Naif Arab University for Security Sciences
Journal of Information Security and Cybercrimes Research
مجلة بحوث أمن المعلومات والجرائم السيبرانية
https://journals.nauss.edu.sa/index.php/JISCR

JISCR

Enhancing Malware Detection by Integrating Machine Learning with Cuckoo Sandbox

Amaal F. Alshmarni*, Mohammed A. Alliheedi
Department of Computer Science, Al-Baha University, Al Bahah, Saudi Arabia.

Received 26 Jan. 2024; Accepted 22 May. 2024; Available Online 20 Jun. 2024

Abstract

In this work, two categories of deep learning and conventional machine learning were used to classify malware using a dataset of all possible API call sequences. Specifically, the objective was to determine the best strategy to tackle the ever-rising menace as malware becomes more complex. A new dataset was created employing Cuckoo Sandbox, where API call sequences originating from both benign and malware samples were recorded. The performance of these algorithms was benchmarked and tested using this dataset, which includes SVM, RF, KNN, XGB, GBC, CNN, and RNN. The study established that both deep learning and conventional machine learning algorithms provided high accuracy above 90%. Specifically, the recurrent neural networks (RNNs) demonstrated high accuracy rates ranging from 95% to 99%. These results are highly indicative of deep learning, especially RNN, as a promising approach to improving the effectiveness of malware detection. The data obtained from dynamic analysis, when integrated into a database, serves as a more reliable source for training and testing of such models, and can improve the model's ability to identify new threats posed by malware. Thus, this work is salient in enhancing the development of new approaches to fight malware that constantly evolve in the modern world.

I. Introduction

In the digital age, the ceaseless evolution of malware remains an omnipresent threat to individuals, organizations, and society at large. Malicious software, or malware, has evolved to become highly sophisticated, elusive, and continually adapts to evade traditional detection mechanisms. Amid this relentless onslaught, the fusion of deep learning techniques with the dynamic analysis capabilities of Cuckoo Sandbox emerges as a beacon of hope in the field of cybersecurity. This paper embarks on a transformative journey by introducing a groundbreaking malware dataset, meticulously curated through dynamic analysis using Cuckoo Sandbox, to drive innovation in malware detection.

Deep learning has emerged as a powerful force in various domains, including computer vision, natural language processing, and speech recognition. Its application to malware detection is compelling, as it enables the automatic extraction of intricate features and behavioral patterns exhibited by malware. However, the performance of deep learning

Keywords: Malware analysis, Machine learning, Deep learning, Malware dataset, Cuckoo sandbox, API call sequencing.

Production and hosting by NAUSS

* Corresponding Author: Amaal F. Alshmarni
Email: amaalalshmarni@hotmail.com
doi: 10.26735/WZNG1384

models is inexorably tied to the quality and diversity of the data on which they are trained. Conventional malware datasets often fall short in providing the breadth and depth required to effectively combat emerging malware threats.

To bridge this gap, this paper pioneers a methodology that leverages the dynamic analysis capabilities of Cuckoo Sandbox, a widely adopted and versatile malware analysis tool. Cuckoo Sandbox simulates the execution of suspicious files within a controlled environment [1], observing their behavior and interactions with the system. This dynamic approach offers an unparalleled opportunity to capture the nuanced tactics and evasion strategies employed by malware, making it an ideal partner for deep learning-based malware detection. The focal point of this paper revolves around the creation of a comprehensive and timely malware dataset, meticulously constructed through the detailed analysis of malware samples using Cuckoo Sandbox. Since 1988, computer security breaches have increased dramatically. All malicious software that infiltrates a computer system without the user's knowledge is referred to as "malware". The terms "malicious" and "software" were combined to create this term. Malware is a significant concern in today's technological environment because it continues to increase in size and complexity. The proliferation of malware-infected websites is on the rise, posing a significant challenge to organizations that attempt to mitigate the problem. The situation is becoming increasingly concerning and can escalate beyond manageable levels. Most malware infects computers when downloading data from the Internet [2].

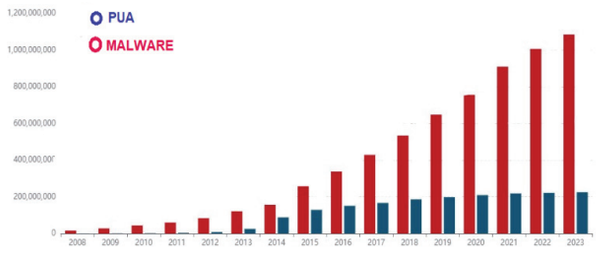

Fig. 1. Number of Malwares Recent Year [3].

