STM JOURNALS

International Journal of
Mobile Computing Technology

Volume 1, Issue 1, 2023
January—June
DOI (Journal): 10.37591/IJMCT

https://journals.stmjournals.com/ijmct

Review **IJMCT**

Malicious Application Detection in Windows Using SVM Algorithm

Manish Kapoor[1,*], R.M. Samant[2], Suraj Sawant[1], Aishwarya Joshi[1], Neha Tawade[1]

Abstract

In recent years, both the development of Windows application clients and the uses of smart mobile phones have increased significantly. As the number of Windows application users continues to grow, there is a rise in malicious individuals who develop harmful Windows applications with the intent of unlawfully obtaining confidential information and engaging in fraudulent activities. These applications are designed to target vulnerable areas such as mobile banking and digital wallets, aiming to deceive users and misuse their sensitive data. There are so many malicious software, tools, and programmers that are available. However, it is essential to establish a system that is capable and effective for identifying and thwarting freshly developed dangerous programmes written by hackers or programmers. This system should be able to recognise and react to sophisticated threats in an efficient manner. The purpose of this study is to identify fraudulent Windows apps using machine learning techniques.

Keywords: Text summarization, abstractive text summarization, extractive text summarization

INTRODUCTION

Malicious apps are expanding quickly because of the expansion of the Windows market and the growing dependence on mobile devices [1]. The efficacy of malicious programme identification must be improved immediately given the current circumstances. Therefore, using SVM (Support Vector Machine) Algorithms to detect harmful applications has been a popular area of research since it can lower personnel costs and increase detection accuracy.

Motivation

For all of their computing devices, Windows has over one billion active users, a market influence that is driving an increase in the volume of data collected from various users, driving the creation of malicious software by cybercriminals to address the issues maliciously produced.
Windows uses a unique architecture and security measures like a user ID.

***Author for Correspondence**
Manish Kapoor
E-mail: manishkapoor3101@gmail.com

[1]Student, Department of Information Technology, NBN Sinhgad School of Engineering, Pune, Maharashtra, India
[2]Head of Department, Department of Information Technology, NBN Sinhgad School of Engineering, Pune, Maharashtra, India

Received Date: June 07, 2023
Accepted Date: July 03, 2023
Published Date: July 20, 2023

Citation: Manish Kapoor, R.M. Samant, Suraj Sawant, Aishwarya Joshi, Neha Tawade. Malicious Application Detection in Windows Using SVM Algorithm. International Journal of Mobile Computing Technology. 2023; 1(1): 30–36p.

Objective

We employ the Windows Dataset to find malicious software. We input a Windows data collection, which is then subjected to pre-processing. After both phases are complete, we employ the SVM technique to classify and detect malicious applications during the segmentation step.

METHODOLOGY

Support vector machines (SVMs) are a group of supervised learning techniques for classifying data, performing regression analysis, and identifying outliers. Support vector machines' benefits include:

- *Effective in high dimensional spaces:* still useful in situations where the number of dimensions exceeds the number of samples. It is also memory efficient because it only uses a portion of the training points (known as support vectors) in the decision function [2].
- *Versatile:* For the decision function, various Kernel functions can be defined. You can choose from a variety of typical kernels or create your own. For the decision function, various Kernel functions can be defined. You can choose from a variety of kernels, including common ones.

System Architecture

First, data has to be obtained from Kaggle as well as from applications. Data preparation is the process that involves cleaning the data. Divide the data into test data and training data after preprocessing. These data are then given as input to the algorithms [3].

The training data trains the algorithm. After training the algorithms, test data is given as input which gives predicted output and then compared with the expected output to calculate the efficiency (Figure 1).

