Android Malware Detection Using Genetic Algorithm

Mr. Mukesh Gilda[1], Yeturu Sai Praneeth Reddy[2], Gurrala Shashi Kumar[3], Penmatsa Shivaram Sandeep Varma[4]

[1]*Asst. Professor, Sphoorthy Engineering College*

[2,3]*Dept of Computer Science and Engineering, Sphoorthy Engineering College*

[4]*Department of Cyber Security (CSE) Sphoorthy Engineering College, Hyderabad, India*

Abstract- This study presents an innovative approach for enhancing Android malware detection through a Genetic Algorithm (GA)-based optimized feature selection coupled with machine learning techniques. Leveraging the evolutionary principles of GA, the proposed method effectively identifies a subset of features from a large pool, maximizing the discriminative power while minimizing computational complexity. By integrating this feature selection mechanism with machine learning classifiers, the system achieves superior performance in distinguishing between, benign and malicious Android applications. Through extensive experimentation and evaluation using real-world datasets, the effectiveness of the proposed framework is demonstrated, showcasing significant improvements in detection accuracy, scalability, and efficiency compared to traditional methods. This research contributes to the advancement of Android security, offering a robust and adaptable solution for combating evolving malware threats in mobile ecosystems.

Keywords— Genetic Algorithm, Machine Learning, Android Malware, Feature Selection Mechanism, Accuracy.

1. INTRODUCTION

Android applications are widely available on Google Play Store and other platforms, but their open-source nature and popularity have made them a prime target for malware developers. Despite Google's efforts to protect users, malicious apps still manage to slip through and compromise personal information, such as contacts, emails, and GPS data, for nefarious purposes.

To combat this, malware analysis, which comes in two main forms- static and dynamic – is essential. Static analysis involves examining the code structure without execution, while of benign and malicious samples. This paradigm shift from static, rule-based detection to dynamic, data-driven approaches has significantly enhanced the efficacy and adaptability of Android malware detection systems.

A. The objective of this project is to develop a robust and efficient Android malware detection system utilizing state-of-the-art machine learning algorithms. By harnessing the power of ML, we aim to create a solution capable of accurately identifying malicious behavior in Android applications while minimizing false positives. This project not only contributes to the advancement of Android security research but also addresses the pressing need for innovative solutions to combat the escalating threat of Android malware.

B. In this introduction, we provide an overview of the challenges posed by Android malware, the limitations of existing detection methods, and the rationale behind leveraging machine learning for malware detection. Subsequently, we outline the goals and objectives of this project, highlighting its significance in the context of contemporary cybersecurity threats. Throughout the remainder of this document, we will delve deeper into the methodologies, experiments, and results that underpin our approach to Android malware detection using machine learning.

2. OBJECTIVE

The primary objective of this project is to develop an effective Android malware detection system using machine learning techniques. Specifically, the goals include:

1. Designing and implementing feature extraction mechanisms to capture relevant characteristics of Android applications.
2. Training machine learning models on labeled datasets of benign and malicious applications to learn patterns indicative of malware behavior.
3. Evaluating the performance of the detection system through rigorous testing and validation, focusing on accuracy, precision, recall, and false positive rates.

4. Enhancing the scalability and efficiency of the detection system to handle large volumes of Android applications in real-time.
5. Contributing to the advancement of Android security research by providing an innovative and adaptive approach to malware detection, capable of addressing evolving threats.

3. VARIOUS TECHINQUES TO INSERT MALWARE

Inserting malware into Android applications involves various techniques aimed at concealing malicious behavior while maintaining the functionality and appearance of legitimate apps. Some common techniques include:

