MACHINE LEARNING APPROACH TO LEARN AND DETECT MALWARE IN ANDROID

Bindu P[1], Chandana K S[2], Ranjith U[3], Chandanraj R J[4]

Professor Krupa K S,
Dept. of Information Science and Engineering,
Global Academy of Technology, Karnataka, India

---***---

Abstract - *Smartphones have become indispensable in modern life as a result of their extensive use in recent years. New solutions have been developed by users to allow them to keep critical data on their mobile devices. Attackers' main focus, however, is on data related to privacy. As a result, hackers constantly develop new methods to steal data from users' devices. To guarantee the defence of users' confidential information from intruders, several antimalware solutions are created. Based on how they detect malware, we classify a lot of recent antimalware techniques. Our goal is to present a clear and brief overview of malware detection and defence procedures. We provide an ANN and SVM-based technique to identify malicious and good apps in this study.*

***Key Words**:* **Android Malware, Smartphones, Machine learning, SVM, ANN**

1. INTRODUCTION

Mobile phones and other smart devices are vital in people's lives due to their features and scalability. They interact with people in a variety of settings, including the workplace, entertainment, money management, etc. Smartphones are the major target for attackers since they are where people save their important data. Every day, new methods for attackers to obtain data from a smartphone are developed. Android malware is software or code created specifically to interfere with, harm, or intrusion into a system. Another name for it is harmful software.

In the area of IT security, malware is a serious hazard and source of concern. In terms of privilege escalation, tariff theft, remote control, and privacy leakage, it poses a threat to system and user security.

The system will be easily targeted by malware because it allows users to install unlicensed or unapproved software. Malware detection for Android devices becomes crucial.

With this project, we offer a workable method for locating mobile apps. Although the design and publication of their Android applications are convenient, the market for third-party applications faces difficulties. This does not guarantee smartphone security. The drawback of signature-based malware detection technology is that it allows users to identify spyware that they are unaware of.

1.1 Malware Classification

Malware is a software program that enters into a user's device without permission and has intentions to cause damage or steal personal information.

Virus: Virens sneak into the system and infect users or spread throughout it by affixing themselves to other executables. The virus "Creeper," one of the earliest varieties, was created in 1971 as a test run for a programme designed by Bob Thomas at BBN. As it did not damage the data, it was not malicious software that was actively operating.

Trojans: Trojans are malicious files that masquerade as helpful files in order to gain access to a system..

Worms: A sort of virus that may duplicate itself automatically on various PCs and gadgets and propagate across the internet. They can spread themselves without becoming a part of other programmes.

Spyware: The term "spyware" refers to a potentially unwanted programme (PUP). It is an undesirable programme that aims to steal personal information (such user passwords or bank account information) and Internet usage data without the user's knowledge or consent.

Adware: A type of spyware, adware. Yet, the adware does not want to harm the PC. Its primary goal is to pique a user's curiosity so that comparable advertising, emails, and pop-ups can be shown to them.

Ransomware: Ransomware is a type of malicious software in which an attacker encrypts all the victims' files and demands payment to get them decrypted.

1.2 Malware Features

Malware identification has two stages: feature extraction and classification/clustering. Malware characteristics refer to the data that can be used for machine learning algorithms. These specify the universe of ideas that can be represented by machine learning. Malware features can be divided into static and dynamic groups. The characteristics that can be extracted without the use of malware are known as static features. These characteristics are derived from malware binaries through analysis. While dynamic characteristics are those that are obtained after a binary file has been executed. The dynamic analysis process is used to extract these characteristics. Malware is categorized and grouped using machine learning methods. Only classification techniques are described in this paper. Some of the features of malware detection were mentioned in this part.

1.2.1 N-Gram

It is one of the more typical static characteristics and comprises of n consecutive bytes taken from the output of the hex dump. The most widely used characteristic, 4 Gram, refers to taking a combination of 4 bytes. You can use N-Gram in overlapping or non-overlapping situations. When there is no crossover, a byte that has already been used once can be used again in the following gramme.

1.2.2 Opcodes

Opcodes refer to various machine-level processes carried out by programmable executables. You can find these opcodes in the assembly code.

