Review on Hybrid Analysis in Android Malware Detection

Asadullah Hill Galib[ac] and B M Mainul Hossain[bc]

[a]Email: bsse0712@iit.du.ac.bd, Cell No: 01678044993, MS Student

[b]Email: mainul@iit.du.ac.bd, Cell No: 01911489984, Associate Professor

[c]Institute of Information Technology, University of Dhaka, Dhaka, Bangladesh;

[a]Address: Shopno Nir Tower, 52/3-C, Balughat, Dhaka Cantonment, Dhaka – 1206, Bangladesh.

A Review on Hybrid Analysis using Machine Learning for Android Malware Detection

Android is the most global mobile operating system nowadays. Its pervasiveness also provokes the enormous growth of Android malware. Largely researchers have focused on static and dynamic analysis using machine learning techniques to detect Android malware. But, different evasion techniques by shrewd malware authors made those techniques inadequate and ineffective. Therefore, recent researchers have turned their attention to the discovery of an effective strategy to combat. Hybrid analysis: a fusion of static and dynamic analysis would be a good candidate for that as it prevails over the individual shortcomings of static and dynamic analysis with the cost of complexity. Hybrid analysis has many opportunities as well as challenges. This research is intended to provide a detailed and systematic review of hybrid analysis using machine learning techniques for Android malware detection. It encompasses leading hybrid analysis research: their contributions, strengths, and weaknesses. This work also discusses the challenges, opportunities, and future directions of hybrid analysis for Android malware detection.

Keywords: Android Malware Detection, Hybrid Analysis, Machine Learning

I Introduction

Android is the most widespread mobile operating system (OS) currently: 72.23% of total mobile OS is Android [1]. With the enormous growth of the Android system, Android malware also has grown significantly as well as upgraded its nature and activities [2]. On average 12,000 new malware instances are found per day [3]. To defend against that malware phenomenon, researchers emphasize on Android malware detection to ensure Android mobile application security.

To detect Android malware, there are three approaches: Static Analysis, Dynamic Analysis, and Hybrid Analysis. The static analysis uses the static features of the Android application such as Permissions, API Calls, etc. The dynamic analysis investigates the dynamic behaviour of the application. These dynamic features/behaviours include System Calls, Network Traffic, etc. Hybrid analysis tends to incorporate both the static and dynamic approaches into a common ground.

Static and dynamic analyses have their limitations. Currently, malware authors are too smart to evade these detection techniques. For static analysis, commonly used evasion techniques by the malware authors are data obfuscation, control flow obfuscation, encryption, reflection, dynamically loaded code, repackaging, etc. [4]. For dynamic analysis, anti-analysis, mimicry, data obfuscation, misleading information flows, and function in-directions, etc. are used as evasion techniques [4]. Besides, limited code coverage lessens the effectiveness of the dynamic analysis.

As static and dynamic analysis have their weaknesses individually, combining both analyses into a common ground would be helpful. The hybrid analysis approach integrates both static and dynamic analyses to mitigate their weaknesses. Though hybrid analysis is complex enough, it is effective and feasible according to related research. But comparatively a few works have been performed in hybrid analysis. Researchers nowadays focuse on it because of its effectiveness and potential.

Though there exist many reviews on Android malware detection, none focuses on hybrid analysis. For instance, Tam et al. [4] depicts the evolution of Android malware and analysis techniques, but they do not give substantial emphasis on hybrid analysis. Qamar et al. [5] presents an all-inclusive review on mobile malware, but they nearly overlook the hybrid analysis. Baskaran et al. [6] covers hybrid analysis imprecisely in their Android malware detection review in parallel with static and dynamic analysis. Naway et al. [7] focuses on deep learning techniques and Feizollah et al. [8] investigates feature selection for malware analysis. None of them presents an in-depth investigation of hybrid analysis.

Due to the potential of hybrid analysis in malware detection, a conclusive review of the existing research is necessary. In this work, we offer a comprehensive and systematic review of the hybrid analysis approach in Android malware detection, analyzed the existing works: their strengths and weaknesses, and discussed challenges, opportunities, and future directions in this regard. This study is an extension of our earlier study [9] and a further exploration of hybrid analysis in Android malware detection.

