iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** XII **Month of publication:** December 2022

DOI: https://doi.org/10.22214/ijraset.2022.47841

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue XII Dec 2022- Available at www.ijraset.com

A Machine Learning Technique to Detect Malware

Ankit Joshi[1], Komesh Borkar[2], Rohit Dhote[3], Saurabh Raut[4], Swapnil Thomare[5], Raghavendra Kulkarni[6], Sharda Chhabria[7]

[1, 2, 3, 4, 5]*Student,* [6, 7]*Mentor, Department Of Artificial Intelligence, G H Raisoni Institute Of Engineering And Technology Nagpur*

Abstract: ***Organizations have been threatened by malware for a long time, but timely detection of the virus remains a challenge. Malware may quickly damage the system by doing pointless tasks that burden it and prevent it from operating efficiently. There are two ways to detect malware: the traditional method that relies on the malware's signature and the behavior-based approach. The malware's behavior is characterized by the action it conducts when active in the machine, such as executing the operating system functions and downloading infected files from the internet. Based on how it behaves, the suggested algorithm finds the virus. The suggested model in this study is a hybrid of Support Vector Machine and Principle Component Analysis. For real Malware, our suggested model obtained an accuracy of 92.70% during validation, with 96% precision, 96.32% recall, and an f1-score of .96.***

Keywords: Malware, Malware Detection, Behavior-based, Principle Component Analysis, Support Vector Machine, Machine Learning

I. INTRODUCTION

With the increased use of the internet and computer systems, data security (both personal and professional) has become a serious issue. Computers using the internet download massive amounts of data from the internet, which may potentially include viruses. Malware is known by many various names, including malicious code, harmful programs, and malicious executable files. As malware assaults have become more common, computer systems have become more vulnerable to the threat. Malware as defined by Kaspersky Labs “a type of computer program designed to infect a legitimate user’s computer and inflict harm on it multiple ways”. With the vast array of malware that is released each day, anti-virus scanning is ineffective. ensure the identification of all types of malware based on its signature, resulting in millions of hosts being targeted and inflicting significant harm to data and other connected systems. Every day, 560,000 new pieces of malware are discovered. There are currently over a billion malware applications on the internet. Every minute, four businesses are targeted by ransomware. Trojans are responsible for 58% of all computer malware. As a result, safeguarding the network and user machines against malware is a critical cyber security duty for a single user or a whole enterprise, because even a single assault may cause substantial loss and harm. The goal of this work is to create a malware detection system that will identify malware based on the actions it may undertake on the machine it is installed on.
Malware can be of different varieties but there are following major categories: -
Virus: is described as a little bit of code that has the ability to duplicate itself. It attaches itself to any valid file and executes its code when the file is downloaded or processed.
Worms: are similar to viruses in that they may multiply themselves. The main distinction between a worm and a virus is that a worm operates on a network and replicates itself by sending copies of itself to machines connected to that network.
Spyware: is software that is frequently included with free software. When the user installs the program, spyware is activated and begins gathering the user's personal information from the system and transmitting it to the host machine through the Internet.
Adware; s described as a malicious piece of code added to any advertisement or 'click me' button that appears on the screen. When a user clicks on a button or advertising, the code linked to it executes and installs a virus or bot into the user's PC.
Trojans; In general, any software that authenticates the user, such as a login page to a website or a contact information form, confuses the user. After the user enters the information, it is collected and sent to the host/hacker.
A Botnet; is defined as a networked group of numerous bots. A single bot is a little piece of code that is assigned the duty of allowing a hacker easy access to a system. A hacker can employ a bot to run n virus on the n user's PC, gather personal information, or damage the performance of the user’s machine.
Malware Detection is done in two phases : -

1) *Malware Analysis:* is the initial stage of detection During this step, previously known malware data is gathered. Traits are produced and retrieved from the virus, and an algorithm based on those features is constructed to detect fresh incoming malware.

©IJRASET: All Rights are Reserved

2) *Malware Detection:* occurs after the investigation and the creation of a competent algorithm capable of identifying malware with high accuracy. The generated method is then applied to incoming packets to determine if they are malware or benign.

