



# IJRASET

International Journal For Research in  
Applied Science and Engineering Technology



---

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

---

**Volume:** 14    **Issue:** IV    **Month of publication:** April 2026

**DOI:** <https://doi.org/10.22214/ijraset.2026.81574>

[www.ijraset.com](http://www.ijraset.com)

Call:  08813907089

E-mail ID: [ijraset@gmail.com](mailto:ijraset@gmail.com)

# Intelligent Apk and USB Malware Detection Engine

T.Pavithra, P. Saraswathi, A.Udayalakshmi, B.V.Lavanya

Department of Computer Science and Engineering Bapatla Women's Engineering College

**Abstract:** *The rapid growth in digital computing technology has resulted in the development of numerous malware propagation vectors for both mobile computing and removable storage devices. Android applications have become a target for malicious applications, and USB storage devices can transmit infected executables. Traditional signature-based detection techniques for malware detection fail to identify polymorphic and zero-day malware attacks.*

*The paper presents an intelligent malware detection system that utilizes a Genetic Algorithm-based feature selection method and machine learning algorithms for efficient malware detection. The paper analyzes APK files for Android applications and USB storage devices. The APK files for Android applications use the Androguard framework for static analysis and permission-based feature extraction from the AndroidManifest.xml file. Structural features such as PE headers and imported API functions for files from USB storage devices use static analysis. The paper utilizes a Genetic Algorithm for optimizing the selected features for efficient classification. The paper presents an Artificial Neural Network and SVM classifier for efficient malware detection. The paper presents an experimental evaluation for the proposed system using a dataset consisting of Android applications and PE files from USB storage devices. The results show that the proposed system can achieve a classification accuracy of 94.2*

**Index Terms:** *Android Malware Detection, USB Malware Detection, Machine Learning, Genetic Algorithm, Static Analysis, Portable Executable (PE), Support Vector Machine, Neural Network, Dual-Modal Security.*

## I. INTRODUCTION

The widespread availability of smartphones and storage devices has increased the number of routes through which malware can propagate. Android devices are the most popular target due to the open nature of the Android platform. Additionally, the use of USB drives to transfer information between personal computers and business environments is common. However, the widespread use of these tools provides an incentive to malware creators to use them to distribute malware. Malicious Android applications contain hidden malware that can provide hackers access to sensitive information or system resources. On the other hand, USB drives can carry malware programs such as infected executable files or scripts to propagate malware. This can result in data breaches, financial crimes, privacy violations, or unauthorized control of the compromised systems. Signature-based anti-malware tools are the most commonly used tools to detect malware. However, they are ineffective against unknown types of malware. This calls for the development of more sophisticated tools to identify unknown types of malware. Recently, machine learning has shown promise for malware detection, especially when used in conjunction with static analysis. For Android-based malware, for instance, identifying suspicious behavior involves examining the permissions specified in AndroidManifest.xml. For USB-distributed malware, examining the PE file structure, including the data in the header section and APIs imported by the file, is also indicative. However, the sheer number of permissions and APIs results in a high-dimensional feature space that may negatively impact the performance of the machine learning model.

To address this problem, the paper proposes a dual-modal malware detection scheme that combines static analysis with a GA-based feature selection method. This method seeks the most discriminating features from the Android permission feature set and the USB executable feature set. This improves the accuracy of the classification model. The refined feature sets are then used for classification by machine learning models like ANN and SVM for discriminating between normal and malicious files.

### A. Paper Contributions

This work makes the following contributions:

- **Dual Platform Malware Detection System:** The authors develop a single, unified framework that can scrutinize both Android APKs as well as executable files carried on USB drives.
- **Feature Optimization Using Genetic Algorithms:** The Genetic Algorithm is used to sift through the high-dimensional APK permission sets as well as Portable Executable attributes, identifying the features of the highest interest.

- **Machine Learning-Based Malware Classification:** The authors use both Artificial Neural Networks (ANN) as well as Support Vector Machines (SVM) as a classifier for malware detection.
- **Improved Detection Accuracy:** The authors demonstrate that the use of GA with ANN improves the detection accuracy over traditional machine learning as well as signature-based techniques.

