Android malware detection: A systematic Literature review

Ojeniyi, J. A.[1], Hussaini, Y.[2], Umar, M.[3], Abdullazeez, A.[4] and Oladipupo, S. F.[5]

[1]Department of Cyber Security Science, Federal University of Technology Minna, Nigeria

[2,3,4,5]Postgraduate students of Department of Cyber Security Science, Federal University of Technology Minna, Nigeria

1) Abstract:

Android malware is growing at alarming rate and spreading rapidly despite on-going mitigating efforts. This brings a necessity to find more effective solutions to detect those malwares and prevent users from any malicious threats. This study uses the PRISMA statement as a reference so to be transparent. This paper uses a SLR to identify where recent studies in Android malware detection have been focused on and offers a broad perspective relating to types of analysis and the dataset sources used in the research area within the range of 2015 to 2020. A total of 58 selected papers met the inclusion criteria based on title of articles, exclusion criteria, reading abstract and content of the selected 58 papers. Different data are extracted from these articles and recorded in an excel sheet for further analysis. Most of the paper discussed about the use of systematic analysis approach to analyze malware using Debrin dataset, Google Play dataset and Virus Share dataset samples. The systematic review carried out would provide information to all researchers and further inform the requirements for future development of enhanced malware analysis and detection methods.

2) Introduction:

The advancement of mobile devices from a simple form of sending Short Message Service (SMS) and phone calls to smartphones particularly android is accelerating the mobile industry and device users are increasing exponentially [1]. For the last few years, Android is known to be the most widely used operating system and this rapidly increasing popularity has attracted the malware developer's attention. Android allows downloading and installation of apps from other unofficial market places. This gives malware developers an opportunity to put repackaged malicious applications in third-party app-stores and attack the Android devices[2]. However, the advent of malicious applications exploiting the ease of access provided for the above-mentioned software and hardware makes a nightmare situation for conventional malware detection systems to handle these malicious apps. Android has 3.5 million applications in its ecology and 99% of the total malware is targeted towards Android[3].

In general, the existing methods for Android malware detection are roughly classified into three categories. The first one is the static detection, including signature-based methods, permission-based methods, bytecode-based methods, and hybrid static analysis methods. These methods check static Android applications to find any potential malicious features without application execution. The second is dynamic detection that executes applications in isolated environments (such as sandbox, simulator, virtual machine) and then determines whether they are malicious or not by monitoring and tracking their behaviors. The combination of static detection and dynamic detection is the third one, called hybrid detection. Traditional malware detection methods rely mainly on the accumulation of signature libraries and human intervention by malware analysts, so it has been difficult to adapt to the explosive growth of Android malware[4]

Static detection approaches are the most common ones used by antiviruses. Technically, these techniques examine the binary code of the targeted file, analyze all possible execution paths, and identify the malicious code without any execution. However, analyzing binary codes turns out to be difficult nowadays because modern compilers and runtime libraries have introduced significant complexities to these codes. This has negatively affected the capabilities of binary analysis toolkits to analyze binary codes, and as a consequent inaccurate analysis can be reported.[5]

This work conducts a systematic Literature review on Android malware detection using the PRISMA framework. Many innovations in the conduct of systematic reviews have occurred since publication of the PRISMA 2009 statement. For example, technological advances have enabled the use of natural language processing and machine learning to identify relevant evidence[6],[7]. To ensure a systematic review is valuable to users, authors should prepare a transparent, complete, and accurate account of why the review was done, what they did (such as how studies were identified and selected) and what they found (such as characteristics of contributing studies and results of meta-analyses). Up-to-date reporting guidance facilitates authors achieving this[8],[6].

3) Methodology

To create a structured and accountable SLR, we use the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analysis) Statement [9]. This statement contains two parts, a checklist and a flow diagram. The checklists areuseful to help a researcher to point the data which should be listed on a SLR. At the same time, flow diagrams helpresearcher to ensure transparency of the literature used by SLRs. This flow diagram consists of four stages;(1)Identification, (2) Screening, (3), Eligibility, and (4) Included.

A) Research Question

The following research questions were developed.

RQ1 – Which is the most frequent used analysis in android malware detection?

We focused on each paper to identify the different analysis used in their research and graphically illustrate the analysis with the highest number of occurrences in the paper reviewed range to answer this question.

RQ2 – What sources of dataset have been used the most in the research area?

We answer this question by looking at the dataset sources used by each research paper and document our findings.

