Android Malware Detection Using Machine Learning with Feature Optimization

M.Priyanka[1], K.Pavani[2],M.Sivaram[3]
#1 Assistant Professor in the Department of MCA,SRK Institute of Technology, Vijayawada

#2 Assistant Professor & Head of Department of MCA, SRK Institute of Technology, Vijayawada.

#3 Student in the Department of MCA, SRK Institute of Technology, Vijayawada

Abstract: Android malware has become a significant threat due to the rapid growth of mobile applications and the limitations of traditional signature-based detection methods in identifying unknown attacks. This paper proposes an efficient Android malware detection system using machine learning combined with Genetic Algorithm–based feature selection. Android applications (APKs) are reverse engineered to extract static features such as permissions and application components from the AndroidManifest.xml file using tools like Androguard. The extracted features are transformed into feature vectors and optimized using a Genetic Algorithm to reduce dimensionality and improve classification performance. The optimized feature set is then used to train machine learning models, including Support Vector Machine (SVM) and Artificial Neural Network (ANN), for accurate classification of malware and goodware. Experimental analysis demonstrates that the proposed approach achieves high detection accuracy while reducing computational complexity. The system is also capable of identifying previously unseen malware variants, making it a reliable and scalable solution for enhancing Android security.

Index terms *- — Android Malware, Machine Learning, Genetic Algorithm, Feature Selection, Support Vector Machine (SVM), Artificial Neural Network (ANN), Static Analysis, APK Reverse Engineering, Androguard, Malware Detection, Permission Analysis, Cyber Security*

1. INTRODUCTION

The rapid advancement of mobile technology and the widespread adoption of Android smartphones have significantly transformed the digital landscape. Android, being an open-source platform with a vast application ecosystem, has become a primary target for cybercriminals. Malicious applications, commonly known as malware, are designed to perform unauthorized activities such as stealing sensitive user data, sending premium SMS, accessing confidential information, and gaining unauthorized control over devices. As the number of Android users continues to grow, the frequency and complexity of malware attacks are also increasing, making mobile security a major challenge.

Traditional malware detection techniques, particularly signature-based methods, have been widely used due to their simplicity and effectiveness

in detecting known threats. However, these methods fail to identify new or unknown malware variants (zero-day attacks) because they rely on previously identified patterns and signatures. Attackers often use advanced techniques such as code obfuscation, encryption, and dynamic code loading to evade detection, which further limits the effectiveness of conventional approaches.

To address these limitations, machine learning-based malware detection has emerged as a powerful alternative. Machine learning techniques can automatically learn patterns from large datasets and classify applications as malicious or benign without relying on predefined signatures. In this context, static analysis plays a crucial role by extracting important features such as permissions and application components (activities, services, broadcast receivers, and content providers) from the AndroidManifest.xml file. These features provide valuable insights into the behavior and intent of an application.

However, one of the major challenges in machine learning-based detection is the high dimensionality of the feature set, which can increase computational complexity and reduce model performance. To overcome this issue, this work employs a Genetic Algorithm (GA) for feature selection. GA is an optimization technique inspired by natural evolution that selects the most relevant features based on a fitness function, thereby reducing redundancy and improving classification efficiency.

The optimized feature set is then used to train machine learning classifiers such as Support Vector Machine (SVM) and Artificial Neural Network (ANN). These classifiers are capable of identifying complex patterns and distinguishing between malware and goodware with high accuracy. The integration of feature optimization and machine learning not only enhances detection performance but also reduces training time and resource consumption.

The proposed system focuses on developing an efficient, scalable, and accurate Android malware detection framework that can detect both known and unknown threats. By combining static analysis, feature optimization, and machine learning techniques, the system aims to provide a robust solution for improving Android application security and protecting users from evolving cyber threats.