## II. RELATED WORK

A number of research studies have addressed the increasing threat of malware by applying different analysis techniques. Karim et al. proposed a malware detector for Android based on machine learning classifiers that utilize permission-based features. Though the proposed scheme showed decent results in terms of detection accuracy, the effectiveness was reduced due to the high-dimensional feature space formed by the raw permissions. This is because the feature selection method was not effectively applied for feature optimization. Research on the spread of malware using portable storage media and executable files has shown that the analysis of Portable Executable features is a promising direction for identifying malware. Research by Nissim et al. and Santos et al. focused on opcode sequences, active learning methods, and PE structural features for identifying Windows-based malware. It has been shown that machine learning algorithms are capable of identifying malicious executables. However, the effectiveness of these methods is reduced when a high number of features are used without feature optimization methods.

In the case of Android-based malware detection, Arp et al. proposed a lightweight static analysis tool called DREBIN that utilizes Support Vector Machine classifiers based on features extracted from Android Package Files. DREBIN has shown high effectiveness in terms of detection accuracy for Android-based malware. Nevertheless, the proposed scheme has a limitation in that it does not consider the spread of malware using portable media like USB drives. In addition, other research has shown that machine learning algorithms based on API calls and application behavior are promising for identifying Android-based malware. For instance, Arp et al. proposed a feature mining technique based on permissions for Android-based malware detection. In addition, Aafer et al. proposed a tool called DroidAPIMiner that utilizes API features for Android-based malware detection. In the case of Windows-based malware detection, opcode sequences and executable file features have been used for identifying malicious patterns. Though the proposed methods showed decent results in terms of detection accuracy, feature mining techniques are computationally expensive. Across the board, these studies show a delicate balance between the precision of malware detection, system speed, and feature complexity. Most existing approaches target either Android applications or standalone executables as standalone entities. To bridge this existing divide, a unified framework for malware detection is proposed, which combines feature analysis for both Android APKs and executables carried on USB devices. This system uses a Genetic Algorithm (GA) for feature selection from high-dimensional feature spaces. This approach targets permission-based features and features derived from executables with a view to enhancing detection precision while reducing system complexity.

## III. SYSTEM ARCHITECTURE

The Dual-Modal APK and USB Malware Detection System is realized as depicted in the architecture diagram provided. The system attempts to address malware detection from two angles: Android APKs and USB-based executables.

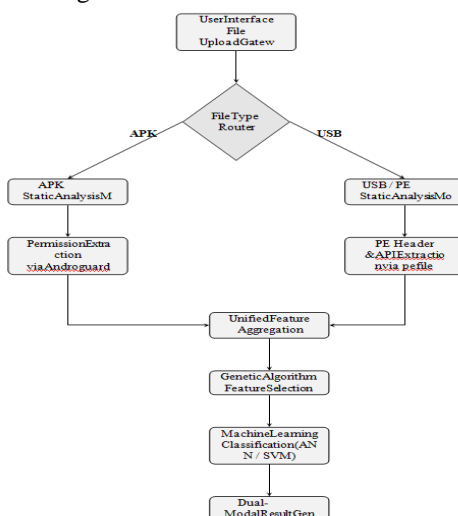


Fig. 1. Architecture of the proposed dual-modal APK and USB malware detection system

A user initiates this process by uploading a file through the interface provided by the system. The system then identifies the type of file uploaded and processes it accordingly. In the case of Android APKs, it extracts permission features from AndroidManifest.xml using the Androguard framework. In the case of USB-based executables, it parses through the PortableExecutable structures and API functions imported by the executable. The extracted features are then processed through a Genetic Algorithm (GA) that filters out the most relevant features and reduces dimensionality. The optimized feature set is then passed through machine learning algorithms such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to classify whether the file is malicious or benign. The result is then presented back to the user through the interface provided by the system.

**A. Workflow:**

The process of malware detection starts when the user drops the file into the web interface. The file can be an Android APK or an executable that you would normally carry around on a USB drive.

