MALWARE IDENTIFICATION USING BEHAVIORAL TRACKING

***Ms. Neethu D S,** Department of Computer Science, Research Scholar, Shri Venkateshwara, University,India

****Dr.Ajay Sharma,** Department of Computer Science, Associate Professor, Shri Venkateshwara University, India

Abstract

Over time, technology has developed to the point where it now significantly influences many facets of our lives. This remarkable expansion has brought about a number of advantages and opportunities, but it has also inadvertently accelerated the transmission of malware and increased the number of malware attacks. The availability of automated and advanced code generators, obfuscators, and packers has led to an increase in complex and sophisticated malware. Additionally, in order to create new malware variants and evade detection techniques, malware makers purposefully include duplicate or needless API calls into instances. As a result, malware identification has become a significant problem for both consumers and companies.

It is not computationally feasible to manually examine each suspicious data due to the growing complexity and amount of malware samples. Machine learning techniques have shown a great deal of promise for automatically identifying dangerous samples by extracting important information from large feature sets of both benign and malicious samples. Therefore, the objective of this study is to create reliable machine learning-based malware detection techniques. The first technique extracts API call sequences by dynamically analyzing malware and benign samples. Purified API call sequences are obtained by removing unnecessary and noisy API calls from them. From these purified sequences, frequent sequential patterns are produced to provide a feature vector.

Therefore, it uses feature selection approaches to reduce the size of the feature vectors and produce an essential feature set that can improve machine learning algorithms' detection skills. A comprehensive collection of tests employing different datasets and machine learning techniques is also used to assess this approach. According to the experimental findings, the hybrid feature selection method significantly decreased the size of the input feature set and enhanced machine learning algorithms' ability while cutting down on training time.

Introduction

Modern technological developments like cloud computing, big data, artificial intelligence (AI), and the Internet of Things (IoT) are making it easier to automate information systems and increase their operational and computational efficiency. The phrase "Industry 4.0" is frequently used for describing this stage of automation and change. These changes have increased the amount of information in information systems, even if they provide many practical advantages to both individuals and enterprises. Numerous security risks, including financial fraud, denial of service assaults, malware attacks, unlawful resource access, and information theft, are brought on by this excess of information. In recent years, malware has emerged as one of the biggest risks to information system security. These malicious computer programs are made to interfere with regular system functions. They engage in a wide range of fraudulent acts, such as gaining illegal access to system resources, stealing private data, disrupting communication networks, initiating denial-of-service attacks, causing financial losses, and more.

Following encouraging outcomes from API call sequences, the second method aims to investigate more API call types for malware detection. Using three different types of API calls—API call usage, API call frequency, and API call sequences—it generates several feature sets and uses machine learning methods to assess each feature set's effectiveness. These feature sets are then combined to create an API integrated feature set, which is a more robust feature set. Additionally, it uses the Term Frequency-Inverse Document Frequency approach to supplement the feature values with useful information. The API integrated feature set outperformed other feature sets during testing, although it was hindered by the feature vector's high dimensionality.

The first malicious program developed in 1972, and malware has been a part of systems for computers for many years. They are becoming much more numerous and sophisticated these days due to the quick development of information technology. The availability of automated malware generation tools has also greatly expanded their complexity and volume, as malware creators frequently employ these tools to create new or unidentified malware 2 types that can take advantage of information system weaknesses. A frightening pace of malware development has been facilitated by these techniques.

Behavior-based Malware Detection Methods

Methods of Behavior-based Malware Detection These methods use dynamic analysis to extract a program's behavioral properties. By running a program in a separate environment, these techniques can watch how it behaves during runtime and extract features such as memory utilization, registry files, changed networks, API calls, processes, threads, and mutexes. It uses genetic algorithms to create synthetic malware patterns in order to enhance the base of malware samples for the identification of previously unobserved malicious behavior, all with the aid of automated tools. Machine learning algorithms are used to evaluate this method's performance, and the findings showed that the chosen features improved the algorithms' performance while also greatly cutting down on the classification model's training and testing times.

These elements help to comprehend a program's true behavior and provide comprehensive information about its workflow. Behavior-based malware detection techniques have been shown to be reliable and successful in detecting the newest, encrypted, obfuscated, and undetected malware. A brief overview of the several dynamic properties that these methods use is given in the part below:

Because they offer comprehensive details about a program's workflow, Application Programming Interface (API) calls are considered to be among its most representative features. They are a collection of operations or processes that allow user programs and operating systems to communicate while they are running. Through file processing, network operations, registry alterations, memory changes, and other means, they provide different degrees of communication between applications and operating systems. The detection of any harmful activity in the program, such as unauthorized access to private resources or odd interactions with system components, can be aided by monitoring such functions.

