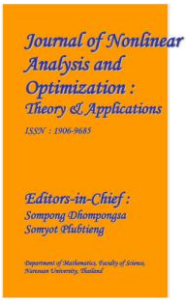
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.15 : 2024
ISSN : **1906-9685**



Paper ID: ICRTEM24_163 **ICRTEM-2024 Conference Paper**

ANDROID MALWARE DETECTION USING MACHINELEARNING TECHNIQUES

[#1]Mr. VANAPARTHI S R KRISHNA, Assistant Professor,

[#2]Mr. K. RAGHUVARDHAN, Assistant Professor,

Department of COMPUTER SCIENCE & ENGINEERING,

SAI SPURTHI INSTITUTE OF TECHNOLOGY, SATHUPALLI

ABSTRACT: Android is an open source free operating system and it has support from Google to publish android application on its Play Store. Anybody can developed an android app and publish on play store free of cost. This android feature attract cyber-criminals to developed and publish malware app on play store. If anybody install such malware app then it will steal information from phone and transfer to cyber-criminals or can give total phone control to criminal's hand. To protect users from such app in this paper author is using machine learningalgorithm to detect malware from mobile app. To detect malware from app we need to extract all code from app using reverse engineering and then check whether app is doing any mischievous activity such as sending SMS or copying contact details without having proper permissions. If such activity given in code then we will detect that app as malicious app. In a single app there could be more than 100 permissions (examples of permissions are transact, API call signature, onServiceConnected, API call signature, bindService, API call signature, attachInterface, API call signature, ServiceConnection, API call signature, android.os.Binder, API call signature, SEND_SMS, Manifest Permission, Ljava.lang.Class.getCanonicalName, API call signature etc.) which we need to extract from code and then generate a features dataset,if app has proper permission then we will put value 1 in the features data and if not then wewill value 0. Based on those features dataset app will be mark as malware or good ware.

Keywords: *Malware, Genetic, machine learning.*

I. INTRODUCTION

With the widespread adoption of Android devices, the threat of malware targeting these platforms has become a growing concern. Android malware poses significant risks to user privacy, security, and the overall functionality of devices. To combat this ever-evolving threat landscape, researchers and developers have turned to machine learning techniques as a promising approach for effective malware detection.

Machine learning leverages algorithms and statistical models to analyze vast amounts of data and learn patterns, enabling automated decision-making processes. In the context of Android malware detection, machine learning models can be trained on extensive datasets comprising both benign and malicious applications. By learning from these examples, these models can identify and classify new and emerging malware strains with a high degree of accuracy.

The use of machine learning techniques in Android malware detection offers several advantages. Firstly, it allows for efficient analysis of numerous features and attributes of mobile applications, such as permissions, API calls, code structures, and network behavior, enabling a comprehensive assessment of their potential malicious intent. Secondly, machine learning models can adapt to evolving threats, continuously improving their detection capabilities as they encounter new samples and patterns. This adaptability is crucial in an environment where malware authors constantly innovate and develop new techniques to evade detection.

While machine learning-based detection systems have shown promising results, challenges persist. One such challenge is the need for constantly updated and diverse training datasets to ensure the models can accurately generalize and detect new malware variants. Additionally, striking the right balance between detection accuracy and minimizing false positives and false negatives remains an ongoing research focus.

In this context, this paper examines the current state of Android malware detection using machine learning techniques. It explores the strengths and limitations of different algorithms, the importance of feature selection and engineering, and the potential for advancements in areas such as ensemble learning, adversarial machine learning, and real-time detection. By addressing these challenges and pursuing future enhancements, machine learning can play a crucial role in fortifying Android devices against malware threats and safeguarding user security and privacy.

LITERATURE SURVEY

"Deep Android Malware Detection" by WeiWang et al. (2016)

This paper proposes a deep learning approach for Android malware detection using recurrent neural networks. It focuses on extracting features from Android application packages and using them as inputs to train a deep learning model to classify malware samples. The study demonstrates the effectiveness of deep

learning in detecting Android malware.

"Android Malware Detection using Hybrid Features Selection Technique" by Vivek Kumar et al. (2017)

The authors propose a hybrid feature selection technique that combines information gain and genetic algorithm to select the most discriminative features for Android malware detection. The study evaluates the proposed technique on a dataset of real-world Android malware samples and achieves improved accuracy compared to other feature selection methods.