II. METHODOLOGY

There are several mechanisms for detecting malware, such as data mining, deep learning, hypothesis exploration, and so on. However, the Machine Learning approach is one of the most used in malware. The malware detection method is divided into two types. The first is the standard signature-based technique, which detects malware based on its signature. The second and most recent strategy for malware detection is the behavior-based approach, in which malware is recognized based on the behaviors it intends to execute on the system it is attempting to attack. The behavior-based technique is more advanced since it can detect newly created malware based on the actions and tasks that they execute on the computer.

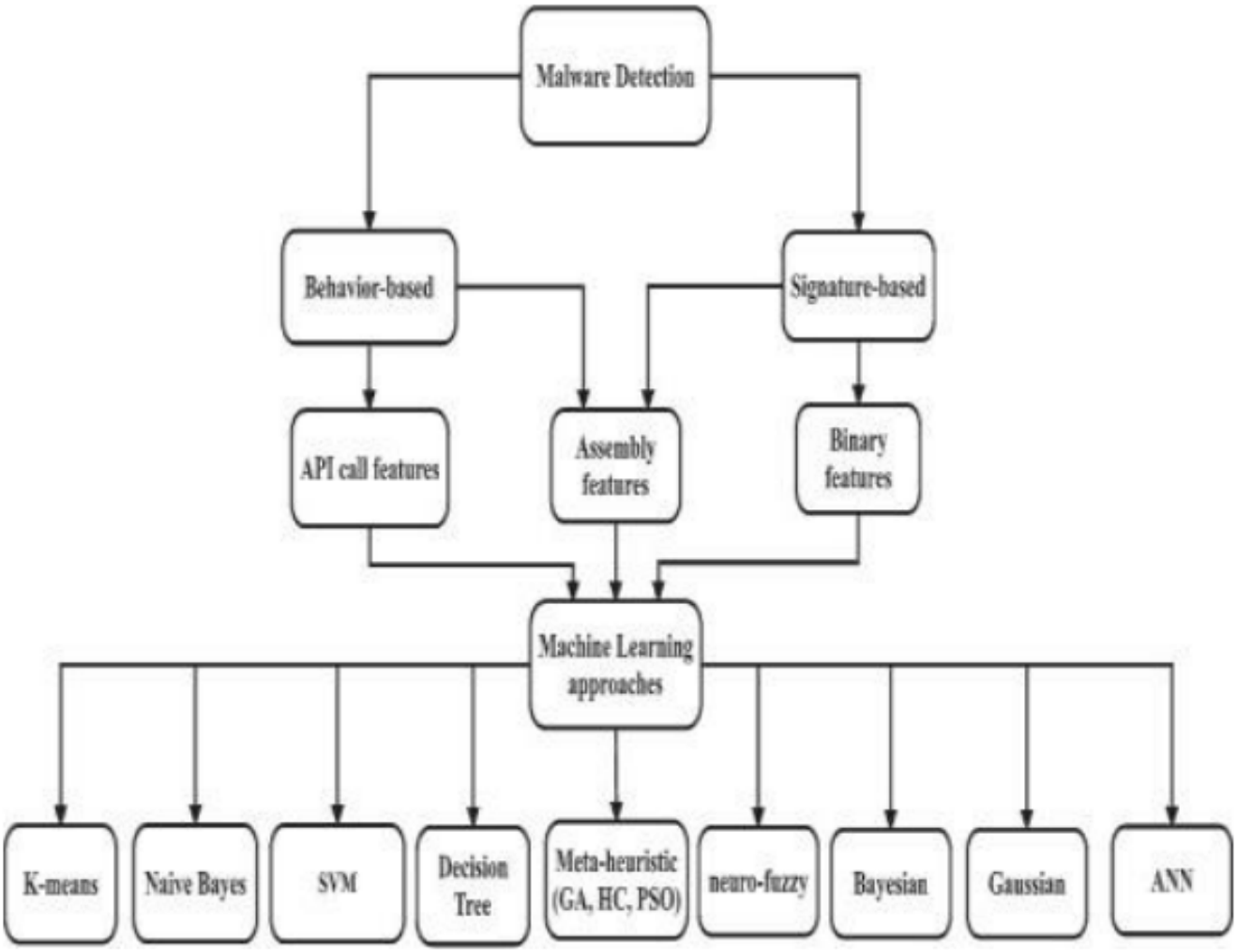


Fig. 1 Malware Detection approaches

A. *Behavior-Based Approach*

The task performed by a file when it is implemented on a computer is specified as its behavior. Payload persistence, stealth techniques, environment mapping, and other such characteristics are examples of these behaviors. Before performing an action, behavior-based malware detection evaluates an incoming file based on its established duties and activities. The