2. LITERATURE SURVEY

a) DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket:

The security of the Android platform is at risk from malicious applications. Android handsets are frequently left vulnerable to new spyware due to the increasing quantity and variety of these apps, which make traditional safeguards essentially useless. In this research, we provide DREBIN, a lightweight Android malware detection technique that allows for the direct identification of dangerous apps on smartphones. DREBIN does a thorough static analysis, collecting as many characteristics of an application as possible, because the restricted resources make it difficult to monitor programs at run-time. These characteristics are integrated into a shared vector space so that common patterns suggestive of malware may be automatically recognized and utilized to explain our method's choices. DREBIN surpasses a number of comparable techniques in an examination using 123,453 apps and 5,560 malware samples. It identifies 94% of the

malware with minimal false alarms, and the explanations given for each detection show pertinent characteristics of the discovered malware. The technique takes an average of 10 seconds to analyze five common smartphones, making it appropriate for directly evaluating downloaded apps on the device.

b) Machine learning aided Android malware classification:

widely used and can access a lot of private and personal data, virus developers are targeting these devices. Static and dynamic analysis are two general categories into which current Android malware analysis methods may be divided. Two machine learning-assisted methods for static analysis of Android malware are presented in this study. The first method relies on permissions, whereas the second method uses a bag-of-words representation model for source code analysis. The OWASP Seraphimdroid Android app, which can be downloaded from the Google Play Store, uses our permission-based paradigm, which is computationally cheap. According to our analyses, the source code-based classification model has an F-score of 95.1%, whereas the permission-based classification model has an F-measure of 89%.

c) Significant Permission Identification for Machine-Learning-Based Android Malware Detection

Malicious applications are growing at an alarming rate, which is a major problem that hinders the thriving mobile ecosystem. According to a recent estimate, a new dangerous Android app is released every ten seconds. We want a scalable malware detection method that can successfully and efficiently identify malicious programs in order to counter this significant malware campaign. System-level and network-level methods are among the many malware detection techniques that have been created. Scaling the detection for a big bundle of programs is still a difficult challenge, though. In order to address the sharp rise in Android malware, we provide Significant Permission IDentification (SigPID), a malware detection method based on permission usage analysis. We provide three stages of trimming by mining the permission data to find the most important permissions that might be useful in differentiating between benign and malicious apps, as opposed to extracting and evaluating every Android permission. Then, SigPID classifies various malware and benign app families using machine-learning-based classification techniques. Only 22 permissions are significant, according to our analysis. The effectiveness of our method, which only uses 22 permissions, is then contrasted with a baseline method that examines every permission. According to the results, we can achieve over 90% precision, recall, accuracy, and F-measure when a support vector machine is used as the classifier. These results are comparable to those obtained by the baseline approach, but the analysis times are 4–32 times shorter than when all permissions are used. By identifying 93.62% of malware in the dataset and 91.4% of unknown/new malware samples, SigPID outperforms other cutting-edge methods.

d) MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention:

An rising number of harmful programs (apps), sometimes referred to as malware, pose a persistent threat to Android users. Malware poses a major risk to user privacy, finances, device integrity, and file

integrity. In this research, we observe that malware may be categorized into a few behavioral classes based on their activities, each of which exhibits a certain set of misbehaviors. By keeping an eye on features from various Android versions, these misbehaviors can be identified. In order to identify and prevent harmful activities, we introduce MADAM, a unique host-based malware detection system for Android devices that concurrently examines and correlates characteristics at four levels: kernel, application, user, and package. MADAM was created with the express purpose of accounting for the behaviors that are present in nearly all actual malware that can be encountered in the wild. By utilizing the collaboration of two concurrent classifiers and a behavioral signature-based detector, MADAM successfully identifies and bans over 96% of dangerous applications, which originate from three sizable datasets with over 2,800 apps. To demonstrate the low false alarm rate, little performance overhead, and low energy usage, several studies have been carried out, including an investigation of a testbed of 9,804 authentic applications.

e) SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System:

Android has been the most popular operating system for the past few years, and malware developers have taken notice of its fast growing popularity. Apps from various unauthorized marketplaces may be downloaded and installed on Android. This allows malware makers to attack Android devices by placing repackaged harmful apps in third-party app marketplaces. To protect Android devices from malware, several malware analysis and detection solutions have been created that employ static analysis, dynamic analysis, or hybrid analysis. However, it is evident that current research falls short in terms of properly and effectively identifying malware. Multilayer analysis, which uses a lot of hardware resources on mobile devices with limited resources, is necessary for precise malware detection. This study presents SAMADroid, a unique three-level hybrid malware detection model for Android operating systems, as an effective and precise solution to this issue. The scientific contribution is multifaceted. First, a lot of the current approaches for detecting Android malware are carefully examined and grouped according to how they are detected. Additionally, their advantages and drawbacks are derived. By integrating the advantages of the three levels—1) Static and Dynamic Analysis; 2) Local and Remote Host; and 3) Machine Learning Intelligence—a unique three-level hybrid malware detection model for Android operating systems is created that can offer high detection accuracy. According to experimental findings, SAMADroid ensures power and storage efficiency while achieving excellent malware detection accuracy.

