IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 6, May 2024

Android Malware Detection using Opcode

Prof. S. S. Raskar[1], Aakash Kumar[2], Arshit Kumar[3], Harshad Jagtap[4], Ketan Chavan[5]
Professor, Department of Computer Engineering[1]
Students, Department of Computer Engineering[2,3,4,5]
Modern Education Society's Wadia College of Engineering, Pune, India

Abstract: *A permission-based system that restricts third-party Android applications' access to essential resources on an Android device is the foundation of Android application security. Before the installation, the user must consent to the set of rights that the programme requests. This procedure attempts to warn users of the risks involved in installing and using an application on their device; however, in the majority of cases, even in cases where the permission system is well understood, users are not sufficiently aware of the threat at stake, and they place their trust in the application's popularity or the application store, accepting the installation without questioning the developer's motivations. More and more methods are being developed to use machine learning classifiers to characterise malware based on its rights, either associatively or individually. This research aims to explore existing literature on malware characterisation and detection strategies based on the above mentioned factors. In order to do so, we highlight the shortcomings of previous studies and offer encouraging ideas for further investigation*

Keywords: Hash Generation , Key Recovery, Blockchain, Government Funding

I. INTRODUCTION

The basis of the Android security concept is permissions. A security feature called permission limits access to specific parts of the device's code or data. The limitation was put in place to prevent sensitive information and code from being misused in order to deceive or impair user experience. Restricted resources and APIs can be accessed or denied based on permissions. For instance, the Android INTERNET permission limits the creation of a network connection since it is required for applications to carry out network communications. A user's phonebook records must be readable by applications, and thus requires the READ CONTACTS permission.

It is the obligation of the designer to decide the consents an application needs. Numerous clients know nothing about what every consent involves and support them automatically, empowering the program to get to delicate data about the client. Another shortcoming is that the client can't pick which privileges to give and which to deny. Numerous customers will, in any case, endorse the establishment of an application regardless of whether it demands problematic consent among numerous clearly legitimate ones. Android has outperformed iOS as the most well known working framework for cell phones and tablets, with an expected piece of the pie of 70% to 80%. With 1 billion Android gadgets expected to be delivered in 2017 and more than 50 billion aggregate application downloads starting from the principal Android telephone was presented in 2008, digital hoodlums have nat-urally directed their concentration toward portable stages. As per portable security trained professionals, there was a surprising development in Android malware from 2012 to 2013, with the quantity of hurtful applications saw as going from 120.000 to 718.000. Numerous endeavors have gone into dissecting the attributes of cell phone stages and their applications somewhat recently to appropriately distinguish malware from official and outsider sources. Bouncer, a Google device, checks applications for possibly hurtful action. Bouncer consequently tests applications submitted to the Android Market by running them in a virtual Android climate facilitated on Google's cloud framework. Albeit how much malware downloads have dropped after Bouncer was introduced, this arrangement doesn't offer security against more up-to-date assault techniques. The consent framework is utilized by the Android stage to restrict application honors to safeguard clients' delicate information. To get to the security pertinent assets, an application should get a

client's consent for the required freedoms. Subsequently, the authorization framework was made to safeguard clients from meddling projects, in spite of the fact that its prosperity is significantly reliant upon the comprehension client might interpret consent endorsement. When a new application is distributed, the framework exhorts checking the Play Store Commercial Center.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 6, May 2024

II. LITERATURE SURVEY

Authorizations important and mentioned are combined with six extra attributes from the manifest and the dismantled code in DREBIN [1]. AI calculations are utilized to naturally grasp the contrast among hazardous and harmless projects. Once prepared disconnected on a committed machine, the Help Vector Machine is simply communicated the learnt model to the cell phone for recognizing perilous applications.

Huang et al. [2] explore the practicality of identifying deceitful Android applications utilizing permis-sions and 20 highlights from application groups. As per their discoveries, a solitary classifier can recognize around 81% of false projects. It very well might be a quick channel to distinguish more suspect applications, as per them, by coordinating discoveries from a few classifiers.

Liu and Liu [3] utilize mentioned and vital consents by an application likewise. AI calculations and consents are utilized in this framework to classify an application as harmless or destructive.

Sanz et al. [4] propose a clever methodology for identifying fake Android applications utilizing AI techniques and looking at the application's separated consents. The presence of labels utilizes authorization and utilizations highlight in the manifest, as well as how much consents conceded to every application, are used to sort them.

As indicated by [5] is a way for building AI classifiers and identifying malware by extricating various properties from the Android manifest. These elements are the particular per-missions looked for and the purposes feature¿¿ tag.

Aung and Zaw [6] present a structure for fostering an AI based malware de-tection framework for Android to recognize malware applications and further develop Cell phone clients' security and protection. This framework gathers various authorization based attributes and occasions from Android applications and examinations them utilizing AI classifiers to decide whether the application is harmless or pernicious.