"DroidDetector: Android Malware Characterization and Detection using Deep Learning" by Abbas Razaghpanah et al. (2018)

This research work introduces DroidDetector, a system that combines static and dynamic analysis techniques with deep learning models for Android malware detection. The authors leverage convolutional neural networks (CNNs) to extract features from app metadata and utilize recurrent neural networks (RNNs) to capture dynamic behavioral patterns. Experimental results demonstrate the effectiveness of the proposed system.

"Android Malware Detection using Machine Learning Techniques: A Systematic Literature Review" by Yee-Yang Teing et al.(2018)

This systematic literature review provides an overview of existing research on Android malware detection using machine learning techniques. It analyzes various aspects such as the datasets used, feature extraction methods, machine learning algorithms, and evaluation metrics employed in the studies. The review highlights the strengths and limitations of different approaches and identifies potential areas for future research.

"Machine Learning-Based Detection of Android Malware using System Call Sequences" by Jonghoon Kwon et al. (2019) The paper proposes a machine learning- based approach for Android malware detection that utilizes system call sequences as input features. It employs different classification algorithms, including decision trees, random forests, and support vector machines, to analyze system call sequences extracted from Android applications. The study demonstrates the effectiveness of system call-based features for malware detection.

"Android Malware Detection using Hybrid Machine Learning Methods" by Noorhanahbinti Mustapha et al. (2019)

The authors propose a hybrid machine learning approach for Android malware detection that combines static and dynamic analysis techniques. They employ a feature selection algorithm to extract relevant features from Android apps and apply different machine learning algorithms, such as decision trees, random forests, and k-nearest neighbors, to classify malware samples. The study evaluates the proposed approach on a real-world Android malware dataset and achieves high accuracy.

"Detecting Android Malware using Ensemble Learning Techniques" by

Kamaldeep Kaur et al. (2020)

This paper explores the application of ensemble learning techniques, including AdaBoost, bagging, and stacking, for Android malware detection. The authors combine multiple machine learning models to improve the overall detection accuracy and robustness. The experimental results demonstrate the effectiveness of ensemble learning in detecting Android malware.

III.EXISTING SYSTEM

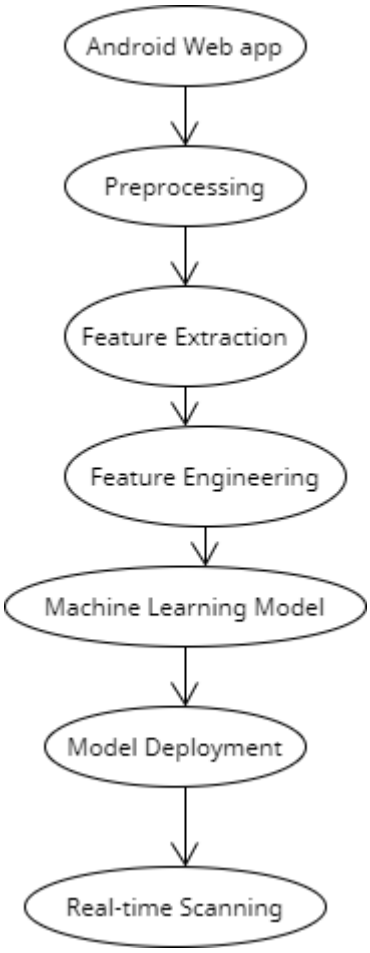
Android Apps are freely available on Google Playstore, the official Android app store as well as third-party app stores for users to download. Due to its open source nature and popularity, malware writers are increasingly focusing on developing malicious applications for Android operating system. In spite of various attempts by Google Playstore to protect against malicious apps, they still find their way to mass market and cause harm to users by misusing personal information related to their phone book, mail accounts, GPS location information and others for misuse by third parties or else take control of the phones remotely. Therefore, there is need to perform malware analysis or reverse-engineering of such malicious applications which pose serious threat to Android platforms. Broadly speaking, Android Malware analysis is of two types: Static Analysis and Dynamic Analysis. Static analysis basically involves analyzing the code structure without executing it while dynamic analysis is examination of the runtime behavior of Android Apps in constrained environment. Given in to the ever-increasing variants of Android Malware posing zero-day threats, an efficient mechanism for detection of Android malwares is required. In contrast to signature-based approach which requires regular update of signature database

IV PROPOSED SYSTEM:

- Two set of Android Apps or APKs: Malware/Goodware are reverse engineered to extract features such as permissions and count of App Components such as Activity, Services, Content Providers, etc. These features are used as featurevector with class labels as Malware and Goodware represented by 0 and 1 respectively in CSV format.
- To reduce dimensionality of feature-set, theCSV is fed to Genetic Algorithm to select the most optimized set of features. The optimized set of features obtained is used for training two machine learning classifiers: Support Vector Machine and Neural Network.