behaviors of a file, or in certain situations a potential danger, are checked for harmful activity. Dynamic analysis is the process of looking for harmful activity forms. While no algorithm is perfect, a behavior-based approach is being used in technology today to detect new and undiscovered dangers in near real-time.

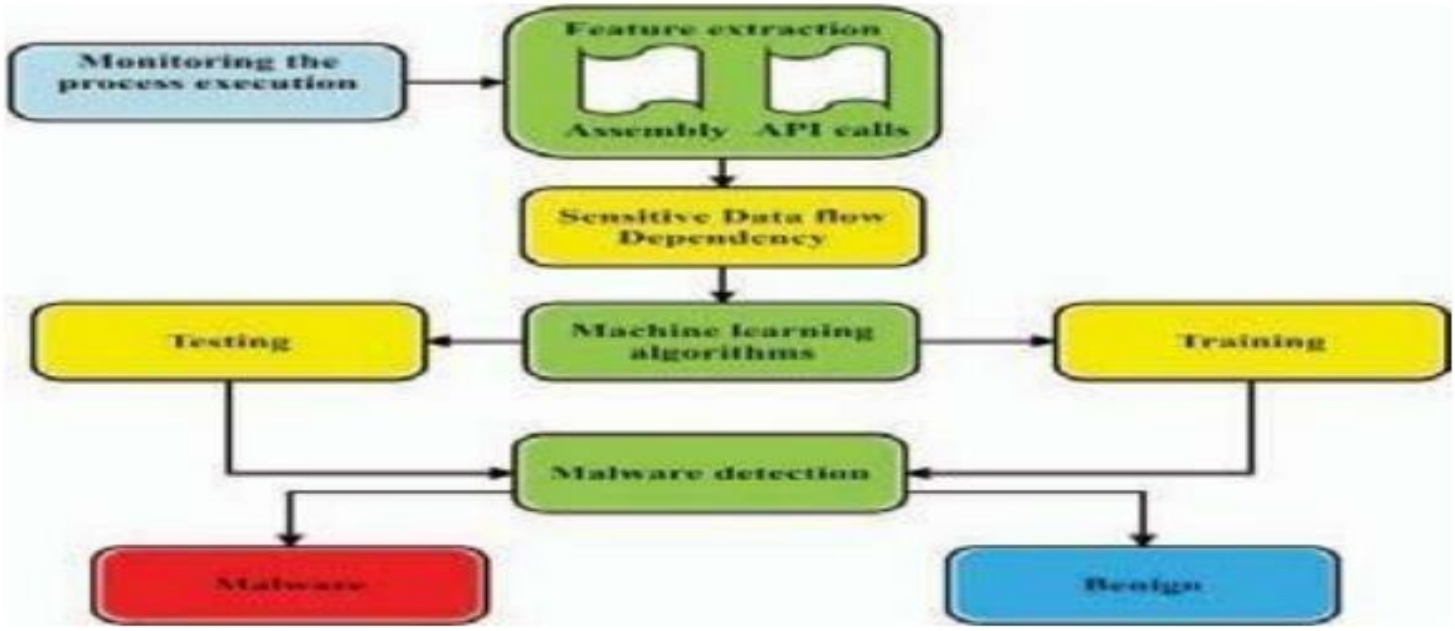


Fig. 1 The Behavior-Based Malware Detection FreamWork

B. Related Work

Malware has plagued people and systems for a long time, therefore much research has been conducted in this field. This part examines the previous work and discusses how to identify malware efficiently. The section will also provide a quick overview of the benefits and drawbacks of some of the research work done in the same area. FAN, Ye, and Chen proposed using the All Nearest Neighbor (ANN) dynamic sequence mining technique to identify malware based on the patterns identified. The malware detection implementation was based on the extraction of Portable Executable (PE) files. The infection was found once IE files were fed into the ANN algorithm. Although filtering the repetitive patterns helped save time and search space, this strategy had certain shortcomings.

