IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 12, Issue 6, June 2023

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.423

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

Malicious Pattern Detection from android API's using Machine Learning

Rohit Thorat, Dr.(Mrs.) S. K.Wagh

Master of Engineering Student, Department of Computer Engineering, Modern Education Society's College of Engineering, Pune, India

Professor, Department of Computer Engineering, Modern Education Society's College of Engineering, Pune, India

ABSTRACT: Smartphone usage and popularity have grown over the last several years, making them the primary communication tool. Attackers continuously keep an eye on cellphones in order to steal sensitive data and information from them using a variety of malware assaults, one of which is Advertising Software (MalPat). Machine Learning (ML) approaches are being developed by researchers and security experts for the purpose of detecting android malware effectively. Using network traffic monitoring, a novel detection technique is suggested in this study to defend smart devices from malware assaults. In the dataset that is being given, malware samples are detected using a variety of data pre-processing, feature selection approaches, and ML algorithms. The performance indicators for malware detection were then compared between the ML classifiers, SVM, and NB.

KEYWORDS: Android applications, malware detection, permission-related APIs, random forests, software security

I. INTRODUCTION

Android OS dominates the world with more than 85% of the market share of smartphones and it keeps growing which made it a main target for attackers. Moreover, android apps are considered easy targets for reverse engineering therefore, it is often exploited by malicious attackers. Attackers are developing malicious apps to target individuals and draining their money, accessing their data and stealing the sensitive information, and destroy the operating systems of the smartphones, IoT, wearable's, and advanced transportation services. Because of the increasing use of the apps that always-connected to the Internet, the networks traffic significantly increased. The fifth generation (5G) connection will generate nearly 3X more traffic than a 4G connection by 2023. This growth is not just in traffic but also in malicious software. Malicious software (malware) is a type of software that is created to disrupt, damage or to unauthorized access to any application, smart device system or network. One kind of malware is Adware (advertising software) which is "a software that displays or downloads advertising material automatically (which are often unwanted) when a user is online".Adware is designed to generate revenue for its developer in every time a user clicks on an advert it shows. Some adware kinds gather user browsing information without permission and utilize it to serve user ads that are more relevant that he/she interesting

II. LITERATURE SURVEY

Permissions necessary and requested are coupled with six additional characteristics from the manifest and the disassembled code in DREBIN [1]. Machine learning algorithms are used to understand the difference between dangerous and benign programmes automatically. Once trained offline on a dedicated machine, the Support Vector Machine is just transmitted the learnt model to the smartphone for identifying dangerous apps.

Huang et al. [2] investigate the feasibility of detecting fraudulent Android apps using permis- sions and 20 features from app bundles. According to their findings, a single classifier can identify around 81 percent of fraudulent programmes. It may be a fast filter to detect more suspect apps, according to them, by integrating findings from several classifiers.

Liu and Liu [3] use requested and necessary permissions by an application in a similar way. Machine learning algorithms and permissions are employed in this system to categorise an app as benign or harmful.

Sanz et al. [4] propose a novel approach for detecting fraudulent Android apps using machine learning methods and examining the application's extracted permissions. The existence of tags uses-permission and uses-feature in the manifest, as well as the amount of permissions granted to each application, are utilised to categorise them.

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

According to [5] is a way for building Machine Learning classifiers and detecting malware by extracting numerous properties from the Android manifest. These features are the specific per- missions sought and the uses-feature¿¿ tag.

Aung and Zaw [6] present a framework for developing a machine learning-based malware de- tection system for Android in order to identify malware apps and improve Smartphone users' security and privacy. This system collects numerous permission-based characteristics and events from Android apps and analyses them using machine learning classifiers to determine if the app is benign or malicious.

Shabtai et al. [7] divide Android apps into two categories: utilities and games. Successful separation between games and tools, in their opinion, should offer a good indicator of such systems' capacity to learn and model Android benign programmes and possibly identify mal- ware files using Machine Learning (ML) techniques on static attributes collected from Android programme files.