TABLE I. Inclusion and Exclusion Criteria

Criteria	
Inclusion	A1. The full article was written in English A2. Only Journal article papers A3. Solutions contain Android malware detection techniques
Exclusion	B1. The article was written outside the range 2015-2020 B2. Book and white paper B3. Duplicate copies indexed in other databases B4. Literature review or overview of other paper B5. Papers not explicitly related to android malware detection

B) Identification Stage

At the identification stage, we develop a search strategy to identify relevant literature. This search strategy was tailored to two databases: Science direct and IEEE, and the search term used were the following: "Android malware detection" NOT Review (included to restrict the search to a particular area). All search spanned from database and by default we implement the exclusion and inclusion filters which A1, A2, and B4 to the journal database shown in Table II. The process was carried out on 6th September, 2021 and the results of the search are shown in Table II.

TABLE II. Related study found using Key words

Database Journal	Journal Articles
Science Direct	27
IEEE Explore	31
Total	58

C) Screening Stage:

At this stage, we implement selection criteria by going through the title, abstract, metadata and conclusions to sort out appropriate articles. At this stage 58 papers will be processed. Duplicate reports were not found (B3), Papers not explicitly related Android Malware detection are not present (B5). A total of 58 papers were still used in the eligibility stage

D) Eligibility stage:

This is the quality assessment stage. The study is based on original journal articles. For maintaining the quality of the review, all duplications were checked thoroughly. Abstract of the articles were checked deeply for the analysis and purification of the articles to ensure the quality and relevance of academic literature included in the review process. A careful evaluation of each research paper was carried out at a later stage. The next exclusion criteria were to limit the papers to those applying Solutions using Android malware detection techniques(A3). Furthermore, after the filtration process no papers were removed from the study. We selected 58 Journal articles after assessing each article on the aforementioned inclusion and exclusion criteria.

This stage is useful to ensure that existing articles can answer our research questions.

E) Included stage:

This is usually the data extraction phase where a total of 58 articles were selected and the characteristic extracted were

1) Article must be Journal paper
2) Article must be in English language
3) Extracted articles were published between 2015-2020
4) Article must proffer Solutions to Android malware detection

At this stage all answers from the RQ will be written on a table which has been collected previously

4) Result and Discussion

A) Overview

Every step in the PRISMA flow diagram is detailed in Fig.1 to keep transparency.

PRISMA 2009 Flow Diagram

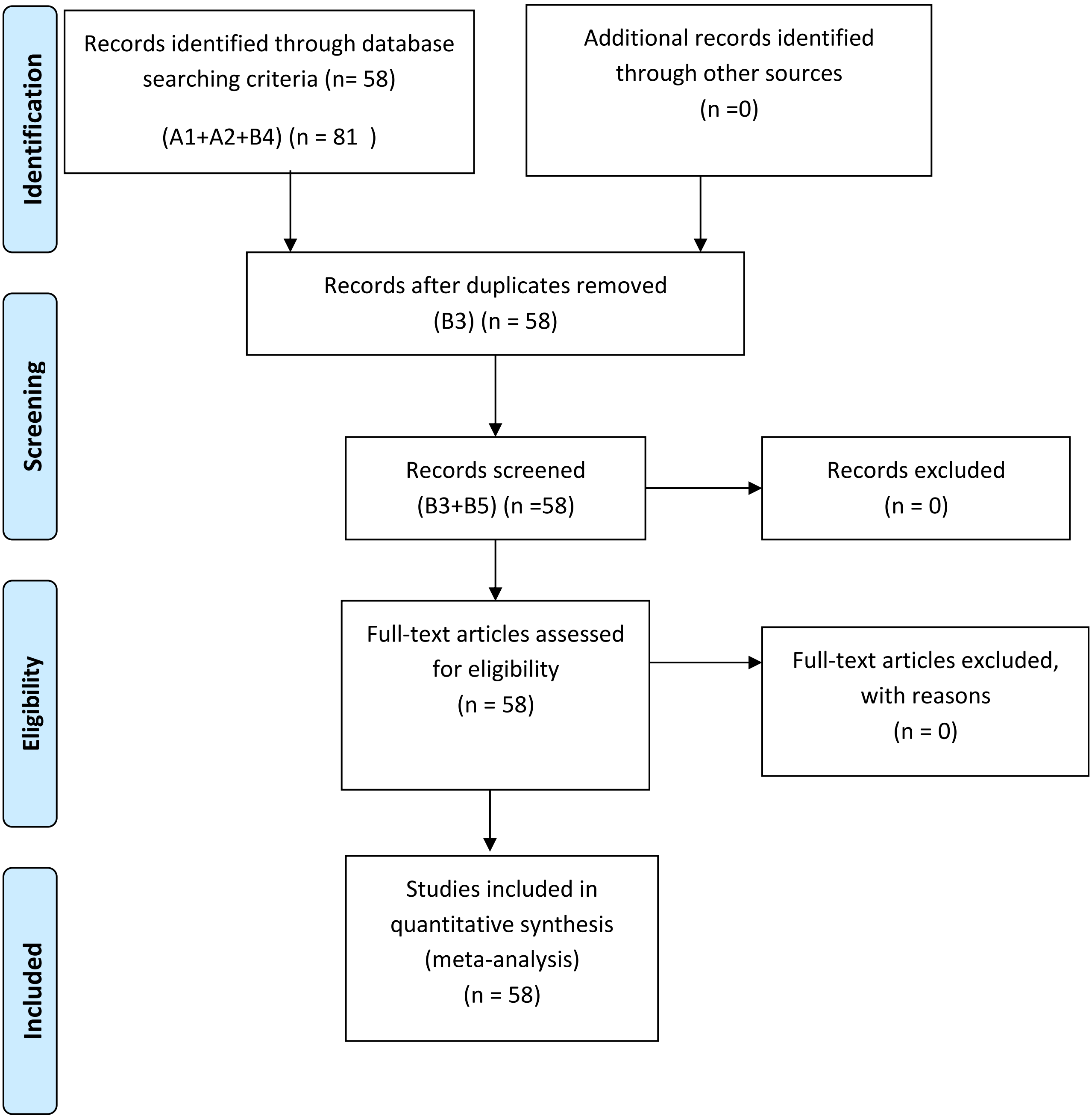
Identification
Screening
Eligibility
Included
Records identified through database searching criteria (n= 58)
(A1+A2+B4) (n = 81)
Additional records identified through other sources
(n =0)
Records after duplicates removed
(B3) (n = 58)
Records screened
(B3+B5) (n =58)
Records excluded
(n = 0)
Full-text articles assessed for eligibility
(n = 58)
Full-text articles excluded, with reasons
(n = 0)
Studies included in quantitative synthesis (meta-analysis)
(n = 58)

