International Conference on Recent Innovations in Science, Engineering & Management
R.R. INSTITUTE OF TECHNOLOGY, Bengaluru (India)
10th - 11th May 2019 www.conferenceworld.in ISBN : 978-93-87793-87-3

A SURVEY ON ANDROID MALWARE DETECTION USING MACHINE LEARNING

Niranjan.M[1],Ksheeraja.P[2],Govardhana.D.K[3],Athira.S[4],

Dr.M.Vinoth Kumar[5]

[1,2,3,4]Department of ISE, Dayananda Sagar Academy of Technology and Management, Bangalore, India.
[5]Associate Professor, [4]Department of ISE,
Dayananda Sagar Academy of Technology and Management, Bangalore, India.

ABSTRACT

The increasing reputation of Android primarily based smartphones caused the spreading of malicious programs advanced through attackers. The solution is to develop refined android malware detection techniques. To detect malware, various techniques which employ static and/or dynamic features extracted from Android applications. To subdue the manual updating overhead, various machine learning techniques are used for malware detection. To represent Android malware patterns based on Android's static features and dynamic behavior, Machine Learning classifiers can be broadly used for this purpose.In this literature survey paper, we goal to briefly discuss the exceptional strategies utilized in Android Malware Detection and attention on their pros and cons.

Keywords: Android, Android malware detection, static features, dynamic features, Machine learning classifiers

I. INTRODUCTION

The Android platform offers a number of competencies at a totally low charge and developed to be the most favoured operating system for smartphones. Android being open supply will increase the chance and increases extreme problems related to malicious packages. Also, the growth in the variety of applications inside the Android marketplace makes it a easy intention for malware authors[1].Having an accurate and deep know-how of the working of malware is important to expand preventive measures. The primary symptoms confirmed through devices when it is inflamed withmalware are slow performance, drop in battery percentage and unusual app behavior.[2]Malware comes in extraordinary forms like Trojans, again-doorways,Worms, Botnets, Spywares, Ransomware and Riskwares. Various procedures have to be devised to combat them based ontheir nature. The 2 predominant methods are static analysisand dynamic analysis. The malware detection packages usethose techniques in their working to alert users if the application suggests malicious behavior [3].

II. ANDROID

The Android operating device is a cell running system advanced with the aid of Google generally for

touchscreen devices, cell telephones, and tablets. Its layout permits users to control cellular gadgets intuitively, with cell phone interactions that mirror commonplace motions, inclusive of pinching, swiping, and tapping. Further to cellular devices, Google employs Android software in televisions, vehicles, and wristwatches, each fitted with specific person interfaces[3].

The Android platform includes a working system based totally upon Linux, a GUI, an internet browser and end-user applications that may be downloaded. Although the initial demonstrations of Android featured a normal QWERTY cell phone and huge VGA display screen, the running gadget changed into written to run on highly inexpensive handsets with traditional numeric keypads[3].

Android has the potential to freely adjust, invent and enforce our own device drivers and capabilities.There are five distinct layers of an Android operating machine[3].

- Android Application Framework: Android applications are programmed in Java language. After programming an app, the Android SDK device help to assemble the statistics and the useful aid files which encompass the XML files, the jar files, manifest documents and special images and stuff into one single archive bundle deal cope with a '.apk' extension. This apk record can be used to put in the app with most effective a single click on at the Android devices. Now, due to the fact the Android-running gadget behaves like a Linux surroundings, the app behavior is likewise the same here. Every app is taken into consideration as a separate individual from the alternative and runs in its very own virtual machine.

- Binder Interprocess Communication: This interface lets in a programmer to make software speak with other application. Greater often, it's no longer the packages that speak, it's the processes.

- System Services: The system services play a key position in exposing the low-diploma capabilities of the hardware and the Linux kernel to the excessive-stage packages. It stays from boot to reboot, i.e., the complete life of the system.

- Hardware Abstraction Layer: This lets in inserting functionality even as no longer having any adjustments to the machine. Each one in every of a kind system has an in another way designed Hardware Abstraction Layer for the cause that they'll be made tool precise. It consists of favoured structures: Module and device.The module structure is saved as a shared library which consists of the primary metadata together with the version wide variety, an author who designed the module and similar stuff. The device is the actual hardware of the product. It defines an extra whole model of the well-known hardware facts which incorporates recommendations and other similar stuff which is probably specific to each hardware.