The writers of these books limit their research to the most often requested permissions (or a cer- tain selection of permissions) [8]. However, depending on the assault, permissions like READ LOGS might be just as dangerous as others (like INTERNET). Every permit should be care- fully assessed as having the potential to be dangerous when paired with another.

This method of selection, according to [9], produces considerably skewed results. Machine learning-based detection techniques are recognised to have two drawbacks: they have a high rate of false alarms, and choosing which characteristics should be learnt during the training phase is a difficult task. The procedure of picking datasets for training is therefore a crucial stage in these systems. The performance of the classifier improves with time: for a particular month Mi, whose apps were used for the training datasets, the resultant classifier becomes less and less capable of identifying all malware in subsequent months. Mk,k is greater than i.

To represent the applications, the majority of these efforts extract a feature set. The information conveyed by such characteristics varies depending on the job. There is no evidence to prove which attributes provide the greatest detection results, however each research takes needed per- missions into account. Moonsamy et al. [10] are interested in using permissions as the sole feature to characterise programmes and identifying certain permission patterns to distinguish between clean and malicious apps.

III. PROPOSED SYSTEM DESIGN

The malware detection techniques are proposed by the system. We used I the permission ranking-based feature selection technique (ii) the similarity-based permission feature selection (iii) the association rule mining technique (iv) the modified random forest classifier parameters) and (v) the modified random forest classifier parameters. The permission ranking-based feature selection strategy and the permission feature selection technique based on similarity rate the characteristics based on frequency. The permissions are deleted using the association rule mining technique, which is popular in malware and benign applications. Furthermore, we increase the random forest algorithm's detection accuracy for permission-induced malware. The upgraded random forest eliminates superfluous characteristics repeatedly. We develop an enhanced random forest technique that comprises fewer but more critical characteristics by comparing essential and non-essential characteristics.

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

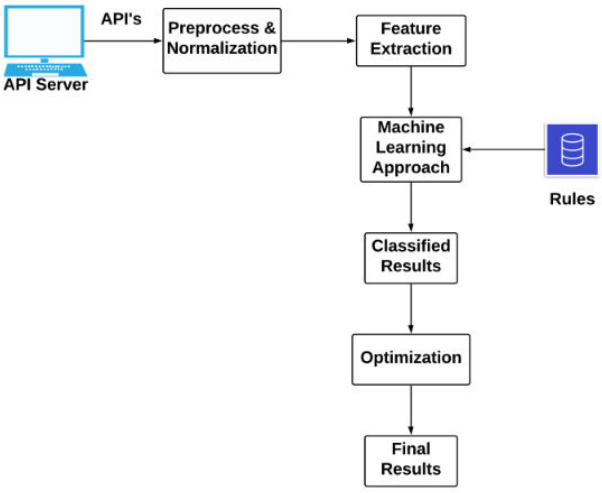

Figure 1: System Architecture

List of Modules and Functionality

- System must takes any kind of .APK file and decompile with respective decompile tool.
- Generate Background Knowledge (BK) for different malicious permissions as rules and generate train module.
- It is must to extract all permission related information from given input .APK files.
- Extracted information has test using any machine learning algorithm and generate the similarity weight for each input.
- According to BK system must show the classification result for individual as well as multiple input objects.
- Try to implement machine learning algorithm for classification and validate the proposed system results with existing approaches.
- Majority voting is also measurable parameter to explore the accuracy of system in case of multiple machines learning algorithms has implemented.

Project Modules

APK Decompilation Module: In this module different APK has decompile using APK Easy software tool. This tool basically converts .apk file into different directories with available class files which also contain manifest.xml, configuration file etc.

Feature Extraction: Various feature extraction techniques have been used to extract the selected features. The entire phase is considered as data pre-processing which deals with data acquisition, classified instance elimination, null value removal and systematic data sampling these techniques have used extract the unique features from input data set

Train module: The feature extraction has been done during the data pre-processing phase and the selected features as used to train the entire module.