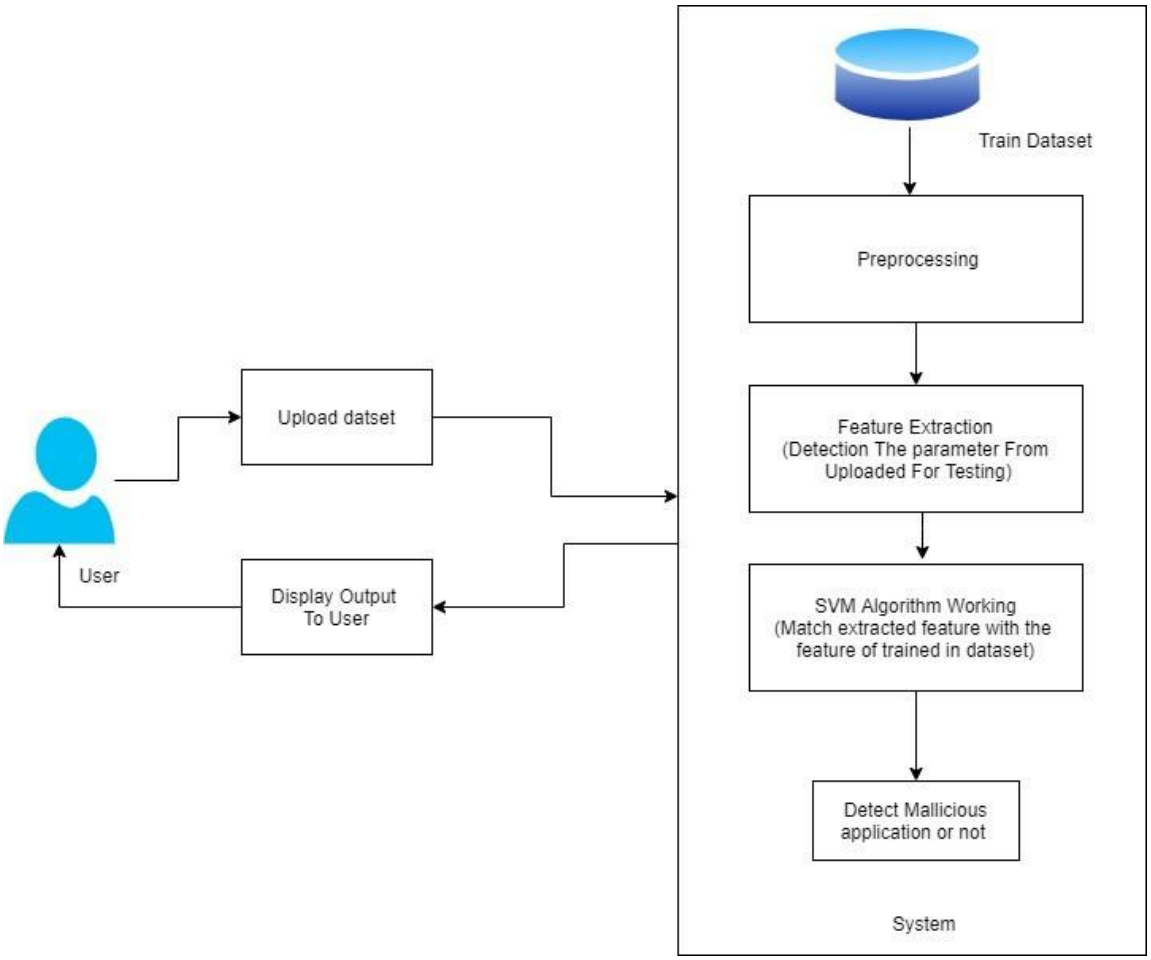

Figure 1. Proposed System architecture.

Data Collection

A data stream is a flow of packets of data. The dataset will be obtained from Kaggle. Each application will be assigned a set of attributes with binary values (0 or 1) to indicate whether it requests permission or not. A value of 0 means the application does not need authorization, whereas a value of 1 means it does [4]. Additionally, there will be a column indicating whether the application is classified as malicious or not. This helps in training as well as testing the efficiency of the algorithms.

Preprocessing

In the dataset there are some blank values other and 1 and 0. So, they need to be suitably adjusted according to the column or the whole row can be removed. Hence data is cleaned, and noisy data is removed from the dataset.

Feature Extraction

There are many attributes present in the dataset, out of these only a few attributes have significant impact on the device. These attributes must be extracted from the dataset and can be utilised as input to the algorithm to improve efficiency. After feature extraction is complete, the data is split into training data and testing data [5].

Efficiency Calculations

After training the algorithm, the test data input is given as input to the algorithm. The projected and actual results are contrasted to assess each algorithm's performance. These results are then presented in the form of graphs (Figure 2).