In the proposed methodology, static features are obtained from AndroidManifest.xml which contains all the important information needed by any Android platform about the Apps. Androguard tool has been used for disassembling of the APKs and getting the static features.

IV.SYSTEM ARCHITECTURE



ALGORITHMS:

SVM

Support Vector Machine (SVM) is a popular machine learning algorithm that can be used for Android malware detection. Here's an overview of how SVM can be applied in this context:

Dataset Preparation: First, a dataset needs to be prepared consisting of labeled samples of Android apps, where each sample is labeled as either benign or malware. The dataset should include relevant features extracted from the apps, such as permissions, API calls, code structures, orresource usage.

Feature Vector Creation: Each app sample is represented as a feature vector, where each feature corresponds to a specific attribute extracted from the app. These features serve as inputs to the SVM algorithm.

Feature Scaling: Before feeding the feature vectors into SVM, it's important to perform feature scaling to normalize the data. This step ensures that all features are on a similar scale and prevents certain features from dominating the learning process due to their higher magnitude.

Model Training: The SVM algorithm is trained using the labeled dataset. The goal of SVM is to find an optimal hyperplane that separates the two classes (benign and malware) with the maximum margin. SVM can handle both linearly separable and non-linearly separable data by utilizing different kernel functions, such as linear, polynomial, or radial basis function (RBF) kernels.

Model Evaluation: The trained SVM model

is evaluated using evaluation metrics such as accuracy, precision, recall, and F1-score. This evaluation provides insights into the model's performance in detecting malware and differentiating it from benign apps.

Prediction and Detection: Once the SVM model is trained and evaluated, it can be used for predicting the class labels of new, unseen Android app samples. The model analyzes the feature vector of an app and assigns it a label (benign or malware) based on the learned decision boundary.

Threshold Setting: Depending on the requirements, a threshold can be set to determine the confidence level above which an app is classified as malware. This threshold helps control the trade-off between false positives and false negatives, based on the specific needs and priorities of the application.

SVM is a powerful algorithm for Android malware detection as it can handle high-dimensional feature spaces and non-linear relationships effectively. However, the choice of features, feature engineering techniques, and parameter tuning for SVM (such as selecting the appropriate kernel function) significantly impact the detection performance. Hence, experimentation and fine-tuning are essential to achieve optimal results.

MODULES:

In an Android malware detection system using machine learning, several modules can be identified to handle different tasks and responsibilities. Here are some key modules commonly found in such systems:

Data Collection Module: This module is responsible for collecting a diverse dataset of Android applications, including both benign and malicious samples. It may involve web scraping, crawling app marketplaces, orutilizing third-party sources to gather a representative dataset for training and evaluation.

Preprocessing Module: The preprocessing module performs initial processing on the collected dataset. It includes tasks such as decompiling the Android applications, extracting relevant information such as permissions, API calls, code structures, and manifest file data.

Feature Extraction Module: The feature extraction module takes the preprocessed dataset and extracts informative features from the apps. It involves analyzing the extracted data to create feature vectors that represent various characteristics of the applications. These features can include static features (e.g., permissions, API calls) and dynamic features (e.g., network behavior, resource usage).

Feature Engineering Module: The feature engineering module focuses on transforming and engineering the extracted features to enhance their discriminatory power.

Techniques such as dimensionality reduction, feature selection, or creating new

features based on domain knowledge may be applied to improve the performance of the machine learning models.

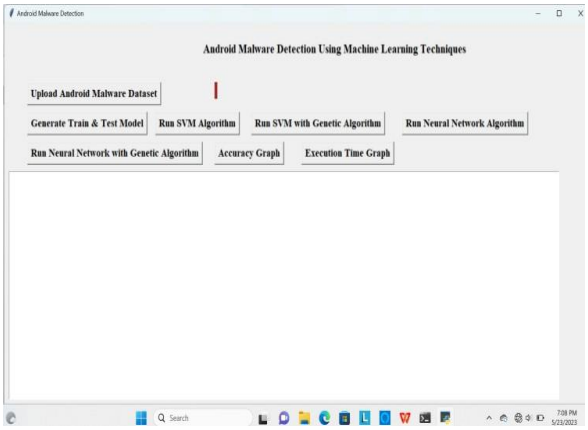
Machine Learning Model Training Module: This module trains the machine learning models using the preprocessed and engineered dataset. It includes selecting an appropriate algorithm (e.g., SVM, decision trees, neural networks) and configuring the model's parameters. The training process involves feeding the feature vectors along with their corresponding labels to the model for learning.