- Linux Kernel: The android kernel includes extra features inclusive of the wakelocks, double-tap to liberate and other comparable capabilities embedded into the mobile operating tool. Features which include wakelock are essential because the kernel is going to paintings on a portable device and it

needs to be a bit extra competitive in memory and battery control; not like the primary Linux where energy control isn't a difficulty. Those additional necessities are prompted within the kernel in place of the system given that this stuff must not affect the integrated drivers.

III. MACHINE LEARNING

Machine learning is an software of artificial intelligence (AI) that provides structures with the potential to routinely analyze and enhance from revel in without being explicitly programmed. System gaining knowledge of focuses on the improvement of computer packages which can get right of entry to statistics and use it learns for themselves [3].

Machine Learning is a concept to analyze from examples and revel in, without being explicitly programmed. Rather than writing code, you feed facts to the ordinary algorithm, and it builds logic based totally at the statistics given.

Machine learning is an interdisciplinary research zone which consolidates thoughts from a few parts of science in particular, artificial intelligence, data hypothesis, arithmetic, and so forth. The prime focal point of machine learning explore is on the improvement of quick and productive learning calculations which can make expectations on information [3].

The categories in Machine Learning are:

- Supervised Learning: The machine attempts to research from the preceding examples which are given.
- Unsupervised Learning: The calculations are left to themselves to find fascinating structures in the information.
- Reinforcement Learning: A program will connect with a dynamic domain in which it must play out a specific objective. The program is furnished with criticism as far as remunerations and disciplines as it explores its concern space. Utilizing this calculation, the machine is prepared to settle on explicit choices. The machine is presented to a domain where it consistently prepares itself utilizing experimentation technique.

These machine learning mechanisms are used in android malware detection where in the malware types become the machine classifiers.

IV. ANDROID MALWARE DETECTION

The different approaches that have been utilized to identify android malware are [3]:

Static Approach: It includes dismantling and analyzing the source code to check functionalities and the nearness of malware without running the code. Static evaluation is done in a non-runtime environment. Typically a static evaluation device will look into software code for all feasible run-time behaviors and are

seeking out coding flaws, again doorways, and doubtlessly malicious code.

Dynamic Approach: It incorporates destroying and analyzing the source code to check functionalities and the closeness of malware with running the code. Dynamic evaluation is done even as a application is in operation. A dynamic check will display device memory, functional conduct, response time, and typical performance of the gadget.

V. LITERATURE SURVEY

Andrea Saracino [4] et al. presented MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention. This paper presents a novel staggered and behavior based, malware finder for Android gadgets called MADAM(Multi-Level Anomaly Detector for Android Malware). Specifically, to distinguish application misbehaviors, MADAM screens the gadget activities, its cooperation with the client and the running applications, by recovering five gatherings of highlights at four distinct dimensions of deliberation, specifically the kernel level, application-level, client level and package level. For a few gatherings of highlights, MADAM applies an anomaly based methodology, for different gatherings, it actualizes a signature based methodology that considers standards of conduct that we have gotten from known malware misbehaviors. In reality, MADAM has been designed to locate malicious behavioral styles extracted from numerous classes of malware. This multi-stage behavioral analysis lets in MADAM to come across misbehaviors standard of just about all malware which can be determined inside the wild [4]. The adequacy of MADAM is assessed against three datasets of malevolent applications, specifically the Genome, Contagio-Mobile, and VirusShare datasets. MADAM has additionally possessed the capacity to recognize 9 malware families which sidestep VirusTotal checks, specifically, the zero-day assault Poder[4].

Moreover, a behavior based scientific classification of existing Android bits of malware into seven classes used to infer the basic examples of misbehaviors over a similar class is proposed[4].The principle oddity of MADAM is its cross-layer approach and a novel incorporation of methods that give high adequacy low overhead. To confirm that such a methodology is for sure practical, a substantial broad arrangement of tests has been performed to demonstrate exactly its adequacy [4].

Ming Fan et al.[5] presented DAPASA: Detecting Android Piggybacked Apps through Sensitive Sub graph Analysis. This work recognizes Android piggybacked applications by using the discernable invocation examples of delicate APIs between the rider and carrier. Sensitive APIs are administered by consents for applications to get to sensitive data or to perform delicate undertakings. To additionally comprehend the recognizable invocation designs, two assumptions are set up dependent on an observational investigation of piggybacked applications.

