INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE
ISSN: 2057-5688

MALWARE PREVENTION IN ANDROID APPLICATIONS USING MACHINE LEARNING ALGORITHMS

[1]Mrs. Geetha, [2]Anugula Akshara, [3]B Sudeeksha Reddy, [4]Bantu sandhya, [5]Venkata laxmi

[1]Assistant Professor, Department of CSE(DS), Malla Reddy Engineering College for Women (Autonomous Institution – UGC, Govt. of India), Hyderabad, INDIA.

[2345]UG,Department of CSE(DS), Malla Reddy Engineering College for Women (Autonomous Institution – UGC, Govt. of India), Hyderabad , INDIA.

Abstract:

Malware has always been a problem in regards to any technological advances in the software world. Thus, it is to be expected that smart phones and other mobile devices are facing the same issues. In this paper, a practical and effective anomaly based malware detection framework is proposed with an emphasis on Android mobile computing platform. A dataset consisting of both benign and malicious applications (apps) were installed on an Android device to analyze the behavioral patterns. We first generate the system metrics (feature vector) from each app by executing it in a controlled environment. Then, a variety of machine learning algorithms: Decision Tree, K Nearest Neighbor, Logistic Regression, Multilayer Perceptron Neural Network, Naive Bayes, Random Forest, and Support Vector Machine are used to classify the app as benign or malware. Each algorithm is assessed using various performance criteria to identify which ones are more suitable to detect malicious software. The results suggest that Random Forest and Support Vector Machine provide the best outcomes thus making them the most effective techniques for malware detection.

I. INTRODUCTION

Malicious software's pervasiveness poses a severe threat to computer systems. This study focuses on how dynamic malware features like API calls, file systems, registry keys, printed string information, and network features are processed. Text analysis techniques are used to process the printable string information features. Over the API calls and PSI features, the Shannon entropy is also applied. Dynamic analysis was used to examine portable executable files in this study. Machine learning algorithms were used to develop malware classifiers after extracting the various run-time data. Despite advancements in the creation of anti-malware systems, malware continues to achieve its harmful goals. Keeping a computer system secure from malware infestation has become a major concern in recent years. Malware's exponential growth and sophistication is a major threat to computer and network

ISSN: 2057-5688

security. The computer's dependence is also a significant element, as almost all tasks, from daily life to business, are now performed on the computer [1]. There are two research communities that are working on the same project at the same time. One is working on dangerous software, while the other is working on defensive software to defend systems from malware

Malware is short for malicious software, which is software that is specifically designed to cause disruption, harm, or obtain unauthorized access to a computer system. Malware is a term used to describe malicious software that is designed to perform damaging actions [2]. On the basis of their behavior and mode of execution, dangerous programmes are classed as worms, viruses, Trojan horses, rootkits, backdoors, spyware, logic bombs, adware, and ransomware.

Malware Detection Techniques: Malware detection systems are designed to not only detect malware, but also to fight against harmful programmes that could destroy a computer system or network assets. The input can be represented in a variety of ways in order to detect and categories malware samples into appropriate families. The appropriate information or knowledge about the harmful file, which represents the malware file's behaviour, is required. Static, dynamic, and hybrid methodologies are used to assess diverse malware samples. The retrieved data is then expressed in the appropriate format, which is then used to train the detection system. Byte sequences, Opcodes, Strings, writing style of code, APIs calls, PE Header information, Memory access operations, Windows Registry, File system accesses, AV/Sandbox submissions, CPU Registers, Length of functions, Network Activities, and Generated Exceptions are among the fifteen different features that have been used by various researchers to develop malware detection systems, according to Ucci et al. (2017). The malware files are evaluated in the most basic sense; features are extracted and represented in an intermediate form before being fed into malware detection. This intermediate stage is crucial in the detecting process. The false positive rate is reduced if the extracted information is appropriate for detecting malware [3].

ISSN: 2057-5688

The following are the three primary categories in which many proposed malware detection approaches are classified.

II. RELATED WORK

This section presents a brief review of the existing schemes for malware detection. We also talk about the gaps in the available literature and how we might close them when creating the malware detection model.

Schultz et al [6] presented a network packet-based malware detection methodology based on cloud computing. They utilized data mining methods to minimize the branching of packets by accumulating packet information, whether or not it is relevant for malware identification. Enck et al. [7] introduced the Kirin framework, which allows us to identify malicious programs by looking at the rights they seek throughout the installation process. Kirin is built on a set of principles that assist us in reducing the impact of malware in Android apps.

Wagener et al. [8] suggested an automated and adaptable method for extracting malware behavior from system calls. To compute related distances, the alignment approach was utilized to find similarities, and the Hellinger distance was determined. The research claims that disguised malware versions with identical characteristics can be discovered. The authors claim that a phylogenetic tree that reflects malware's common functions can help with categorization.

Naval et al. [9] proposed a dynamic malware detection system that gathers system calls and creates a graph that discovers semantically meaningful pathways between them. A NP-complete issue is locating all semantically meaningful routes in a network. As a result, the authors assessed the most relevant pathways, which define malware behaviors that aren't seen in benign samples, to decrease the time complexity. According to the authors, the suggested approach beats its competitors because it can identify malware utilizing system-call injection assaults at a high rate, whereas other methods can't. The article contains a number of flaws, including performance overhead during path calculation, vulnerability to call-injection attacks, and the inability to effectively discover

ISSN: 2057-5688

all semantically meaningful pathways. The performance may be improved if these constraints are removed.