In the following forms, they are widely used in dynamic analysis-based malware detection approaches and are thought to be the most accurate and discriminative dynamic features for differentiating between malware and benign samples:

i. Activities on File Systems:

Various file activities carried out by the program while it was running are represented by the file system. These operations cover a variety of file system tasks, including adding new files, editing and removing old ones, transferring data between locations, and more.CreateFile(), DeleteFile(), MoveFile(), CopyFile(), GetFileSize(), and other file system activity-based API methods are provided.

ii. Changes to Registry:

Important details about computer programs installed in the operating system, such as user preferences and hardware and software configuration settings, are stored in Windows registry files. Along with their key values, they indicate the different registry operations that the application performs while it is running, such as creating, deleting, altering, and so forth. RegCloseKey(), RegCreateKey(), RegDeleteKey(), RegDeleteTree(), RegDeleteValue(), RegDisablePredefinedCache(), and others are some of the crucial registry methods. These features aid in identifying malicious activity by reflecting registry modifications made to the program during examination.

iii. Network Activities:

It keeps an eye on all of the network operations that the program carries out while it is running in order to identify any malicious activity, including unauthorized domain name requests, dubious IP addresses, network protocols, monitoring ports, and other actions. HttpSendRequest(), HttpOpenRequest(), HttpEndRequest(), InternetConnectA(), and others are among the API functions used for network operations. iv. Memory Analysis: It keeps track of various memory operations carried out by the programs in order to comprehend how they behave. These operations include memory allocations, memory accesses, memory dump analysis, and other memory-related tasks. However, because they require a lot of memory and processing power to analyze the executables, these methods are computationally more expensive than static analysis-based ones. Furthermore, it is difficult to examine these executables without the assistance of a human.

1. An Overview of Behavior-based Malware Identification

- The meaning and significance of behavior-based approaches.
- comparison with heuristic and signature-based detection techniques.
- Behavior-based detection challenges.

2. Essential Elements utilized in patterns and sequences of Behavior-based Detection API calls.

- registry and file system operations.
- Patterns of communication and network behavior.
- abnormalities in CPU and memory use.

3. Methods of Dynamic Analysis

- emulation-based malware analysis and sandboxing.
- Malware behavior is monitored during runtime.
- dynamic analysis's benefits and drawbacks.

4. Detection Based on Behavior and Machine Learning

- Techniques for feature extraction and selection.
- algorithms for classification (e.g., Neural Networks, SVM, Decision Trees).
- designing features with behavioral data.

5. Hybrid Techniques to Improve Detection

- combining static and behavior-based analysis methods.
- combining signature-based techniques with machine learning.
- hybrid system case studies.

6. Strategies for Evasion and Countermeasures

- Malware frequently employs evasion tactics, such as code obfuscation and anti-sandboxing.
- Techniques for identifying and combating evasion.
- Adaptive malware detection advances.

7. Practical Uses

- Examples of behavior-based malware detection programs are CrowdStrike and Windows Defender.
- Use cases in threat intelligence and business security.

8. Sophisticated Methods of Behavioral Analysis

RNNs and LSTMs are used in sequence modeling.
behavioral representations based on graphs.
Spatial and temporal investigation of malware activity.

9. Creation and Administration of Behavioral Datasets

- techniques for gathering and annotating behavioral data.
- issues with labeling and dataset scalability.
- datasets of behavioral malware that are openly accessible.

10. Behavior-based Detection Evaluation Metrics

- F1 score, recall, accuracy, and precision.
- examination of both false negatives and false positives.
- benchmarks for malware detection performance.

11. Developing Patterns for Behavior-based Malware Identification

- Deep learning and AI's role.
- developments in behavioral analysis in real time.
- difficulties using behavior-based techniques to identify novel malware strains.

Future Scope

The malware detection techniques described in this thesis are intended to handle a number of malware-related problems. The current study has the potential to significantly advance the field of malware detection, as evidenced by a thorough examination of the experimental data. This work, however, can be expanded to investigate the categorization of malware into distinct family groups. Malware analysts are better able to identify dangerous patterns and create countermeasures when they group malware samples based on shared traits. Furthermore, classical machine learning methods are the main tool used in this work to detect malware. Deep learning techniques may be used in future studies to automatically extract features from samples for malware categorization and detection.

Conclusion

Ultimately, this thesis advances our knowledge of malware behavior by introducing dynamic analysis-based malware detection techniques. Utilizing API calls recovered from both benign and malicious samples, the proposed techniques thoroughly examine these properties to ascertain how they contribute to malware identification. Additionally, these approaches apply a variety of feature selection strategies to extract reliable and pertinent information, which are then fed into a number of machine learning algorithms. The outcomes of the experiment demonstrated that ML algorithms can perform remarkably well with these characteristics. Ransomware is a particular kind of malware that is used to further evaluate the detection performance of the suggested approaches. Additionally, this thesis investigates the application of evolutionary algorithms to identify hidden virus behavior.

References

- Behavioral analysis. (2014). *Android Malware and Analysis*, 112-149. https://doi.org/10.1201/b17598-12
- Chawla, K. (2023). A study of malware analysis and malware detection methods in cyber-security. *International Journal of Scientific Engineering and Research*, *11*(5), 51-56. https://doi.org/10.70729/se23515123724
- Jayaswal, A. (n.d.). Malware detection on virtual environments based on behavioral anomalies. https://doi.org/10.17918/00000066
- Noorani, M. (n.d.). On the detection of malware on virtual assistants based on behavioral anomalies. https://doi.org/10.17918/ckv0-8x43
- Saxena, S. (n.d.). Malware detection using behavioral Whitelisting of software. https://doi.org/10.17918/a4gh-dw65