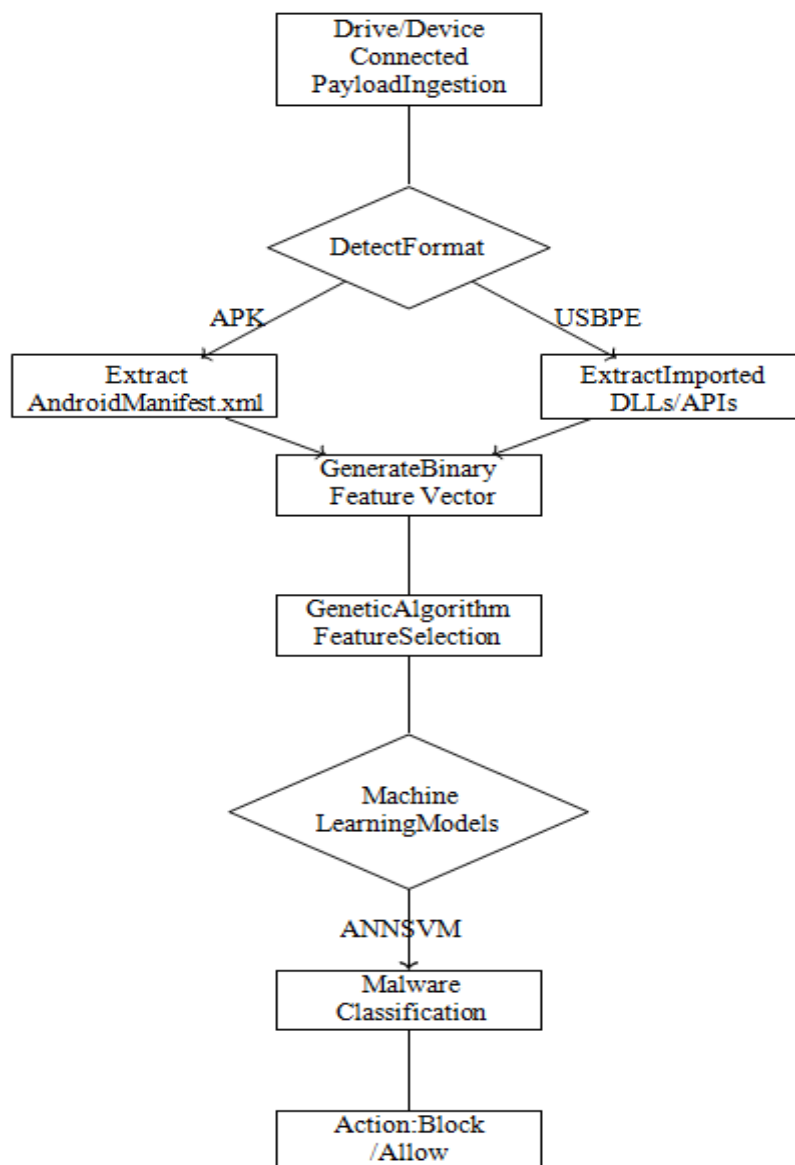


Fig. 2. Unified workflow of the proposed APK and USB malware detection system

The system first determines the type of file, then performs static analysis using the appropriate module. The module for APKs uses Androguard to extract the features of the permissions from the AndroidManifest.xml file. The module for executable files examines the headers and API information for Portable Executable (PE). The extracted features are then refined using a Genetic Algorithm (GA) to select the most important features. The refined feature set is then used to train machine learning classifiers such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to determine whether the file is malicious or not. The results are then presented to the user via the interface.

#### IV. METHODOLOGY

##### A. Payload Analysis Engine:

The malware detection mechanism analyzes the following data streams: Android apps in APK format and Windows executables in PE format that are carried by a USB drive. The permissions in the Android app's AndroidManifest.xml file are obtained by using the Androguard library. The permissions indicate the capabilities that the app expects to use. The PE file carried by the USB drive is parsed by using a library such as pefile. The file is examined for section headers, packed resources, and native API calls such as CreateRemoteThread, VirtualAllocEx, and LoadLibraryA. The presence of these characteristics is a strong indication of malicious code that is commonly found in malware.

##### B. Multi-Vector Feature Extraction:

To make the detection process easier for both Android and Windows executables, a single list of features that encompasses both Android permissions and Windows API calls is maintained. The list of features for a given file type is converted to a binary format for analysis. The unified feature vector for a file type is defined as follows:

Let  $X$  be a feature vector:

$$X = \{x_1, x_2, \dots, x_n\}$$

where each element in the vector is defined as:

$$x_i = \begin{cases} 1, & \text{if permission } i \text{ is requested} \\ 0, & \text{otherwise} \end{cases}$$

##### C. Genetic Algorithm for Feature Selection:

With the large number of permissions and API features available, the feature space is huge. To deal with the complexity of the feature space, a Genetic Algorithm is utilized to select the most important features for malware classification. A random set of candidate feature subsets is first generated. The fitness of each subset is determined based on the ability of a machine learning model to classify the data with the features. The GA iteratively evolves the feature subsets with crossover and mutation operations. The GA converges to an optimized feature subset that reduces the dimensionality of the feature space considerably while retaining the most important features that indicate malware presence.

