International Journal of Gender, Science and Technology
ISSN: 2040-0748
UGC Care Group I Journal.
Vol-13 Issue-01 May 2024
https://ijgst.com.2024.v13.i1.pp125-131

DETECTION AND ANALYSIS OF MALICIOUS CODE ON ANDROID

TADIKAMALA VINEELA[1] JAINA SAICHAND[2] SYED RESHMA[3] P.RAJASEKHARAN[4]
[1]ASST.PROFESSOR, DEPARTMENT OF ARTIFICIAL INTELLIGENCE,
[2] ASST. PROFESSOR, DEPARTMENT OF ARTIFICIAL INTELLIGENCE,
[3]PROFESSOR, DEPARTMENT OF ARTIFICIAL INTELLIGENCE,
[4] ASST. PROFESSOR, DEPARTMENT OF ARTIFICIAL INTELLIGENCE,
[1,2,3,4] SRI MITTAPALLI COLLEGE OF ENGINEERING

Abstract: One of the most significant problems with operating systems and software is malware. These problems are also affecting the android foundation. It has been observed that several signature-based malware detection algorithms have been used. Even still, the algorithms failed to identify cryptic spyware. The accuracy of new malware discoveries remains a critical problem, despite the existence of many finding and analysis methodologies. Here we take a look at the methods currently used to detect and analyse malicious code on Android and highlight some of the best ones. In addition to thinking about it, we provide machine learning methods that may be used to analyse this kind of malware, and we'll also be doing semantic analysis. For harmful apps, we will have a database of authorisations. We want to compare this with the permissions retrieved from the program. In the end, the customer wants to know how much bad authorisation is in the software, and we help them out by analysing it via comments.

1.Introduction

Malicious software, also known as spyware, Trojan horses, backdoors, rootkits, or simply malicious software, encompasses a wide range of potentially harmful programs. Damage, theft, upset, or other evil deeds are the primary goals of malware. Malware is powerful enough to infect any computing device running an application, and anti-malware software is now universally available for personal computers (PC). The location processes used by smartphone devices are falling far behind the rapid growth of the mobile population. According to recent research, there are over 2.1 million Android apps available for download. More malware targeting the Android platform has been released as a result of the growing use of the Android framework. The outside parties developing these apps are responsible for their widespread distribution in the market. Even Google's own Android Market doesn't promise that every app is safe to use. There have been rumours of Trojans apps that, when installed, also install harmful malware and are difficult for Google's security measures to detect when published in the Google Play market. Android users are at risk from banking Trojans, bots, root exploits, phishing, SMS fraud, and phoney installers, among other dangers. Customers may freely download Android apps from the official Android app store, Google Playstore, or from third-party app shops. Android is a popular and open-source OS, which means that malware

developers are focussing on making dangerous apps for it. In spite of Google Playstore's best efforts to prevent malicious apps, some manage to make it to the market and inflict harm on users by stealing their phone numbers, email addresses, GPS coordinates, and other personal information or even taking over their phones remotely. Malware analysis, or the identification of such harmful apps, is therefore necessary since they pose a real danger to Android systems. Static analysis and dynamic analysis are the two main categories of Android malware. While dynamic analysis involves looking at how Android apps behave when run in limited environments, static analysis mostly involves analysing the code structure without running it. Given the ever-growing number of Android malware variants that pose zero-day vulnerabilities, it is crucial to have a reliable method for discovering these threats. Machine learning combined with static and dynamic analysis may detect new Android malware variants that pose zero-day risks, as opposed to signature-based approaches that need frequent updates to signature databases. Using the Support Vector Machine technique, a wide-ranging yet lightweight static analysis was carried out in [5], with a respectable discovery accuracy of 94%. Two approaches were presented by Nikola Milosevic et al. [6] for static analysis-based classification: one relied on consents and the other featured representing the source code as a bag of words. The authors of [7–11] provide an alternative method that relies on identifying the most important consents and then using machine learning to assess them. The primary goal of this work is to use Genetic Algorithm to reduce the feature measurement to approximately half of the original feature set. This will allow machine learning classifiers to train with less complexity while still accurately classifying malware. As an alternative to the exhaustive strategy, which involves testing for 2N different combinations of features (where N is the number of features), a heuristic searching approach based on previous work called Genetic Algorithm has been used for feature choosing. In order to train two machine learning algorithms—Neural Network and Backing Vector Machine—using the improved feature set acquired via the Genetic method. While working on a much reduced feature measurement, a reasonable classification accuracy of over 94% is maintained, thus reducing the training time complexity of classifiers.