Statistics show that many malwares and potentially unwanted applications (PUAs) have been identified in recent years see Fig. 1 [3].

In recent decades, there has been a significant increase in the development of methods for detecting malware and its components. Malware detection encompasses the steps taken to identify malicious software, utilizing either signature-based or anomaly-based approaches [4]. Signature-based detection relies on a database of known malware signatures, comparing suspicious patterns against this database for identification. While accurate for known malware, this method consumes system resources and is ineffective against zero-day attacks [4]. Anomaly-based detection, on the other hand, can be particularly useful in identifying novel or previously unseen malware by flagging deviations from normal behavior patterns. However, a major challenge with anomaly-based detection is the high rate of false positives [4].

Effective computer system security necessitates the identification and understanding of potential malware threats. Malware analysis aims to uncover the origin, functionality, and impact of malware on a system, aiding security analysts in identifying vulnerabilities. Malware analysis techniques can be categorized as static or dynamic [5]. Static analysis involves examining the code of suspected malware without executing it [6], while dynamic analysis involves running the malware in a controlled environment to observe its behavior [7].

Deep learning, a subset of machine learning, has emerged as a promising approach for malware detection. By learning abstract representations of data through network layers, deep learning can identify hidden patterns and characteristics in various data types, including images, sounds, and text [8].

Several studies have explored the application of machine learning and deep learning in malware detection as shown in Table I. Patil et al. [9] evaluated various algorithms, demonstrating the superior performance of deep learning, particularly Convo-

lutional Neural Networks (CNNs), which achieved 98.4% accuracy. Singh and Singh [10] proposed a behavior-based method using dynamic analysis in the Cuckoo sandbox, achieving a high detection accuracy of 99.54%, but with potential overfitting issues. Lajevardi et al. [11] introduced a dynamic approach using system call data, achieving 98.2% accuracy, while Catak et al. [12] utilized LSTM and TF-IDF models with a focus on API call sequences, achieving 95% accuracy. Liu et al. [13] proposed a method using graph convolutional networks (GCNs) on API call sequences, achieving high accuracy up to 98%, but potentially overlooking the benefits of integrating multiple AI techniques.

TABLE I

Related Works

Author	Pros and Cons
Patil et al. [9]	Pros: RF, SVM, CNNs utilized. Cons: lacks sufficient details on data.
Singh and Singh. [10]	Pros: Dynamic analysis for runtime features. Cons: Lacks thorough analysis of potential overfitting issues given the high accuracy.
Lajevardi et al. [11]	Pros: Dynamic behavior-based malware detection utilizes system call data and control dependency sequences. Cons: Lacks comprehensive details on dataset diversity and representativeness.
Catak et al.[12]	Pos: Employed LSTM and TF-IDF for malware behavior detection from API call sequences. Cons: Limiting the approach to LSTM for malware classification may miss out on potential benefits offered by ensemble methods.
Liu et al. [13]	Pos: GCNs classify malware via API call sequences, using directed cycle graphs, Markov chain, and PCA for robust detection. Cons: Overlooking benefits of integrating diverse AI techniques by solely focusing on graph convolutional networks for malware classification.

In this work, we focused on dynamic malware analysis and presented the following contributions:

1) The creation of a novel behavioral dataset comprising API call sequences extracted from both benign and malware files.
2) An evaluation of the performance of both machine learning and deep learning algorithms to demonstrate their effectiveness in malware detection.
3) Achieving a high level of accuracy in all algorithms, surpassing 90%.

The rest of this paper is organized as follows: The methodology of our work is described. Then, the results of this study are detailed, followed by a discussion of the results. Lastly, a section consisting of the conclusion and future work of this study is presented.

II. Methodology

This section will detail the data collection and preprocessing steps, including the sources of malware and benign software (goodware) samples used in the study, and the methods employed to extract relevant features from these samples.

We collected 2576 malware samples [14] and 1080 goodware samples [15]. These samples were analyzed using Cuckoo Sandbox, which executed the malware in a controlled environment and recorded all activities. Cuckoo Sandbox generated detailed reports including information on file type, size, network traffic, system activity, and behavior. These reports were exported and used to create a new malware dataset.

All 3656 exported reports were then processed. Relevant features were extracted from the reports using JavaScript, and the resulting dataset was converted into a format suitable for machine learning algorithms. Before these raw features could be used for training and testing the classification algorithm, they were categorized and cleaned.

