IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 11, Issue 4, April 2023

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

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

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |

Permission-Based APK Analysis and Machine Learning for Android Virus Detection

Bhanupriya BH[*1], Dr. Channakrishna Raju[*2]

II nd Year, Master of Technology, Department of Computer Science Engineering, Sri Siddhartha Institute of Technology, Tumakuru, Karnataka, India

*Professor, Department of Computer Science Engineering, Sri Siddhartha Institute of Technology, Tumakuru, Karnataka, India

ABSTRACT : The highest market share worldwide belongs to the Google-backed and open source Android platform. Unfortunately, due to its broad use, malicious programmers are now being distributed on this platform by cybercriminals. Due to its ubiquity among smartphone users, Android has turned into a top target for hackers. Malware is increasingly difficult for security providers to detect and identify due to the complex ways in which it is incorporated into Android apps. Traditional detection methods become inadequate as Android malware becomes increasingly complex and challenging to detect.

The complexity and inventiveness of Android development are better addressed by machine learning-based methods. These techniques recognize malware activity patterns and use this knowledge to discriminate between known and unidentified threats. This research suggests an evolutionary genetic feature selection strategy based on machine learning for Android malware detection. The genetic algorithm evaluates the classification accuracy before and after feature selection to determine which characteristics are the most effective for training a machine learning classifier. According to the experimental findings, the genetic algorithm chooses the most efficient feature subset and cuts the feature dimension to just under half of what it was initially. After feature selection, the machine learning classifier retains an accuracy rate of over 94%, lowering computing complexity.

KEYWORDS: Malware, Android, cyber criminals, MLA, Genetic Algorithm

I. INTRODUCTION

The Google Play Store and other third-party app shops accept Android apps for free distribution because it is an open-source operating system. Yet, Android is susceptible to viruses because of its openness. Malware has the ability to access users' private data, including phone numbers, email addresses, and GPS position information, as well as take over their devices.Reverse engineering and malware analysis are needed to counter these threats. The two types of Android malware analysis are static analysis and dynamic analysis. While dynamic analysis analyses an Android application's runtime behavior in a limited context, static analysis analyses the structure of code without ever running it. There is a need for a reliable detection system since Android malware varieties keep multiplying. Static and dynamic analysis, along with a machine learning-based methodology, can be used to identify new Android malware variants that represent zero-day threats. This method is more effective than signature-based methods, which call for frequent modifications to the signature database. uses a light static analysis method called the support vector machine algorithm, and its detection accuracy rate is 94%.Two static analysis-based categorization techniques are presented: one bases classification on the license, while the other models the source code as a collection of words. A different strategy is to rank the most crucial permissions and use machine learning to assess them.

II. ANDROID MALWARE DETECTION BASED ON APK PERMISSIONS

[1]. Feizollah et al. classified Android malware using static analysis and got a 91% detection rate, which climbed to 95.5% when intents and permissions were combined. But, relying solely on intentions is insufficient and needs be supplemented with other traits.

[2]. Nisha et al. suggested leveraging mutual information and chi-square approaches for feature selection to identify repackaged Android malware. The highest accuracy was 91.76%, attained by the random forest classifier. The method should be improved to include only the detrimental permissions since it only focuses on 88 recognized permissions.

[3]. Deep learning methods were utilized by Sandeep to use Exploratory Data Analysis (EDA) to find malware as it

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |

was being installed. Using Random Forest as the classifier, the framework achieved 94.6% accuracy by using permissions to mimic application behavior. The method's 331 features for categorization could be improved upon.
[4]. SIGPID, a permission-based detection system for Android malware that uses a multi-level data pruning technique to pick important features, was proposed by Li et al. Using Permission Ranking with Negative Rate, Support Based Permission Ranking, and Permission Mining with Association Rules, they discovered 22 important permissions. Using an SVM classifier, the SIGPID approach obtained 90% precision, accuracy, recall, and F-measure.
[5]. To automatically recognize permission interactions that differentiate between good and bad apps, Wang et al. used Multilevel Permission Extraction (MPE). They evaluated 9736 apps, and their detection rate was 97.88%.
[6].The fregraphs strategy, which builds frequent subgraphs to describe typical behaviors of malware belonging to the same family, was proposed by Fan et al. for identifying Android malware. They also suggested FalDroid, a technique for detection based on fregraphs. With an average of 4.6 seconds per programmer, FalDroid can classify up to 94.2% of malware samples into the appropriate categories.