Suleiman Y. Yerima et al.[6] presented DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection. This paper displays and examines a novel classifier fusion approach that uses a multilevel architecture to expand the prescient intensity of machine learning calculations. The system, called DroidFusion, is intended to prompt a classification Model for Android malware recognition by training

various base classifiers at the lower level. A set of positioning based calculations are then used to infer combination plans at the larger amount, one of which is chosen to assemble the last model. The structure is able of utilizing not just traditional singular learning calculations like Decision Trees or Naive Bayes, yet in ensemble learning calculations like Random Forest, Random Subspace, Boosting and so forth for enhanced grouping exactness trained on a training set using a stratified N-fold cross-validation technique to assess their relative prescient correctness's. The results are used by four distinctive positioning based calculations that characterize certainly criteria for the choice and ensuing mix of a subset of the relevant base classifiers. The results of the positioning, calculations are consolidated in sets with the end goal to discover the most grounded combine, which is in this manner used to construct the last DroidFusion demonstrate.

Jin Li et al.[7] presented Significant Permission Identification for Machine Learning Based Android Malware Detection. This paper presents SIGPID, a methodology that extracts significant permissions from applications, and utilizes the extricated data to successfully recognize malware utilizing supervised learning algorithms. The goal of SIGPID is to recognize malware productively and precisely. This methodology investigates permissions and at that point recognizes just the ones that are noteworthy in recognizing malignant and benign applications. In particular, a multilevel data pruning approach including permission ranking with negative rate, permission mining with association rules and support based permission ranking to extract significant permissions strategically is proposed. At that point, machine learning based characterization calculations are utilized to arrange distinctive kinds of malware and benign applications.

As an information-driven methodology, SIGPID progressively decides critical authorizations based on significant permissions by the applications rather than statically characterizing hazardous consents dependent on their expected administrations. This key contrast enables our way to deal with recognize more malware than the methodology that utilizes the dangerous list alone.

Nikola Milosevic presented Machine learning aided malware classification ofAndroid applications[8].This paper uses two static malware examination approaches: in one authorization the application is asking for while the second we are the entire source code of the application. With the end goal to computerize the investigation process, two machine learning approaches are considered, to be specific clustering and classification. Classification is utilized for distinguishing to which of an arrangement of class or subpopulation new observation has a place. It utilizes supervised machine learning the approach in which the model is made out of existing, labeled observation examples. Since software can be arranged into malware also, goodware, the assignment of malware recognition can be demonstrated as a classification issue. Clustering is a strategy for unsupervised machine discovering that is ready to make groups of comparable elements. Clustering algorithms are valuable when there is just a little part of dataset marked. In view of the labeled examples, it is conceivable to surmise the class of the clustered data in indistinguishable bunches from the named information. he practical implication is that labels obtained in unsupervised learning, during the clustering can be later used to retrain classification model with more data. This methodology is a semi-supervised learning.

Fei Tong et al.[9] presented A Hybrid Approach of Mobile Malware Detection in Android. This paper proposed a novel hybrid approach for mobile malware detection by embracing both dynamic and static analysis. Accumulation of execution data of sample malware and benign applications utilizing a net_link technology to create patterns of system calls identified with document and system get to is done. Moreover, a malicious pattern set and an ordinary pattern set is developed by looking at the examples of malware and benign applications with one another. For identifying an obscure application, a dynamic technique to gather its system calling information is utilized. At that point, they are contrasted and both the malicious and ordinary pattern sets disconnected with the end goal to pass judgment on the unknown application.

Rashmi Rupendra Chouhan et al. [10] presented A Preface on Android Malware: Taxonomy, Techniques and Tools.This paper describes the various approaches are presented to detect malware at two stages,

I. Before execution i.e. Static approach

II. At the time of execution i.e. Dynamic approach.

Detailed description of techniques and tools used for malware detection in Android is given. This paperacts as a base to understand the taxonomy of malwares in Android.

Manar Mohamed et al. [11] presented SMASheD: Sniffing and Manipulating Android Sensor Data for Offensive Purposes.This paper portrays SMASheD, a real structure under the present Android environment that can be utilized to stealthily sniff and also control huge numbers of the Android's confined sensors (even touch input). SMAShed endeavours the Android Debug Bridge (ADB) functionality and empowers a malevolent application with just the INTERNET authorization to read, and write to, various distinctive sensor information records voluntarily. Crushed is the system that can sniff and control ensured sensors on unrooted Android gadgets, without client mindfulness, without consistent gadget PC association and without the need to contaminate the PC.