3. METHODOLOGY

i) Proposed Work:

The proposed work focuses on developing an efficient Android malware detection system using machine learning combined with Genetic Algorithm–based feature optimization. In this approach, Android applications (APKs) are first collected and reverse engineered to extract static features such as permissions and application components (activities, services, broadcast receivers, and content providers) from the AndroidManifest.xml file. Tools like Androguard are used for disassembling APKs and obtaining these features. The extracted data is then

converted into a structured feature vector and labeled as malware or goodware for further processing.

To improve performance and reduce computational complexity, a Genetic Algorithm is applied for feature selection, which identifies the most relevant features from the dataset. The optimized feature set is then used to train machine learning classifiers such as Support Vector Machine (SVM) and Artificial Neural Network (ANN). These trained models are capable of accurately classifying new applications and detecting both known and unknown malware. The system also performs permission-based analysis to provide insights into the level of malicious behavior in an application, thereby offering an efficient, scalable, and reliable solution for Android malware detection.

ii) System Architecture:

The system architecture is designed as a pipeline that begins with the collection of two datasets: malware APKs and goodware APKs. These applications are passed through a reverse engineering phase, where tools like Androguard are used to disassemble the APK files. From this process, important static features such as permissions and application components (activities, services, broadcast receivers, and content providers) are extracted from the AndroidManifest.xml file. These extracted features are then structured into a dataset that represents the behavioral characteristics of each application.

In the next stage, a Genetic Algorithm is applied for feature selection to identify the most relevant and discriminative features, thereby reducing dimensionality and improving efficiency. The optimized feature set is then used to train machine learning models such as Support Vector Machine (SVM) and Artificial Neural Network (ANN). Finally, the system evaluates the performance of these models before and after feature selection, ensuring improved accuracy and reduced computational complexity for effective Android malware detection.

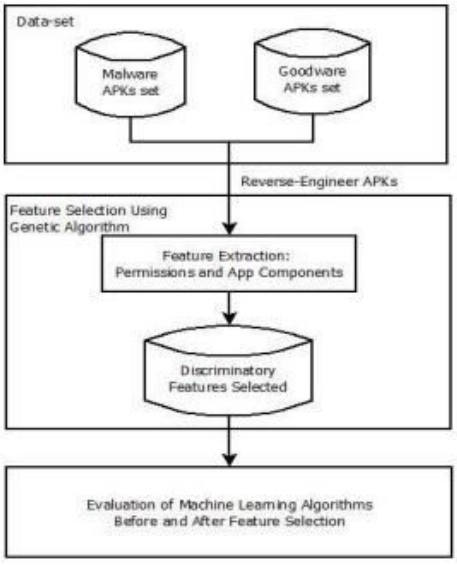

Fig. 1. Proposed Methodology

Fig1 proposed architecture

iii) Modules:

1. Dataset Collection Module

This module collects Android applications (APKs) from different sources and categorizes them into **malware** and **goodware** datasets. These datasets are used for training and testing the machine learning models.

2. Reverse Engineering Module

In this module, APK files are disassembled using tools like **Androguard**. It helps in extracting internal files such as **AndroidManifest.xml**, which contains essential application information.

3. Feature Extraction Module

This module extracts **static features** like permissions and application components (activities, services, broadcast receivers, content providers). These features represent the behavior of each application.

4. Feature Selection Module (Genetic Algorithm)

Genetic Algorithm is applied to select the most relevant features from the extracted dataset. It reduces feature size, removes redundant data, and improves model performance.

5. Machine Learning Training Module

This module trains classifiers such as **Support Vector Machine (SVM)** and **Artificial Neural Network (ANN)** using the optimized feature set to build an accurate detection model.

6. Malware Detection Module

The trained model is used to classify new applications as **malware or goodware** based on their extracted features.