Test Module: This model basically works for classifying the test instances from the extracted feature vector, the outcome of this module provides the specific APK contains some malicious information and classify it normal or abnormal respectively**.**

Analysis: After completion of the entire execution finally, the system will illustrate the classification accuracy of proposed system as well as explore and validate the proposed system with some existing machine learning algorithms and show how to propose day planning algorithm effective than classical machine learning algorithms

Algorithm

SVM

Input: Train features TF [], Test features Ts[], threshold T,

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

Output: Classification of weight

Step 1: vector is given as an input

Step 2: Each values in the given vector is extracted

Step 3: The extracted values is searched in the dataset

Step 4: for each (x[] into TF[] when!=null)

Step 5: Get all features x1[]=Ts[]

Step 6: w=CalDistance(x[], x1[])

Step 7: evaluate w with T

Step 8: Classify weight

NB Algorithm

Input: Selected feature of all test instances D[i….n], Training database policies {T[1]…….T[n]}

Output: No. of probable classified trees with weight and label.

Step 1: Read (D into D[i])

V ←Extract features (D)

Step 2: N ←Count Features(D)

Step 3: for each (c into Train DB)

Step 4: Nc[i] ←Ext Features(c)

Step 5: select relevant features of w= {Nc[i], N}

Step 6: Statement (w>t)

Step 9: Return Label { label}

IV. RESULT

The implementation process was completed in a Java open-source setting. The device operates on the Java 3-tier analytics platform with a distributed INTEL 3.0 GHz i5 CPU and 4 GB RAM. Whether an email is spam or not has been determined uses the APK dataset. We have performed experiment analysis on ensemble machine implementation to verify the outcomes**.**

The detection accuracy of APK in malicious or not detection using different machine learning and deep learning classifications.

To ensure higher accuracy, our model was trained using many classifiers, which were then checked and compared. The

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

user will get each classifier's assessed results. The user may compare the result with other results to determine if the data is APK in malicious or not when all classifiers have returned their findings to them. For easier comprehension, graphs and tables will be used to display each classification result.

Table I. Comparison Machine Learning algorithms Table

Classifiers	Accuracy	Precision	Recall	F-score
SVM	80.50	100	89.50	85.40
NB	81.40	100	87.40	82.40

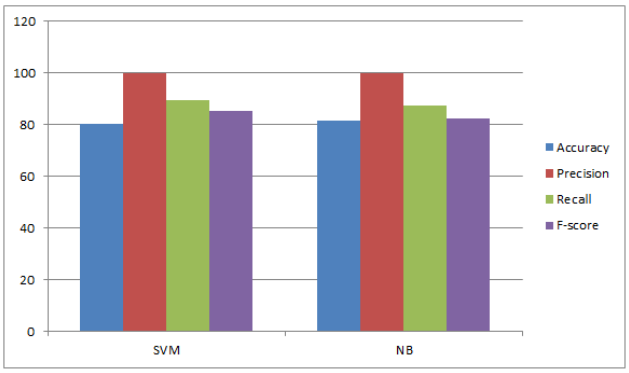

Figure 2: Comparison of Machine Learning algorithms

V. CONCLUSION

The number of Android smartphones connecting to the Internet has recently increased. For connectivity and data sharing, the majority of these Android devices rely on Android Apps. The Android platform's permissions system limited the apps' access. Permission may be used as a feature in Android apps to distinguish between good and bad apps. Our efforts, on the other hand, lowered the number of permissions required to maintain accuracy and efficiency. In this project, harmful and benign Android applications are used in the real world to uncover hidden malware patterns. Previous research has mostly focused on permissions, sensitive resources, intents, and the like, with relatively few attempts to solve the malware detection issue from an API standpoint. To close this gap, we compare the behaviour of malicious and benign applications when it comes to API use. We can mine malware patterns and retrieve extremely sensitive APIs by training the random forests classifier with fine-grained features. We propose an automated malware detection method 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.

|e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 6, June 2023 ||

| DOI:10.15680/IJIRSET.2023.1206091 |

[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

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor: 8.423

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details