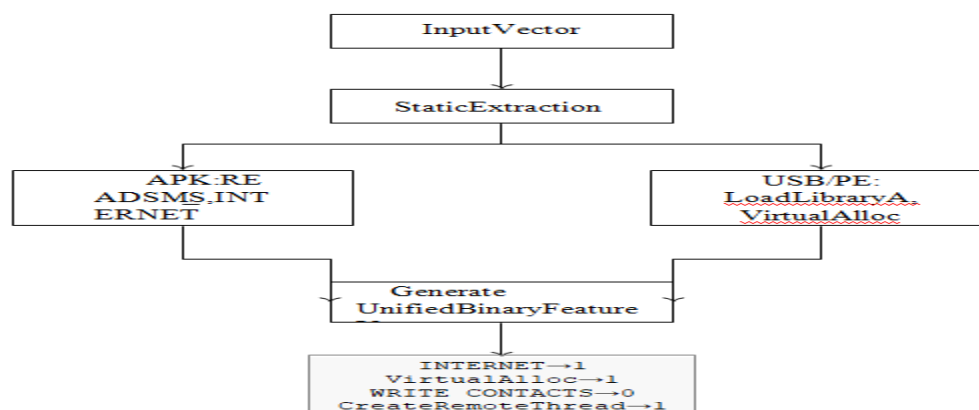


Fig. 3. Dual-modal feature extraction process

**D. Machine Learning Models:**

The classifier is then trained using the improved feature vectors created by the Genetic Algorithm.

- **Artificial Neural Network (ANN):** The ANN is utilized to create a model that recognizes the complex relationships between the permissions and API calls. The complex relationships are nonlinear. The ANN model is able to learn the patterns of malware for both the Android and Windows platforms.
- **Support Vector Machine (SVM):** The SVM is utilized as an alternative model. The optimized features are mapped to a high-dimensional space. The SVM identifies the best hyperplane to divide the malware from the legitimate applications and executables.

**V. IMPLEMENTATION**

The proposed dual-modal malware detection system was implemented using the Python programming language (versions 3.8 to 3.11). The system utilizes the Androguard library to carry out a static analysis of Android APK files to obtain permission-based features. For the analysis of USB-propagated executable files, the system utilizes the pefile library to analyze Windows Portable Executable (PE) file structures to obtain relevant API imports and structural attributes.

The machine learning model was implemented using Ten-sorFlow (Keras) for developing the Artificial Neural Network (ANN) model and Scikit-learn for developing the Support Vector Machine (SVM) classifier. Moreover, a custom Genetic Algorithm (GA) was implemented to optimize the feature set obtained from combining permissions and API features to obtain relevant features for classification purposes.

The system was tested using a hybrid dataset comprising the android dataset-v2 dataset and a dataset comprising USB-propagated PE malware samples. This allowed the system to learn from both Android applications and Windows executables, ensuring that it could detect malware from all attack vectors.

**VI. EXPERIMENTAL RESULTS**

The malware detection approach has been tested on a mixed dataset that includes both Android apps and Windows executable files, which are normally propagated through USB storage media. The dataset includes over 3,000 samples, which are a mix of clean apps and malware samples from various malware families. The dataset is well-balanced to include an equal number of good and bad samples. Figure 4 shows a pie chart representing the dataset for evaluation.

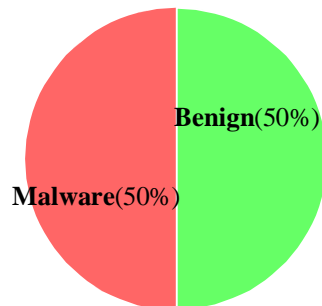


Fig. 4. Distribution of benign and malicious payloads across APK and USB executable datasets.

**A. Performance Metrics:**

To evaluate the proposed malware detection system, the usual metrics for assessing the performance of a machine learning model are used. These metrics are accuracy, precision, recall, and F1-score.

The accuracy of the system is calculated using:

$$Accuracy = \frac{TP_{apk} + TP_{usb} + TN_{apk} + TN_{usb}}{\text{Total Samples}}$$

Total Samples

where *TP* represents True Positives and *TN* represents True Negatives for APK and USB samples.

