JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

UTILIZING MACHINE LEARNING-BASED REVERSE ENGINEERING TECHNIQUES TO IDENTIFY MALWARE ON MOBILE DEVICES WITHIN ANDROID APPLICATIONS

K. Nishitha[1], G.Rajesh[2], N.Haritha[3]
Assistant Professor, Department of CSE, Audisankara College of Engineering & Technology, Gudur[1].
Associate Professor, Department of CSE, Audisankara College of Engineering & Technology, Gudur[2].
Assistant Professor, Department of Computer Science, Sri Harsha Institute of PG Studies, Nellore[3].

Abstract:

The current trend shows a significant increase in the popularity of smart phone operating systems. However, this rise has also made them a prime target for hackers and attackers, particularly in the case of Android applications. As a result, there has been a noticeable surge in malicious code attacks on the Android platform, making it an appealing target for such perpetrators. These malevolent actors have developed sophisticated techniques to hide harmful algorithms within seemingly innocent Android applications, posing a challenge for security firms in identifying and classifying these apps as malware. Android malware has evolved to evade traditional detection methods due to its increasingly unique characteristics.

Machine learning-based techniques have emerged as a more effective solution for addressing the complex challenge of emerging Android threats. These approaches analyze the behaviors exhibited by existing malware patterns and leverage this data to differentiate between known threats and novel risks.This study proposes a novel approach to identifying weaknesses in mobile apps. The key contributions are twofold:First, the study presents a model that combines more innovative feature sets with the largest existing malware sample records, going beyond traditional methods.

Second, the model employs ensemble learning with machine learning algorithms such as AdaBoost and SVM to enhance its accuracy and performance.The test results demonstrate that the proposed model achieves a 96.24% accuracy rate in identifying malware samples from Android apps, with a low 0.3 false positive rate. Crucially, the model incorporates overlooked malicious features such as permissions, targets, and API calls, and is trained using a single reverse-engineered sample as input.Overall, this study offers a significant advancement in mobile app security, providing a robust and comprehensive solution for detecting malware.

I. Introduction

Cybersecurity is increasingly a top priority for network engineers and computer scientists, which is a relief. The associate editor in charge of reviewing and approving this manuscript has encountered several issues. As a result, rapid advancements in technology and their widespread integration into daily life, along with the rise of various

malware applications and their targets, have been extensively researched. Among these, Android malware has attracted considerable attention online. Android is the most popular operating system, holding a dominant position in the market. Malware that invades systems to evade detection has become more sophisticated, with some malware applications containing over 50 parameters, complicating the detection process. Therefore, it's crucial to develop strategies to address the ongoing increase in Android malware, aiming to identify, neutralize, or eliminate it effectively.

These challenges motivate academics to delve deeper into the subject, pushing them to pursue further studies aimed at discovering malware and effectively handling it. Consequently, researchers have devised three strategies for locating Android malware, including dynamic, static, and hybrid approaches. Static analysis involves identifying characteristics that help in pinpointing malicious behavior in apps without the need for a full application installation. However, this method faces obstacles due to code obfuscation techniques that malware authors use to evade detection through static analysis. Dynamic analysis, on the other hand, is employed to identify malware within apps as they run. Static analysis can locate malware in source code, while dynamic analysis can identify malware in a running environment. However, Android developers and users may still be exposed to risks from malware.

This study covers methods for detecting malware. It examines using machine learning (ML) models to analyze Android Application Packages (APKs) and derive relevant features. Both deep learning (DL) and traditional ML approaches can be used to recognize malicious APKs.

Similarly, detecting vulnerabilities in software code involves two stages: training ML models on derived attributes to identify vulnerable code segments, and generating features through code analysis.

The paper presents an Automated Android Malware Detection technique using an Optimal Ensemble Learning Approach for Cybersecurity (AAMD-OELAC).

The AAMDOELAC technique is utilized for data preprocessing in the initial stage. In the process of detecting Android malware, the AAMD-OELAC technique employs an ensemble learning approach using three ML models: Least Square Support Vector Machine (LS-SVM), kernel extreme learning machine (KELM), and Regularized random vector functional link neural network (RRVFLN). Additionally, the hunter-prey optimization (HPO) algorithm is employed to optimize the parameters of the three DL models, resulting in enhanced malware detection outcomes. To demonstrate the superiority of the AAMD-OELAC approach, a comprehensive experimental analysis is conducted. In summary, the main contributions can be summarized as follows.