1. Code Injection: Malicious code can be injected into legitimate Android applications, either statically or dynamically, to add malicious functionalities while retaining the original functionality of the app.
2. Steganography: Malware authors may embed malicious code within image or audio files, using steganography techniques to conceal the presence of the payload. Once the app is installed, the hidden code can be extracted and executed.
3. Dynamic Code Loading: Malware can dynamically load additional code from remote servers after installation. This allows attackers to modify the behavior of the app post-installation, making it challenging to detect malicious activities during static analysis.
4. Reflection: Malware can use Java reflection to dynamically invoke methods and access resources, enabling it to bypass static analysis checks that rely on straightforward code execution paths.
5. Encryption and Obfuscation: Malicious code can be encrypted or obfuscated to evade static analysis techniques. Decrypting and deobfuscating the code dynamically at runtime allows the malware to execute its malicious behavior without detection.
6. Exploiting Native Code: Malware authors may leverage native code libraries (e.g., using the Android Native Development Kit - NDK) to execute malicious code directly in native languages like C/C++, bypassing some of the security mechanisms enforced by the Android Runtime (ART) or Java Virtual Machine (JVM).
7. Packers and Protectors: Malware can be packed or protected using commercial or custom packers to obfuscate the code and make static analysis more difficult. These packers compress, encrypt, or modify the executable file, requiring unpacking or decryption at runtime.

8. Permission Abuse: Malicious apps may request excessive permissions during installation, exploiting users' trust in legitimate apps. This allows the malware to access sensitive data or perform unauthorized actions without raising suspicion.

4. LITERATURE REVIEW

1. Android Malware Detection Techniques:

A Review, by Sherly Elizabeth and Dr.P. Sheik Abdul Khader (International Journal of Computer Applications, 2015): This paper provides an extensive review of various techniques and methodologies employed for Android malware detection. It covers both traditional signature-based methods and emerging approaches based on machine learning and behavioural analysis.

2. A Survey of Machine Learning Techniques for Android Malware Detection by Arpita Gandhi and Hemant Patel (International Journal of Computer Applications, 2016):

This survey paper explores the application of machine learning techniques for Android malware detection. It categorizes different ML algorithms used in existing research and evaluates their effectiveness in detecting Android malware based on features extracted from applications.

3. Android Malware Detection Using Machine Learning Algorithms:

A Review, by S. Sharmeela and Dr. P. Sheik Abdul Khader (International Journal of Computer Applications, 2016) : Focusing specifically on machine learning-based approaches, this review paper examines the use of various ML algorithms for Android malware detection. It discusses the strengths and limitations of different algorithms and provides insights into future research directions.

4. A Review of Machine Learning Approaches to Android Malware Detection"** by Shashikant Patil and Dr. Suryakanth Biradar (International Journal of Engineering Research and Applications, 2017):

This paper provides a comprehensive review of machine learning techniques applied to Android malware detection. It discusses feature selection methods, dataset

construction, and the performance evaluation of ML models in detecting Android malware.

5. A Survey of Machine Learning Techniques for Android Malware Detection"** by Amir R. Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani (IEEE Communications Surveys & Tutorials, 2018):

This survey article offers an in-depth analysis of machine learning approaches for Android malware detection. It covers various aspects such as feature selection, dataset construction, and the comparative evaluation of different ML algorithms.

6. Machine Learning-Based Android Malware Detection:

A Survey by Salih Abdulrahman Saleh, Muneera Alshammari, and Mohammed Khalid (IEEE Access, 2019): Focusing on recent advancements, this survey paper provides insights into the application of machine learning for Android malware detection. It discusses the challenges posed by evolving malware threats and explores potential solutions offered by ML techniques.

5. EXISTING SYSTEMS

The primary achievement of this research is the significant reduction of feature dimensions to less than half of the original set using Genetic Algorithms. This reduction allows for inputting into machine learning classifiers, simplifying the training process while retaining accuracy in classifying malware. Unlike exhaustive methods, which require testing numerous combinations, Genetic Algorithms employ a heuristic approach based on fitness functions for feature selection. The optimized feature set obtained through this method is then utilized to train Support Vector Machine and Neural Network algorithms. Remarkably, a classification accuracy exceeding 79% is maintained despite the reduced feature dimensionality, effectively lowering the training complexity of the classifiers.

6. PROPOSED SYSTEM

The proposed system aims to enhance Android malware detection using machine learning techniques, specifically Support Vector Machines (SVM). The primary focus is on improving the accuracy and efficiency of malware detection by utilizing a more data-driven and adaptive approach.