1.2.3 Strings

The definition of a string is a group of printable letters. They discovered that the headers in PE-format contain plain text that can be used to retrieve data. Additionally, non-PE executables have encoded strings as well that can be used with their information. A more precise meaning of strings states that they can only be used if they are "interpretable" and make some sort of semantic sense.

1.2.4 Memory Access

The primary memory is used to store a lot of information, including configuration, network activity, and window registry keys. Thus, by looking at how memory is utilized, crucial information about malware can be gleaned.

1.2.5 API Calls

The Application Programming Interface (API) acts as a link between the operating system and applications. Some duties, such as writing a file to the disc, can only be completed directly by the operating system, and the system library has a library of these functions. A call to the system library made by an application is known as an API call. For instance, the CopyFileW API will be used when a file needs to be copied. Many experts use API calls, which are a crucial component of malware detection. An API trace is what we term a sequence of API calls that can be preserved. This API trace can identify advanced malware behaviors like "walking through folders" and "copying itself to disc".

2. LITERATURE SURVEY

[1] **"Android Malware Detection through Machine Learning Techniques: A Review":** The author of this article used a variety of methodologies, including Random Forest, SVM, and Decision Tree, and determined that both high accuracy and efficiency can be attained.

[2] "**On building machine learning pipelines for Android malware detection: a procedural survey of practices, challenges and opportunities":** An innovative procedural taxonomy for ML-based Android malware detection was presented by the author in this article. He talked about the sources of malicious and benign APKs as well as the kinds of static and dynamic characteristics that researchers have gleaned from them. He also looked into how to get rid of the less useful aspects. It is possible to use and classify ML systems.

[3] "**Analysis of Android Malware Detection Techniques: A Systematic Review":** The author of this paper provided a comparative analysis of various Android mobile malware detection methods. Through a critical analysis, this research was able to identify all of the weaknesses and advantages of each detection method. The findings support the claim that detection techniques created for Android viruses do not always result in 100% accurate detection.

[4] **"Detection of Android Malware using Machine Learning and Deep Learning Review":** Based on the results of this study, experiments were conducted in this paper to select authorization and API-related information features for machine learning. According to the findings, evolutionary automated process feature selection was more advantageous than a common knowledge gain. The genetic algorithm still beats non-selection in terms of model generation time, even though its attribute selection performance was generally reduced by less than 3%.

[5] **"Android Malware Detection Using Machine Learning Classifiers":** The author suggests that category-based machine learning classifiers boost the effectiveness of the classification. Machine learning algorithms have been used to train classifiers with attributes of malicious apps and construct models that are capable of detecting dangerous patterns in the static analysis of Android malware. The author creates a profile of the set of features for the category of the top-rated applications in that category. To determine whether an app has benign characteristics, we compare its features to those that are required to provide the functionality of the category to which the app belongs.

[6] **"Android Mobile Malware Detection Using Machine Learning: A Systematic Review":** According to the author, classification accuracy is improved by using category-based machine learning models. In order to build models that can identify potentially hazardous patterns in the static analysis of Android malware, machine learning algorithms have been used to teach classifiers with characteristics of malicious apps. The author compiles a list of features for the most popular applications in each area. By contrasting an app's features with those needed to provide the utility of the

category to which it belongs, we can determine whether it has benign characteristics.

[7] **"An Android Malware Detection Leveraging Machine Learning":** The author looked into the effectiveness of four machine learning algorithms that try to identify malware based on permissions and action repetition, also known as static and dynamic features. He divided the undertaking into three phases. The findings show that classification precision was very good. The results also demonstrated a great level of accuracy. Therefore, for categorization, using static analyses alone should be effective and inexpensive.

[8] **"A Review of Android Malware Detection Approaches based on Machine Learning":** The study into Android malware detection is done in this paper. Authors used supervised learning techniques like Multinomial NB, Random Forest, and SVM. In addition to execution time, metrics like precision, recall, and F-measure are used to assess the outcomes. In comparison to other models, the SVM model performed better in terms of processing time.