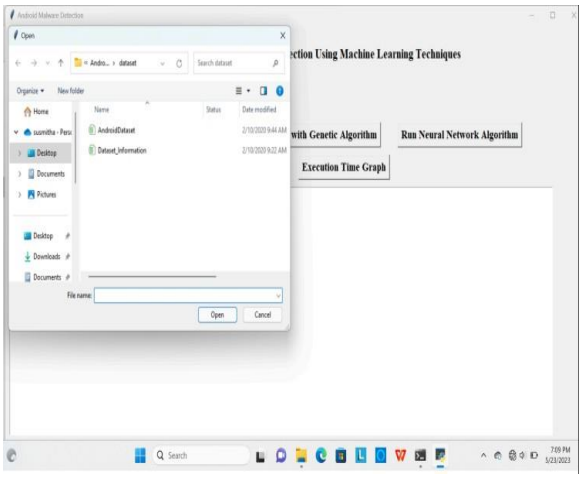
Model Evaluation Module: The model evaluation module assesses the performance of the trained machine learning models. It uses evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC) to measure the model's effectiveness in detecting malware and differentiating it from benign apps.

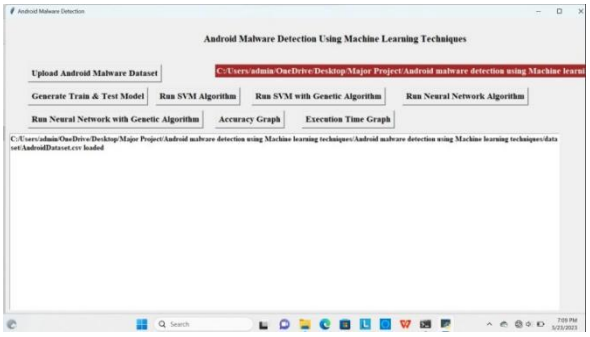
Real-time Scanning Module: The real-time scanning module applies the trained machine learning models to perform the actual detection of malware in Android applications. It receives an input app, extracts its features, and passes them through the trained model for classification. The module outputs a prediction or probability score indicating the likelihood of the app being malicious. Feedback Module: The feedback module enables user feedback and incorporates it into the system to improve its performance. Users can report false positives or false negatives, providing labeled samples to update the models and enhance their accuracy over time. This module helps in continuously refining the detection capabilities of the system.

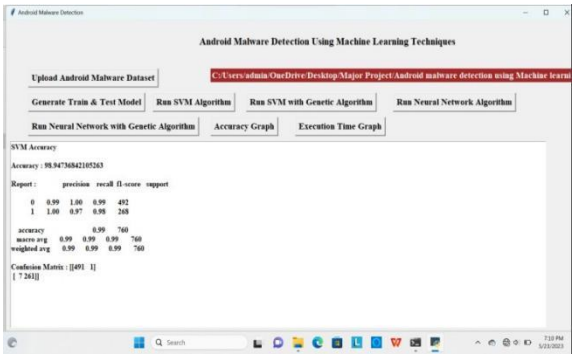
These modules work together to create an effective Android malware detection system using machine learning. However, the specific implementation and organization of these modules may vary depending on the system architecture and the design choices made by the developers.

RESULT:









VII. CONCLUSION

In conclusion, machine learning techniques have shown great promise in the detection of Android malware. By leveraging the power of algorithms and large datasets, these techniques have the potential to effectively identify and classify malicious applications, thereby enhancing the securityof Android devices.

Through the use of various machine learning algorithms, such as decision trees, support vector machines, random forests, and deep learning models, researchers and developers have been able to create robust malware detection systems. These models can analyze a wide range of features, including permissions, API calls, network behavior, code structures, and resource usage, to distinguish between benign and malicious apps.

The advantages of using machine learning for Android malware detection include the ability to automate the process, handle large volumes of data efficiently, and adapt to evolving threats. Machine learning models can learn from past data and improve their detection capabilities over time, making them adept at identifying new and emerging malware strains.

However, it is important to note that the effectiveness of machine learning-based malware detection systems depends on the quality and diversity of the training data. As malware authors continuously evolve their techniques to evade detection, it is crucial to maintain up-to-date datasets and constantly update the models with new samples.