To be specific, this work makes the following contributions:

1) It presents the significance of hybrid analysis over static analysis and dynamic analysis by

assessing their weaknesses and limitations.

2) It analyses the existing works on hybrid analysis and presents a review of the research.
3) It triggers a discussion about the hybrid analysis challenges, opportunities and future directions.

II Background

A. Android Malware

Android malware is an application running on the Android OS that implicitly or explicitly performs malicious activities. It includes viruses, worms, ransomware, spyware, and other malicious applications. It tends to cause - disrupting normal functioning, taking access controls, leaking information, root exploitation, manipulating data, private content exposed, phishing, disruption of services, etc. [5].

Moreover, malware is growing exceedingly to keep pace with the immense growth of Android applications. In each month, on average almost 10 million new malware is introduced [10]. New malware is found in every 10 seconds [11].

B. Detection Techniques

Researchers generally analyze Android malware with the following three approaches: Static Analysis, Dynamic Analysis, and Hybrid Analysis.

In static analysis, various static features are extracted from source code and meta-data. According to the static features, a detection model is built using machine learning techniques to classify Android malware. Researchers used Androguard, ApkTool, Appknox, DroidMat, etc. tools for static analysis. According to the existing research [12–16], the most used static features are as follows: Permissions, Intents, Instructions, Hardware Usage Analysis, Meta-data, Intents, API Calls, Intents, Suspicious Files, and Potentially Dangerous Functions and Methods.

The dynamic analysis deals with the dynamic behaviours of an application. In doing so, the application is to be run in an emulated environment or on a real device. A detection model is also built here according to the dynamic features. Researchers commonly used Droidbox, Marvin, Cuckoo Sandbox, AppsPlayground, DroidLogger, etc. tools for dynamic analysis. According to research [17–20], the most used dynamic features are System Calls, Network Traffic, Running Services, File Operations, Network Operations, and Phone Events.

Hybrid Analysis incorporates both static and dynamic features for detecting Android malware. As it deals with both static and dynamic features, it is computationally more complex. Andrubis, AndroData, etc. are used by the researchers for hybrid analysis.

C. Drawbacks of Static and Dynamic Analysis

The most alarming fact is that the noxious malware authors are aware of the malware detection system and they use many novel and crafty evasion techniques to avoid detection. Static Analysis faces many troubles such as data obfuscation, control flow obfuscation, encryption, reflection, dynamically loaded code, repackaging, etc. [4]. On the other hand, Dynamic Analysis also has some drawbacks. To evade dynamic analysis, the anti-analysis technique is used frequently by malware authors to detect virtual machines or emulated environments. If the application detects emulated environments in advance, they will act as a benign application. By doing so, the dynamic analysis might fail to detect Android malware. Besides, malware authors use mimicry, data obfuscation, misleading information flows, and function indirections, etc. to evade dynamic analysis [4]. The biggest weakness of dynamic analysis is limited code coverage: covering all paths is not feasible when investigating the dynamic behaviours.

III HYBRID ANALYSIS USING MACHINE LEARNING

The hybrid analysis integrates both static and dynamic features for effectiveness. Firstly, it seeks to extract the static and dynamic features of Android applications. After that, those extracted static and dynamic features are combined to build a detection model. Finally, according to the static and dynamic features, a detection model is built using machine learning techniques to classify Android malware. By incorporating static and dynamic approaches into a common ground, the hybrid analysis leads to more complexity in Android malware detection. The detection process is more likely to take more time and effort. Though the hybrid approach might be more effective for Android malware detection than the static or dynamic approach, accomplishing a viable malware detection technique is challenging.

As the hybrid approach is the combination of static and dynamic approaches, this approach can overcome the individual weakness as well as can accumulate the advantages of them. Thereby, the hybrid approach strengthens the detection process with the cost of time and complexity. Hybrid methods can also increase robustness, monitor edited apps, increase code coverage, and find vulnerabilities [4].

IV METHODOLOGY

To build up a systematic literature review, a state-of-the-art guideline presented by Kitchenham and Stuart [21] is followed. According to the guideline, developing a review protocol is compulsory to shape a systematic review. The review protocol includes the rationale for the review, research questions, search strategy, study selection criteria, study selection procedures, and study quality assessment, Data extraction, and Data synthesis. Relevant steps in this work are described in the following subsections.