III. OBJECTIVES

1. Develop an Android application that detects the presence of malware in Android applications installed on smartphones.
2. The application provides real-time analysis of the presence of malware and alerts the user, as well as a classification of malware families.

IV. EXISTING SYSTEM

1. Existing systems use signature-based approaches for malware detection by comparing digital signatures of known malware to scanned files or applications. However, it may not be effective against new or evolving threats. Therefore, researchers are exploring alternative detection methods to improve system and network security.
2. Signature-based approaches use patterns extracted from known malware to create databases of signatures to detect new instances of malware. However, this approach may not be effective against new or unknown threats. Therefore, researchers are exploring alternative malware detection methods to improve overall system security.

V. PROPOSED SYSTEM

The proposed method optimizes the input of a machine learning classifier by reducing feature dimensionality to less than half of the original feature set using a genetic algorithm. This lessens the complexity of training while keeping malware classification accuracy. When using support vector machine and neural network algorithms, the method achieves over 94% classification accuracy, outperforming exhaustive feature selection techniques.

VI. PROPOSED SOLUTION

The goal of this project is to classify Android malware using genetic algorithms by reducing the dimensionality of the feature set. This approach reduces the complexity of machine learning classifiers without sacrificing accuracy. Instead of testing 2N combinations in an exhaustive feature selection method, a heuristic search method and a genetic algorithm based on the fitness function are used. The resulting optimized feature set is then used to train various machine learning classifiers, such as support vector machines, neural networks, and potentially random forest or decision tree classifiers. The results show that classification accuracy remains above 94%, while significantly reducing feature dimensionality, thereby minimizing training time complexity.

VII. ANDROID MALWARE DETECTION ARCHITECTURE

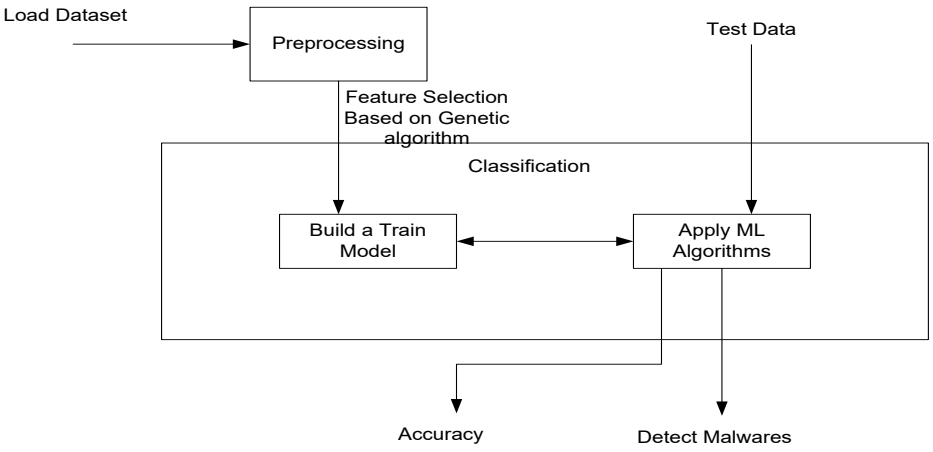
A conceptual framework for explaining the composition and operation of a system is called system architecture. There is no one method that works for everyone. A formal description of a system that arranges its parts and connections to make structural analysis easier is known as an architectural description. It acts as a template for building products and systems to accomplish system objectives. Information on the environment, interface, and operations may be included.
A high-level view of the system is provided by the system architecture, which directs system design and development.