Fig 1. SLR transparency process using Prisma Flow Diagram[9]

In the Identification phase, after some basic inclusions and exclusions were applied in two journal databases, there were 58 articles matching our criteria to answer the RQ. No articles were added from other sources, we got 58 papers to be processed. In the Screening and Eligibility step, we excluded no paper. At the end, 58 articles will be included in this report.

Those articles were separated into the database distribution. The results are shown in Fig. 2. The data also proceed to see the distribution per annum and presented in Fig. 3

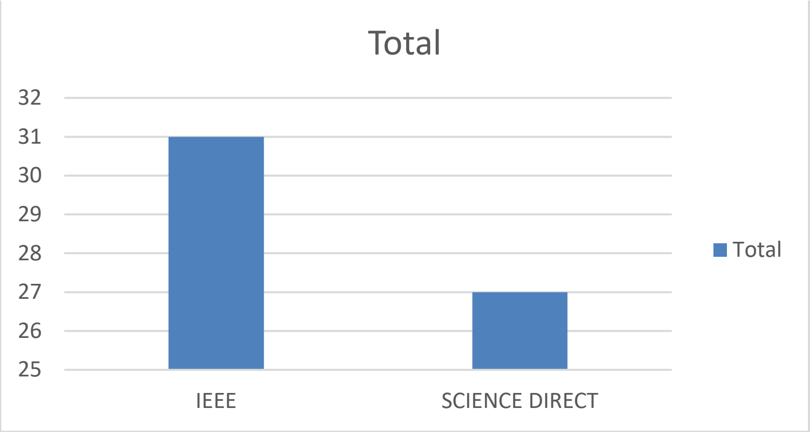

Fig.2 Total Journal articles distribution per database

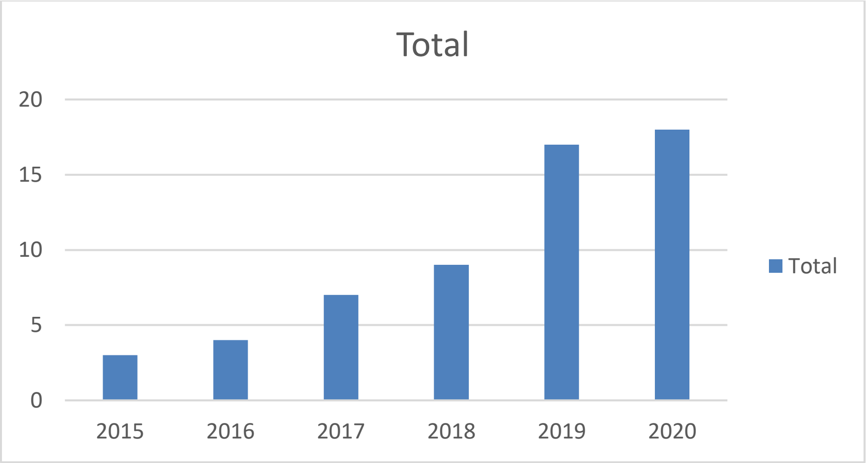

Fig.3 Total Journal articles distribution per year.