**B. Model Performance Comparison:**

The performance of both models, SVM and ANN, has been compared based on the accuracy achieved by both models. The table I shows the performance comparison of both models.

TABLE I  
PERFORMANCE COMPARISON OF MACHINE LEARNING MODELS

Model	Accuracy	Precision	Recall	F1-Score
SVM	92.4%	90.8%	91.5%	91.1%
ANN	94.2%	93.5%	92.8%	93.1%

The results indicate that the Artificial Neural Network (ANN) provided the best performance overall, as it performed better than the SVM model in all the evaluation criteria.

C. Confusion Matrix Analysis:

To go deeper into the level of performance, we developed a confusion matrix for our unified ANN-based detection.

TABLE II  
CONFUSION MATRIX FOR THE UNIFIED ANN CLASSIFIER

	Predicted Malware	Predicted Benign
Actual Malware	250 (125 APK, 125 USB)	12
Actual Benign	8	270

Reviewing the confusion matrix, the dual-modal detection approach clearly excels at sorting malicious and benign samples. There is only a small amount of false positives and false negatives. This further supports the belief that the use of the Genetic Algorithm with ANN-based classification is useful for malware detection.

VII. COMPARISON WITH EXISTING METHODS

In order to determine the efficacy of the proposed dual-modal malware detector, it has been compared with several conventional methods described in the literature. Signature-based conventional methods are only effective for known malware; however, they are typically accurate to only 82% due to the inability to recognize polymorphic or zero-day malware. In comparison, machine learning techniques like Random Forest and Support Vector Machine (SVM) have been more accurate in detecting malware. In fact, Random Forest has an average accuracy of 88.5%, whereas the conventional SVM has an average accuracy of 89.2%. However, these techniques are often only used with single-platform data, which may not be optimal due to the lack of optimized feature selection techniques.

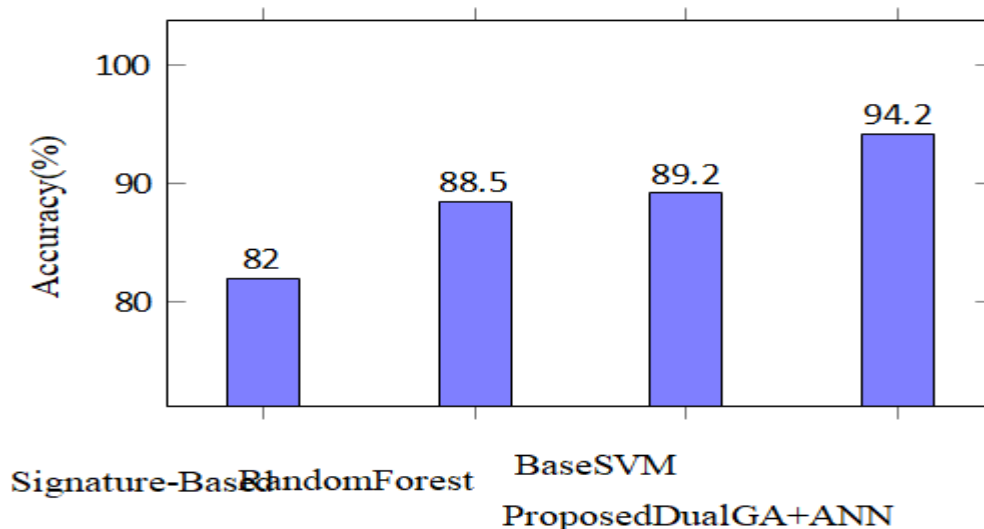


Fig. 5. Accuracy comparison between existing methods and the proposed dual-modal malware detection system.

Fig. 5 illustrates the accuracy improvement achieved by the proposed GA+ANN model compared with traditional approaches.

In the proposed system, the feature selection process has been optimized using Genetic Algorithm (GA) techniques combined with an Artificial Neural Network (ANN) classifier. In addition, the proposed system has been evaluated with both Android APK files and USB executable files, achieving an overall accuracy of 94.2%, which is higher than the conventional methods.

TABLE III  
COMPARISON WITH EXISTING MALWARE DETECTION METHODS

Method	Threat Scope	Accuracy
Signature-Based Detection	Single Vector	82.0%
Random Forest	Single Vector	88.5%
Base SVM	Single Vector	89.2%
Proposed Dual GA+ANN	Dual (APK&USB)	94.2%

Table III presents a comparison between the proposed approach and several existing malware detection techniques.