Module

Admin

- The admin must connect into this module with a legitimate login and password. Upon successful login, the admin gains access to various operations, including the ability to view all users and authorize them, view all e-commerce websites and authorize them, view all products and reviews, view early reviews for all products, view details of keyword searches, view product search ratios, view keyword search results, and view product review rank results.
- *User Management:* In this module, the admin is able to view the list of registered users and their details such as username, email, and address. The admin also has the authority to authorize users.
- *Chart Results:* This module allows users to view various charts and results, including the product search ratio, keyword search results, and product review rank results.
- *E-commerce User:* Within this module, there is a registration process for users. Once registered, user details are stored in the database. Users can log in after successfully registering by using their approved username and password. Operations available to users include adding products, viewing all products with reviews, viewing early product reviews, and viewing purchased transactions.
- *End User:* This module is designed for registered users. Users can log in using their authorised username and password after successfully registering. Operations available to users include managing their account, searching products by keyword, making purchases, and viewing product details.

Figure 2. Level 0 DFD.

Your Search Transactions, View

- View Charts Results.
- View all keyword search results, product review rank results, and product search ratios.

Ecommerce User

- Within this module, there is a group of users. Prior to engaging in any activities, users are required to complete a registration process. Once registered, their information is stored in the database. Upon successful registration, users are prompted to log in using their authorized credentials. Once logged in, users gain access to various functions, including adding products, viewing all products along with their reviews, browsing through early product reviews, and reviewing all completed transactions.

End User

Within this module, there is a presence of numerous users. After successfully registering, users can log in using their authorised username and password. Upon successful registration, their information is securely stored in the database. Subsequently, users are required to login using their authorized username and password [6]. Once successfully logged in, users gain access to various functionalities, including managing their account, conducting keyword-based product searches, making purchases, and viewing their search transactions (Figure 3).

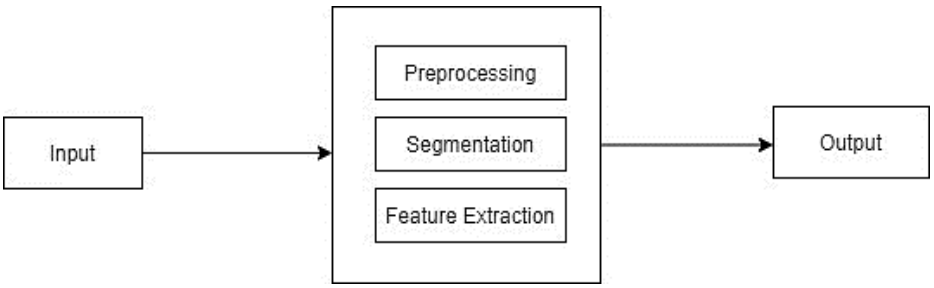

Figure 3. Internal Working layers.

LITERATURE REVIEW

1. 'A malicious application detection model to remove the influence of interference API sequence' by Tian and Huang [7]

Methodology

This paper presents a novel approach to identify Windows malicious applications through a detection model. The model captures the API call sequences during application runtime and extracts relevant features from them. These features demonstrate a strong correlation with detecting malicious attributes and exhibit minimal redundancy among each other. It is observed that API subsequences generated by normal behaviour, which might be present in malicious applications, can hinder the training process of the detector. To overcome this interference and enhance detection accuracy, a combination of VSM, K-means, and the GBDT algorithm is employed. The efficiency of this strategy in reducing the negative effects of interference API sequences and enhancing detection accuracy is supported by experimental data.

2. 'Dynamic detection of unknown malicious executables is based on API interception' by Chen and Fu [8]

Methodology

This research introduces a novel approach for dynamically detecting malicious executables on the Windows platform. The method focuses on extracting signatures from these executables and utilizes API interception techniques to uncover the behaviours of unknown malicious executables. The dynamic detection process consists of three main steps: capturing the API function call sequence of the executable, transforming the API sequence into a vector, and comparing the vector with a feature library constructed based on security policies to determine the maliciousness of the executable. Results from experiments show how well this technique works for locating unknown harmful executables.

3. 'Study on the application of Dalvik injection technique for the detection of malicious programs in Android' Li *et al.* [9]

Methodology

With the rising prevalence of computers and laptops in everyday life, there has been a continuous influx of malicious software specifically targeting smartphones. As Windows holds the largest market share among phone systems, it is confronted with a comprehensive security challenge. This article concentrates on examining the utilization of the Dalvik injection technique for detecting Windows Malicious applications. By employing the Dalvik injection technique, it becomes possible to modify

the system's Application Program Interface (API) and directly detect programs on Windows computers. By analysing the list of sensitive API calls made by potentially malicious programs, it becomes feasible to determine whether the target program is malicious or not.