Jeremy Martin et al.[12] presented A Study of MAC Address Randomization in Mobile Devices and When it Fails.This paper exhibits the principal wide-scale investigation of MAC address randomization in the wild, including a definite breakdown of various randomization procedures by operating systems, manufacture, and model of the device. It recognizes various flaws in the executions which can be exploited to defeat randomization as performed by existing gadgets. It is demonstrated that gadgets usually make improper utilization of randomization by sending wireless frames with the genuine, worldwide location when they ought to utilize a randomized location .This imperfection allows a functioning assault that can be utilized in specific situations to follow any existing wireless device.

YinxingXue et al.[13] presented Auditing Anti-Malware Tools by Evolving Android Malware and Dynamic Loading Technique. In this paper, a methodology named MYSTIQUE-S is executed, as a service-oriented malware generation system. MYSTIQUE-S consequently chooses attack features under different client situations and conveys the corresponding malicious payloads at runtime. Depending on the dynamic code

binding and loading (through reflection) procedures, MYSTIQUE-S empowers dynamic execution of payloads on client gadgets at runtime.

Jing Chen et al.[14] presented Uncovering the Face of Android Ransomware: Characterization and Real-time Detection.This paper centers around the Android platform and aims to describe existing Android ransomware. These samples are systematically characterized based on several aspects, including the timeline and malicious features. To identify ransomware that extorts clients by encoding information, a novel real-time detection system is proposed, called RansomProber. By breaking down the UI gadgets of related exercises and the directions of clients' finger developments, RansomProber can induce whether the document encryption tasks are started by clients.

Ali Feizollah et al.[15] presented AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection. This paper assesses the adequacy of Android Intents (explicit and implicit) as a distinctive element for distinguishing malicious applications. Intents are semantically rich highlights that can encode the aims of malware when contrasted with other very much contemplated highlights, for example, permissions. The paper additionally contends that this kind of highlight isn't the ultimate solution. It ought to be utilized related to other known features.

Karthick S et al.[16] presented Static Analysis Tool for Identification of Permission Misuse byAndroid Applications.This paper, for the most part, centers around distinguishing how the permissions granted to an explicit application is abused by another application utilizing SharedUserID. The paper likewise proposes a security apparatus that distinguishes a list of applications which are abusing the permissions in a client's Android cell phone. The practicality of the device is tried by utilizing a Proof-of-Concept (PoC) usage of the security apparatus.

Jixin Zhang et al.[17] presentedDalvikOpcode Graph Based Android Malware Variants Detection Using Global Topology Features. This paper proposes a novel technique to assemble a graph of Dalvikopcode and break down its worldwide topology properties, which will initially develop a weighted probability graph of operations, and afterwards utilize data entropy to prune this graph while holding data as additional as could be allowed, the following extraction of a few worldwide topology highlights of the graph to represent malware is done, and finally, search the similarities with these highlights between programs. These worldwide topology highlights detail the abnormal state qualities of malware. This methodology gives a lightweight a system to recognize Android malware variations dependent on graph hypothesis and data hypothesis.

Tejaswini Bhandarkar et al. [15] presented Enhancing Security of Android Phones. The Android operating system is broadly being utilized in mobile phones. This paper exhibits an Android Security Design with quickly disclosing security threats to Android phones and proposing secured solutions of Root Access Authorization and Permission Mechanisms for improving the security of Android Phones.

Ignacio Martín et al. [19] presented Android Malware Characterization Using Metadata and Machine Learning

Techniques. This paper analyzes metadata to discover a subset of features which have demonstrated prescient power and utilize them to create and test unique Machine Learning (ML) models. The investigations in this paper utilize Logistic Regression (LR), Support Vector Machines (SVMs), and Random Forests (RF) as three surely understood directed Machine Learning algorithms.