III. MACHINE LEARNING TO DETECT ANDROID MALWARE

The other method of performing static analysis to detect An- droid malware with ML is code-based analysis. TinyDroid [4] provided a model that looked at the most recent malware in the Drebin dataset. The model makes advantage of instruction simplification and machine learning. The opcode sequence was abstracted from the decompiled DEX files by converting APK to smali codes. The features were then retrieved using N-gram and combined with the example selection approach. For intrusion detection, the exemplar selection method used a clustering algorithm called Affinity Propagation to build a good representation of data (AP). This is because in AP, determining or estimating the number of clusters is not required prior to launching the programme. The resulting 2,3, and 4-gram sequences were then input into classifiers such as SVM, KNN, RF, and NB ML. The RF method was found to be the best for this scenario, with a True Positive Rate of 0.915, a False Positive Rate of 0.106, a Precision of 0.876, and a Recall of 0.915 for a 2-gram sequence. In comparison to the examined ML algorithms, high accuracy rates for the other 3 and 4-grams were also achieved. The proposed method, however, still has flaws, such as the use of malware samples from only a few research studies and few organizations, as well as the lack of metamorphic malware samples. As a result, some malware may go undetected.

IV RESULTS

A deep learning-based static analysis approach with an accuracy of 99.9% and an F1-score of 0.996 was tested with an accuracy of 99.9% and an F1-score of 0.996. A dataset of over 1.8 million Android apps was employed in this method. Malware characteristics were discovered using vectorised opcode retrieved from the bytecode of APKs using one-hot encoding. After testing models such as Recurrent Neural Networks, Long Short Term Memory Networks, Neural Networks, Deep Convents, and Diabolo Networks, it was determined that Bidirectional Long

ISSN: 2057-5688

Short-Term Memory (BiLSTMs) is the optimum model for this method. To construct a more thorough malware detection tool based on deep learning approaches, it is preferable to analyze the entire byte code with static analysis and verify the app behavior with dynamic analysis.

V CONCLUSION

Any smartphone is subject to security vulnerabilities, but attackers are particularly interested in Android smartphones. This is owing to the fact that it is open-source and has a bigger market share than other mobile operating systems. The Android architecture and security model, as well as potential threat vectors for the Android operating system, were discussed in this paper. A thorough assessment of the state-of-the-art ML-based Android malware detection algorithms was conducted based on the available literature, including the most recent research from 2016 to 2021. It went over the various ML and DL models and their performance in detecting Android malware, as well as code and APK analysis approaches, feature analysis and extraction methods, and the proposed methodologies' strengths and limitations. Malware aside, if a developer makes a mistake, a hacker will have an easier time finding and exploiting these flaws. As a result, strategies for detecting source code vulnerabilities using machine learning were discussed. The analysis found potential gaps in prior research as well as future research possibilities for improving Android OS security. Malware for Android, as well as detection approaches for it, are constantly growing. As a result, we feel that further assessments covering these developing concerns and their detection methods will be necessary. Since DL approaches have proven to be more accurate than typical ML models, more complete systematic reviews concentrating solely on DL-based malware detection on Android will be valuable to the research community, according to our findings in this study. Another area of interest for systematic reviews and studies is the prospect of employing reinforcement learning to uncover source code vulnerabilities.

VI REFERENCES

[1] Cui, Baojiang, et al. "Service-

ISSN: 2057-5688

oriented mobile malware detection system based on mining strategies." Pervasive and Mobile Computing 2015.

[2] Enck, William, MachigarOngtang, and Patrick McDaniel. "On lightweight mobile phone application certification." Proceedings of the 16th ACM conference on Computer and communications security. 2009.

[3] El Attar, Ali, RidaKhatoun, and Marc Lemercier."A Gaussian mixture model for dynamic detection of abnormal behavior in smartphone applications." 2014 global information infrastructure and networking symposium (GIIS).IEEE, 2014.

[4] Feizollah, Ali, et al. "A study of machine learning classifiers for anomaly-based mobile botnet detection." Malaysian Journal of Computer Science 2013.

[5] Shabtai, Asaf, et al. "Mobile malware detection through analysis of deviations in application network behavior." Computers & Security 2014.

[6] Stevanovic, Matija, and Jens MyrupPedersen."Machine learning for identifying botnet network traffic" (2013).

[7] Schultz, Matthew G., et al. "Data mining methods for detection of new malicious executables." Proceedings 2001 IEEE Symposium on Security and Privacy.S&P 2001.IEEE, 2000.

[8] Wei, Te-En, et al. "Android malware detection via a latent network behavior analysis." 2012 IEEE 11th international conference on trust, security and privacy in computing and communications.IEEE, 2012.

[9] Karnik, Abhishek, SuchandraGoswami, and RatanGuha. "Detecting obfuscated viruses using cosine similarity analysis." First Asia International Conference on Modelling& Simulation (AMS'07). IEEE, 2007.

[10] Moser, Andreas, Christopher Kruegel, and EnginKirda. "Exploring multiple execution paths for malware analysis." 2007 IEEE Symposium on Security and Privacy (SP'07). IEEE, 2007.

[11] Zhang, Boyun, et al. "Malicious codes detection based on ensemble learning." International conference on autonomic and trusted computing.Springer, Berlin, Heidelberg, 2007.

[12] Wagener, Gérard, and AlexandreDulaunoy. "Malware behaviour analysis." Journal in computer virology 2008.

ISSN: 2057-5688

[13] P. Beaucamps and J. Marion, ‘‘On behavioral detection,’’ in Proc. EICAR, vol. 9, 2009.

[14] Cha, Sang Kil, et al. "SplitScreen: Enabling efficient, distributed malware detection." Journal of Communications and Networks 2011.