4. 'Hybrids of support vector machine wrapper and filter-based framework for malware detection' by Huda *et al*. [10]

Methodology

Due to the rapid growth of windows malicious application samples, traditional detection methods need to spend a lot of time for training, a detecting method for malicious windows application based on incremental SVM was proposed to achieve incremental learning of the detection system. The method used the SVM as the classification and training algorithm and extracted sensitive permissions and APIs as application characteristics. On the basis of SVM, a dual weight function was designed to filter the historical training samples to avoid redundant samples, and the incremental learning method of SVM was implemented in combination with KKT conditions. Therefore, the training time could be reduced, and the learning efficiency of the malicious application detection system could be improved without reducing the training accuracy.

5. 'Application layer anomaly detection based on HSMM' by Bailin *et al*. [11]

Methodology

At the application layer, network-based assaults are becoming increasingly prevalent today. These attacks may not have significant malicious actions, according to observations made at the network layer and transport layer, and they produce unusual network traffic. However, traditional security techniques usually detect attacks from those two layers. Although some security techniques can detect some application layer attacks, these techniques can only detect some known attacks and these techniques cannot detect the unknown or novel attacks happened on application layer. In theory, application layer anomaly detection can detect unknown and novel attacks happened on application layer, so the research of application layer anomaly detection is very important. This paper presents a new application layer anomaly detection method which is based on HSMM. The results of the experiments reveal that this method has a high detection accuracy and a low false positive ratio.

6. 'Anomaly detection of malicious users' behaviours for web applications based on web logs' by Gao *et al*. [12]

Abstract

Web application security concerns have been more urgent as the number of web applications keeps expanding. Traditional intrusion detection systems primarily focus on identifying cyberattacks based on individual requests, rather than analysing user behaviours. Additionally, these systems are limited in their ability to protect web applications against unknown or zero-day attacks. To address this, we propose an approach that involves analysing web server logs and categorizing user behaviours as either normal or malicious. By leveraging the characteristics of web resources to define user behaviours, we achieve higher accuracy rates and lower false alarm rates in intrusion detection.

7. 'Real-time detection system against malicious tools by monitoring DLL on client computers' by Matsuda *et al*. [13]

Methodology

The targeted attacks cause severe damage worldwide. Detecting targeted attacks is challenging because the attack methods are very sophisticated. Network-based solutions such as Firewall, Proxy Server, and Intrusion Detection System (IDS) have been widely used. In addition to this, recently, detection methods for malicious programs by monitoring behaviour on the endpoints called Endpoint Detection and Response (EDR) have been proposed. Also, some researchers introduce detection methods using DLLs by analysing suspicious files on the sandbox, such as Cuckoo. Using Cuckoo is one of the solutions for analysing files that are already identified as malicious. In this study, we provide a real-time strategy for detecting malicious software that makes use of DLL data gathered by

System Monitor (Sysmon), a free logging tool offered by Microsoft. Our technique's main objective is to spot fresh malicious processes in the real-world setting. We focus on DLLs commonly loaded by malicious tools regardless of the environments, then propose "the common DLL lists" for detection. Additionally, we offer a detection method that is applicable and uses Elastic Stack for Security Information and Event Management (SIEM). By using Elastic Stack, DLL information loaded on computers can be uniformly monitored and enables real-time detection by comparing logs with the common DLL lists. We evaluate the effectivity of the proposed method using four free malicious tools introduced by US-CERT: China Chopper, Mimi Katz, PowerShell Empire, and HUC Packet Transmitter. As a result, our method detected China Chopper, Mimi Katz, PowerShell Empire with 100 false positive occurred for HUC Packet Transmitter, and false positive rate was 0.55 lists are useful for detecting malicious tools in real time using Elastic Stack.