Hui-Juan Zhu et al. [20] presented DroidDet: effective and robust detection of Android malware using staticanalysis along with rotation forest model. Due to the characteristics of the Android platform, for example, supporting the informal App stores, open source strategy and the extraordinary resistance for App confirmation, it is inescapable that it faces difficult issues of malicious software intrusion. With the end goal to shield the clients from the genuine harms caused by Android malware,a minimal effort and high-proficient technique to extract permissions, sensitive APIs, monitoring system events and permission-rate as key highlights is proposed in this paper, and the ensemble Rotation Forest (RF) is utilized to build a model to recognize whether an Android App is malicious or not. To additionally assess the execution of the proposed model, it is contrasted with the cutting edge Support Vector Machine (SVM) model under the equivalent trial conditions.

Sen Chen et al. [21] presented Automated poisoning attacks and defensesin malware detection systems: An adversarial machine learning approach. Today, attackers can adapt by maximally subverting machine-learning classifiers through polluting training data, rendering latest machine learning-based malware recognition devices, (for example, DREBIN, DROIDAPIMINER, and MAMADROID) ineffective. This paper investigates the attainability of developing crafted malware samples; analyze how machine-learning classifiers can be misled under three diverse risk models; at that point reason that infusing precisely made data into training data can altogether diminish detection accuracy. To handle the issue, KUAFUDET, a two-stage learning enhancing methodology that learns versatile malware by adversarial detection is proposed. KUAFUDET incorporates an offline training stage that chooses and extracts features from the training set and an online detection stage that uses the classifier trained by the main stage. To additionally address the adversarial condition, these two stages are interwoven through a self-versatile learning plan, wherein automated camouflage detector is introduced to filter the suspicious false negatives and feed them once again into the training stage.

Tieming Chen et al.[22] presented TinyDroid: A Lightweight and Efficient Model for Android Malware Detection and Classification. With the end goal to detect Android malware viably, this paper proposes a novel lightweight static identification model, TinyDroid, utilizing instruction simplification and machine learning procedure. Initial, a symbol-based simplification method is proposed to abstract the opcode succession decompiled from Android Dalvik Executable records. At that point, N-gram is utilized to extract features from the improved opcode arrangement, and a classifier is trained for the malware discovery and classification tasks. Machine learning algorithms like K-Nearest Neighbors, Random Forest, Naive Bayes, and so forth are used. To enhance the proficiency and adaptability of the proposed model, a compression system is likewise used to reduce features and select models for the malware test dataset.

Zahoor-Ur Rehman et al. presented Machine learning-assisted signature and heuristic-baseddetection of malwares in Android devices [23].In this paper, a proficient hybrid system is exhibited for the detection of malware in Android Apps. The proposed system considers both signature and heuristic-based investigation for Android Apps. Reverse Engineering of the Android Apps is done to extract manifest files, and binaries and best in class machine learning algorithm are utilized to proficiently recognize malware. For this reason, a thorough arrangement of tests are performed utilizing different classifiers, for example, SVM, Decision Tree, W-J48 and KNN. The paper says that SVM if there should arise an occurrence of binaries and KNN if there should be an occurrence of manifest.xml documents, are the most appropriate choices in vigorously distinguishing the malware in Android gadgets.

VI. CONCLUSION

In this paper, the different methodologies used to distinguish malware utilizing machine learning and different techniques are examined. It is seen that the detection of a malicious substance utilizing the static approach is less productive when contrasted with the dynamic approach which continues checking the applications remotely. However, the vast majority of these approaches neglect to identify the parts of the malignant code that isn't executed. So it is smarter to join the best features in both the methodologies and to build up a hybrid technique which gives much better execution. The diverse machine learning calculations help in various malware identification be it static or dynamic methodology. We can presume that no single methodology is sufficient to make a framework secure and no single machine learning algorithm can give the required effectiveness.

REFERENCES

[1] Noei, E., Syer, M.D., Zou, Y., Hassan, A.E. and Keivanloo, I.,, March. A study of the relation of mobile device attributes with the user-perceived quality of Android apps. In Software Analysis, Evolution and Reengineering (SANER), 2018, pp. 469-469.

[2] Arshad, S., Shah, M.A., Wahid, A., Mehmood, A., Song, H. and Yu, H., SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for AndroidOperating System". IEEE Access, 2018, 6, pp.4321-4339.

[3] Zhou, L., Pan, S., Wang, J. and Vasilakos, A.V.,Machine learning on big data: Opportunities and challenges. Journal of Neurocomputing, 237, pp.350-361.

[4] Saracino, A., Sgandurra, D., Dini, G. and Martinelli, F.Madam: Effective and efficient behavior-based android malware detection and prevention". IEEE Transactions on Dependable and Secure Computing, 15(1), 2018, pp.83-97.