All completed JSON behavior reports were stored in a local directory. The "Behaviors" object within each JSON report contained all relevant behavioral features. Our focus was on extracting API call sequences from both malware and goodware files. Since the raw API call features in JSON format were incompatible with deep learning techniques,

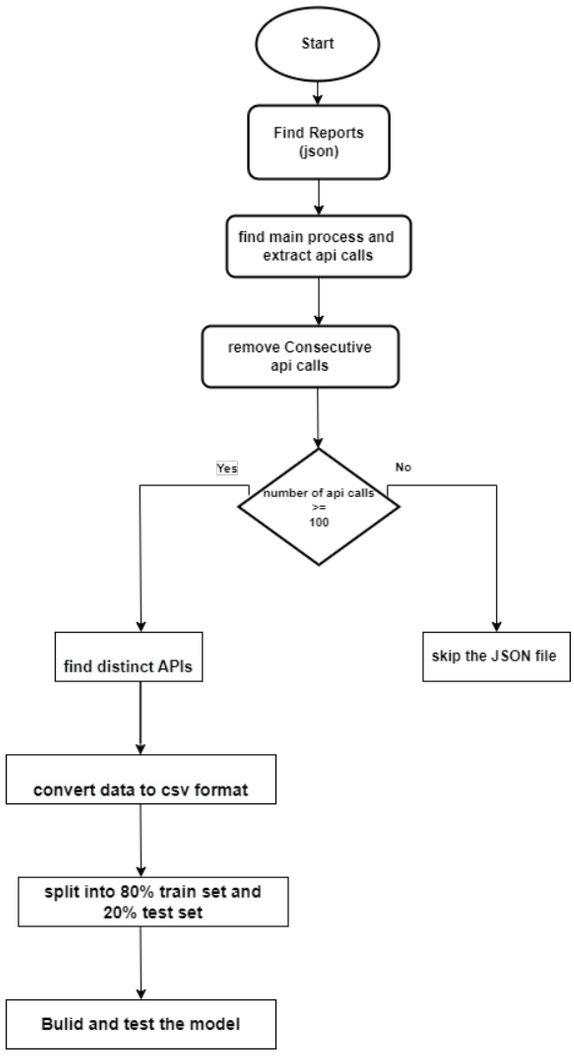

Fig. 2. Pre-processing JSON behavioral reports.

they underwent a two-stage conversion process:

1) Stage One: The JSON files were processed to generate a numerical representation of each API call feature.
2) Stage Two: All JSON reports were processed to generate a comma-separated value (CSV) file.

After extracting the data, we preprocessed it to handle missing values and address the issue of unbalanced data. This research utilized deep learning algorithms, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to detect malware. Additionally, we compared the performance of these deep learning algorithms with traditional machine learning algorithms.

A. Deep Learning Algorithms

This section will discuss the deep learning models employed in this research for malware detection. CNNs were used to detect malware by processing data through convolutional, pooling,

Convolutional Neural Networks (CNNs) distinguish input images and assign importance to various aspects [16], utilizing convolutional, pooling, and fully connected layers. Recurrent Neural Networks (RNNs) process sequential data, retaining information over time with memory, hidden layers, and weights [17]. This memory helps categorize data based on previous time steps, creating a linear graph structure.

B. Machine Learning Algorithms

The results of the deep learning algorithms were compared with several traditional machine learning algorithms: Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Gradient Boosting (GB), and XGBoost. GridSearchCV was used to perform a grid search across a defined hyperparameter space. The optimal estimator found during the grid search was used to fit the model to the complete training set and generate predictions for the test set. The model's effectiveness was evaluated using a classification report.

III. Results

This section presents the results of this study. We first describe the metrics used to evaluate the performance of the machine learning models. These metrics, including accuracy, allow us to compare various ML techniques and determine their relative superiority. Accuracy is calculated as the ratio of correct predictions across all samples to the total sample size [16]. The accuracy calculation formula is:

$$\frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

- Precision is defined as the proportion of predicted positive samples that are positive. The following is the precision formula:

$$\frac{TP}{TP + FP} \quad (2)$$

- The True Positive Rate is another term for recall (TPR).) The recall formula is as follows:

$$\frac{TP}{TP + FN} \quad (3)$$

- F1-Score is a unique statistic for assessing a model's performance that combines a harmonic measure of recall and precision:

$$2 \times \frac{recall \times precision}{precision + recall} \quad (4)$$

- Area under the curve: The area under the curve (AUC), also known as the area under the ROC curve, summarizes the performance of a binary classifier at various thresholds. Finding the value involves taking the ROC and calculating its area. The AUCs can be anywhere from 0 to 1. (If our AUC score increases it means that our classifier is more accurate). A higher AUC score indicates that our classifier is more accurate at predicting positive and negative examples.