8. 'Mapldroid: Malicious android application detection based on naive bayes using multiple' by Bhat *et al.* [14]

Methodology

Android is currently the most popular operating system for mobile devices in the market. Android devices are being used by every other person for everyday life activities and it has become a centre for storing personal information. Because of these reasons it attracts many hackers, who develop malicious software for attacking the platform; thus, a technique that can effectively prevent the system from malware attacks is required. This research paper presents Map Droid, a novel technique for detecting malware applications specifically designed for the Android platform. The proposed technique statically analyses the application files using features which are extracted from the manifest file.

A Naive Bayes-based supervised learning model is utilized to classify the applications as either benign or malicious in this study. Map Droid demonstrated an impressive recall score of 99.12%.

CONCLUSION

The dynamic detection of applications based on Hook technology is used to acquire the system API call sequence after a thorough study of the application monitoring principle. The feature selection technique is expanded, and the most beneficial detecting features are extracted on the basis of the already available research. A detection model is created and put into practise with the goal of addressing the issue of interference in the application API sequence. A plan for removing the interference API sequence utilising the vector space model is put forth.

REFERENCES

1. Karbab EB, Debbabi M, Derhab A, Mouheb D. MalDozer: Automatic framework for android malware detection using deep learning. Digit Investig. 2018 Mar 1; 24: S48–59.
2. Ferrante A, Medvet E, Mercaldo F, Milosevic J, Visaggio CA. Spotting the malicious moment: Characterizing malware behavior using dynamic features. In 2016 IEEE 11th International Conference on Availability, Reliability and Security (ARES). 2016 Aug 31; 372–381.
3. Canfora G, Medvet E, Mercaldo F, Visaggio CA. Detecting android malware using sequences of system calls. In Proceedings of the 3rd International Workshop on Software Development Lifecycle for Mobile. 2015 Aug 31; 13–20.
4. Enck W, Gilbert P, Han S, Tendulkar V, Chun BG, Cox LP, Jung J, McDaniel P, Sheth AN. Taintdroid: an information-flow tracking system for real time privacy monitoring on smartphones. ACM Trans Comput Syst (TOCS). 2014 Jun 1; 32(2): 1–29.
4. Ferrante A, Medvet E, Mercaldo F, Milosevic J, Visaggio CA. Spotting the malicious moment: Characterizing malware behavior using dynamic features. In 2016 11th International Conference on Availability, Reliability and Security (ARES). 2016 Aug 31; 372–381.
5. Ni Z, Yang M, Ling Z, Wu JN, Luo J. Real-time detection of malicious behavior in android apps. In 2016 IEEE International Conference on Advanced Cloud and Big Data (CBD). 2016 Aug 13; 221–227.

6. Feldman S, Stadther D, Wang B. Manilyzer: automated android malware detection through manifest analysis. In 2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems. 2014 Oct 28; 767–772.
7. Tian P, Huang X. A malicious application detection model to remove the influence of interference API sequence. In 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS). 2017 Nov 24; 501–505.
8. Chen F, Fu Y. Dynamic detection of unknown malicious executables base on API interception. In 2009 IEEE 1st International Workshop on Database Technology and Applications. 2009 Apr 25; 329–332.
9. Li Y, Fang J, Liu C, Liu M, Wu S. Study on the application of Dalvik injection technique for the detection of malicious programs in Android. In 2015 IEEE 5th International Conference on Electronics Information and Emergency Communication. 2015 May 14; 309–312.
10. Huda S, Abawajy J, Alazab M, Abdollalihian M, Islam R, Yearwood J. Hybrids of support vector machine wrapper and filter based framework for malware detection. Future Gener Comput Syst. 2016 Feb 1; 55: 376–90.
11. Bailin X, Shunzheng Y, Tao W. Application layer anomaly detection based on hsmm. In 2010 International Forum on Information Technology and Applications. 2010 Jul 16; 2: 411–414.
12. Gao Y, Ma Y, Li D. Anomaly detection of malicious users' behaviors for web applications based on web logs. In 2017 IEEE 17th International Conference on Communication Technology (ICCT). 2017 Oct 27; 1352–1355.
13. Matsuda W, Fujimoto M, Mitsunaga T. Real-time detection system against malicious tools by monitoring DLL on client computers. In 2019 IEEE Conference on Application, Information and Network Security (AINS). 2019 Nov 19; 36–41.
14. Bhat P, Dutta K, Singh S. Mapldroid: Malicious android application detection based on naive bayes using multiple. In 2019 IEEE 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT). 2019 Sep 28; 49–54.