A novel approach called AAMD-OELAC is introduced in this study, which combines data preprocessing, ensemble learning, and HPO-based hyper parameter tuning for the purpose of Android malware detection. It is worth noting that the AAMDOELAC technique has not been previously documented in any existing literature. The classification process for Android malware detection involves the utilization of LS-SVM, KELM, and RRVFLN models within the ensemble learning framework. The integration of the HPO algorithm and ensemble learning process leads to an enhanced accuracy in detecting Android malware. Through the utilization of multiple classifiers and optimization strategies, the proposed model effectively identifies malicious patterns and behaviors exhibited by Android applications.

Android now controls the vast majority of mobile devices, with Android devices accounting for an average of 80% of the global market share over the past years. With the ongoing plan of Android to a growing range of smart phones and consumers around the world, malware targeting Android devices has increased as well. Since it is an open-source operating system, the level of danger it poses, with malware authors and programmers implementing unwanted permissions, features and application components in Android apps.

When the number of smart phone apps increases, so does the security problem with unnecessary access to different personal resources. As a result, the applications are becoming more insecure, and they are stealing personal information, SMS frauds, ransom ware, etc. In contrast to static analysis methods such as a manual assessment of AndroidManifest.xml, source files and Dalvik Byte Code and the complex analysis of a managed environment to study the way it treats a program, Machine Learning includes learning the fundamental rules and habits of the positive and malicious settings of apps and then data enabling. Machine learning employs a range of methodologies for data classification. SVM (Support Vector Machine) is a strong learner that plots each data item as a point in n-dimensional space with the value of each feature becoming the vector value.

Infected file extensions, files received via Bluetooth, links to infected code in SMS, and MMS application links are all ways that mobile malware can propagate . Machine learning algorithms and also implemented a Boosting ensemble learning approach (AdaBoost) with a Decision Tree based on the binary classification to enhance our prediction rate.Feature selection is an essential step in all machine-based learning approaches. The optimum collection of features will not only help boost the outcomes of tests but will also help to reduce the compass of most machine-based learning algorithms. Reverse engineer Android applications and extract features and do static analysis from them without having to execute them.

This method entails examining the contents of two files: AndroidManifest.xml and classes.dex, and working on the file with the .apk extension. Feature selection techniques and classification algorithms are two crucial areas of feature-based types of fraudulent applications. Feature filtering methods are used to reduce the dimension size of a dataset. Parsing tools can help learn which permissions, packages or services an application offers by analyzing the AndroidManifest.xml file, such as permission android. permission. call phone, which allows an application to misuse calling abilities.The fact explained that Android applications pose a lot of threats to its user because of the unnecessary programs compiled inside them and explained why it is necessary to automate the process of static analysis for the efficient detection of malware applications based on the extracted features.

II.LITERATURE SURVEY AND EXISTING SYSTEM

1. **Adversarial superiority in Android malware detection: Lessons from reinforcement learning based evasion attacks and defenses, Forensic Sci. Int., Digit. Invest., vol. 44, Mar. 2023, Art. no. 301511 Author: H. Rathore, A. Nandanwar, S. K. Sahay, and M. Sewak.** Now-a-days android smart phones are being used by billions of users and thus have become a lucrative target of malware designers. Therefore being one step ahead in this zero-sum game of malware detection between the anti-malware community and malware developers is more of a necessity than a desire. This work focuses on a proactive adversary-aware framework to develop adversarially superior android malware detection models. We first investigate the adversarial robustness of thirty-six distinct malware detection models constructed using two static features (permission and intent) and eighteen classification algorithms. We designed two Targeted Type-II Evasion Attacks (TRPO-MalEAttack and PPO-MalEAttack) based on reinforcement learning to exploit vulnerabilities in the above malware detection models. The attacks aim to add minimum perturbations in each malware application and convert it into an adversarial application that can fool the malware detection models. The TRPO-MalEAttack achieves an average fooling rate of 95.75% (with 2.02 mean perturbations), reducing the average accuracy from 86.01% to 49.11% in thirty-six malware detection models. On the other hand, The PPO-MalEAttack achieves a higher average fooling rate of 96.87% (with 2.08 mean perturbations), reducing the average accuracy from 86.01% to 48.65% in the same thirtysix detection models. We also develop a list of the TEN most vulnerable android permissions and intents that an adversary can use to generate more adversarial

applications. Later, we propose a defense strategy (MalVPatch) to counter the adversarial attacks on malware detection models. The MalVPatch defense achieves higher detection accuracy along with a drastic improvement in the adversarial robustness of malware detection models. Finally, we conclude that investigating the adversarial robustness zof models is necessary before their real-world deployment and helps achieve adversarial superiority in android malware detection.