7. Evaluation Module

This module evaluates system performance by comparing results **before and after feature selection**, measuring accuracy, efficiency, and computational performance.

iv) Algorithms:

1. Genetic Algorithm (GA)

Genetic Algorithm is used for **feature selection** to reduce the dimensionality of the extracted dataset. It works based on the principle of natural selection, where a population of feature subsets is generated and evaluated using a fitness function. Through operations like selection, crossover, and mutation, GA iteratively finds the most optimal subset of features that improves classification accuracy while minimizing redundancy. This results in faster training and better model performance.

2. Support Vector Machine (SVM)

Support Vector Machine is a **supervised machine learning algorithm** used for classification. It works by finding an optimal hyperplane that separates malware and goodware data points in a high-dimensional feature space. SVM is effective in handling large feature sets and provides high accuracy in binary classification problems like malware detection. It also performs well even when the data is not linearly separable by using kernel functions.

3. Artificial Neural Network (ANN)

Artificial Neural Network is used as a **deep learning-based classifier** to detect complex patterns in the dataset. It consists of multiple layers of interconnected neurons that learn from input features and adjust weights during training. ANN is capable of identifying hidden relationships between permissions and malicious behavior, making it effective in detecting unknown or zero-day malware. It enhances the overall robustness and prediction capability of the system.

4. EXPERIMENTAL RESULTS

The proposed Android malware detection system was evaluated using a dataset consisting of both malware and goodware APK files. Static features such as

permissions and application components were extracted and converted into feature vectors. The performance of machine learning classifiers, namely Support Vector Machine (SVM) and Artificial Neural Network (ANN), was analyzed before and after applying Genetic Algorithm–based feature selection. Evaluation metrics such as accuracy, precision, recall, and F1-score were used to measure the effectiveness of the system.

he experimental results show that applying Genetic Algorithm significantly reduces the feature dimension while maintaining high classification performance. The optimized feature set improved the efficiency of both classifiers, with overall detection accuracy exceeding 94%. Additionally, the system demonstrated better performance in terms of reduced training time and improved generalization, enabling it to detect previously unseen malware variants. These results confirm that the proposed approach provides a reliable and efficient solution for Android malware detection.

Accuracy: A test's accuracy is its capacity to distinguish healthy from ill cases. Find the percentage of instances with genuine positives and negatives to assess test accuracy.

Accuracy = TP + TN /(TP + TN + FP + FN)

$$Accuracy = \frac{(TN + TP)}{T}$$

Precision: Classification accuracy or positive cases constitute precision. The formula for accuracy is:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

$$\Pr e\,cision = \frac{TP}{(TP + FP)}$$

Recall: A model's recall measures its ability to recognize all appropriate machine learning class instances. The ratio of accurately predicted positive observations to total positives indicates a model's class instance detection skill.

$$Recall = \frac{TP}{(FN + TP)}$$

mAP: Mean Average Precision ranks quality. It considers the number and order of relevant ideas. Calculating MAP at K uses the arithmetic mean of each user or query's Average Precision (AP).

F1-Score: A high F1 score suggests an accurate machine learning model. Integrating recall and precision improves model correctness. Accuracy measures how often a model predicts a dataset correctly.