Shabtai et al. [7] partition Android applications into two classes: utilities and games. Fruitful partition among games and devices, as they would see it, ought to offer a decent mark of such frameworks' ability to learn and show Android harmless projects and perhaps recognize mal-product documents utilizing AI (ML) procedures on static credits gathered from Android program records.

The essayists of these books limit their examination to the most frequently mentioned consents (or a cer-tain choice of authorizations) [8]. Be that as it may, contingent upon the attack, consents like READ LOGS may be similarly all around as hazardous as others (like Web). Each grant ought to be care-completely evaluated as having the capacity to be hazardous when matched with another.

This strategy for choice, as indicated by [9], creates extensively slanted results. AI based identification methods are perceived to have two disadvantages: they have a high pace of phony problems, and picking which qualities ought to be mastered during the preparation stage is a troublesome undertaking. The strategy of picking datasets for preparing is hence a pivotal stage in these frameworks. The presentation of the classifier improves with time: for a specific month Mi, whose applications were utilized for the preparation datasets, the resultant classifier turns out to be less and less fit for distinguishing all malware in ensuing months. Mk,k is more prominent.

To address the applications, most of these endeavors separate a list of capabilities. The data conveyed by such attributes shifts relying upon the gig. There is no proof to demonstrate which credits give the best discovery results, but each examination considers required per-missions. Moonsamy et al. [10] are keen on involving consents as the sole component to portray programs and recognizing specific authorization examples to recognize perfect and malignant applications.

III. PROPOSED METHODOLGY

The system recommends the malware detection methods.We employed the following feature selection techniques: I permission ranking based feature selection; (ii) similarity based permission feature selection; (iii) association rule mining; (iv) modified random forest classifier parameters; and (v) modified random forest classifier parameters.Both the permission feature selection approach based on similarity rate and the permission ranking based feature selection strategy grade the characteristics based on frequency.The association rule mining method, which is widely used in malicious and benign programmes, is used to remove the rights. Additionally, we improve the detection accuracy of permission induced malware with the random forest approach.The random forest that has been improved consistently

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 6, May 2024

removes unnecessary features. By comparing important and nonessential properties, we create an improved random forest approach with fewer but more critical attributes.

TECHNICAL DETAILS

Opcode Definition and Types

1. Opcode Definition:

- An opcode (operation code) is the portion of a machine language instruction that specifies the operation to be performed. It is the fundamental part of the machine code instruction used by the CPU to execute a particular task.

2. Types of Opcodes:

- Arithmetic Opcodes:Perform arithmetic operations like addition, subtraction, multiplication, and division.
- Logic Opcodes: Handle logical operations such as AND, OR, XOR, and NOT.
- Data Transfer Opcodes:Move data between registers or between memory and registers.
- Control Transfer Opcodes: Direct the flow of execution through jumps, calls, and returns.
- I/O Opcodes:Manage input and output operations.

Machine Learning Algorithms

1. Support Vector Machine (SVM):

- SVM is a supervised learning model that analyzes data for classification and regression analysis. It works well for binary classification problems like distinguishing between malware and benign applications.
- Kernel Functions:Use kernel functions (like linear, polynomial, radial basis function) to transform data into higher dimensions where it becomes easier to classify using a hyperplane.

2. Random Forest:

- A versatile machine learning method that operates by constructing multiple decision trees during training and outputting the mode of the classes (classification) or mean prediction (regression) of the individual trees.
- Advantages: Handles overfitting better than individual decision trees, provides feature importance, and works well with large datasets.

3. Neural Networks:

- Particularly useful for handling complex patterns and non-linear relationships within the data. Deep neural networks (DNNs) can be employed for their ability to learn intricate representations from opcode sequences.

Dataset Description

1. Sources:

- Malware Datasets: Publicly available datasets like Drebin, AMD (AndroidMalware Dataset), and VirusShare.
- Benign Datasets: Collection of applications from the Google Play Store or curated collections of legitimate apps.

2. Size:

- Typically, datasets can range from a few thousand to hundreds of thousands of samples, balancing between malware and benign samples to avoid bias.

3. Preprocessing Steps:

- Disassembly: Convert the APK files into readable opcode sequences.
- Opcode Extraction: Extract opcode sequences from disassembled files.
- Feature Encoding: Encode opcodes into numerical vectors suitable for machine learning models (e.g., one-hot encoding, word embeddings).
- Normalization: Scale features to ensure uniformity and improve model performance.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 6, May 2024

IV. METHODOLOGY ENHANCEMENTS

Feature Extraction Techniques

1. Opcode Frequency:

- Count the frequency of each opcode in the application's code to create a feature vector.
- Term Frequency-Inverse DocumentFrequency (TF-IDF): A statistical measure used to evaluate the importance of a word (or opcode) in a document (or app) relative to a collection of documents (entire dataset).