A. The Rationale for the Review: The hybrid analysis using machine learning for Android malware detection is a promising research domain because the weaknesses of static and dynamic analysis have lessened here. The potentiality of this domain needs a brief review of the existing literature.

B. Research Questions: We have identified the following research questions for the review:

1) What are the size and source of the dataset used in the existing research?

2) What are the features used in hybrid analysis using machine learning?

3) Which techniques are used in the existing research?

4) Which evaluation metrics are used in the existing research?

5) What are the outcomes of the existing research?

6) What are the strengths and limitations of the existing research?

C. Study Selection Criteria: The inclusion and exclusion criteria are as follows:

1) Inclusion Criteria:
   a. Journal, Conference Proceedings of hybrid analysis using machine learning
   b. Date (year) of publication: 2012-2020

2) Exclusion Criteria:
   a. Research that uses - hybrid keyword, but not directed to the hybrid analysis
   b. Research that incorporates hybrid analysis, but not using machine learning
   c. Research that lacks a well-defined methodology and unambiguous contributions

D. Study Quality Assessment: We have scrutinized the selected papers for bias, internal validity, and external validity. Though there is no consensus about the interpretation of - quality, the CRD Guidelines [22] and the Cochrane Reviewers Handbook [23] suggest that quality correlates insofar as the study minimizes bias and maximizes internal and external validity [21].

V SYSTEMATIC LITERATURE REVIEW

In the following section, we have resolved the research questions and presented an inclusive systematic review of the consequential research in hybrid analysis. Table 1 depicts the literature overview of hybrid analysis using machine learning.

One of the state-of-the-art study in hybrid analysis, Marvin [24] employs a lot of static and dynamic features to detect malware. It uses SVM and Linear Classifiers to build a detection model where Linear Classifiers can detect more accurately but SVM is faster comparatively. To avoid the obsolescence of its classification model in the future, it presents a retraining strategy. Marvin's performance is sound enough as its accuracy is 98.24 % with less than 0.04% false-positive rate. But for previously unseen malware, its accuracy is close to 90%. Though Marvin considers a lot of features, it overlooks system-level events such as System Calls: an integral part of the behavioural aspects (dynamic features).

Samadroid [25] presents an on-device malware detection architecture which ensures the resource efficiency by reducing memory overhead of local devices. It uses a subset of Drebin's [12] features (6 out of 8) as static features and 10 predefined System Calls as dynamic features. Its accuracy is about 98% with a false positive rate of 0.1%. Though it outperforms Drebin, it used an outdated dataset. Thereby it would fail to fight against recent malware as malware behaviour changes frequently over time. It also overlooks any additional dynamic features except System Calls.

BRIDEMAID [26] proposes a framework using multi-level and multi-feature analysis. It is capable of detecting polymorphic and composition malware to avoid zero-day attacks. Its accuracy is relatively high regarding existing works. However, it does not use any benchmark dataset. Also, it reports only accuracy and FPR, other metrics should be reported to properly evaluate the framework.

OmniDroid [27] fuses several prior tools to extract many static and dynamic features and employs ensemble-based classifiers. Though they considered only a large feature-set, their performance is relatively lower than existing works. MADAM [28] simultaneously analyses and correlates static and dynamic features at four levels: kernel, application, user, and package, to detect and stop malicious behaviours. Though it gains accuracy of 96.9%, it has high memory overhead and limited scope (only run in the rooted device, works on post-installed apps).

Hadm [29] incorporates Deep Neural Network for feature extraction. It exhibits that combining advanced features derived by deep learning with the original static and dynamic features gives substantial returns. It achieves 94.7% accuracy with a false positive rate of 1.8% while with the original features the best accuracy is 93.5%., an improvement of 1.2% with the cost of high complexity. Droid-detector [30] combines static and dynamic analyses which extracted more than 200 features by using deep neural network. It achieves 96.5% accuracy in detection. However, it uses a limited dataset and limited types of features.