7. SYSTEM ARCHITECHTURE

A web server has been developed to handle various functions related to Android malware detection. The server is designed to accept all information related to malware detection, accessing and processing data from a web database. Users can log in to the system, where service providers can train and test data sets, and view the accuracy of their trained and tested models through bar charts and detailed results. Additionally, users can view predicted Android malware detection details, including detection ratios, and download predicted datasets for further analysis. The system allows remote users to register, log in, predict Android malware types, and view their profiles, ensuring a comprehensive and user-friendly experience for all.

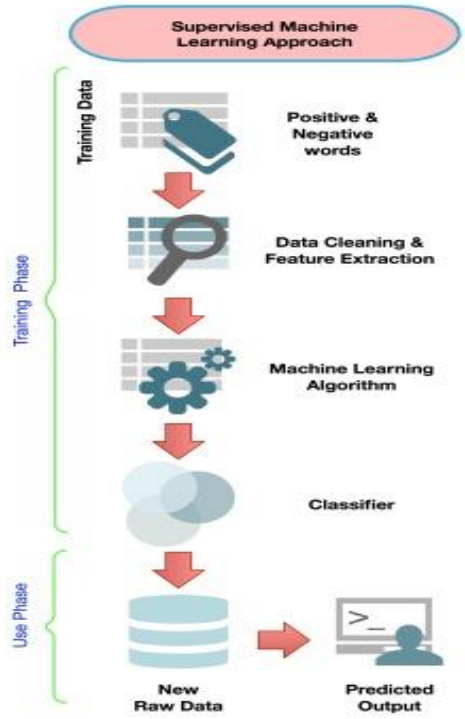


Figure 1: Architecture diagram