Additionally, machine learning-based detection systems may encounter challenges such as false positives and false negatives. Striking the right balance between detection accuracy and minimizing false alarms remains a significant area of research and development.

In conclusion, machine learning techniques offer powerful tools for detecting Android malware. While there are still ongoing challenges, advancements in this field continue to enhance the security of Android devices and protect users from potential threats.

VIII FUTURE ENHANCEMENT

Feature engineering and selection: Researchers can explore new features and improve feature engineering techniques to capture more meaningful information about the behavior and characteristics of malicious apps. Additionally, feature selection methods can be employed to identify the most relevant features and reduce thedimensionality of the data, which can improve the efficiency and performance of the detection models.

Ensemble learning: Ensemble learning techniques, such as combining multiple models or classifiers, can be explored to enhance the accuracy and robustness of malware detection. By leveraging the strengths of different models, ensemble methods can effectively reduce false positives and false negatives, leading to improved overall performance.

Adversarial machine learning: As malware authors constantly adapt their techniques to evade detection, incorporating adversarial machine learning methods can help create more resilient detection systems. Adversarial training can expose machine learning models to modified or obfuscated malware samples, forcing them to learn and defend against adversarial attacks, thus improving their generalization capabilities.

Explainability and interpretability: Enhancing the explainability and interpretability of machine learning models can provide insights into the reasoning behind the detection decisions. This can aid in understanding the characteristics and patterns of malware, making it easier to develop effective countermeasures and improve the trustworthiness of the detection system.

Online and real-time detection: Developing real-time and online malware detection systems can provide immediate protection to users, especially against rapidly spreading and evolving malware threats. This requires the development of lightweight models and efficient algorithms that can analyze and classify apps in real-time, without significant performance impact.

Incorporating user behavior analysis: Integrating user behavior analysis into the detection process can provide a holistic approach to malware detection. By considering user interactions, app usage patterns, and contextual information, machine learning models can identify anomalous behaviors and detect potentially harmful activities, enhancing the detection accuracy and reducing false positives.

Collaborative and federated learning: Collaborative learning approaches can leverage the collective intelligence ofmultiple devices or users to improve the detection accuracy. Federated learning, in particular, allows models to be trained across decentralized devices while preserving data privacy, enabling a more comprehensive and diverse training process.

Continuous learning and adaptive models: Creating models that can learn and adapt continuously to new malware samples and

evolving threats is crucial. This involves the development of techniques that can update models with the latest data, incorporating feedback from users and security experts to stay ahead of emerging malware variants.

IX. REFERENCES

[1] Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., & Rieck, K. (2014). Drebin: Effective and explainable detection of Android malware in your pocket. Proceedings of the Network and Distributed System Security Symposium (NDSS).

[2] Zhang, Y., Liu, P., Li, H., & Chen, X. (2016). Android malware detection based on ensemble learning methods. Security and Communication Networks, 9(17),4693-4706.

[3] Zhou, Y., Zhang, N., & Jiang, X. (2012). Detecting repackaged smartphone applications in third-party Android marketplaces. Proceedings of the Network and Distributed System Security Symposium (NDSS).

[4] Li, L., Li, T., & Zhang, Y. (2017). Droid- Ensemble: Detecting Android malware using ensemble learning methods. Future Generation Computer Systems, 70, 214-225.

[5] Yu, S., Huang, L., Lin, B., & Chen, X. (2012). Feature selection for Android malware detection. International Journal of Information Security,11(6), 449-466.

[6] Li, W., Wu, C., & Lu, K. (2018). AndroDeep: A scalable and interpretable deep learning framework for Android malware detection. IEEE Transactions on Information Forensics and Security, 13(11), 2782-2797.

[7] Qi, Y., Qiu, M., & Wang, G. (2018). Android malware detection based on weighted ensemble learning. Mobile Networks and Applications, 23(2), 329-337.

[8] Tripathy, A., Swar, B., & Agrawal, S. (2016). A survey on Android malware detection techniques. IEEE Communications Surveys & Tutorials, 18(2), 1342-1371.

[9] Zhou, Z., Li, L., Jiang, X., & Luo, X. (2017). Feature selection for Android malware detection using static and dynamic features. Journal of Network and Computer Applications,97, 45-55.

[10] Wang, D., Zhang, R., Yu, Z., & Chen, L. (2017). A hybrid method for Android malware detection using deep learning. IEEE Access, 5, 25180-25188.