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

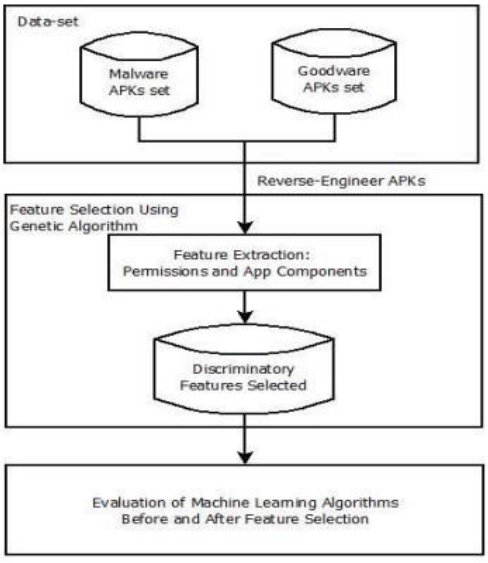
|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |



VII. METHODOLOGY

The proposed Android malware detection method involves reverse engineering malware and genuine APK files to extract features such as permissions and application components, which are then used to create feature vectors with software malware and item class labels. This vector is used to train a machine learning classifier that can accurately distinguish between the two.



VIII. PROPOSED ALGORITHM

a. KNN Algorithm:

K nearest neighbors is one of the simplest machine learning algorithms based on supervised learning techniques.The K-NN algorithm accepts the similarity between the new case/new data and the available cases and places the new case in the category most similar to the available categories.

1. Select Kth from the neighbors
2. Get the K neighbors Euclidean distances.
3. Pick the K neighbors that are closest to you based on the determined Euclidean distance.
4. Count the number of each sort of data point inside these k neighbors.
5. Provide fresh data points to the class with the most neighbors.
6. Finished is our model.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |

b. Random Forest:

An incredibly common supervised machine learning technique used for Classification and Regression issues in machine learning is called the Random Forest Algorithm. A forest is made up of many different types of trees, and the more trees there are, the more robust the forest will be. Similar to this, the accuracy and problem-solving capacity of a Random Forest Algorithm increase with the number of trees in the algorithm. In order to increase the dataset's predictive accuracy, a classifier called Random Forest uses many decision trees on different subsets of the input data. It is based on the idea of ensemble learning, which is the practice of integrating various classifiers to solve a challenging problem and enhance the model's performance.

1.Choose k features at random from all m features, where k << m.
2.Calculate the node "d" using the optimal split point among the "k" characteristics.
3.Use the optimum split to divide the node into daughter nodes.
4.Up until the "l" number of nodes, repeat steps 1 through 3 as necessary.
5. To produce a "n" number of trees, repeat steps 1 through 4 a "n" number of times.

c. SVM

Support Vector Machine, sometimes known as SVM, is a linear model used to solve classification and regression issues. It works well for many real-world issues and can solve both linear and non-linear problems. The SVM concept is straightforward: A line or a hyperplane that divides the data into classes is produced by the algorithm.

1. Input: Test instance X', Training set S, Base classifiers N.
2. Training Process:
For i=1:N
Employ the Bootstrap method to create the training subset Si from the input S;
Develop the SVM classifier SVM_i;
End by instructing the SVM algorithm to modify the classification hyperplane of SVM_i to create $SVM\text{-}OTHR_i$.
3. Testing Process:
For i=1:N
To predict its class label yi', enter x' into the classifier SVM-OTHRi;
end
To determine the final class label, y', use majority voting;
4. Output: y', which is the test instance x's anticipated class label.

d. Naive Bayes

It is a classification method built on the Bayes Theorem with the assumption of predictor independence. A Naive Bayes classifier, to put it simply, believes that the presence of one feature in a class has nothing to do with the presence of any other feature.