Based on Fig.3 it can be seen that Android Malware detection research that fits our filter increased significantly across 2018-2020. While in Fig. 2 the two Journal databases shows that IEEE has more Journal papers than Science direct in the research area. IEEE has a total of 31 Journal papers which were used in this review while Science Direct has a total of 27 papers.

A) RQ1 – Which is the most frequent used analysis in android malware detection?

We use a total of 49 available articles to answer RQ1. We created a graphical and tabular analysis to describe the types of android malware analysis included in the research report as shown in Fig. 4 and TABLE III. It can be seen from Fig. 4 that static analysis has the highest occurrences in our search**.**

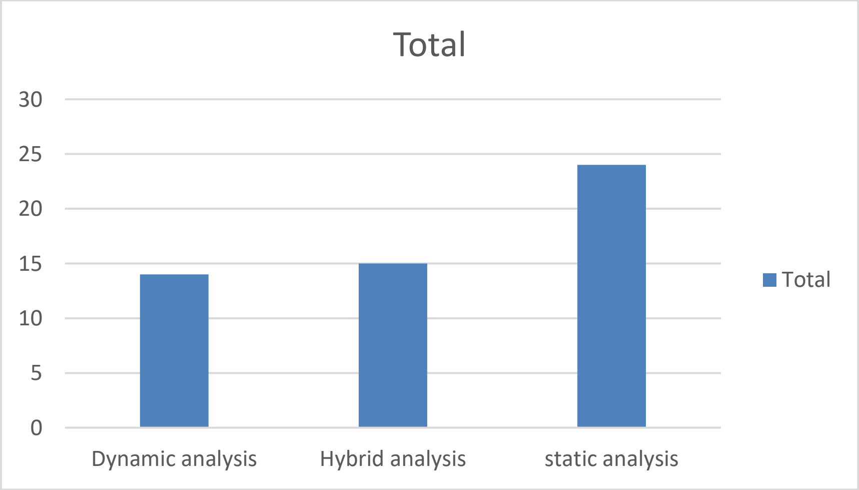

Fig.4 Android malware analysis frequency distribution

TABLE III. Overview of RQ1

Analysis Type	Literature
Dynamic	[5], [10], [11], [12], [13], [14],[15], [16], [17], [18], [19], [20],[21],[22]
Static	[23], [24] ,[25], [26], [27],[28],[29], [30],[31],[32],[33],[34],[35] ,[36], [37],[38],[39],[40],[41],[42],[43], [44],[3], [4]
Hybrid	[45],[46], [47], [48], [49], [50], [51],[52],[53],[54],[55],[56], [57],[58],[59],[60],

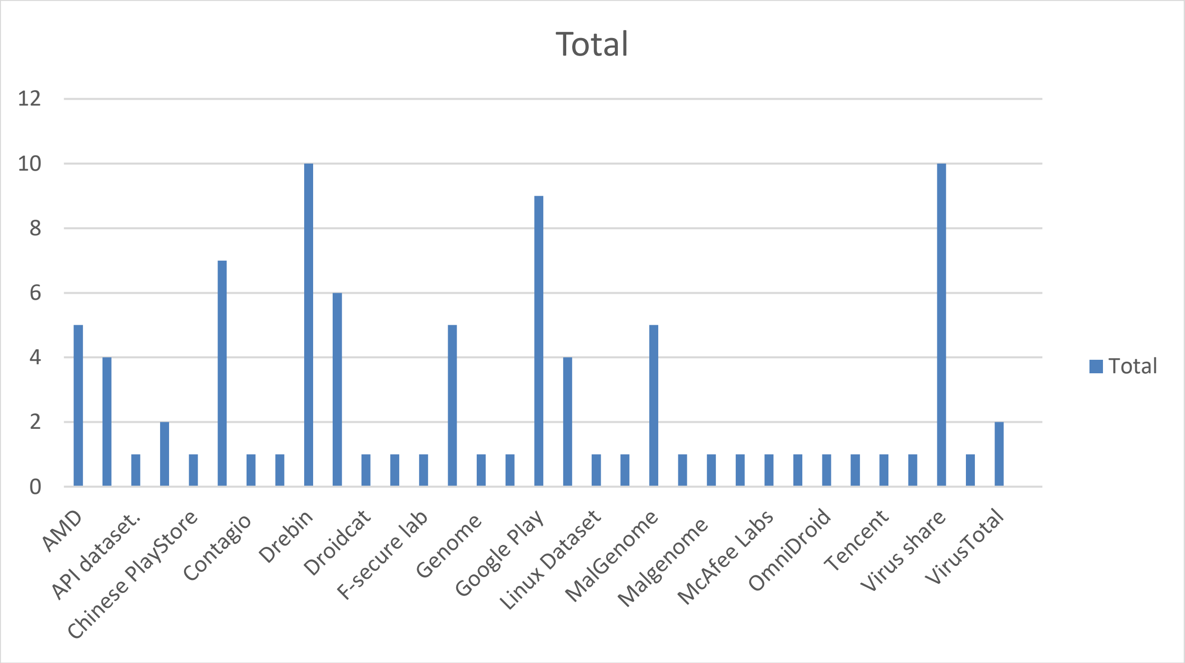

Fig.5 Frequency distribution of dataset sources used in Android malware detection