[5] Ming Fan, Jun Liu, Wei Wang, Haifei Li, ZhenzhouTian and Ting Liu.DAPASA: Detecting Android PiggyBacked Apps. IEEE Transactions on Information Forensics and Security. Volume 12, Issue 8.2017.

[6] Yerima, S.Y. and Sezer, S., DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection. IEEE Transactions on Cybernetics.2018.

[7] Li, J., Sun, L., Yan, Q., Li, Z., Srisa-an, W. and Ye, H. "Significant Permission Identification for Machine

Learning Based Android Malware Detection". IEEE Transactions on Industrial Informatics, 2018.

[8] Milosevic, N. and Dehghantanha, A., 2017. Machine learning aided malware classification of Android applications,Computer and Electrical Engineering, 61, 2017, pp.266-274.

[9] Tong, F. and Yan, Z., A hybrid approach of mobile malware detection in Android. Journal of Parallel and Distributed Computing, 103, 2017, pp.22-31.

[10] Chouhan, R.R. and Shah, A.K., A Preface on Android Malware: Taxonomy, Techniques and Tools. International Journal on Recent and Innovation Trends in Computing and Communication, 5(6),2017, pp.1111-1117.

[11] Mohamed, M., Shrestha, B. and Saxena, N., Smashed: Sniffing and manipulating android sensor data for offensive purposes. IEEE Transactions on Information Forensics and Security, 12(4), 2017, pp.901-913.

[12] Martin, J., Mayberry, T., Donahue, C., Foppe, L., Brown, L., Riggins, C., Rye, E.C. and Brown, D., A study of MAC address randomization in mobile devices and when it fails. Proceedings on Privacy Enhancing Technologies, 2017, pp.365-383.

[13] Xue, Y., Meng, G., Liu, Y., Tan, T.H., Chen, H., Sun, J. and Zhang, J., Auditing Anti-Malware Tools by Evolving Android Malware and Dynamic Loading Technique,. IEEE Trans. Information Forensics and Security, 12(7), 2017, pp.1529-1544.

[14] Chen, J., Wang, C., Zhao, Z., Chen, K., Du, R. and Ahn, G.J., Uncovering the face of android ransomware: Characterization and real-time detection. IEEE Transactions on Information Forensics and Security, 13(5), 2018, pp.1286-1300.

[15] Feizollah, A., Anuar, N.B., Salleh, R., Suarez-Tangil, G. and Furnell, S., Androdialysis: Analysis of android intent effectiveness in malware detection. Computers& security, 65, 2017, pp.121-134.

[16] Karthick, S. and Binu, S., Static Analysis Tool for Identification of Permission Misuse by Android Applications. International Journal of Applied Engineering Research, 12(24), 2018, pp.15169-15178.

[17] Zhang, J., Qin, Z., Zhang, K., Yin, H. and Zou, J., DalvikOpcode Graph Based Android Malware Variants Detection Using Global Topology Features. IEEE Access, 6, 2018, pp.51964-51974.

[18] Bhandarkar, T. and Babu, G.R., Enhancing Security of Android Phones,. International Journal on Recent and Innovation Trends in Computing and Communication, 5(6), 2017, pp.467-470.

[19] Martín, I., Hernández, J.A., Muñoz, A. and Guzmán, A., Android Malware Characterization Using Metadata and Machine Learning Techniques. Security and Communication Networks, 2018.

[20] Zhu, H.J., You, Z.H., Zhu, Z.X., Shi, W.L., Chen, X. and Cheng, L.,DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model. Neurocomputing, 2018, pp.638-646.

[21] Chen, S., Xue, M., Fan, L., Hao, S., Xu, L., Zhu, H. and Li, B., Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach. Computers& security, 73, 2018, pp.326-344.

[22] Chen, T., Mao, Q., Yang, Y., Lv, M. and Zhu, J., TinyDroid: A Lightweight and Efficient Model for Android Malware Detection and Classification. Mobile Information Systems, 2018.

[23] Rehman, Z.U., Khan, S.N., Muhammad, K., Lee, J.W., Lv, Z., Baik, .W., Shah, P.A., Awan, K. and

Mehmood, I., Machine learning-assisted signature and heuristic-based detection of malwares in Android devices. Computers & Electrical Engineering, 69, pp.828-841.