1. Input:Training set T, F=(f1, f2,..., fn)
2. Output: A subset of test data.
3. Step: The following steps should be repeated:
a. Read the training dataset T;
b. compute the mean and standard deviation of the predictor variables for each class
c.Repeat, Once the probabilities of all the predictor variables (f1, f2,...,fn) have been determined, calculate the probability of fi using the Gauss density equation for each class.
d. Determine the likelihood in each scenario;
e. Find the best chance;

VIII. RESULT

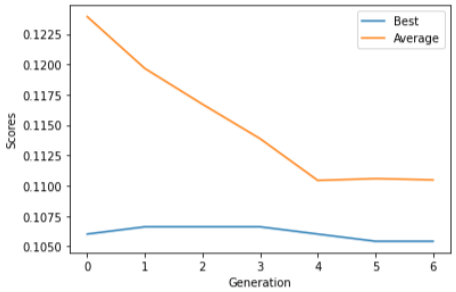
1. To determine if an apk file contains malware or legitimate software, it is necessary to download and scan the file for potential threats.
2. To make predictions based on the input data, machine learning algorithms are used. The algorithm is trained on a set of labeled data to learn patterns and make predictions on new unlabeled data. To evaluate the performance of an algorithm, a performance metric such as accuracy is used to determine how well the algorithm is able to correctly classify the output. By measuring the accuracy of an algorithm's predictions, we can determine the effectiveness of a machine learning approach in solving a problem.

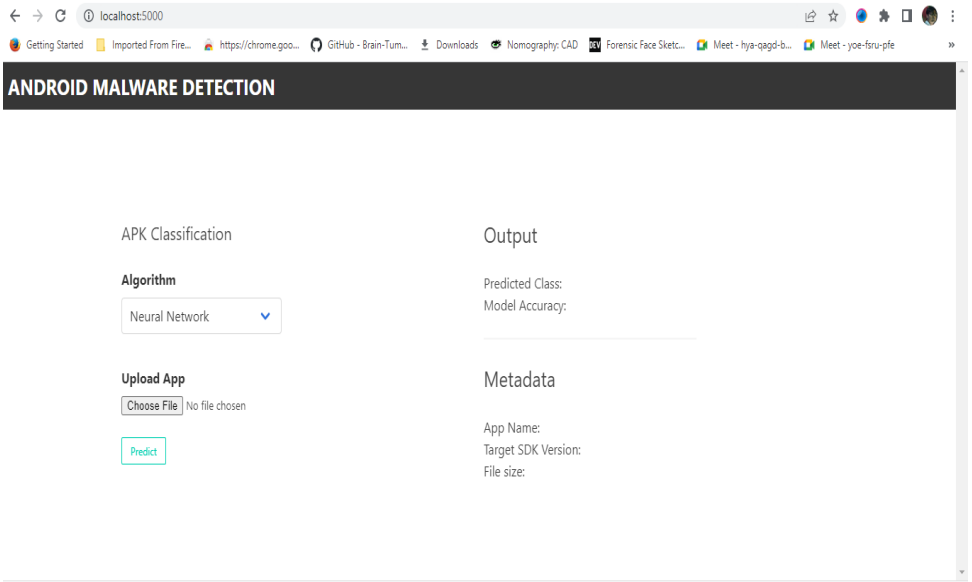
| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

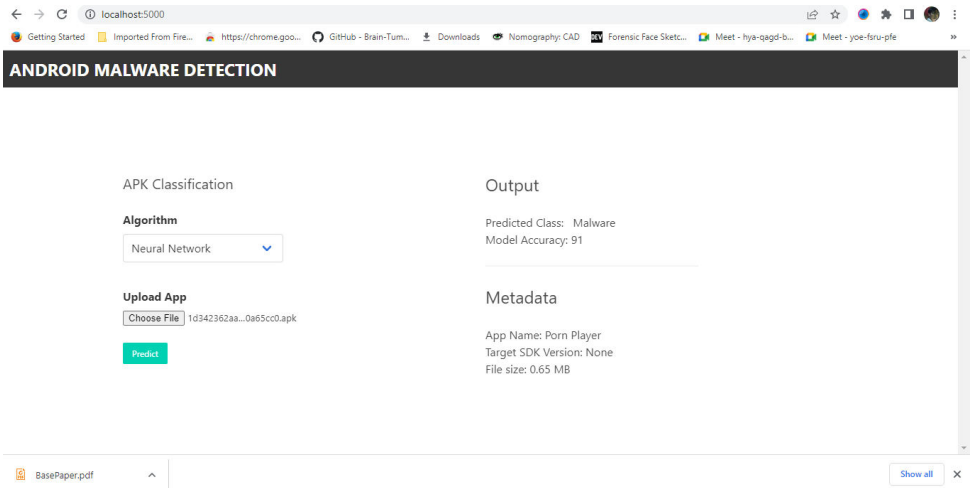
|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |

```
generation: 1
generation: 2
generation: 3
generation: 4
generation: 5
generation: 6
generation: 7
```