2. ‘ ‘ You are what the permissions told me!Android malware detection based on hybrid tactics,’ ’ J. Inf. Secur. Appl., vol. 66, May 2022, Art. no. 103159 Author: H. Wang, W. Zhang, and H. He,

Recent years have witnessed a significant increase in the use of Android devices in many aspects of our life. However, users can download Android apps from third-party channels, which provides numerous opportunities for malware. Attackers utilize unsolicited permissions to gain access to the sensitive private intelligence of users. Since signature-based antivirus solutions no longer meet practical needs, efficient and adaptable solutions are desperately needed, especially in new variants. As a remedy, we propose a hybrid Android malware detection approach that combines dynamic and static tactics. We firstly adopt static analysis inferring different permission usage patterns between malware and benign apps based on the machine-learning-based method. To classify the suspicious apps further, we extract the object reference relationships from the memory heap to construct a dynamic feature base. We then present an improved state-based algorithm based on DAMBA. Experimental results on a realworld dataset of 21,708 apps show that our approach outperforms the wellknown detector with 97.5% F1-measure. Besides, our system is demonstrated to resist permission abuse behaviors and obfuscation techniques.

3. ‘ ‘ Metaheuristics with deep learning model for cybersecurity and Android malware detection and classification,’ ’ Appl. Sci., vol. 13, no. 4, p. 2172, Feb. 2023 Author: A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, and S. Shamsudheen,

Since the development of information systems during the last decade, cybersecurity has become a critical concern for many groups, organizations, and institutions. Malware applications are among the commonly used tools and tactics for perpetrating a cyberattack on Android devices, and it is becoming a challenging task to develop novel ways of identifying them. There are various malware detection models available to strengthen the Android operating system against such attacks. These malware detectors categorize the target applications based on the patterns that exist in the features present in the Android applications. As the analytics data continue to grow, they negatively affect the Android defense mechanisms. Since large numbers of unwanted features create a performance bottleneck for the detection mechanism, feature selection techniques are found to be beneficial. This work presents a Rock Hyrax Swarm Optimization with deep learning-based Android malware detection (RHSODL-AMD) model. The technique presented includes finding the Application Programming Interfaces (API) calls and the most significant permissions, which results in effective discrimination between the good ware and malware applications. Therefore, an RHSO based feature subset selection (RHSO-FS) technique is derived to improve the classification results. In addition, the Adamax optimizer with attention recurrent autoencoder (ARAE) model is employed for Android malware detection. The experimental validation of the RHSODL-AMD technique on the Andro-AutoPsy dataset exhibits its promising performance, with a maximum accuracy of 99.05%

4. ‘ ‘ A method for automatic Android malware detection based on static analysis and deep learning,’ ’ IEEE Access, vol. 10, pp. 117334– 117352, 2022. Author: M. Ibrahim, B. Issa, and M. B. Jasser.

The computers nowadays are being replaced by the smartphones for the most of the internet users around the world, and Android is getting the most of the smartphone systems’ market. This rise of the usage of smartphones

generally, and the Android system specifically, leads to a strong need to effectively secure Android, as the malware developers are targeting it with sophisticated and obfuscated malware applications. Consequently, a lot of studies were performed to propose a robust method to detect and classify android malicious software (malware). Some of them were effective, some were not; with accuracy below 90%, and some of them are being outdated; using datasets that became old containing applications for old versions of Android that are rarely used today. In this paper, a new method is proposed by using static analysis and gathering as most useful features of android applications as possible, along with two new proposed features, and then passing them to a functional API deep learning model we made. This method was implemented on a new and classified android application dataset, using 14079 malware and benign samples in total, with malware samples classified into four malware classes. Two major experiments with this dataset were implemented, one for malware detection with the dataset samples categorized into two classes as just malware and benign, the second one was made for malware detection and classification, using all the five classes of the dataset. As a result, our model overcomes the related works when using just two classes with F1-score of 99.5%. Also, high malware detection and classification performance was obtained by using the five classes, with F1-score of 97%

