UNIVERSITÉ DE RENNES 1

Direction de la Recherche et de l'Innovation

Appel à projets - Année 2022

Contrats doctoraux «Ordinaires»

ECOLE DOCTORALE : MathSTIC **POLE DE RECHERCHE : IRISA**

Unité de recherche d'accueil : IRISA

Unité de Recherche : IRISA, UMR 6074

Equipe : WIDE

Encadrement

Directeur de thèse : David Bromberg **Unité de recherche** : IRISA UMR 6074

Employeur/Établissement d'origine : Université de Rennes 1

HDR : x oui ☐ non **Nb de thèses en cours** : 0

Mail : david.bromberg@irisa.fr **Tél.** :

Co-directeur de thèse : Julia Lawall **Unité de recherche** : Inria Paris

Employeur/Établissement d'origine : Inria

HDR : xoui ☐ non **Nb de thèses en cours** :

Mail : julia.lawall@inria.fr **Tél.** :

Sujet

Titre : The next gen on device ANDROid MAlware protection

Mots clés : Android, Machine Learning, Malware, Système

Cotutelle internationale (oui/non) : non **Si oui, préciser l'établissement et le pays de rattachement** :

Descriptif (15-30 lignes) :
Contexte socioéconomique et scientifique ; Les hypothèses et questions posées ; Les grandes étapes de la thèse et démarche ; Approches méthodologiques et techniques envisagées

Context. Android is now the most used mobile platform, with a market share of 86%. Its Google Play store holds 3.3 million applications, with a rate of more than 50,000 submissions per month. Estimations indicate that more than 75 billion Android apps were downloaded in 2016. This enormous popularity has, however, led to Android becoming a prime target for hackers. Malware can steal personal data, hold critical data hostage, trick users into making unintended purchases, etc. As illustrated by the recently detected Pegasus spyware, such malware can lurk on a mobile device for years.

Objectives. The main objective of this thesis is to counter the proliferation of Android malware, and propose a practical and effective Android malware detection system. The current state of the art in Android malware detection relies on static analysis [6], which collects software features (such as counts of the number of various library calls) in the code without running it, thus avoiding the need to request specific privileges on a modified Android system. However, this technique is known for its limitations if applications are obfuscated and/or if their malicious code is downloaded dynamically at runtime. Another alternative to overcome these limitations is to rely on dynamic analysis, that enables to [6] analyze the actual behavior of the application during its execution. However, due to its inherently high resource consumption, most dynamic analysis is performed in a lab environment [1, 3, 4, 5] rather than on off-the-shelf devices. Further, it requires specific privileges on a modified Android system [7], which are not possible on consumer phones without breaking the warranty and being an expert.

Challenges. The challenge is to provide a practical and effective Android malware detection system designed around dynamic analysis, i.e., running applications in a realistic sandboxed environment, tracing the operations performed, and analyzing the resulting traces to learn the characteristics that indicate that malware is present. For training, this analysis must furthermore be done at a very large scale, to capture the very wide variety of both

benign and malware apps. We emphasize that the state of the art is typically trained on datasets that are small, scattered, and not representative of the latest evasion techniques used in the malware community, drastically reducing the effectiveness of the developed techniques [2].

Accordingly, two major challenges must be overcome :
— Providing an off-the-shelf Android-device execution profiler that an anti-virus could leverage for implementing dynamic analysis techniques;
— Providing a meaningful and sufficient large-scale dataset through the use of a malware generator, built over the state-of-the-art work on generative adversarial networks (GAN) applied to code instrumentation, and code analysis techniques, to automatically generate functioning malware that is targeted to evade detection.

Candidate Profile. The candidate will have experience with systems programming and will ideally have experience with the Linux kernel, the Android framework, ARM assembly language, and machine learning.

Références

[1] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "Dynalog : an automated dynamic analysis framework for characterizing android applications." https://doi.org/10.1109/CyberSecPODS.2016.7502337, 2016.
[2] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Dos and don'ts of machine learning in computer security," in USENIX Security, 2022.
[3] M. Backes, S. Bugiel, O. Schranz, P. von StypRekowsky, and S. Weisgerber, "Artist : The android runtime instrumentation and security toolkit," in EuroS&P. IEEE, 2017, pp. 481–495.
[4] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid : An information-flow tracking system for realtime privacy monitoring on smartphones," in USENIX OSDI, 2010, pp. 393–407.
[5] P. Lantz, "Droidbox : An Android application sandbox for dynamic analysis." in Master's Thesis at Department of Electrical and Information Technology, 2011.
[6] L. Li, T. F. Bissyandé, M. Papadakis, S. Rasthofer, A. Bartel, D. Octeau, J. Klein, and L. Traon, "Static analysis of Android apps : A systematic literature review," Information and Software Technology, vol. 88, pp. 67 – 95, 2017. [Online]. Available : http://www.sciencedirect.com/science/article/pii/S0950584917302987
[7] L. Qiu, Z. Zhang, Z. Shen, and G. Sun, "Apptrace : Dynamic trace on Android devices," in 2015 IEEE International Conference on Communications (ICC), 2015, pp. 7145–7150.