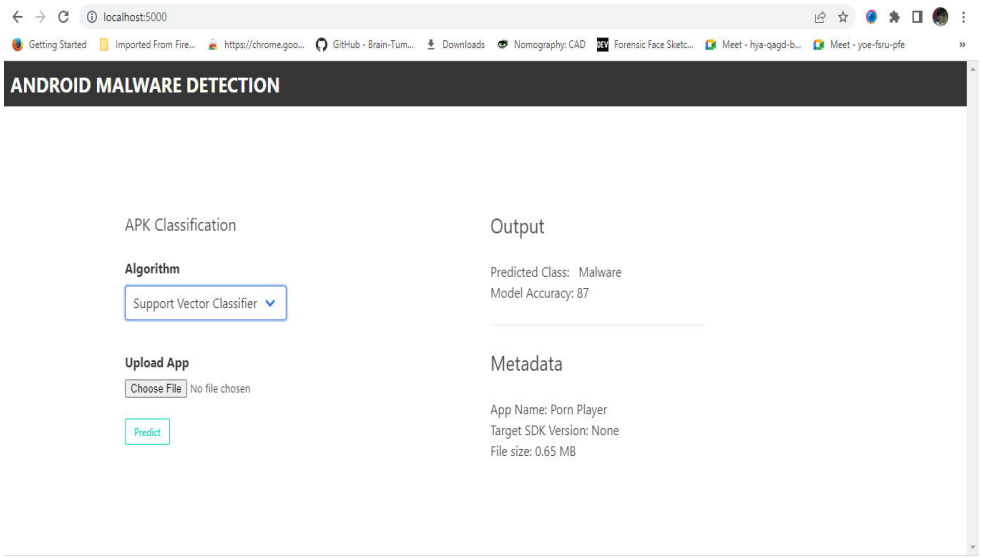


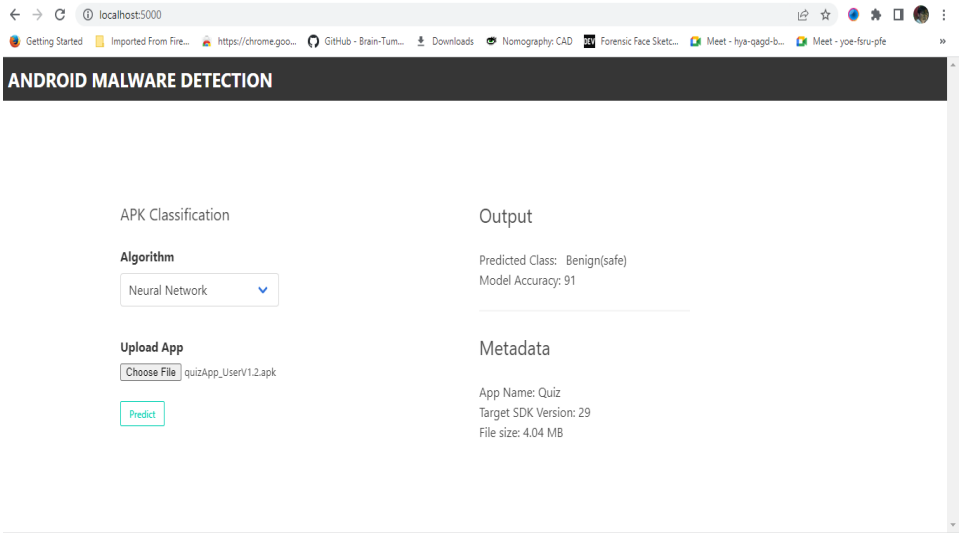
International Journal of Innovative Research in Computer and Communication Engineering