B) RQ2 – What were the most frequently sources of dataset found in the review

To answer this question, we use 53 Journal articles published within the year 2015-2020 to identify the sources of the dataset used in their research and the findings to answer RQ2 is graphically represented in Fig. 5. It can be seen that Drebin, Google play and Virus share are the most common dataset sources used in the area of Android malware detection. The literatures that used these database sources have been summarized in TABLE IV Showing references of the journals. We bench mark a minimum of 8 journals that have used at least one category of the dataset sources mentioned above in their experiment.

TABLE IV. Overview of RQ2

S/N	Dataset type	LITERATURE
1	Google Play	[40],[43], [17], [16], [15], [51], [31], [50], [52], [42],[4],[34], [14],
2	Drebin	[40],[39],[47],[2],[21],[20],[19],[37],[49],[38],[48],[35],[36],[10], [44],
3	Virus share	[22],[44], [43], [18], [42], [4],[61],[26], [25], [24], [23]

5) Limitation

The limitation of this paper is the journal database used. The more journal databases used, the more diverse answers from the RQ are. Besides, the use of keywords in the journal database is also another limitation coming from the journal database's engine. Applying the same keywords in different journal databases will result in the different conclusion.

6) Conclusion

This SLR research conducted in September 2021 on the two journal databases (IEEE and Science Direct) could filter 58 papers using the search criteria were used in this study.In RQ1, there were 53 journals accessed to conclude on the research question. The majority of researchers are spread across a period of 3 years i.e 2018,2019 and 2020. RQ2 has mapped out the recent and most frequently used dataset sources in the research area, which were discussed in the article.

Our results indicate that the majority of research done in the area of Android malware detection was carried out between 2018-2020. Fig. 2 summarizes the result of the majority of the research carried out within the search range of 6 years. Majority of the dataset used was the Debrin dataset, Google Play and Virus share.

References:

[1] H.-S. Ham and M.-J. Choi, “Analysis of Android malware detection performance using machine learning classifiers,” in *2013 International Conference on ICT Convergence (ICTC)*, Oct. 2013, pp. 490–495. doi: 10.1109/ICTC.2013.6675404.

[2] S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song, and H. Yu, “SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System,” *IEEE Access*, vol. 6, pp. 4321–4339, 2018, doi: 10.1109/ACCESS.2018.2792941.

[3] M. Amin, T. A. Tanveer, M. Tehseen, M. Khan, F. A. Khan, and S. Anwar, “Static malware detection and attribution in android byte-code through an end-to-end deep system,” *Future Gener. Comput. Syst.*, vol. 102, pp. 112–126, 2020, doi: https://doi.org/10.1016/j.future.2019.07.070.

[4] Z. Ren, H. Wu, Q. Ning, I. Hussain, and B. Chen, “End-to-end malware detection for android IoT devices using deep learning,” *Ad Hoc Netw.*, vol. 101, p. 102098, 2020, doi: https://doi.org/10.1016/j.adhoc.2020.102098.

[5] M. Jerbi, Z. C. Dagdia, S. Bechikh, and L. B. Said, “On the use of artificial malicious patterns for android malware detection,” *Comput. Secur.*, vol. 92, p. 101743, 2020, doi: https://doi.org/10.1016/j.cose.2020.101743.

[6] “Using text mining for study identification in systematic reviews: a systematic review of current approaches | SpringerLink.” https://link.springer.com/article/10.1186/2046-4053-4-5?utm_source=getftr&utm_medium=getftr&utm_campaign=getftr_pilot (accessed Nov. 14, 2021).

[7] M. J. Page *et al.*, “The PRISMA 2020 statement: An updated guideline for reporting systematic reviews,” *J. Clin. Epidemiol.*, vol. 134, pp. 178–189, Jun. 2021, doi: 10.1016/j.jclinepi.2021.03.001.

[8] D. Moher, “Reporting guidelines: doing better for readers,” *BMC Med.*, vol. 16, no. 1, p. 233, Dec. 2018, doi: 10.1186/s12916-018-1226-0.

[9] “moher2009.pdf.”

[10] R. Surendran, T. Thomas, and S. Emmanuel, “GSDroid: Graph Signal Based Compact Feature Representation for Android Malware Detection,” *Expert Syst. Appl.*, vol. 159, p. 113581, 2020, doi: https://doi.org/10.1016/j.eswa.2020.113581.