2. Background Work

There are various ways and techniques through malware or any malicious record can enter your framework or application. A portion of the basic strategies of malware getting interfered into the framework are as per the following: -

- Penetration: Malicious apps often use repackaging, upgrading, and downloading as penetration processes for activating installation and operating on the Android foundation.
- Repackaging: Installing harmful apps on an Android platform is a common practice for malware developers.Use a repackaging strategy to infect widely used apps with malware. An engineer may get their hands on these kinds of apps, modify them with malicious code, and then release them to the public or the official Android app store.
- Updating: When trying to identify malware, this process is even more

challenging.The malware developer can still use repackaging, but this time, instead of embedding the code within the application, they may include an update component that can download dangerous code while the program is running.

- Downloading: Here we have the tried-and-true method of assault. In order to get clients to download malicious software, malware developers must make the apps seem appealing.

3.Literature Review

Circulation of the article "Android Malware Detection Using Machine Learning on Image Patterns"[1] occurred in 2018. They ran the malware detection using 300 malicious records and 300 benign apk files; they got 183 malicious records and 300 positive greyscale pictures. The remaining 117 malware samples could not be turned into pictures due to corrupted apk documents or records without the classes.dex file. On top of that, none of the algorithms they used were really accurate. They were able to identify it by using three distinct classifier strategies: k-nearest neighbour (KNN), random forest (RF), and decision tree (DT).

"Android Malware Detection Using Machine Learning on Image Patterns"[1] was circulated in 2018. They sent 300 malicious records and 300 benign apk files through the malware detection process; they obtained 183 malicious records and 300 positive greyscale photos. Because of broken apk files or records missing the classes.dex file, the other 117 malware samples could not be transformed into images. Additionally, all of the algorithms used were woefully inaccurate. Using k-nearest neighbour (KNN), random forest (RF), and decision tree (DT) as their classifier algorithms, they successfully identified it.

Distribution of the publication "An Android Behaviour-Based Malware Detection Method using Machine Learning"[3] occurred in 2016. In an Android sandbox, they've suggested a Robotium program that can automatically launch and monitor the actions of any Android app. The software includes an automated trigger program for UI identification, which allows users to touch on mobile apps in a meaningful way. More extensive testing might be conducted by the software. In addition, they used the random forest technique to try to build a decision model using collected behaviour. Both its certainty value and the ability to judge if the mysterious software is malware have been shown. They could keep the result and the confidence level of the mysterious apk file in their records.

The 2018 research was published in "RanDroid: Android malware detection using random machine learning classifiers" [4]. They have put forward an Android malware detection framework that makes use of APIs, consents, and unique app data (such as dynamic code, reaction code, native code, cryptographic code, databases, etc.) to train and build a classification model that can detect malicious Android apps (malware) separately from legitimate ones.

4.System Analysis

Existing System

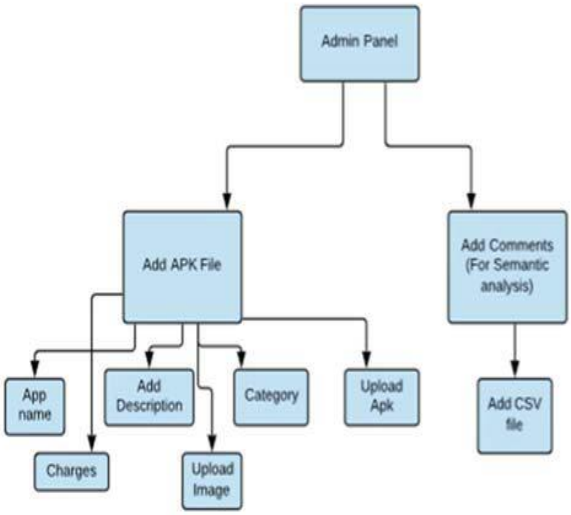
In order to train machine learning classifiers with less complexity while keeping their

https://ijgst.com.2024.v13.i1.pp125-131

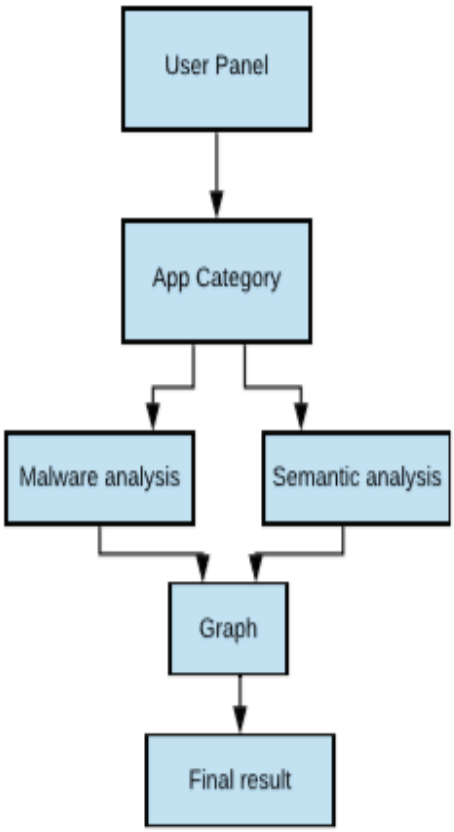
performance in malware classification, the study primarily contributes by reducing the feature dimension to less than half of the original feature-set using Genetic Algorithm. To avoid testing for 2N distinct combinations—where N is the number of features—the exhaustive feature selection technique was replaced with a heuristic search strategy based on fitness functions—the Genetic Algorithm—to choose features. Two machine learning algorithms, Support Vector Machine and Neural Network, are trained using the optimised feature set produced via Genetic algorithm. Classifiers' training time complexity is reduced while a respectable classification accuracy of over 94% is maintained by operating on a much smaller feature dimension.