1) Excessive time consumption;
2) Reduced classification accuracy;
3) Inability to identify new and sophisticated viruses.

Fan, Hsiao, Chou, and Tseng used the Application Programmable Interface (API) function to assess and detect harmful software. For malware identification, the author employed SVM, Nave Bayesian, and Decision Tree techniques. The application of the Attribute Selection Methodology with a lesser number, The authors Baldangombo, Jambaljav, and Horng proposed performing static malware detection by evaluating the raw properties of PE files. The selection, data transformation, mac, and machine learning approaches were used in this research. By picking the necessary traits, the irrelevant attributes were deleted. The primary disadvantage of this technology was its high computational and memory complex its.Sanz and colleagues suggested a new machine-learning methodology, to detect malicious codes in a system, based on incoming package features.

The main contributions of this article were 1. It illustrated how to extract characteristics from Android. 2. It demonstrated a technique to malware detection for Android applications. 3. The identification of malware was quite accurate. jerlin and Marimuthu have advocated a strong Rate based MDNBS approach for detecting fraudulent code and classifying API sequences. The primary goal of this technique was to improve malware detection accuracy for the supplied dataset. The benefits of this strategy include: -

a) Improved Detection Rate
b) Reduced time consumption

The disadvantage of this paper was the limited dataset available. So, the issue of underfitting, outliers and noise arise.

This section will describe the detailed description of the proposed work done for the detection of malware. Dataset: - Various malware and benign samples that have previously been discovered by various sources were gathered and subjected to feature extraction.

When the feature extraction was finished, a total of 77 features were generated, which would be utilized to train the model. Software and library used: - Python: a high-level object-oriented programming language that is interactive and interpreted. Modules, exceptions, dynamic typing, extremely high-level dynamic data types, and classes are all included. It is an extremely flexible open-source language. Pandas: is a Python library that provides quick, flexible, and expressive data structures for working with "relational" or "labeled" data in a simple and straightforward manner. Pandas' two core data structures, 'Series' and 'DataFrame,' are of attributes for training that considerably improved the performance of the classification algorithms. The benefit of this technology was that it improved the detection model's performance while simultaneously reducing detection complexity. used to handle missing data, data alignment, merging, and dataset joining.

Scikit-Learn: is an open-source software library developed in Python that contains numerous machine learning methods for model, design, and implementation assessment. It also provides compatibility with other Python libraries such as SciPy and NumPy. Pycharm: is an open-source IDE that includes plugins for several programming and debugging languages. It includes a Python plugin called PyDev, which aids in the implementation of many Python difficulties. Algorithms used: Principle Component Analysis: is used to reduce dimensionality. PCA aids in the elimination of characteristics that are unimportant in categorization. PCA's goal is to find the main components of all accessible characteristics. As a result, by lowering the number of characteristics, the machine can learn more quickly. Support Vector Machine: SVM is generally used for performing binary classification based on the separation by the hyperplane. The hyperplane is selected in such a manner that it maximizes the distance between separation and the closest vectors known as support vectors. The support vectors are the data points that are closest to the hyperplane. By employing kernel functions SVM can be used with non-linear data.

C. Implementation Flow

The suggested model first preprocesses the dataset for missing values by either deleting them or replacing them with the mean values. Once the missing values are removed, the datatype of the entire dataset is synchronized using the panda's library's asType("float64") function in Python. After cleaning, the dataset was submitted to normalization and scaling. After the data has been preprocessed, it is given to PCA Analysis for Dimensionality Reduction. The results of the PCA analysis revealed that just 40 features were required for the model's training, validation, and testing. These 40 characteristics were retrieved from the DataFrame using Panda's library and put into SVM with a Gaussian kernel (C=1 and gamma=0.1). Other kernels, such as the linear kernel and the poly kernel, were tested. The model obtained requires less time to train and has proven to be superior. The flowchart seen below depicts the proposed technique.