[11] H. Papadopoulos, N. Georgiou, C. Eliades, and A. Konstantinidis, “Android malware detection with unbiased confidence guarantees,” *Neurocomputing*, vol. 280, pp. 3–12, 2018, doi: https://doi.org/10.1016/j.neucom.2017.08.072.

[12] E. B. Karbab, M. Debbabi, and D. Mouheb, “Fingerprinting Android packaging: Generating DNAs for malware detection,” *Digit. Investig.*, vol. 18, pp. S33–S45, 2016, doi: https://doi.org/10.1016/j.diin.2016.04.013.

[13] C. Sun, H. Zhang, S. Qin, N. He, J. Qin, and H. Pan, “DexX: A Double Layer Unpacking Framework for Android,” *IEEE Access*, vol. 6, pp. 61267–61276, 2018, doi: 10.1109/ACCESS.2018.2875694.

[14] F. Shen, J. D. Vecchio, A. Mohaisen, S. Y. Ko, and L. Ziarek, “Android Malware Detection Using Complex-Flows,” *IEEE Trans. Mob. Comput.*, vol. 18, no. 6, pp. 1231–1245, Jun. 2019, doi: 10.1109/TMC.2018.2861405.

[15] T. Lei, Z. Qin, Z. Wang, Q. Li, and D. Ye, “EveDroid: Event-Aware Android Malware Detection Against Model Degrading for IoT Devices,” *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6668–6680, Aug. 2019, doi: 10.1109/JIOT.2019.2909745.

[16] L. Cen, C. S. Gates, L. Si, and N. Li, “A Probabilistic Discriminative Model for Android Malware Detection with Decompiled Source Code,” *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 4, pp. 400–412, Jul. 2015, doi: 10.1109/TDSC.2014.2355839.

[17] P. Feng, J. Ma, C. Sun, X. Xu, and Y. Ma, “A Novel Dynamic Android Malware Detection System With Ensemble Learning,” *IEEE Access*, vol. 6, pp. 30996–31011, 2018, doi: 10.1109/ACCESS.2018.2844349.

[18] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, “MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention,” *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 83–97, Jan. 2018, doi: 10.1109/TDSC.2016.2536605.

[19] H. Zhang, S. Luo, Y. Zhang, and L. Pan, “An Efficient Android Malware Detection System Based on Method-Level Behavioral Semantic Analysis,” *IEEE Access*, vol. 7, pp. 69246–69256, 2019, doi: 10.1109/ACCESS.2019.2919796.

[20] C. Li, K. Mills, D. Niu, R. Zhu, H. Zhang, and H. Kinawi, “Android Malware Detection Based on Factorization Machine,” *IEEE Access*, vol. 7, pp. 184008–184019, 2019, doi: 10.1109/ACCESS.2019.2958927.

[21] X. Chen *et al.*, “Android HIV: A Study of Repackaging Malware for Evading Machine-Learning Detection,” *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 987–1001, 2020, doi: 10.1109/TIFS.2019.2932228.

[22] H. Cai, N. Meng, B. Ryder, and D. Yao, "DroidCat: Effective Android Malware Detection and Categorization via App-Level Profiling," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 6, pp. 1455–1470, Jun. 2019, doi: 10.1109/TIFS.2018.2879302.

[23] Y. Pang, L. Peng, Z. Chen, B. Yang, and H. Zhang, "Imbalanced learning based on adaptive weighting and Gaussian function synthesizing with an application on Android malware detection," *Inf. Sci.*, vol. 484, pp. 95–112, 2019, doi: https://doi.org/10.1016/j.ins.2019.01.065.

[24] S. Wang *et al.*, "Deep and broad URL feature mining for android malware detection," *Inf. Sci.*, vol. 513, pp. 600–613, 2020, doi: https://doi.org/10.1016/j.ins.2019.11.008.

[25] S. Wu, P. Wang, X. Li, and Y. Zhang, "Effective detection of android malware based on the usage of data flow APIs and machine learning," *Inf. Softw. Technol.*, vol. 75, pp. 17–25, 2016, doi: https://doi.org/10.1016/j.infsof.2016.03.004.

[26] K. O. Elish, X. Shu, D. (Daphne) Yao, B. G. Ryder, and X. Jiang, "Profiling user-trigger dependence for Android malware detection," *Comput. Secur.*, vol. 49, pp. 255–273, 2015, doi: https://doi.org/10.1016/j.cose.2014.11.001.

[27] S. Zhang, X. Xie, and Y. Xu, "A Brute-Force Black-Box Method to Attack Machine Learning-Based Systems in Cybersecurity," *IEEE Access*, vol. 8, pp. 128250–128263, 2020, doi: 10.1109/ACCESS.2020.3008433.