5. **Machine learning-based adaptive genetic algorithm for Android malware detection in auto-driving vehicles, Appl. Sci., vol. 13, no. 9, p. 5403, Apr. 2023.Author: L. Hammood, İ. A. Doğru, and K. Kılıç.**

The growing trend toward vehicles being connected to various unidentified devices, such as other vehicles or infrastructure, increases the possibility of external attacks on“vehicle cybersecurity (VC). Detection of intrusion is a very important part of network security for vehicles such as connected vehicles, that have open connectivity, and self-driving vehicles. Consequently, security has become an important requirement in trying to protect these vehicles as attackers have become more sophisticated in using malware that can penetrate and harm vehicle control units as technology advances. Thus, ensuring the vehicles and the network are safe is very important for the growth of the automotive industry and for people to have more faith in it. In this study, a machine learning-based detection approach using hybrid analysis-based particle swarm optimization (PSO) and an adaptive genetic algorithm (AGA) is presented for Android malware detection in auto-driving vehicles. The “ CCCS-CIC-AndMal-2020” dataset containing 13 different malware categories and 9504 hybrid features was used for the experiments. In the proposed approach, firstly, feature selection is performed by applying PSO to the features in the dataset. In the next step, the performance of XGBoost and random forest (RF) machine learning classifiers is optimized using the AGA. In the experiments performed, a 99.82% accuracy and F-score were obtained with the XGBoost classifier, which was developed using PSO-based feature selection and AGA-based hyper parameter optimization. With the random forest classifier, a 98.72% accuracy and F-score were achieved. Our results show that the application of PSO and an AGA greatly increases the performance in the classification of the information obtained from the hybrid analysis.

EXISTING WORK: Various methods proposed in related research to improve malware detection in android applications. Some methods aim to increase accuracy, while others focus on providing large datasets or utilizing different set of features. The methods used the random forest algorithm and introduced the PDMS approach for detecting malware in android applications. The experiments demonstrated high accuracy rates in detecting malware sample, and PDMS proved effective in identifying previously unknown malware with low false positive rates.

III.METHODOLOGY AND DISCUSSION

Detecting malware on mobile devices through machine learning-based reverse engineering of Android applications involves a comprehensive methodology that encompasses several key steps. Initially, researchers collect a diverse dataset of Android applications, comprising both benign and malicious samples, covering various categories and versions to capture a broad spectrum of behaviors. The next step involves feature extraction, where relevant static (e.g., permissions, API calls, manifest details) and dynamic features (e.g., runtime behavior, system interactions) are extracted from the APKs. Following this, the extracted features undergo pre-processing to clean and prepare them for analysis, including handling missing or corrupted data and normalization for machine learning algorithms. Machine learning models are then carefully selected and trained using the prepared dataset, with common choices including supervised (e.g., Random Forests, SVM) and unsupervised learning (e.g., clustering). The trained models are evaluated using validation data, with performance metrics like accuracy and recall assessed to optimize the model. During the discussion phase, researchers explain the rationale behind feature selection, analyze model performance, discuss challenges and limitations encountered, compare their approach with existing methods, and suggest future directions for improvement. Additionally, ethical considerations related to privacy and data security are addressed, alongside practical implications for deploying such systems in real-world settings. Through this rigorous methodology and discussion, researchers can develop effective machine learning-based solutions for detecting mobile malware.