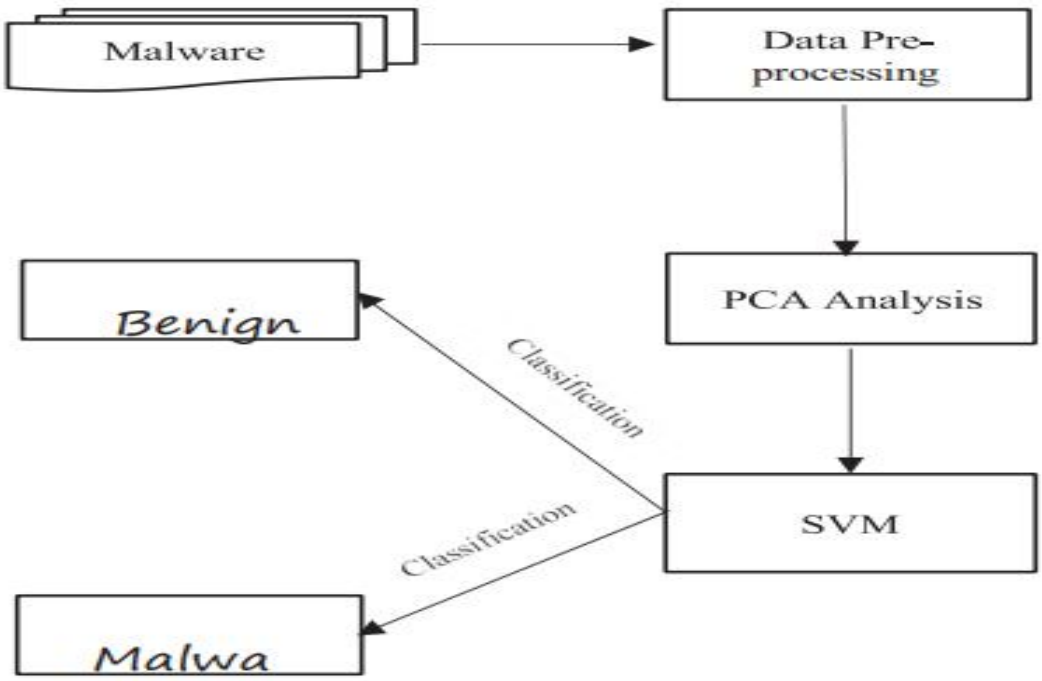


fig 3: - Detailed methodology of Malware Detection

III. RESULTS

The model provided the following results by using SVM and, after applying PCA and feeding the results to train SVM. The accuracy of prediction by only using SVM is found to be 90.4% whereas the accuracy obtained by first applying PCA and then SVM with linear kernel function to the dataset is found to be 97.75%. The Confusion Matrix and Classification Report are shown below:

Table 1: Confusion Matrix

Actual Predicted	Benign	Malware
Benign	1416	115
Malware	43	4314

Table 2: Classification Report

Classification Report	Precision	Recall	f – 1 Score	Support
Benign	0.96	0.92	0.94	1593
Malware	0.96	0.98	0.97	4556
Micro Average	0.96	0.96	0.96	5883
Macro Average	0.96	0.95	0.98	5887
Weighted Average	0.96	0.98	0.95	5884

IV. CONCLUSION

The suggested methodology will be able to more simply and quickly discriminate between malware and benign files and packets. Because the suggested model employs Principle Component Analysis, the computational complexity is reduced. Because it is based on SVM with a Gaussian kernel, the suggested model is also extremely accurate and has excellent performance accuracy.

It is an extremely efficient model for malware detection this model we integrate the backend with an interactive frontend with the help of flask with this integration the uses of the model is very easy for general users to Behavior-based malware detection evaluates an object-based on its intended actions before it can actually execute that behavior.An object's behavior, or in some cases its potential behavior, is analyzed for suspicious activities. Attempts to perform actions that are clearly abnormal or unauthorized would indicate the object is malicious, or at least suspicious. There's a multitude of behaviors that point to potential danger.