[9] **"A survey of Android Malware Detection Technology Based on Machine Learning":** In this paper, the author used the vertical comparison technique to analyse the algorithm model, fundamental concepts, datasets, and performance metrics of the existing methods. In comparison, the machine learning-based static detection technique has advantages in accuracy and requires fewer detection experts.

[10]**"Machine -Learning based analysis and classification of Android malware signatures":** This study examined 259,608 malware signatures that were detected in a variety of Android apps. To allow cross-engine analysis, the signatures have been normalised into a common namespace using the Signature Miner tool. Then, malware signatures have been examined by grouping the families into three categories, according to the hazard and nature of each threat: Adware, Harmful, and Unknown.

3. EXISTING METHODOLOGY

After looking over the contributions made by different authors in the literature study, we discovered that malware detection in the current system is done using either a signature-based method or a heuristic method. The existing models' training is primarily based on simple classification algorithms, which makes it difficult to more accurately optimise the dataset. High false positive rates may result from some new malware signatures that are not yet in malware repositories failing to be recognised and as a result failing to identify the malware.

4. PROPOSED METHODOLOGY

The proposed approach determines whether an application is malware or not by classifying it using the ideal machine learning algorithm. In our undertaking, we use Grid search and SVC to fine-tune the model. We primarily concentrate on increasing the precision of the classification of the Android application in the provided dataset using Grid search.

A. Dataset Creation

We will first download the dataset of benign and malicious software. Adware, Ransomware, Scareware, PremiumSMS, SMS Malware, and Benign2015, Benign-2016, and Benign-2017 apks are all included in the collection. The downloaded dataset will then be unzipped, and malicious and benign folders will be created for the appropriate applications. Adware, Ransomware, Scareware, PremiumSMS, and SMS Malware are now included in the sources for malware, while Benign-2015, Benign-2016, and Benign2017 are included in the sources for innocuous software.

B. Data Pre-processing

The number of both malicious and positive applications will then be counted, and both counts will be returned. Then, we'll use the androguard tool to extract the rights and look only for those that begin with permission. The file "permissions.txt" contains all the permissions that have been extracted from all applications. We will now individually check each apk for rights. It is marked as "1" if the specific apk is using that authorization; otherwise, it is marked as "0". The CSV file labels these 0s and 1s with rights. The CSV's final entry falls under the benign or malignant category. A CSV file containing this preprocessed information is kept for use in future model training.

C. Training the model

Support Vector Machine (SVM) and artificial neural networks are used to develop the model (ANN). From the labeled examples, these models are trained to extract the required weights and biases. Using binary cross-entropy, the model is built, and only the epochs that generate better data than the previous epoch are chosen. Over 100 epochs have been used to teach the model. If there is no improvement in accuracy over a set number of epochs, the model's training will end immediately.

D. Evaluation of the model against the validation set and predicting the accuracy and loss.

The model's performance is measured after it has been compared to the validation and test datasets. The F1 value is computed. Finally, the accuracy and cost are calculated. The model is then preserved and transferred into apk file for front-end use with Flask.

E. Fine-tuning to find the best possible accuracy.

In this section, we attempt to modify the model in a few ways that will help us improve its overall accuracy. In SVM, we've used SVC with the Grid Search technique, which employs the best parameters to be specified in the kernel 'rbf' to fine-tune the model. If we were to run the model right now, the accuracy would have improved, increasing its efficiency.

D. Implementation of a front-end application to determine whether a particular apk contains malware or not.

After being preserved, the model is dropped into a front-end application. To submit the apk file for which the prediction is to be made, a user-friendly and interactive website using web programming languages like HTML, CSS, and Flask must be created. The user is then shown the forecast.