System Architecture

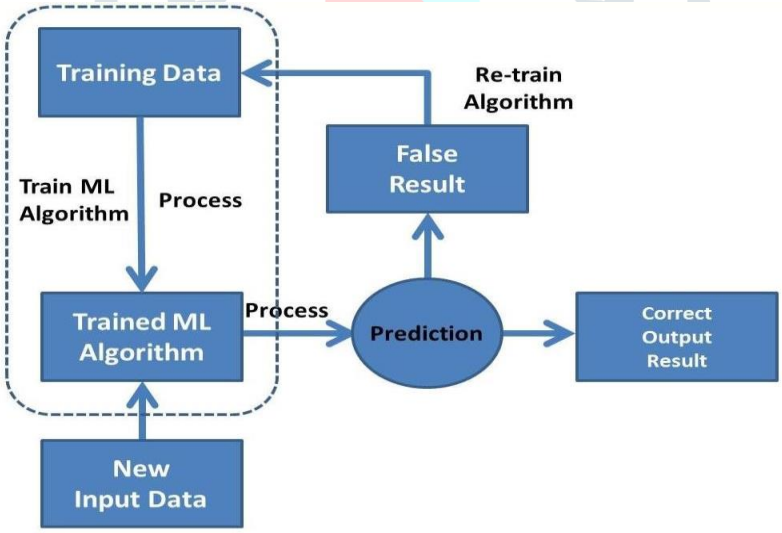


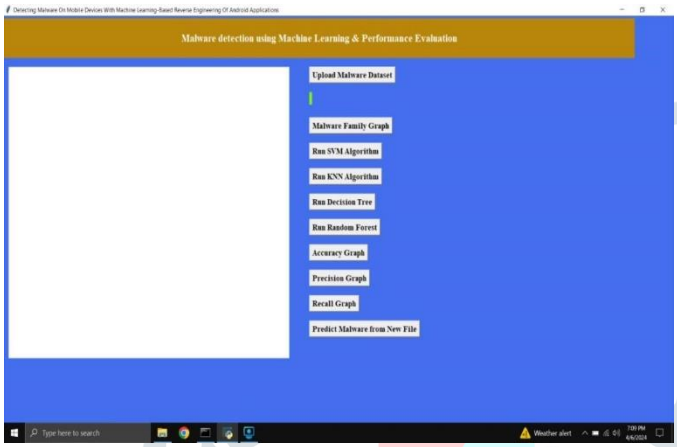
Fig 1: System Architecture

The authors present a unique Android malware detection approach dubbed Permissionbased Malware Detection Systems (PMDS) based on a study of 2950 samples of benign and malicious Android applications. In PMDS, requested permissions are viewed as behavioral markers, and a machine learning model is built on those indicators to detect new potentially dangerous behavior in unknown apps depending on the mix of rights they require. PMDS identifies more than 92– 94% of all heretofore unknown malware, with a false positive rate of 1.52– 3.93%. The authors of this article is solely use the machine learning ensemble learning method Random Forest supervised classifier on Android feature malware samples with 42 features respectively. Their objective was to assess Random Forest's accuracy in identifying Android application activity as harmful or benign. Dataset 1 is built on 1330 malicious apk samples and 407 benign ones seen by the author. This is based on the collection of feature vectors for each application. Based on an ensemble learning approach, Congyi proposes a concept for recognizing and

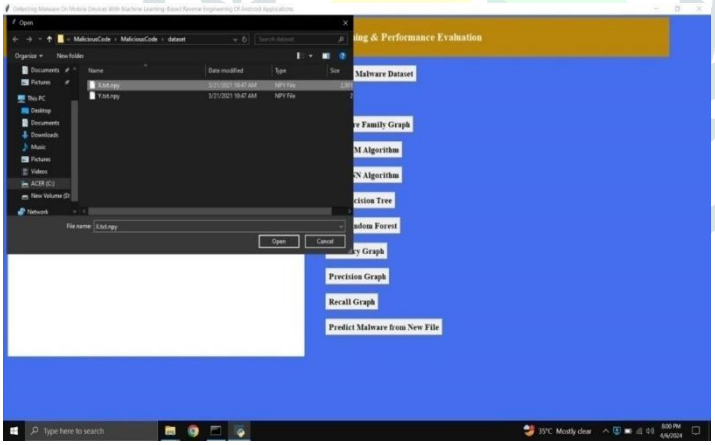
distinguishing Android malware. To begin, a static analysis of the Android Manifest file in the APK is done to extractsystem characteristics such as permission calls, component calls, and intents. Then, to detect malicious apps, they deploy the XGBoost technique, which is an implementation of ensemble learning. Analysing more than 6,000 Android apps on the Kaggle platform provided the initial data for this experiment. They tested both benign and malicious apps based on 3 feature sets for a testing set of 2,000 samples and used the remaining data to create a training set of 6,315 samples. Additional approaches include, an SVM-based malware detection technique for the Android platform that incorporates both dangerous permission combinations and susceptible API calls as elements in the SVM algorithm.