Some examples include any attempt to discover a sandbox environment, disabling security controls, installing rootkits, and registering for autostart. Evaluating for malicious behavior as it executes is called dynamic analysis. Threat potential or malicious intent can also be assessed by static analysis, which looks for dangerous capabilities within the object's code and structure. While no solution is completely foolproof, behavior-based detection still leads technology today to uncover new and unknown threats in near real-time. Some examples of where behavior-based technology succeeds when signature-based systems fail.

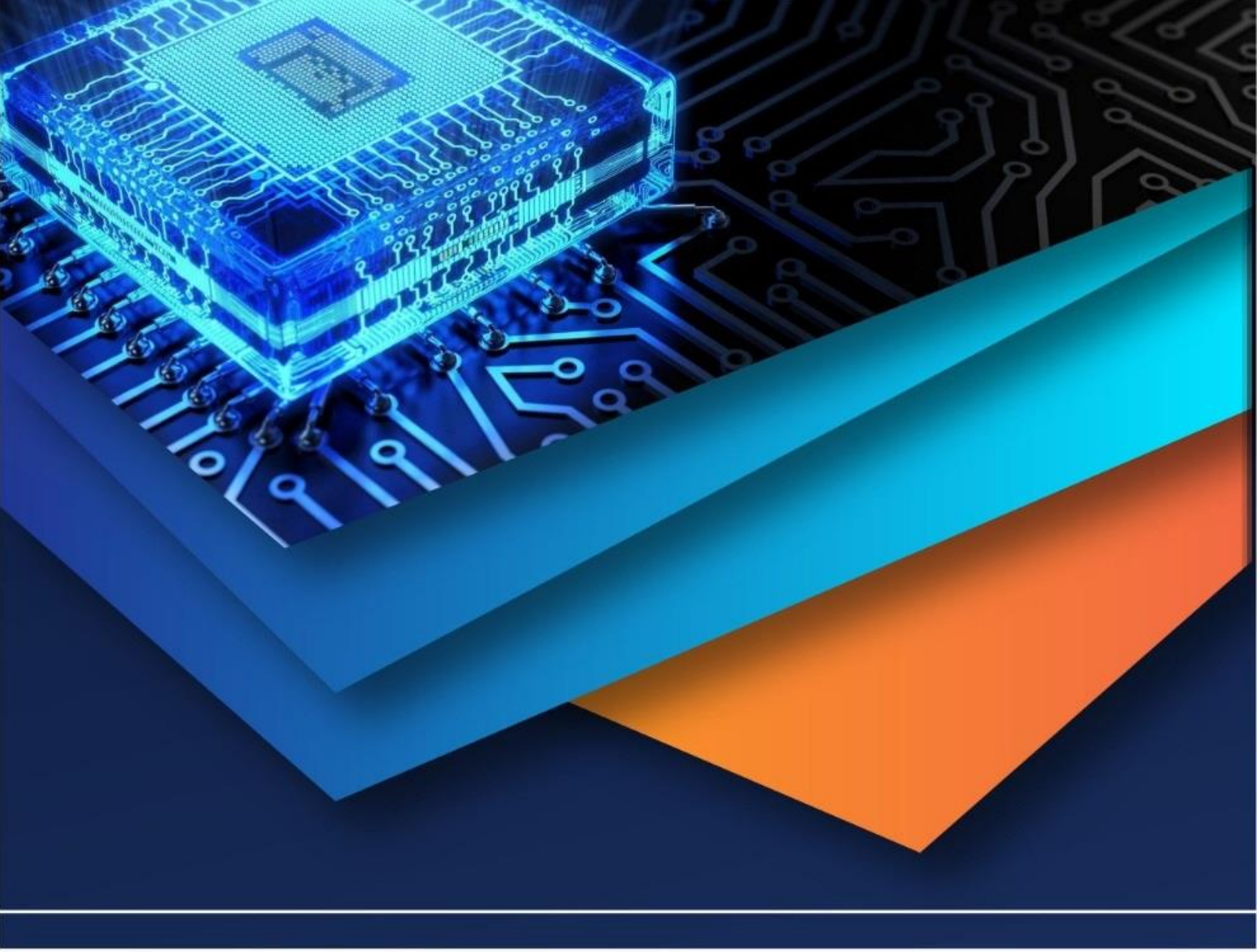
V. FUTURE SCOPE

We also acknowledge the limitations of our current approach and would like to point out several important future directions to make the method more applicable to the real world. 1.The project can be extended from simple malware detection to both integrations of the behaviour-based method and signature-based method, which will involve a more complicated but more realistic solution. 2.The project address a simplified one i.e the file included for testing is from very limited types and sources for the development of real-world application the project must be trained under various file formats and execution environment. 3.The approach is still developed under the development environment to it is not able to work a in the production environment to of the system. 4.The integration of this approach with the different APM (Application Performance Monitoring) tool which helps us to monitor the performance of any application. 5.The integration may cause damage the system to avoid this the project must create the isolated environment for the execution of this project. 6.The end-to-end integration of the project with could can help us to make the project production ready.

REFERENCES

[1] J. Language and M. Wankhvade, "Malware and Malware Detection Techniques: A Survey," Int. J. Eng. Res. Technol., no. 12, pp. 61–68, 2013.
[2] P. Kaur and S. Sharma, "Literature Analysis on Malware Detection," Int. J. Electron. Electr. Eng., vol. 7, no. 7, pp. 717–722, 2014.
[3] I. A. Saeed, A. Selamat, and A. M. A. Abuagoub, "2013-A Survey on Malware and Malware Detection Systems.pdf," vol. 67, no. 16, pp. 25–31, 2013.
