applied sciences

MDPI

Article

Call Graph and Model Checking for Fine-Grained Android Malicious Behaviour Detection

Giacomo Iadarola [1,*,†], Fabio Martinelli [1,†], Francesco Mercaldo [2,1,*,†] and Antonella Santone [2,†]

[1] Institute for Informatics and Telematics, National Research Council of Italy, 56124 Pisa Italy; fabio.martinelli@iit.cnr.it

[2] Department of Medicine and Health Sciences "Vincenzo Tiberio", University of Molise, 86100 Campobassoo, Italy; antonella.santone@unimol.it

* Correspondence: giacomo.iadarola@iit.cnr.it (G.I.); francesco.mercaldo@unimol.it or francesco.mercaldo@iit.cnr.it (F.M.)

† All authors contributed equally to this work.

Received: date; Accepted: date; Published: date

Abstract: The increasing diffusion of mobile devices, widely used for critical tasks such as the transmission of sensitive and private information, corresponds to an increasing need for methods to detect malicious actions that can undermine our data. As demonstrated in the literature, the signature-based approach provided by antimalware is not able to defend users from new threats. In this paper, we propose an approach based on the adoption of model checking to detect malicious families in the Android environment. We consider two different automata representing Android applications, based respectively on Control Flow Graphs and Call Graphs. The adopted graph data structure allows to detect potentially malicious behaviour and also localize the code where the malicious action happens. We experiment the effectiveness of the proposed method evaluating more than 3000 real-world Android samples (with 2552 malware belonging to 21 malicious family), by reaching an accuracy ranging from 0.97 to 1 in malicious family detection.

Keywords: malware; model checking; formal methods; security; Android; mobile

---

Funding: This work has been partially supported by MIUR - SecureOpenNets, EU SPARTA, CyberSANE and E-CORRIDOR projects.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bhat, H.J. Investigate the Implication of "Self-service Business Intelligence (SSBI)"—A Big Data Trend in Today's Business World. *Curr. Trends Inf. Technol.* **2020**, *10*, 17–22.
2. Taylor-Quiring, D.P.; Wiebe, N.J. Cognitive and Predictive Analytics on Big Open Data. In Proceedings of the Cognitive Computing—ICCC 2020: 4th International Conference, Held as Part of the Services Conference Federation, SCF 2020, Honolulu, HI, USA, 18–20 September 2020; Springer Nature: London, UK, 2020; Volume 12408, p. 88.
3. Gautam, A.; Chatterjee, I. Big Data and Cloud Computing: A Critical Review. *Int. J. Oper. Res. Inf. Syst. (IJORIS)* **2020**, *11*, 19–38.
4. Chatfield, A.T.; Reddick, C.G. Cybersecurity innovation in government: A case study of US pentagon's vulnerability reward program. In Proceedings of the 18th Annual International Conference on Digital Government Research, Staten Island, NY, USA, 7–9 June 2017; pp. 64–73.
5. Langner, R. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Secur. Priv.* **2011**, *9*, 49–51.

6. Liagkou, V.; Kavvadas, V.; Chronopoulos, S.K.; Tafiadis, D.; Christofilakis, V.; Peppas, K.P. Attack detection for healthcare monitoring systems using mechanical learning in virtual private networks over optical transport layer architecture. *Computation* **2019**, *7*, 24.
7. Cimino, M.G.; De Francesco, N.; Mercaldo, F.; Santone, A.; Vaglini, G. Model checking for malicious family detection and phylogenetic analysis in mobile environment. *Comput. Secur.* **2020**, *90*, 101691.
8. Cimitile, A.; Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Talos: no more ransomware victims with formal methods. *Int. J. Inf. Secur.* **2018**, *17*, 719–738.
9. Statista.com. Development of New Android Malware Worldwide from June 2016 to March 2020. Available online: https://www.statista.com/statistics/680705/global-android-malware-volume/ (accessed on 20 September 2020).
10. Fan, M.; Liu, T.; Liu, J.; Luo, X.; Yu, L.; Guan, X. Android malware detection: A survey. *Sci. Sin. Inf.* **2020**, *50*, 1148–1177.
11. Casolare, R.; Martinelli, F.; Mercaldo, F.; Santone, A. Android Collusion: Detecting Malicious Applications Inter-Communication through SharedPreferences. *Information* **2020**, *11*, 304.
12. Bayazit, E.C.; Sahingoz, O.K.; Dogan, B. Malware Detection in Android Systems with Traditional Machine Learning Models: A Survey. In Proceedings of the 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 26–28 June 2020; pp. 1–8.
13. Pedreschi, D.; Giannotti, F.; Guidotti, R.; Monreale, A.; Ruggieri, S.; Turini, F. Meaningful explanations of Black Box AI decision systems. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 9780–9784.
14. Milner, R. *Communication and Concurrency*; PHI Series in Computer Science; Prentice Hall: Upper Saddle River, NJ, USA, 1989; pp. I–XI, 1–260.
15. Camilleri, J.; Winskel, G. CCS with priority choice. *Inf. Comput.* **1995**, *116*, 26–37.
16. Milner, R.; Parrow, J.; Walker, D. A calculus of mobile processes, I. *Inf. Comput.* **1992**, *100*, 1–40.
17. Stirling, C. An Introduction to Modal and Temporal Logics for CCS. In *Concurrency: Theory, Language, And Architecture*; Springer: Berlin/Heidelberg, 1989; pp. 2–20.
18. Emerson, E.A. Temporal and modal logic. In *Formal Models and Semantics*; Elsevier: Amsterdam, The Netherlands, 1990; pp. 995–1072.
19. Gabbay, D.M.; Hodkinson, I.; Reynolds, M.A.; Finger, M. *Temporal Logic: Mathematical Foundations and Computational Aspects*; Clarendon Press Oxford: Oxford, UK, 1994; Volume 1.
20. Ben-Ari, M.; Pnueli, A.; Manna, Z. The temporal logic of branching time. *Acta Inform.* **1983**, *20*, 207–226.
21. January 27 – February 1, 2019, Hilton Hawaiian Village, Honolulu,Hawaii, USA Francesco, N.D.; Lettieri, G.; Santone, A.; Vaglini, G. Heuristic search for equivalence checking. *Softw. Syst. Model.* **2016**, *15*, 513–530. doi:10.1007/s10270-014-0416-2.
22. Orailoglu, A.; Gajski, D.D. Flow graph representation. In Proceedings of the 23rd ACM/IEEE Design Automation Conference, Las Vegas, NV, USA, 29 June–2 July 1986; pp. 503–509.
23. Allen, F.E. Control flow analysis. *ACM Sigplan Not.* **1970**, *5*, 1–19.
24. Cesare, S.; Xiang, Y. Classification of malware using structured control flow. In Proceedings of the Eighth Australasian Symposium on Parallel and Distributed Computing-Volume 107, Darlinghurst, Australia, 3-7 February 2010; pp. 61–70.
25. Bruschi, D.; Martignoni, L.; Monga, M. Detecting self-mutating malware using control-flow graph matching. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 129–143.
26. LaToza, T.D.; Myers, B.A. Visualizing call graphs. In Proceedings of the 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Pittsburgh, PA, USA, 18–22 September 2011; pp. 117–124.
27. Kinable, J.; Kostakis, O. Malware classification based on call graph clustering. *J. Comput. Virol.* **2011**, *7*, 233–245.
28. Canfora, G.; Martinelli, F.; Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Leila: formal tool for identifying mobile malicious behaviour. *IEEE Trans. Softw. Eng.* **2018**, *45*, 1230–1252.
29. VirusTotal. VirusTotal Developer Hub. Available online: https://developers.virustotal.com/reference (accessed on 28 September 2020).