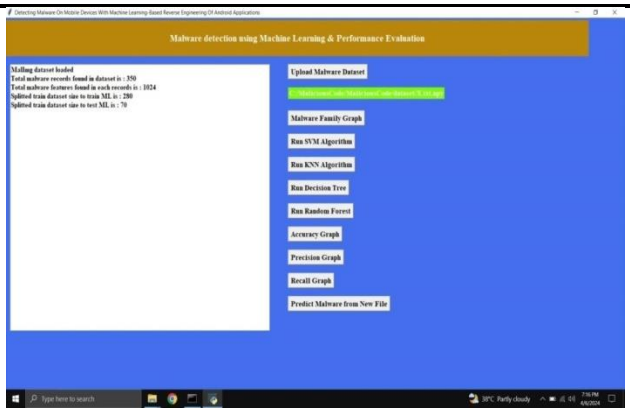
IV.RESULTS

LOGIN PAGE

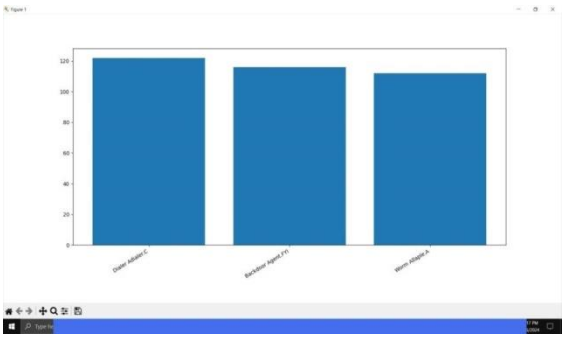


Entered uploading Malware dataset

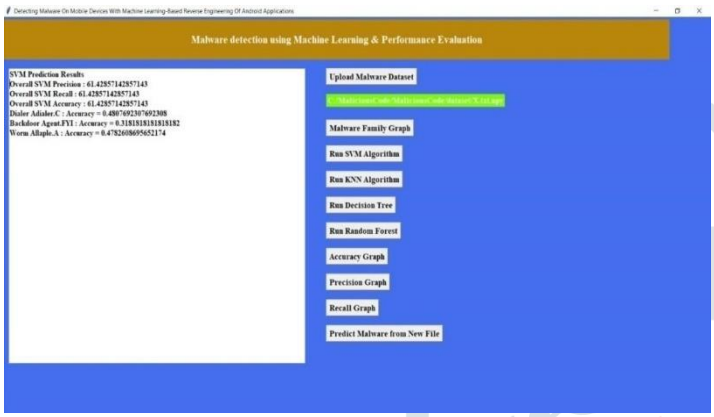




Uploaded dataset



Graph Showing Different Malwares



SVM Accuracy of Trained Malware Dataset

V.CONCLUSION

The proposed novel application of machine learning to identify malware in mobile apps offers a robust and efficient solution for combating the growing threat of malicious software on mobile devices. By leveraging advanced machine learning algorithms and effective feature selection techniques, the application demonstrates high accuracy in distinguishing between benign and malicious apps. Designed with a focus on scalability, efficiency, and continuous improvement, the application provides a proactive defense mechanism to safeguard users against evolving cyber threats, thereby enhancing the security and integrity of mobile ecosystems.

VI. References

[1] H. Rathore, A. Nandanwar, S. K. Sahay, and M. Sewak, ‘ ‘ Adversarialsuperiority in Android malware detection: Lessons from reinforcementlearning based evasion attacks and defenses,’ ’ Forensic Sci. Int., Digit.Invest., vol. 44, Mar. 2023, Art. no. 301511.

[2] H. Wang, W. Zhang, and H. He, ‘ ‘ You are what the permissions told me!Android malware detection based on hybrid tactics,’ ’ J. Inf. Secur. Appl.,vol. 66, May 2022, Art. no. 103159.

[3] A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, and S. Shamsudheen,‘ ‘ Metaheuristics with deep learning model for cybersecurity and Androidmalware detection and classification,’ ’ Appl. Sci., vol. 13, no. 4, p. 2172,Feb. 2023.

[4] M. Ibrahim, B. Issa, and M. B. Jasser, ‘ ‘ A method for automatic Androidmalware detection based on static analysis and deep learning,’ ’ IEEEAccess, vol. 10, pp. 117334– 117352, 2022.

[5] L. Hammood, İ. A. Doğru, and K. Kılıç, ‘‘Machine learning-based adaptivegenetic algorithm for Android malware detection in auto-driving vehicles,’’Appl. Sci., vol. 13, no. 9, p. 5403, Apr. 2023.

[6] P. Bhat and K. Dutta, ‘‘A multi-tiered feature selection model for Androidmalware detection based on feature discrimination and information gain,’ ’ J. King Saud Univ.-Comput. Inf. Sci., vol. 34, no. 10, pp. 9464– 9477,Nov. 2022.