2. Opcode Sequences:

- Analyze sequences or patterns of opcodes rather than individual opcodes. This can capture more contextual information.
- N-grams: Extract n-grams (sequences of n opcodes) to understand opcode patterns and transitions.

3. Control Flow Graphs (CFG):

- Construct CFGs from the opcode sequences to analyze the control flow within the application's code. This helps in understanding the structural patterns in malicious applications.

Model Evaluation Techniques

1. Precision, Recall, and F1-Score:

- Precision: The ratio of correctly predicted positive observations to the total predicted positives.
- Recall: The ratio of correctly predicted positive observations to all observations in the actual class.
- F1-Score: The weighted average of Precision and Recall.

2. Accuracy:

- The ratio of correctly predicted observations to the total observations.

3. ROC-AUC (Receiver Operating Characteristic - Area Under Curve):

- Measures the ability of the model to distinguish between classes. A higher AUC indicates a better performing model.

4. Confusion Matrix:

- Provides a detailed breakdown of true positives, false positives, true negatives, and false negatives, helping in assessing the model's performance.

Feature Selection

1. Permission Ranking Based Feature Selection:

- Rank permissions based on their frequency and correlation with malware presence.

2. Similarity-Based Permission Feature Selection:

- Group permissions based on similarity metrics (e.g., cosine similarity) to identify and select relevant features.

3. Association Rule Mining:

- Identify frequent item sets (permissions) and derive association rules to filter out non-essential permissions.

4. Random Forest Feature Importance:

- Use the feature importance scores provided by the random forest to select the most impactful features.

Improved Random Forest Classifier

1. Parameter Tuning:

- Optimize hyperparameters like the number of trees, maximum depth, minimum samples split, and feature subsets to enhance the classifier's performance.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 6, May 2024

2. Feature Reduction:

- Eliminate redundant and non-informative features based on importance scores to reduce complexity and improve efficiency.

3. Balanced Training:

- Ensure balanced training data to prevent bias, using techniques like SMOTE (Synthetic Minority Over-sampling Technique) if necessary.

Real-Time Detection

1. Incremental Learning:

- Implement models capable of learning incrementally to adapt to new malware patterns in real-time without the need for retraining from scratch.

2. Low-Latency Inference:

- Optimize models for low-latency inference to facilitate real-time detection on resource-constrained devices.

V. CONCLUSION

Prior studies have mostly concentrated on permissions, sensitive resources, intents, etc., and not many solutions have been put out to tackle the malware detection issue from an API standpoint. In order to close this disparity, we examine how dangerous applications use APIs in comparison to benign apps. By training the random forests classifier with fine-grained features, we are able to extract highly sensitive APIs and mine malware patterns. We suggest an automated malware detection solution using a machine learning algorithm to help Android app markets.

REFERENCES

[1] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, "DREBIN: Effective ¨ and Explainable Detection of Android Malware in Your Pocket," 2014.
[2] C.-Y. Huang, Y.-T. Tsai, and C.-H. Hsu, "Performance Evaluation on Permission-Based Detection for Android Malware," in Advances in Intelligent Systems and Applications- Volume 2, pp. 111–120, Springer, 2013.
[3] X. Liu and J. Liu, "A Two-Layered Permission-Based Android Malware Detection Scheme," in Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2014 2nd IEEE International Conference on, pp. 142–148, IEEE, 2014.
[4] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. Alvarez, "Puma: Permission ´ usage to detect malware in android," in International Joint Conference CISIS12-ICEUTE´ 12-SOCO´
[5] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, J. Nieves, P. G. Bringas, and G. Al- varez Mara ´ n˜on, ´ "MAMA: Manifest Analysis for Malware Detection in Android,"Cybernetics and Systems, vol. 44, no. 6-7, pp. 469–488, 2013
[6] Z. Aung and W. Zaw, "Permission-based android malware detection," International Journal Of Scientific Technology Research, vol. 2, no. 3, 2013.
[7] A. Shabtai, Y. Fledel, and Y. Elovici, "Automated static code analysis for classifying An- droid applications using machine learning," in Computational Intelligence and Security (CIS), 2010 International Conference on, pp. 329–333, IEEE, 2010.12 Special Sessions,pp. 289–298, Springer, 2013.
[8] R. Sato, D. Chiba, and S. Goto, "Detecting Android Malware by Analyzing Manifest Files," Proceedings of the Asia-Pacific Advanced Network, vol. 36, pp. 23–31, 2013.
[9] K. Allix, T. F. D. A. Bissyande, J. Klein, and Y. Le Traon, "Machine Learning-Based Malware Detection for Android Applications: History Matters!," 2014.
[10] V. Moonsamy, J. Rong, and S. Liu, "Mining permission patterns for contrasting clean and malicious android applications," Future Generation Computer Systems, vol. 36, pp. 122–132, 2014