Mobile-SandBox [31] uses Permissions, Services, Receivers, Intents, potentially dangerous functions as static features and investigates Native Code (Native API Calls) and Network Traffic as dynamic features to classify malware. However, its evaluation is insufficient as no detection metric is given. Kapratwar et al. [32] uses Permissions and System Calls for hybrid analysis. Its performance (AUC) is significantly better for static features in comparison with dynamic features. But it uses a small (200 apps) and old dataset and overlooks other static and dynamic features. Dhanya et al. [33] uses Permissions and API Calls for hybrid analysis. Separability assessment Criteria is used for feature selection. Their performance is insufficient as no accuracy measure is given. Besides, they do not consider any other features. Liu et al. [34] proposes a hybrid malware detecting scheme for Android where Permissions and API Calls are used as static features and System Calls used as dynamic features. Their scheme's detection accuracy is from 93.33% to 99.28% according to experimental results. Nevertheless, they consider only a small feature-set and their dataset is also limited. Patel et al. [35] uses Genetic algorithm for rule-based classification of malware using static and dynamic features. Analyses of more than 231 features achieves 96.4% accuracy in detection. But it uses a limited dataset and its execution time and resource consumption are high. Yusof et al. [36] uses Permissions, API Calls, and System Calls for malware detection. It makes sound performance in terms of accuracy, precision, and recall. Yet, the FPR is high enough. Also, its model is trained with the malware samples only which would lead to a biased model.

In short, Permissions and API Calls as static features and System Calls as dynamic features are the most frequently used features according to the existing research. The most common datasets are Drebin, Contagio, and Android Malware Genome Project. Besides, most researchers use the Google Play Store and local app stores to collect benign applications. VirusTotal, VirusShare, etc. sources are also used for malware samples. Support Vector Machine (SVM) is the most frequently used machine learning technique. Besides, Naive Bayes, Random Forest, J48, Logistic Regression, etc. are also common in the existing research. Accuracy, True Positive Rate (TPR), False Positive Rate are the most common evaluation metrics.

Table 1: SYSTEMATIC LITERATURE OVERVIEW OF HYBRID ANALYSIS USING MACHINE LEARNING

Ref.	**Static Features**	**Dynamic Features**	**Dataset Source**	**Dataset Size**	**ML Model**	**Results**	**Limitations**
Mobile-SandBox (2013) [31]	Permissions, Services, Receivers, Intents, Potentially Dangerous Functions	Native Code (Native API Calls) and Network Traffic	Asian markets and Google Play Store	40,000 apps			Insufficient evaluation. No detection performance is given. Old dataset.
Patel et al. (2015) [35]	Permissions, Intents, Receivers	SMS, File Operations, Native Code, Network Data,	Droid-Kin, Contagio	755 apps	Genetic Algorithm, Information Gain	Accuracy 96.4%	High execution time and resource consumption. Limited dataset.
Marvin (2015) [24]	Permissions, Intents, Suspicious Files, API Calls, Developer's Certificate	File Operations, Network Operations, Phone Events, Dynamically Loaded Code	Google Play Store, Virus-Total, GenomeProject, Contagio	150,000 apps: 135,000 benign, 15,000 malware	SVM and Linear Classifier	Accuracy: 98.24%, FPR: <0.04%	Overlooking system-level events such as System Calls. Too many features. Higher complexity.
Droid-detector (2016) [30]	Permissions, API Calls	File Operations, Network Traffic, Phone Events, SMS	Google Play Store, Genome Project Contagio	21,760 apps: 20,000 benign, 1761 malware	Deep Belief Network	Accuracy: 96.7%	Overlooking many static features and system-level events such as System Calls.
MADAM (2016) [28]	Permissions, API Calls	System Calls, SMS, Phone Event	Genome Project, Virus-Share, Contagio	2800 apps: 125 malware families	KNN	Accuracy: 96.9%	High memory overhead. Limited scope (only run in rooted device, post-installed apps).
Liu et al. (2016) [34]	Permissions	System Calls	Gnome Project, Wandoujia App Market	1000 apps: 1000 benign, 1000 malware	SVM, KNN	Accuracy: 93.33%~99.28%, TPR: 94.59%~99.47%, FPR: 0.20%~ 11.01%	Using limited dataset, considering few features
Hadm (2016) [29]	Permissions, API Calls, Intent	System Call Sequences	Google Play and Virus-Share	5888 apps: 4002 benign, 1886 malware	Deep Neural Network, SVM, Hierarchi-cal MKL	Accuracy: 94.7%, FPR: 1.8%	Higher complexity with respect to accuracy gains. No benchmark dataset. Limited features set.
BRIDE-MAID (2016) [26]	Permissions, Meta Info., Opcodes	System Calls, SMS	Google Play Store, Virus-Total	12,598 apps: 9804 benign, 2794 malware	SVM	Accuracy: 99.7%, FPR: 0.2%	No benchmark dataset. Insufficient evaluation
Kapra-twar et al. (2017) [32]	Permissions	System Calls	Google Play Store, Virus-Total, Drebin	200 apps: 103 benign, 97 malware	IBk, Nave Bayes, J48, Random Forest, Logistic	AUC: 0.5844~0.9660	Overlooking many static and dynamic features. Limited and old dataset. Insufficient evaluation
Sama-droid (2018) [25]	Permissions, API Calls, Intents, App Components	System Calls (10)	Drebin	5,560 malware	SVM, Naïve Bayes, Decision Tree and Random Forest	Accuracy: 91.6%~ 98.97%, TPR: 81.1%~ 98.5%, FPR: 0.03%~ 7.8%	Overlooking many dynamic features. Limited and old dataset.