[7] D. Wang, T. Chen, Z. Zhang, and N. Zhang, ‘ ‘ A survey of Android malwaredetection based on deep learning,’ ’ in Proc. Int. Conf. Mach. Learn. CyberSecur. Cham, Switzerland: Springer, 2023, pp. 228– 242.

[8] Y. Zhao, L. Li, H. Wang, H. Cai, T. F. Bissyandé, J. Klein, and J. Grundy,‘‘On the impact of sample duplication in machine-learning-based Androidmalware detection,’’ ACM Trans. Softw. Eng. Methodol., vol. 30, no. 3,pp. 1–38, Jul. 2021.

[9] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, ‘‘Deep learning basedmalware detection for Android systems: A comparative analysis,’’ Tehničkivjesnik, vol. 30, no. 3, pp. 787–796, 2023.

[10] H.-J. Zhu, W. Gu, L.-M. Wang, Z.-C. Xu, and V. S. Sheng, ‘‘Androidmalware detection based on multi-head squeeze-and-excitation residualnetwork,’ ’ Expert Syst. Appl., vol. 212, Feb. 2023, Art. no. 118705.

[11].Russello G, Jimenez AB, Naderi H, van der Mark W. Firedroid: hardening security in almost-stock android. In: Proceedings of the 29th annual computersecurity applications conference. ACM; 2013. p. 319–28.

[12].JacobG, Compare ttiPM, Neug schwandtner M, KruegelC , Vigna G.Astatic, packer-agnostic filter to detect similar malware samples. In: Detection of intrusions and malware, and vulnerability assessment. Springer; 2012. p. 102–22.

[13].SanzB, SantosI, Ugarte PedreroX, LaordenC, NievesJ, Bringas PG. Anomaly detection using string analysis for android malware detection. In: Inter-national joint conference SOCO’13-CISIS’13-ICEUTE’13. Springer; 2014. p. 469–78.

[14].ArpD, Spreitzenbarth M, HubnerM, GasconH, RieckK. DREBIN: effective and explainable detection of android malware in your pocket. Symposiumon network and distributed system security (NDSS); 2014.

[15].YerimaSY, SezerS, McWilliamsG, MuttikI. A new android malware detection approach using bayesian classification. In: 2013 IEEE 27th international conference on advanced information networking and applications. IEEE; 2013. p. 121–8.

[16].ArztS,etal. Flow droid: precise context, flow, field, object-sensitive and life cycle-awaretaint analysis for android apps. In: ACMSIGP LAN notices, Vol.49. ACM; 2014. p.259–69.

[17]. ArmandoA, ChiarelliG, CostaG, DeMaglieG, MammolitiR, MerloA. Mobile app security analysis with the MAVeriC static analysis module. JoWUA 2014; 5(4):103–19.

[18].FarukiP, GanmoorV, LaxmiV, GaurMS, BharmalA. Androsimilar: robust statistical feature signature for android malware detection. In: Proceedings of the 6th international conference on security of information and networks. ACM; 2013. p. 152–9.

[19].ZhengM,SunM,LuiJ.Droidanalytics:asignaturebasedanalyticsystemtocollect,extract,analyzeandassociateandroid malware.In:12thIEEEinternationalconferenceontrust,securityandprivacyincomputingandcommunications.IEEE;201 3.p.163–71.

[20].McClurgJ,FriedmanJ,NgW.Androidprivacyleakdetectionviadynamictaintanalysis.Electric.Eng.Comput.Sci.201 3;450.[22].YanLK,YinH.Droidscope:seamlesslyreconstructingtheOSanddalviksemanticviewsfordynamicandroidma lwareanalysis.In:21stUSENIXsecuritysymposium (USENIX security 12); 2012. p. 569–84.

[23].RastogiV,ChenY,EnckW.AppsPlayground:automaticsecurityanalysisofsmartphoneapplications.In:Proceedingso fthethirdACMconferenceondata and application security and privacy. ACM; 2013. p. 209–20.

[24].Spreitzen barth M, FreilingF, EchtlerF, SchreckT, Hoffmann J. Mobiles and box: having a deeper look into android applications. In: Proceedings of the 28th annual ACM symposium on applied computing. ACM; 2013. p. 1808–15.

[25].Weich selbaumL, Neug schwand tnerM, LindorferM, FratantonioY, vander VeenV, Platzer C.Andrubis. Andrubis,1. Vienna University of Technology; 2014. Tech. Rep. TRISECLAB-0414.