [1], and AMD [5], to check whether our approach (a) manages to correctly classify Android repackaged malware under the different detection settings defined in [3], and (b) manages to isolate the code segments that reveal the malignance of such elusive breed of malware.

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. Gather static and dynamic representations of Android apps.
   a. Decompile and retrieve source code for static experiments.
   b. Execute Android apps and stimulate them for dynamic experiments.
2. Design the training, validation, and test processes.
   a. Decide upon the features to extract from app representations and classifiers to utilize.
3. Implement the proposed approach.
4. Evaluate the implemented technique.
   a. Identify evaluation criteria and design experiments.
   b. Prepare the evaluation datasets.
   c. Perform static and dynamic experiments on the aforementioned datasets.
5. Document the design, implementation, and evaluation of the proposed approach.

Required Skills

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

- Very good understanding and familiarity with the Android platform.
- Good programming skills (particularly Python).
- Good understanding of machine learning fundamentals.
- Self motivation and ability to work independently.

Deliverables

- The source-code and design of the implemented tool(s).
- The data used during evaluation and the recorded results.
- 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] Symphony Luo and Peter Yan. Fake apps: Feigning legitimacy. Technical report, Trend Micro, 2014.

[3] Aleieldin Salem and Alexander Pretschner. Poking the bear: Lessons learned from probing three android malware datasets. In *Proceedings of the 1st International Workshop on Advances in Mobile App Analysis*, pages 19–24. ACM, 2018.

[4] Simon Tong. *Active learning: theory and applications*. Stanford University, 2001.

[5] Fengguo Wei, Yuping Li, Sankardas Roy, Xinming Ou, and Wu Zhou. Deep ground truth analysis of current android malware. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 252–276. Springer, 2017.

[6] Wu Zhou, Yajin Zhou, Xuxian Jiang, and Peng Ning. Detecting repackaged smartphone applications in third-party android marketplaces. In *Proceedings of the second ACM conference on Data and Application Security and Privacy*, pages 317–326. ACM, 2012.