8. USE CASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

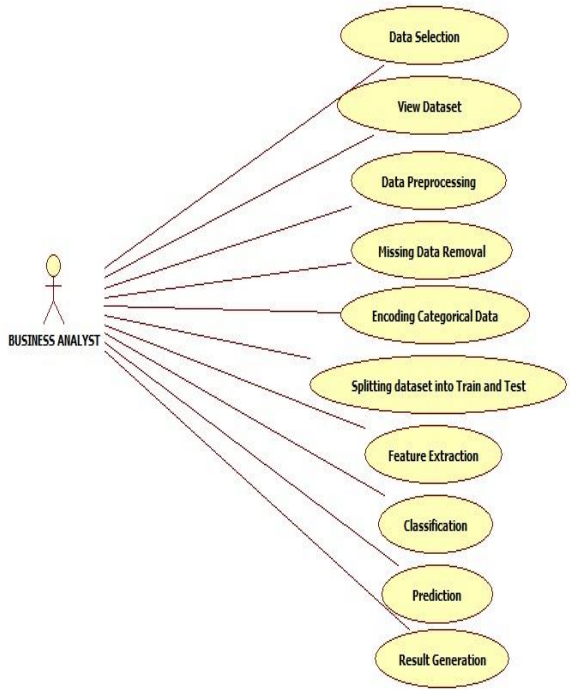


Figure 2: Use Case Diagram

9. CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

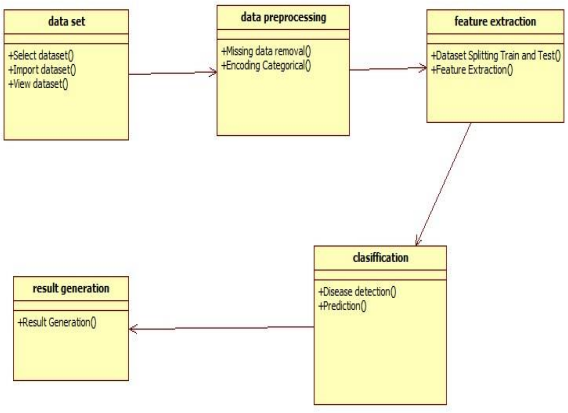
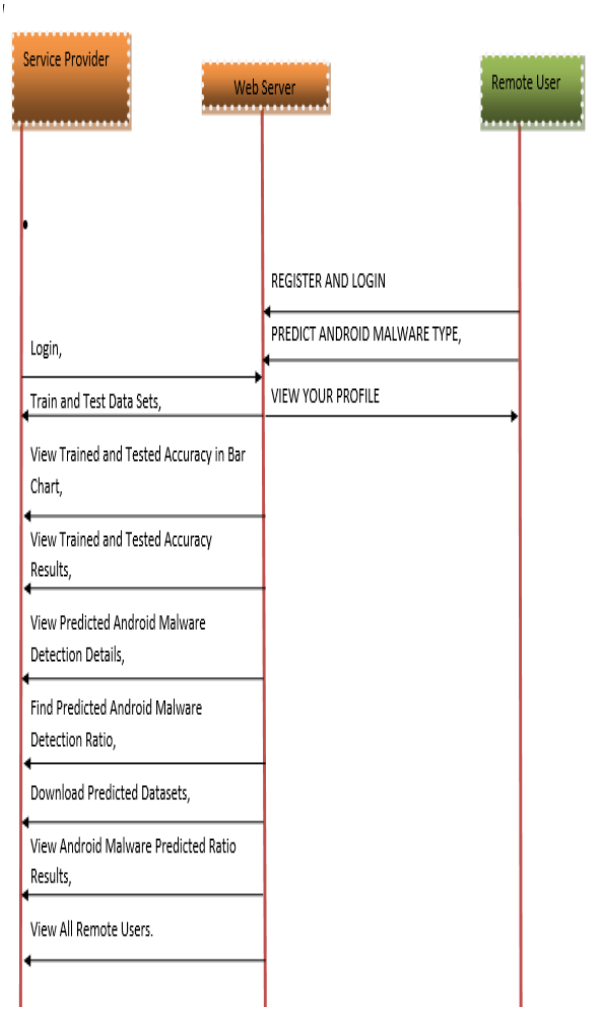


Figure 3. Class Diagram

10 . SEQUENCE DIAGRAM

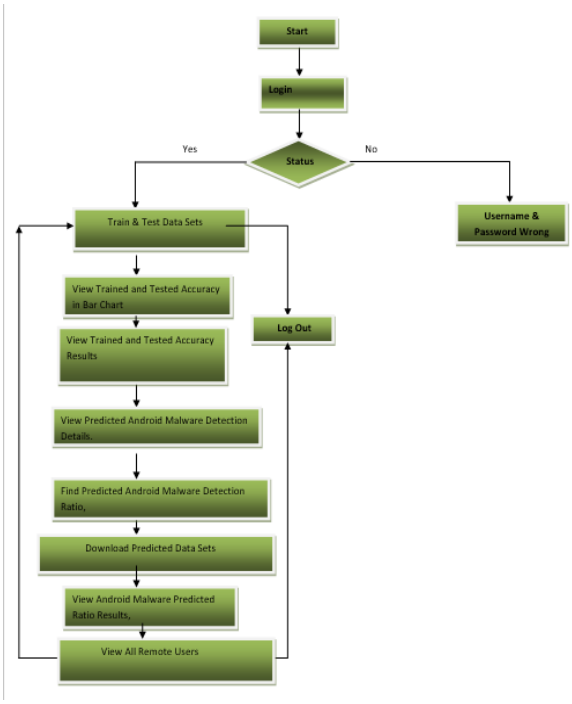


11. METHODOLOGY

Modules

a. Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Train and Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Predicted Android Malware Detection Details, Find Predicted Android Malware Detection Ratio, Download Predicted Datasets, View Android Malware Predicted Ratio Results, View All Remote Users.



View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorize the users.

Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT ANDROID MALWARE TYPE, VIEW YOUR PROFILE.