Proposed system :-

Two Packages of Android Applications (APKs): Through reverse engineering, features like permissions and the count of app components like activity, services, content providers, etc., may be extracted from malware and goodware. The characteristics are saved in CSV format as a feature vector with the class labels "Malware" (represented by 0) and "Goodware" (represented by 1) correspondingly. We feed the CSV into the Genetic Algorithm to find the best collection of features and lower the dimensionality of the feature set. Two machine learning classifiers—Neural Network and Support Vector Machine—are trained using the optimised set of features that were acquired. The suggested technique relies on static characteristics extracted from AndroidManifest.xml, a file that every Android platform needs to know about apps. The APKs were disassembled and their static features were extracted using the Androguard tool. We have included both an admin and a user panel in our system. The administrator may submit the apk files, along with information and classifications, and also provide comments for semantic analysis in the admin interface.The user may see the application's data, including price, description, and name, in the user-panel after selecting the category. The proportion of the program that is harmful may be seen by the user. Plus, the user will get an accurate assessment of the program thanks to the graph presentation of the processed semantic analysis data.



Admin Panel 1



User Panel 1

5.Results

Malware Detection is able to identify various types of permissions based on the kind of permissions requested and then granted. Similarly, the application's appropriateness may be determined by using semantic analysis to gather the right comments. In order to determine if the client can use those particular apps or not, the appropriate yield is provided by the consent-based examination and the semantic investigation.

6. **Conclusion**

A methodology for investigating authorisation and semantics is presented in our study. By comparing it to a dataset, our technology can also detect malware authorisations that are application-dependent. The security framework and other client-side applications, such as malware detection software, may benefit from the suggested framework. However, our system does have limitations. Although the authorisations we are describing may differ from one customer to another, they are based on our standards. Even when one customer thinks it's completely free of malware, another might see the same consents as malicious software. The next version will include enhancements to it.

References

[1] Darus,Fauzi Mohd,Salleh Noor Azurati Ahmad,and Aswami Fadillah Mohd Ariffin.”Android Malware Detection Using Machine Learning on Image Patterns”2018 Cyber Resilience Conference(CRC).IEEE,2018.

[2] Vrama,P.Ravi Kiran,Kotari Prudvi raj,and KV Subba Raju.”Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms.”2017 InternationalConference on I-SMAC (IoT in Social,Mobile,Analytics and Cloud)(I-SMAC).IEEE2017.

[3] Chang,Wei-Ling, Hung-Min Sun,and Wei Wu.”An Android Behaviour-Based Malware Detection Method using Machine Learning.” 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC).IEEE, 2016.

[4] Koli, J. D. “RanDroid: Android malware detection using random machine learning classifiers.” 2018 Technologies for Smart-City Energy Security and Power (ICSESP).IEEE,2018.

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

[6] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, “A Multimodal Deep Learning Method for Android Malware Detection using Various Features,” vol. 6013, no. c, 2018

[7] A. Martin, F. Fuentes-Hurtado, V. Naranjo, and D. Camacho, “Evolving Deep Neural Networks architectures for Android malware classification,” *2017 IEEE Congr. Evol. Comput. CEC 2017 - Proc.*, pp. 1659–1666, 2017.

[8] X. Su, D. Zhang, W. Li, and K. Zhao, “A Deep Learning Approach toAndroid Malware Feature Learning and Detection,” 2016 IEEE Trust.,pp. 244–251, 2016.

[9] K. Zhao, D. Zhang, X. Su, and W. Li,

https://ijgst.com.2024.v13.i1.pp125-131

"Fest : A Feature Extraction and Selection Tool for Android Malware Detection," *2015 IEEE Symp. Comput. Commun.*, pp. 714–720, 4893.

[10] A. Feizollah, N. B. Anuar, R. Salleh, and A. W. A. Wahab, "A review on feature selection in mobile malware detection," *Digit. Investig.*, vol. 13, pp. 22–37, 2015.

[11] Srinu, Nidamanuri, Sampathi Sivahari, and Mastan Rao Kale. "Leveraging Radial Basis Function Neural Networks for Rainfall Prediction in Andhra Pradesh." *2022 International Conference on Computer, Power and Communications (ICCPC)*. IEEE, 2022.

[12] Muppavarapu, Rajasekhar, and Mastan Rao Kale. "An Effective Live Video Streaming System