[28] Y.-S. Yen and H.-M. Sun, "An Android mutation malware detection based on deep learning using visualization of importance from codes," *Microelectron. Reliab.*, vol. 93, pp. 109–114, 2019, doi: https://doi.org/10.1016/j.microrel.2019.01.007.

[29] Y. Zhang, W. Ren, T. Zhu, and Y. Ren, "SaaS: A situational awareness and analysis system for massive android malware detection," *Future Gener. Comput. Syst.*, vol. 95, pp. 548–559, 2019, doi: https://doi.org/10.1016/j.future.2018.12.028.

[30] S. Y. Yerima and S. Sezer, "DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 453–466, Feb. 2019, doi: 10.1109/TCYB.2017.2777960.

[31] K. A. Talha, D. I. Alper, and C. Aydin, "APK Auditor: Permission-based Android malware detection system," *Digit. Investig.*, vol. 13, pp. 1–14, 2015, doi: https://doi.org/10.1016/j.diin.2015.01.001.

[32] H. Kato, T. Sasaki, and I. Sasase, "Android Malware Detection Based on Composition Ratio of Permission Pairs," *IEEE Access*, vol. 9, pp. 130006–130019, 2021, doi: 10.1109/ACCESS.2021.3113711.

[33] Y. Du, J. Wang, and Q. Li, "An Android Malware Detection Approach Using Community Structures of Weighted Function Call Graphs," *IEEE Access*, vol. 5, pp. 17478–17486, 2017, doi: 10.1109/ACCESS.2017.2720160.

[34] F. Idrees, M. Rajarajan, M. Conti, T. M. Chen, and Y. Rahulamathavan, "PIndroid: A novel Android malware detection system using ensemble learning methods," *Comput. Secur.*, vol. 68, pp. 36–46, 2017, doi: https://doi.org/10.1016/j.cose.2017.03.011.

[35] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb, "MalDozer: Automatic framework for android malware detection using deep learning," *Digit. Investig.*, vol. 24, pp. S48–S59, 2018, doi: https://doi.org/10.1016/j.diin.2018.01.007.

[36] R. Taheri, M. Ghahramani, R. Javidan, M. Shojafar, Z. Pooranian, and M. Conti, "Similarity-based Android malware detection using Hamming distance of static binary features," *Future Gener. Comput. Syst.*, vol. 105, pp. 230–247, 2020, doi: https://doi.org/10.1016/j.future.2019.11.034.

[37] A. Demontis *et al.*, "Yes, Machine Learning Can Be More Secure! A Case Study on Android Malware Detection," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 4, pp. 711–724, Jul. 2019, doi: 10.1109/TDSC.2017.2700270.

[38] H. Zhou, X. Yang, H. Pan, and W. Guo, "An Android Malware Detection Approach Based on SIMGRU," *IEEE Access*, vol. 8, pp. 148404–148410, 2020, doi: 10.1109/ACCESS.2020.3007571.

[39] A. Feizollah, N. B. Anuar, R. Salleh, G. Suarez-Tangil, and S. Furnell, "AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection," *Comput. Secur.*, vol. 65, pp. 121–134, 2017, doi: https://doi.org/10.1016/j.cose.2016.11.007.

[40] S. Alam, Z. Qu, R. Riley, Y. Chen, and V. Rastogi, "DroidNative: Automating and optimizing detection of Android native code malware variants," *Comput. Secur.*, vol. 65, pp. 230–246, 2017, doi: https://doi.org/10.1016/j.cose.2016.11.011.

[41] P. Palumbo, L. Sayfullina, D. Komashinskiy, E. Eirola, and J. Karhunen, "A pragmatic android malware detection procedure," *Comput. Secur.*, vol. 70, pp. 689–701, 2017, doi: https://doi.org/10.1016/j.cose.2017.07.013.

[42] H.-J. Zhu, Z.-H. You, Z.-X. Zhu, W.-L. Shi, X. Chen, and L. Cheng, "DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model," *Neurocomputing*, vol. 272, pp. 638–646, 2018, doi: https://doi.org/10.1016/j.neucom.2017.07.030.

[43] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A Multimodal Deep Learning Method for Android Malware Detection Using Various Features," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 3, pp. 773–788, Mar. 2019, doi: 10.1109/TIFS.2018.2866319.

[44] X. Liu, X. Du, Q. Lei, and K. Liu, “Multifamily Classification of Android Malware With a Fuzzy Strategy to Resist Polymorphic Familial Variants,” *IEEE Access*, vol. 8, pp. 156900–156914, 2020, doi: 10.1109/ACCESS.2020.3019282.

[45] Z. Yuan, Y. Lu, and Y. Xue, “Droiddetector: android malware characterization and detection using deep learning,” *Tsinghua Sci. Technol.*, vol. 21, no. 1, pp. 114–123, Feb. 2016, doi: 10.1109/TST.2016.7399288.