12. EVALUATION

a) Algorithms: -

i. Decision Tree: -

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision-making knowledge from the supplied data. Decision tree can be generated from training sets. The procedure for such generation based on the set of objects (S), each belonging to one of the classes C1, C2, …, Ck is as follows:

Step 1. If all the objects in S belong to the same class, for example Ci, the decision tree for S consists of a leaf labelled with this class

Step 2. Otherwise, let T be some test with possible outcomes O1, O2, On. Each object in S has one outcome for T so the test partitions S into subsets S1, S2, Sn where each object in Si has outcome Oi for T. T becomes the root of the decision tree and for each outcome Oi, we build a subsidiary decision tree by invoking the same procedure recursively on the set Si.

ii. SVM: -

In classification tasks a discriminant machine learning technique aims at finding, based on an *independent and identically distributed* (*iid*) training dataset, a discriminant function that can correctly predict labels for newly acquired instances. Unlike generative machine learning approaches, which require computations of conditional probability distributions, a discriminant classification function takes a data point *x* and assigns it to one of the different classes that are a part of the classification task. Less powerful than generative approaches, which are mostly used when prediction involves outlier detection, discriminant approaches require fewer computational resources and less training data, especially for a multidimensional feature space and when only posterior probabilities are needed. From a geometric perspective, learning a classifier is equivalent to finding the equation for a multidimensional surface that best separates the different classes in the feature space.

SVM is a discriminant technique, and, because it solves the convex optimization problem analytically, it always returns the same optimal hyperplane parameter—in contrast to *genetic algorithms* (*GAs*) or *perceptron's*, both of which are widely used for classification in machine learning. For perceptron's, solutions are highly dependent on the initialization and termination criteria. For a specific kernel that transforms the data from the input space to the feature space, training returns uniquely defined SVM model parameters for a given training set, whereas the perceptron and GA classifier models are different each time training is initialized. The aim of GAs and perceptron's is only to minimize error during training, which will translate into several hyperplanes' meeting this requirement.

iii. Logistic Regression Classifiers: -

Logistic regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name *logistic regression* is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name *multinomial logistic regression* is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divorced, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar.

Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modeling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does.

This program computes binary logistic regression and multinomial logistic regression on both numeric and categorical independent variables. It reports on the regression equation as well as the goodness of fit, odds ratios, confidence limits, likelihood, and deviance. It performs a comprehensive residual analysis including diagnostic residual reports and plots. It can perform an independent variable subset selection search, looking for the best regression model with the fewest independent variables. It provides confidence intervals on predicted values and provides ROC curves to help determine the best cutoff point for classification. It allows you to validate your results by automatically classifying rows that are not used during the analysis.

*12.*RESULT

The overall outcomes of the Malware detection system on Android platform using Genetic Algorithm method with other models. The outcomes identified that the NB approach has poor performance, whereas the AdaBoostM1 model gains slightly enhanced results. Along with that, the Machine Learning models accomplish moderately closer performance. However, the AAMD-OELAC technique offers better results with increased accu_y of 98.93%, prec_n of 99.15%, reca_l of 98.93%, and F_score of 99.04%. Finally, the computational time (CT) analysis of the Malware detection system on Android platform using Genetic Algorithm technique is compared with recent approaches in Table 4. The outcomes exhibited that the Malware detection system on Android platform using Genetic Algorithm technique reaches the least CT value of 8s. At the same time, the existing models have reached increased CT values. These results highlighted that the Malware detection system on Android platform using Genetic Algorithm technique shows maximum performance over other models on malware classification.

13. REFERENCE

[1] H. Rathore, A. Nandanwar, S. K. Sahay, and M. Sewak, ‘‘Adversarial superiority in Android malware detection: Lessons from reinforcement learning based evasion attacks and defenses,’’ *Forensic Sci. Int., Digit. Invest.*, vol. 44, Mar. 2023, Art. no. 301511.

[2] H. Wang, W. Zhang, and H. He, ‘‘You are what the permissions told me! Android malware detection based on hybrid tactics,’’ *J. Inf. Secur. Appl.*, vol. 66, May 2022, Art. no. 103159.

[3] A. Taha and O. Barukab, ‘‘Android malware classification using optimized ensemble learning based on genetic algorithms,’’ *Sustainability*, vol. 14, no. 21, p. 14406, Nov. 2022.

[4] O. N. Elayan and A. M. Mustafa, ‘‘Android malware detection using deep learning,’’ *Proc. Comput. Sci.*, vol. 184, pp. 847–852, Jan. 2021.

[5] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, ‘‘MAPAS: A practical deep learning-based Android malware detection system,’’ *Int. J. Inf. Secur.* Vol. 21, no. 4, pp. 725–738, Aug. 2022.