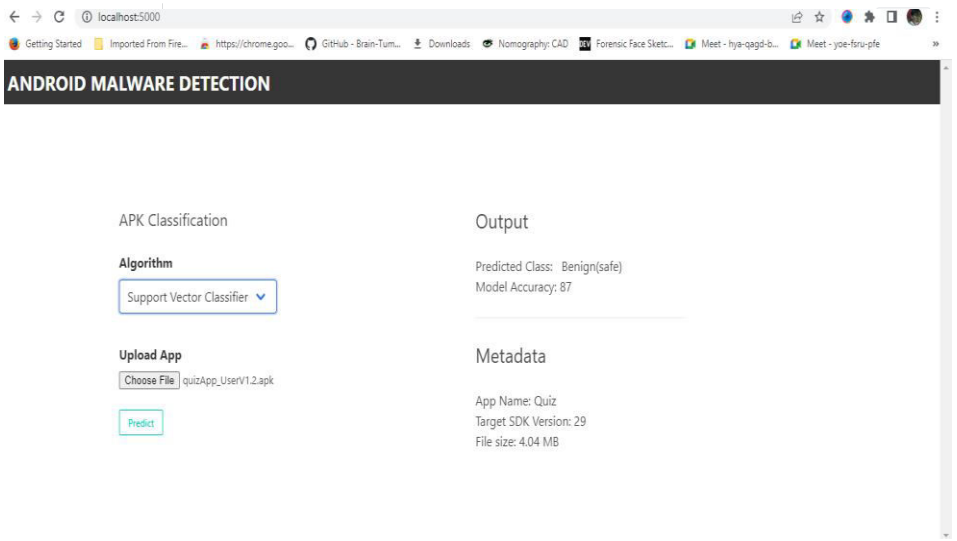
| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |







| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104047 |

IX. CONCLUSION AND FUTURE DISCUSSION

As the Android platform faces an increasing risk of rogue apps and malware, a reliable framework is needed to accurately detect these threats. Signature-based methods are insufficient to detect new and unknown malware variants, also known as zero-day threats. Therefore, machine learning based methods are used to solve these problems. To achieve this goal, researchers have proposed a novel method that uses an evolutionary genetic algorithm to optimize a subset of functionality for a machine learning algorithm. The goal is to improve the accuracy of malware detection.Experimental results show that support vector machines and neural network classifiers can achieve over 94% classification accuracy when using low-dimensional feature sets. This not only improves the accuracy of classifiers, but also simplifies their training process. Future research could focus on using larger datasets to further improve results, as well as analyzing the effectiveness of other machine learning algorithms when combined with genetic algorithms. Overall, this proposed approach provides a promising solution for malware detection on the Android platform.

REFERENCES

Here is a list of six research articles focused on detecting Android malware:
[1] Feizollah, A. et al. (2017). "AndroDialysis: Analyzing the Effectiveness of Android Malware Detection." This article examines the effectiveness of different Android malware detection techniques, including static analysis and dynamic analysis.
[2] Jannath, N.O.S. enBhanu, S.M.S. (2018). "Detection of modified Android apps based on app permissions." This research paper explores a method of detecting malware in patched Android apps by scanning permissions.
[3] Sandeep, H.R. (2019). “Statistical Decomposition for Android Malware Detection Using Deep Learning.” This article explores the use of deep learning techniques to detect Android malware, where a statistical decomposition is a key approach.
[4] Lee, J.wait. (2018). "Key Permission Identification for Machine Learning-Based Android Malware Detection." This article focuses on the application of machine learning algorithms to Android malware detection, focusing on identifying key permissions-related features.
[5] Wang Zhongwait. (2019). "Multi-Level Privilege Extraction in Android Applications for Malware Detection." This research paper proposes a multi-level privilege extraction method to detect Android malware.
[6] Fan, male. wait. (2018). "Classification of Android Malware Families and Selection of Representative Samples by Conventional Subgraph Analysis." This article proposes a method to classify Android malware families and select representative samples using conventional subgraph analysis.

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor: 8.165

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details