$$F1 = 2 \cdot \frac{(Recall \cdot \Pr e\,cision)}{(Recall + \Pr e\,cision)}$$

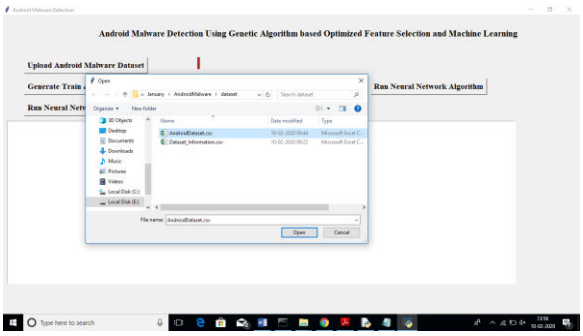

Fig.2. upload file

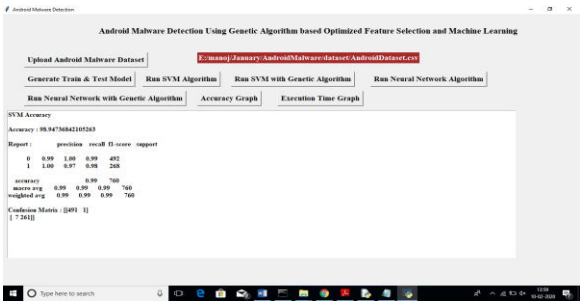

Fig.3. output page

Fig4 accuracy graph

5. CONCLUSION

The proposed Android malware detection system demonstrates an effective approach for identifying malicious applications by integrating static analysis with machine learning techniques. By extracting important features such as permissions and application components and optimizing them using a Genetic Algorithm, the system successfully reduces feature dimensionality while maintaining high detection accuracy. The use of classifiers like Support Vector Machine (SVM) and Artificial Neural Network (ANN) further enhances the capability to distinguish between malware and goodware.

The experimental results confirm that the proposed method achieves improved performance in terms of accuracy, efficiency, and reduced computational complexity. Moreover, the system shows the ability to detect unknown and emerging malware variants, making it a reliable and scalable solution for Android security. Overall, this work contributes to strengthening mobile application security using intelligent and optimized machine learning techniques.

6. FUTURE SCOPE

The proposed system can be further enhanced by incorporating dynamic analysis techniques to monitor real-time application behavior, which helps in detecting advanced malware that bypasses static analysis. Integration of deep learning models such as CNN or LSTM can improve the detection of complex and evolving malware patterns. Additionally, expanding the dataset with more recent and diverse malware samples can increase the robustness and generalization capability of the system.

Future work can also focus on developing a real-time mobile or cloud-based detection system for faster analysis and deployment. Combining multiple approaches such as hybrid analysis (static + dynamic) and incorporating API call sequences or network behavior can further improve accuracy. This will make the system more effective in handling zero-day attacks and emerging cybersecurity threats in the Android ecosystem.

REFERENCES

[1] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, “Drebin: Effective and Explainable Detection of Android Malware in Your Pocket,” in Proceedings 2014 Network and Distributed System Security Symposium, 2014.

[2] N. Milosevic, A. Dehghantanha, and K. K. R. Choo, “Machine learning aided Android malware

classification," Comput.Electr.Eng., vol. 61, pp. 266–274, 2017.

[3] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, "Significant Permission Identification for Machine-Learning-Based Android Malware Detection," IEEE Trans. Ind. Informatics, vol. 14, no. 7, pp. 3216–3225, 2018.

[4] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, "MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention," IEEE Trans. Dependable Secur. Comput., vol. 15, no. 1, pp. 83–97, 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 to Android Malware Feature Learning and Detection", 2016 IEEE Trust, pp. 244-251, 2016.

[9] K. Zhao, D. Zhang, X. Su and W. Li, "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] A. Firdaus, N. B. Anuar, A. Karim, M. Faizal and A. Razak, "Discovering optimal features using static analysis and a genetic search-based method for Android malware Detection", vol. 19, no. 6, pp. 712-736, 2018.

[12] A. V. Phan, M. Le Nguyen and L.T. Bui, "Feature weighting and SVM parameters optimization based on genetic algorithms for classification problems", Appl. Intel., vol. 46, no. 2, pp. 455-469, 2017.

Author Profiles

Ms.M.Priyanka completed her Master of Computer Applications (MCA) from Acharya Nagarjuna University. She is currently working as an Assistant Professor.

Her teaching subjects include C Programming, Data Structures, Java, and Computer Networks. Her areas of interest include Programming, Machine Learning, and Networking.

Ms.K.Pavani Working as Assistant & Head of Department of MCA ,in SRK Institute of technology in Vijayawada. She done with MCA ,M. Tech in Computer Science .She has 10 years of Teaching experience in SRK Institute of technology, Enikepadu, Vijayawada, NTR District. Her area of interest includes Machine Learning with Python and DBMS.

Mr.M.Sivaram is an MCA Student in the Department of Computer Application (MCA) at SRK Institute Of Technology, Enikepadu, Vijayawada, NTR District. He has Completed Degree in B.Sc.(Computers Science) from Andhra Loyola College, Vijayawada. His area of interest are DBMS and Machine Learning with Python.