[46] W. Zhang, H. Wang, H. He, and P. Liu, “DAMBA: Detecting Android Malware by ORGB Analysis,” *IEEE Trans. Reliab.*, vol. 69, no. 1, pp. 55–69, Mar. 2020, doi: 10.1109/TR.2019.2924677.

[47] L. Zhang, V. L. L. Thing, and Y. Cheng, “A scalable and extensible framework for android malware detection and family attribution,” *Comput. Secur.*, vol. 80, pp. 120–133, 2019, doi: https://doi.org/10.1016/j.cose.2018.10.001.

[48] R. Surendran, T. Thomas, and S. Emmanuel, “A TAN based hybrid model for android malware detection,” *J. Inf. Secur. Appl.*, vol. 54, p. 102483, 2020, doi: https://doi.org/10.1016/j.jisa.2020.102483.

[49] H. Bai, N. Xie, X. Di, and Q. Ye, “FAMD: A Fast Multifeature Android Malware Detection Framework, Design, and Implementation,” *IEEE Access*, vol. 8, pp. 194729–194740, 2020, doi: 10.1109/ACCESS.2020.3033026.

[50] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, “Significant Permission Identification for Machine-Learning-Based Android Malware Detection,” *IEEE Trans. Ind. Inform.*, vol. 14, no. 7, pp. 3216–3225, Jul. 2018, doi: 10.1109/TII.2017.2789219.

[51] J. Feng, L. Shen, Z. Chen, Y. Wang, and H. Li, “A Two-Layer Deep Learning Method for Android Malware Detection Using Network Traffic,” *IEEE Access*, vol. 8, pp. 125786–125796, 2020, doi: 10.1109/ACCESS.2020.3008081.

[52] R. Kumar, X. Zhang, W. Wang, R. U. Khan, J. Kumar, and A. Sharif, “A Multimodal Malware Detection Technique for Android IoT Devices Using Various Features,” *IEEE Access*, vol. 7, pp. 64411–64430, 2019, doi: 10.1109/ACCESS.2019.2916886.

[53] Z. Wang, C. Li, Z. Yuan, Y. Guan, and Y. Xue, “DroidChain: A novel Android malware detection method based on behavior chains,” *Pervasive Mob. Comput.*, vol. 32, pp. 3–14, 2016, doi: https://doi.org/10.1016/j.pmcj.2016.06.018.

[54] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, “DL-Droid: Deep learning based android malware detection using real devices,” *Comput. Secur.*, vol. 89, p. 101663, 2020, doi: https://doi.org/10.1016/j.cose.2019.101663.

[55] J. H. Abawajy and A. Kelarev, “Iterative Classifier Fusion System for the Detection of Android Malware,” *IEEE Trans. Big Data*, vol. 5, no. 3, pp. 282–292, Sep. 2019, doi: 10.1109/TBDATA.2017.2676100.

[56] F. Tong and Z. Yan, “A hybrid approach of mobile malware detection in Android,” *J. Parallel Distrib. Comput.*, vol. 103, pp. 22–31, 2017, doi: https://doi.org/10.1016/j.jpdc.2016.10.012.

[57] Z.-U. Rehman *et al.*, “Machine learning-assisted signature and heuristic-based detection of malwares in Android devices,” *Comput. Electr. Eng.*, vol. 69, pp. 828–841, 2018, doi: https://doi.org/10.1016/j.compeleceng.2017.11.028.

[58] A. Martín, R. Lara-Cabrera, and D. Camacho, “Android malware detection through hybrid features fusion and ensemble classifiers: The AndroPyTool framework and the OmniDroid dataset,” *Inf. Fusion*, vol. 52, pp. 128–142, 2019, doi: https://doi.org/10.1016/j.inffus.2018.12.006.

[59] T. Gao, W. Peng, D. Sisodia, T. K. Saha, F. Li, and M. Al Hasan, “Android Malware Detection via Graphlet Sampling,” *IEEE Trans. Mob. Comput.*, vol. 18, no. 12, pp. 2754–2767, Dec. 2019, doi: 10.1109/TMC.2018.2880731.

[60] A. Narayanan, M. Chandramohan, L. Chen, and Y. Liu, “Context-Aware, Adaptive, and Scalable Android Malware Detection Through Online Learning,” *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 1, no. 3, pp. 157–175, Jun. 2017, doi: 10.1109/TETCI.2017.2699220.

[61] H. Li, S. Zhou, W. Yuan, J. Li, and H. Leung, “Adversarial-Example Attacks Toward Android Malware Detection System,” *IEEE Syst. J.*, vol. 14, no. 1, pp. 653–656, Mar. 2020, doi: 10.1109/JSYST.2019.2906120.