Yusof et al. (2018) [36]	Permissions, API Calls	System Calls	Drebin, Google Play Store	Train: 5,560 malware, Test: 800 benign	SVM, Naïve Bayes, KNN and Random Forest	Accuracy 97.9%, Pre: 98.2%, Rec: 99.4, TPR: 99.4, FPR: 12.4	Model trained with only malware which may lead to biasness. High false positive rate.
Dhanya et al. (2019) [33]	Permissions	API Calls	Drebin	400 apps: 200 benign, 200 malware	Nave Bayes, SVM, J48 & Random Forest	F-score: 0.71%~ 0.975, Precision:74.7 %~ 97.6%, Recall: 72.5%~ 97.5%	Limited and old dataset. Considering few features. No accuracy given.
Omni-Droid (2019) [27]	Permissions, API Calls, Intents, Meta Info., Opcodes	System Calls, Network Data	Omni-Droid	22,000 apps: 11,000 benign, 11,000 malware	Random Forest, Bagging, Voting	Accuracy: 89.7%, Precision: 89.7%,	Too many features, Higher complexity regarding performance gain

VI DISCUSSION

In this section, we have pointed out the opportunities, challenges, limitations, and future directions of hybrid analysis in Android malware detection.

A. Dataset Inadequacy: Almost 10 million new malware are found each month [10]. But there does not exist any up-to-date dataset of malware. So, their performance in malware detection is doubtful considering the vast population of the new malware. Dataset inadequacy is a vital factor as an appropriate dataset is required for research evaluation. So, the malware dataset should be updated on a regular basis to assure the effectiveness of the new research and to justify the feasibility of the existing research.

B. Exploring New Feature: Most of the existing research deals with some common features such as Permissions, API Calls, System Calls, File Operations, Network Operations, etc. But it would be possible that there exist more distinguishable features to detect malware. For instance, Talha et al. [37] reveals many unknown characteristics of Android malware, however, it did not integrate any machine learning technique to detect malware. They reveal that over-privileged permissions are one of the characteristics of malware. Besides, they uncover that malware's average number of incoming and outgoing connections, the average size of download and upload, the average number of INTERNET CLOSE action are distinguishable features in malware. Likewise, looking for more discernible features would create new opportunities in Android malware detection.

C. New Malware Family: As existing malware’s behaviour is decoded by the existing tool or research outcome; malware authors update existing malware families and create new malware families frequently to evade detection. They try to trick existing detection systems by introducing new behaviour as well as exhibiting benign behaviour. So, researchers should consider this issue carefully to ensure security. *How do we detect new malware families effectively?* - would be a promising research question.

D. Reducing Complexity: Since the hybrid approach combines static and dynamic approaches, its overall complexity is higher with respect to time, cost, and effort. *How do we reduce the complexity of hybrid analysis?* - would be a potential direction for future researchers.

E. Better Performance: Hybrid analysis exhibits better performance on average than the static and dynamic approaches and provokes a lot of opportunities. By taking those opportunities and overcoming the challenges ahead, the hybrid analysis would be a vanguard for malware detection in the future.

F. Lack of Research: Though hybrid analysis is a promising and effective approach in Android malware detection, there is not enough research in hybrid analysis. A lot of opportunities and research directions are available right now. Researchers' enthusiastic focus on this field would have been beneficial to fight against the rising malware authors community.