30. Arp, D.; Spreitzenbarth, M.; Hubner, M.; Gascon, H.; Rieck, K.; Siemens, C. Drebin: Effective and explainable detection of android malware in your pocket. *NDSS* **2014**, *14*, 23–26.
31. Martinelli, F.; Mercaldo, F.; Nardone, V.; Santone, A.; Vaglini, G. Model Checking and Machine Learning techniques for HummingBad Mobile Malware detection and mitigation. *Simul. Model. Pract. Theory* **2020**; Volume 105, 102169.
32. Google Play Unofficial Python API. Available online: https://github.com/egirault/googleplay-api (accessed on 28 September 2020).
33. dex2jar—Tools to Work with Android .dex and java .class Files. Available online: https://github.com/pxb1988/dex2jar (accessed on 20 September 2020).
34. Vallée-Rai, R.; Gagnon, E.; Hendren, L.; Lam, P.; Pominville, P.; Sundaresan, V. Optimizing Java bytecode using the Soot framework: Is it feasible? In *International Conference on Compiler Construction*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 18–34.
35. Soot Java Optimization Framework. Available online: https://github.com/soot-oss/soot(accessed on 20 September 2020).
36. Iadarola, G. graph4apk - Analysis for Generating Graphs from Apk File. Available online: https://github.com/Djack1010/graph4apk (accessed on 20 September 2020).
37. Iadarola, G.; Martinelli, F.; Mercaldo, F.; Santone, A. Formal Methods for Android Banking Malware Analysis and Detection. In Proceedings of the 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Granada, Spain, 22–25 October 2019; pp. 331–336.
38. Battista, P.; Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Identification of Android Malware Families with Model Checking. In Proceedings of the ICISSP, Rome, Italy, 19–21 February 2016; pp. 542–547.
39. Song, F.; Touili, T. Model-checking for android malware detection. In *Asian Symposium on Programming Languages and Systems*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 216–235.
40. Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Ransomware Steals Your Phone. Formal Methods Rescue It. In Proceedings of the Formal Techniques for Distributed Objects, Components, and Systems—36th IFIP WG 6.1 International Conference, FORTE 2016, Held as Part of the 11th International Federated Conference on Distributed Computing Techniques, DisCoTec 2016, Heraklion, Crete, Greece, 6–9 June 2016; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9688, pp. 212–221.
41. Mercaldo, F.; Nardone, V.; Santone, A.; Visaggio, C.A. Download malware? no, thanks: how formal methods can block update attacks. In Proceedings of the 4th FME Workshop on Formal Methods in Software Engineering, FormaliSE@ICSE 2016, Austin, TX, USA, 15 May 2016; ACM: New York, NY, USA, 2016; pp. 22–28.
42. Jasiul, B.; Szpyrka, M.; Śliwa, J. Formal specification of malware models in the form of colored Petri nets. In *Computer Science and its Applications*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 475–482.
43. Beaucamps, P.; Gnaedig, I.; Marion, J.Y. Abstraction-based malware analysis using rewriting and model checking. In *European Symposium on Research in Computer Security*; Springer: Berlin/Heidelberg, Germany 2012; pp. 806–823.
44. Bai, G.; Ye, Q.; Wu, Y.; van der Merwe, H.; Sun, J.; Liu, Y.; Dong, J.S.; Visser, W. Towards Model Checking Android Applications. *IEEE Trans. Software Eng.* **2017**, *44*, 595–612.