[4] U. Baldangombo, N. Jambaljav, and S.-J. Horng, "A Static Malware Detection System Using Data Mining Methods," 2013.

[5] Y. Saint Yen and H. M. Sun, "An Android mutation malware detection based on deep learning using visualization of importance from codes," Microelectron. Reliab., vol. 93, no. October 2018, pp. 109–114, 2019.
[6] S. Sohrabi, O. Udrea, and A. V. Riabov, "Hypothesis Exploration for Malware Detection Using Planning," Twenty-Seventh AAAI Conf. Artif. Intell., pp. 883– 889, 2013.
[7] J. C. Rosales, "Rehumanización y metáfora religiosa en Luis Rosales," Insula, vol. 767, pp. 32–34, 2010.
[8] D. Nieuwenhuizen, "A behavioral-based approach to ransomware detection," Whitepaper. MWR Labs Whitepaper, 2017.
[9] H. S. Galal, Y. Bassiouni, and M. A. Aiea, "Behavior based features model for malware detection," J. Comput. Virol. Hacking Tech., no. April 2018.
[10] Y. Fan, Y. Ye, and L. Chen, "Malicious sequential pattern mining for automatic malware detection," Expert Syst. Appl., vol. 52, pp. 16–25, 2016.
[11] C. I. Fan, H. W. Hsiao, C. H. Chou, and Y. F. Tseng, "Malware detection systems based on API log data mining," Proc. - Int. Comput. Softw. Appl. Conf., vol. 3, pp. 255–260, 2015.
[12] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. Álvarez, "PUMA: Permission usage to detect malware in android," Adv. Intell. Syst. Comput., vol. 189 AISC, pp. 289–298, 2013.
[13] M. A. Berlin and K. Marimuthu, "A New Malware Detection System Using Machine Learning Techniques for API Call Sequences," J. Appl. Secure. Res., vol. 13, no. 1, pp. 45–62, 2018.
[14] "General Python FAQ — Python 3.7.3 documentation." [Online]. Available: https://docs.python.org/3/faq/general.html
[15] documentation." [Online]. Available: http://pandas.pydata.org/pandas docs/stable/getting_started/overview. html.
[16] "An introduction to machine learning with scikit- learn — sci-kit-learn 0.20.3 documentation."
[17] D. J. Bartholomew, "Principal components analysis," Int. Encycl. Educ., vol. 2, no. June 2001, pp. 374–377, 2010.
[18] D. Srivastava, "Data Classification Using Support Vector," Journal of Theoretical and Applied



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)