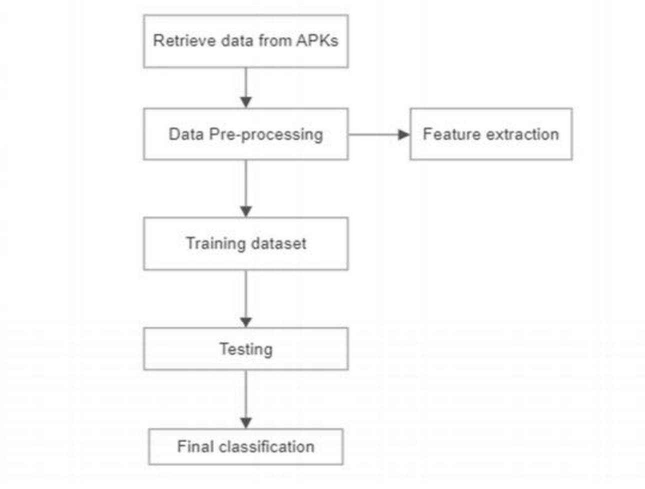

Fig - 1: Process Flow Diagram

5. CONCLUSIONS

On the Android platform, researchers have looked into the use of ML for automated malware detection. Building an intricate, multi-staged workflow is necessary for machine learning. Because of this, it has been challenging for new researchers to understand the state-of-the-art in this area. Over time, there has been a sharp rise in smartphone usage. Malware writers have used the significant increase in Android users to simultaneously target and harm users. This paper reviewed current frameworks and algorithms for machine learning-based malware detection. 100% accuracy with a 0% false alarm rate was the best. The identification rates were respectively 83% and 90%. To identify malware with high accuracy and detection rate, we need a good machine-learning algorithm. However, adding a couple of effective methods and algorithms will raise their efficiency, accuracy, and detection rates.

REFERENCES

[1]. Abikoye, Oluwakemi & Gyunka, Benjamin & OLUWATOBI, AKANDE. (2020). Android Malware Detection through Machine Learning Techniques: A Review. International Journal of Online and Biomedical Engineering (iJOE). 16. 14. 10.3991/ijoe.v16i02.11549.

[2]. Koushki, Masoud & Abualhaol, Ibrahim & Durai Raju, Anandharaju & Zhou, Yang & Giagone, Ronnie & Shengqiang, Huang. (2022). On building machine learning pipelines for Android malware detection: a procedural survey of practices, challenges and opportunities. Cybersecurity. 5. 16. 10.1186/s42400-022-00119-8.

[3]. Ashawa, Moses & Morris, Sarah. (2019). Analysis of Android Malware Detection Techniques: A Systematic Review. International Journal of Cyber-Security and Digital Forensics. 8. 177-187. 10.17781/P002605.

[4]. Joshi, Prof. (2022). Detection of Android Malware using Machine Learning and Deep Learning Review. International Journal of Recent Technology and Engineering (IJRTE). 11. 134-139. 10.35940/ijrte.A6963.0511122.

[5]. Ali, Huda & Oh, Tae & Fokoue, Ernest & Stackpole, Bill. (2016). Android Malware Detection Using Category-Based Machine Learning Classifiers. 54-59. 10.1145/2978192.2978218.

[6]. Senanayake, Janaka & Kalutarage, Harsha & Al-Kadri, M. Omar. (2021). Android Mobile Malware Detection Using Machine Learning: A Systematic Review. Electronics. 10. 1606. 10.3390/electronics10131606.

[7]. Shatnawi, Ahmed & Jaradat, Aya & Bani Yaseen, Tuqa & Taqieddin, Eyad & Al-Ayyoub, Mahmoud & Mustafa, Dheya. (2022). An Android Malware Detection Leveraging Machine Learning. Wireless Communications and Mobile Computing. 2022. 1-12. 10.1155/2022/1830201.

[8]. Liu, Kaijun & Xu, Shengwei & Xu, Guoai & Zhang, Miao & Sun, Dawei & Liu, Haifeng. (2020). A Review of Android Malware Detection Approaches Based on Machine Learning. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.3006143.

[9]. Wu, Qing, Xueling Zhu, and Bo Liu. "A survey of android malware static detection technology based on machine learning." *Mobile Information Systems* 2021 (2021): 1-18.

[10]. Martín, Nacho & Hernández, José & Santos, Sergio. (2019). Machine-Learning based analysis and classification of Android malware signatures. Future Generation Computer Systems. 97. 10.1016/j.future.2019.03.006.