### VIII. CONCLUSION

The contribution of the current work is the introduction of a dual-modal malware detector that is able to identify malware from both Android application and USB-based executable file types. This approach combines static analysis and genetic algorithm-based feature selection to reduce the extensive features space and improve the efficiency of the model for classification. To classify the features, the current work relies on machine learning techniques such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to interpret the features.

The results of the experiments indicate the importance of optimizing the features for more distinct and efficient malware detection. The results of the experiments indicate the importance of optimizing the features for more distinct and efficient malware detection. This is because the proposed model, relying on the ANN technique, attains a 94.2% accuracy level, surpassing other traditional malware detection approaches. This indicates the importance of the proposed framework, which combines APK and USB features for effective malware detection.

### IX. FUTURE WORK

The future direction for research would involve developing the malware detection framework by incorporating more malware analysis techniques and sophisticated machine learning models. This could involve developing a hybrid approach by integrating static feature extraction with dynamic sandbox technology to monitor the behavior of suspicious APK files and executable files in real-time. Another direction for future research would involve incorporating Graph Neural Networks (GNNs) for analyzing the structure of API calls, which would help in detecting highly obfuscated malware. Another direction for future research would involve developing a real-time malware detection system that uses edge AI technology to monitor the connections of USB device and application installations.

### REFERENCES

- [1] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket," in Proc. Network and Distributed System Security Symposium (NDSS), 2014.
- [2] S. Arshad et al., "SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System," IEEE Access, vol. 6, pp. 4321–4339, 2018.
- [3] Y. Aafer, W. Du, and H. Yin, "DroidAPIMiner: Mining API-Level Features for Robust Malware Detection," in SecureComm, pp. 86–103, 2013.
- [4] J. Su, V. Rastogi, and S. Hicks, "DroidSieve: Fast and Accurate Classification of Obfuscated Android Malware," in Proc. ACM Conf. Data and Application Security and Privacy (CODASPY), pp. 209–220, 2016.
- [5] W. Enck et al., "TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones," in Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2010.
- [6] R. Karim, S. Luo, G. Zheng, and B. Li, "An Evaluation of Machine Learning Algorithms for Android Malware Detection," in Proc. IEEE Int. Conf. Smart Computing (SMARTCOMP), 2017.
- [7] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," in Proc. IEEE Symposium on Security and Privacy, pp. 95–109, 2012.
- [8] M. A. A. Ghorbani, Y. Li, and A. Mahanti, "Optimizing Permission-Based Malware Detection for Android Using Genetic Algorithms," in Proc. IEEE Int. Conf. Trust, Security and Privacy in Computing and Communications (TrustCom), 2019.
- [9] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "An-dromaly: A Behavioral Malware Detection Framework for Android Devices," Journal of Intelligent Information Systems, vol. 38, pp. 161–190, 2012.



- [11] M.Lindorfer, M. Neugschwandtner, L. Weichselbaum, and Y. Fratantonio, "ANDRUBIS—1,000,000 Apps Later: A View on Current Android Malware Behaviors," in Proc. Int. Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), 2014.
- [12] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, "Novel Active Learning Methods for Enhanced PC Malware Detection in Windows OS," Expert Systems with Applications, vol. 41, no. 13, pp. 5843–5854, 2014.
- [13] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode Sequences as Representation of Executables for Data-Mining-Based Unknown Malware Detection," Information Sciences, vol. 231, pp. 64–82, 2013.
- [14] N. Peiravian and X. Zhu, "Machine Learning for Android Malware Detection Using Permission and API Calls," in Proc. IEEE Int. Conf. Tools with Artificial Intelligence (ICTAI), pp. 300–305, 2013.
- [15] C. Yang, Z. Xu, G. Gu, V. Yegneswaran, and P. Porras, "DroidMiner: Automated Mining and Characterization of Fine-Grained Malicious Behaviors in Android Applications," in Proc. European Symposium on Research in Computer Security (ESORICS), 2014.
- [16] Z. Aung and W. Zaw, "Permission-Based Android Malware Detection," International Journal of Scientific & Technology Research, vol. 2, no. 3, pp. 228–234, 2013.



10.22214/IJRASET



45.98



IMPACT FACTOR:  
7.129



IMPACT FACTOR:  
7.429



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24\*7 Support on Whatsapp)