VII CONCLUSION

Detecting Android malware effectively and feasibly is one of the crucial challenges of this fast-growing digital world. The hybrid analysis technique has the capability and can offer a sound direction in this field. By exploring this field, researchers have already published several research. This work tends to highlight those research by providing a thorough and systematic review of them. It encompasses the static features, dynamic features, dataset, algorithms, metrics considered in those research. It also focuses on the individual strengths and limitations of them. Besides, it points out the specific challenges, limitations, and future directions in the hybrid analysis technique. By doing so, this research seeks to contribute to academia as well as raise concern for Android mobile application security

REFERENCES

[1] N. G., 2019, “Android: Market share other stats [infographic]”.

[2] Popper, B., 2017, “Google announces over 2 billion monthly active devices on Android”, The Verge.

[3] Lueg, C., 2018, “Cyber attacks on android devices on the rise”, G DATA.

[4] Tam, K., Feizollah, A., Anuar, N.B., Salleh, R. and Cavallaro, L., 2017, “The evolution of android malware and android analysis techniques”, ACM Computing Surveys (CSUR), 49(4), pp.1-41.

[5] Qamar, A., Karim, A. and Chang, V., 2019, “Mobile malware attacks: Review, taxonomy & future directions”, Future Generation Computer Systems, 97, pp.887-909.

[6] Baskaran, B. and Ralescu, A., 2016, “A study of android malware detection techniques and machine learning”.

[7] Naway, A. and Li, Y., 2018, “A review on the use of deep learning in android malware detection”, arXiv preprint arXiv:1812.10360.

[8] Feizollah, A., Anuar, N.B., Salleh, R. and Wahab, A.W.A., 2015, “A review on feature selection in mobile malware detection”, Digital investigation, 13, pp.22-37.

[9] Galib, A.H., Hossain, B. M., 2019, December, “A Systematic Review on Hybrid Analysis using Machine Learning for Android Malware Detection”, In International Conference on Innovation in Engineering and Technology (ICIET) 2019.

[10] Av-test, 2019, April, “Malware statistics trends report: Av-test”.

[11] New Jersey Cybersecurity and Communications Integration Cell (NJCCIC), 2018, “Android malware threat profile”, NJCCIC The Weekly Bulletin.

[12] Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., Rieck, K. and Siemens, C.E.R.T., 2014, February, “Drebin: Effective and explainable detection of android malware in your pocket”, In Ndss (Vol. 14, pp. 23-26).

[13] Yerima, S.Y., Sezer, S., McWilliams, G. and Muttik, I., 2013, March, “A new android malware detection approach using bayesian classification”, In 2013 IEEE 27th international conference on advanced information networking and applications (AINA) (pp. 121-128), IEEE.

[14] Mariconti, E., Onwuzurike, L., Andriotis, P., De Cristofaro, E., Ross, G. and Stringhini, G., 2016, "Mamadroid: Detecting android malware by building markov chains of behavioral models", arXiv preprint arXiv:1612.04433.

[15] Ghorbanzadeh, M., Chen, Y., Ma, Z., Clancy, T.C. and McGwier, R., 2013, January, "A neural network approach to category validation of android applications", In 2013 International Conference on Computing, Networking and Communications (ICNC) (pp. 740-744), IEEE.

[16] Jerome, Q., Allix, K., State, R. and Engel, T., 2014, June, "Using opcode-sequences to detect malicious Android applications", In 2014 IEEE International Conference on Communications (ICC) (pp. 914-919), IEEE.

[17] Yan, L.K. and Yin, H., 2012, "Droidscope: Seamlessly reconstructing the {OS} and dalvik semantic views for dynamic android malware analysis", In Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12) (pp. 569-584).

[18] Amos, B., Turner, H. and White, J., 2013, July, "Applying machine learning classifiers to dynamic android malware detection at scale", In 2013 9th international wireless communications and mobile computing conference (IWCMC) (pp. 1666-1671), IEEE.

[19] Wu, W.C. and Hung, S.H., 2014, October, "DroidDolphin: a dynamic Android malware detection framework using big data and machine learning", In Proceedings of the 2014 Conference on Research in Adaptive and Convergent Systems (pp. 247-252).

[20] Enck, W., Gilbert, P., Han, S., Tendulkar, V., Chun, B.G., Cox, L.P., Jung, J., McDaniel, P. and Sheth, A.N., 2014, "TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones", ACM Transactions on Computer Systems (TOCS), 32(2), pp.1-29.

[21] Kitchenham, B. and Charters, S., 2007, "Guidelines for performing systematic literature reviews in software engineering".

[22] Khan, K.S., Ter Riet, G., Glanville, J., Sowden, A.J. and Kleijnen, J., 2001, "Undertaking

systematic reviews of research on effectiveness: CRD's guidance for carrying out or commissioning reviews (No. 4 (2n)", NHS Centre for Reviews and Dissemination.

[23] Higgins, J.P., Thomas, J., Chandler, J., Cumpston, M., Li, T., Page, M.J. and Welch, V.A. eds., 2019, "Cochrane handbook for systematic reviews of interventions", John Wiley & Sons.

[24] Lindorfer, M., Neugschwandtner, M. and Platzer, C., 2015, July, "Marvin: Efficient and comprehensive mobile app classification through static and dynamic analysis", In 2015 IEEE 39th annual computer software and applications conference (Vol. 2, pp. 422-433), IEEE.

[25] Arshad, S., Shah, M.A., Wahid, A., Mehmood, A., Song, H. and Yu, H., 2018, "SAMADroid: a novel 3-level hybrid malware detection model for android operating system", IEEE Access, 6, pp.4321-4339.

[26] Martinelli, F., Mercaldo, F., Saracino, A. and Visaggio, C.A., 2016, December, "I find your behavior disturbing: Static and dynamic app behavioral analysis for detection of android malware", In 2016 14th Annual Conference on Privacy, Security and Trust (PST) (pp. 129-136), IEEE.

[27] Martín, A., Lara-Cabrera, R. and Camacho, D., 2019, "Android malware detection through hybrid features fusion and ensemble classifiers: the AndroPyTool framework and the OmniDroid dataset", Information Fusion, 52, pp.128-142.

[28] Saracino, A., Sgandurra, D., Dini, G. and Martinelli, F., 2016, "Madam: Effective and efficient behavior-based android malware detection and prevention", IEEE Transactions on Dependable and Secure Computing, 15(1), pp.83-97.

[29] Xu, L., Zhang, D., Jayasena, N. and Cavazos, J., 2016, September, "Hadm: Hybrid analysis for detection of malware", In Proceedings of SAI Intelligent Systems Conference (pp. 702-724), Springer, Cham.

[30] Yuan, Z., Lu, Y. and Xue, Y., 2016. "Droiddetector: android malware characterization and detection using deep learning", Tsinghua Science and Technology, 21(1), pp.114-123.

[31] Spreitzenbarth, M., Freiling, F., Echtler, F., Schreck, T. and Hoffmann, J., 2013, March, "Mobile-sandbox: having a deeper look into android applications", In Proceedings of the 28th Annual ACM Symposium on Applied Computing (pp. 1808-1815).

[32] Kapratwar, A., Di Troia, F. and Stamp, M., 2017, February, "Static and dynamic analysis of android malware", In ICISSP (pp. 653-662).

[33] K. D. T. Gireesh Kumar, 2019, "Efficient android malware scanner using hybrid analysis," International Journal of Recent Technology and Engineering (TM), vol. 7, pp. 76–80.

[34] Liu, Y., Zhang, Y., Li, H. and Chen, X., 2016, January, "A hybrid malware detecting scheme for mobile Android applications", In 2016 IEEE International Conference on Consumer Electronics (ICCE) (pp. 155-156). IEEE.

[35] Patel, K. and Buddadev, B., 2015, August, "Detection and mitigation of android malware through hybrid approach", In International Symposium on Security in Computing and Communication (pp. 455-463), Springer, Cham.

[36] Yusof, M., Saudi, M.M. and Ridzuan, F., 2018, "Mobile Botnet Classification by using Hybrid Analysis", International Journal of Engineering & Technology, 7(4.15), pp.103-108.

[37] A. T. Kabakus and I. A. Dogru, 2015, "An in-depth analysis of android malware using hybrid techniques", Digital Investigation, vol. 24, pp. 25– 33, 2018.