A. Experiment 1

In the previous section we talked about how we created a new dataset that contained 2576 malware and 1080 goodware. We then applied some algorithms of deep learning, namely CNN and RNN, as well as machine learning, such as SVM and KNN, on the dataset we created. We repeated the following experiment on all the aforementioned algorithms:

- Randomly select training data 80% and test data 20%.
- Optimal hyperparameters using grid search in machine learning techniques.

The results were as shown in Table II. The effectiveness of deep learning algorithms is tested using conventional machine learning techniques including XGB, GBC, KNN, RF, and SVM. High degrees of accuracy have been attained by all the machine learning and deep learning techniques where it was the highest accuracy RNN and among them all, SVM had nearly the lowest accuracy.

B. Experiment 2

To enhance the performance of the machine learning algorithms, we utilized a different dataset from Kaggle [1]. This dataset contains 1,079 API call sequences for goodware and 42,797 for malware. Each sequence consists of the first 100 consecutive, non-repeated API calls connected

TABLE II
RESULTS OF EXPERIMENT 1

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	ROC-AUC score (%)
CNN	98	98	98	98	97
RNN	99	98	99	99	98
SVM	91	91	91	91	96
KNN	92	92	92	92	95
XGB	93	92	93	92	97
RF	95	95	95	95	98
GBC	95	95	96	95	96

TABLE III
RESULTS OF EXPERIMENT 2

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	ROC-AUC score (%)
CNN	98	98	98	98	98
RNN	99	99	99	99	99
SVM	95	96	94	95	96
KNN	99	99	99	99	98
XGB	96	97	95	96	97
RF	98	98	98	98	98
GBC	99	99	98	99	99

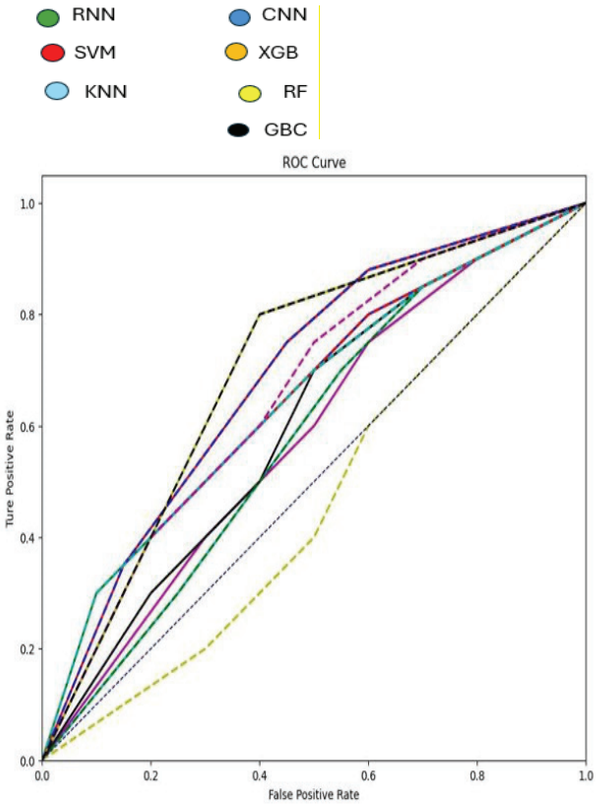

Fig. 3. ROC Curve.

to the parent process, as derived from the 'calls' portion of Cuckoo Sandbox reports [18]. We selected this larger dataset to assess the efficacy and accuracy of all algorithms, as it provided a more representative distribution of classes compared to our previous dataset. Table III demonstrates that deep learning algorithms maintained consistent results compared to Experiment 1. However, machine learning algorithms showed improved performance with this larger dataset, likely due to the increased exposure to a wider range of samples.

Fig. 3 shows the ROC curve to assess the performance of our classification model and determines its effectiveness in distinguishing between the positive and negative outcomes.

IV. DISCUSSION

This section discusses the results of this study. Deep learning, utilizing neural networks, has opened new avenues for malware detection. Our study developed and evaluated a deep learning-based malware detection system using a large dataset, achieving impressive results. The deep learning model, specifically CNN and RNN due to their ability to handle time series data, outperformed traditional machine learning methods in terms of accuracy, precision, recall, and F1-score.

These findings indicate that various machine learning and deep learning techniques can effectively identify malware. However, careful tuning of hyperparameters, such as batch size and the use of optimizers like Adam, is crucial for optimal performance.

Despite its high performance, our approach has limitations. The model heavily relies on a good and updated dataset, which is challenging to maintain due to the dynamic nature of malware. This may

hinder its ability to recognize new threats or adapt to evolving evasion techniques. Additionally, deep learning models demand substantial computational resources for training and inference, and their "black box" nature makes them difficult to interpret.

To address these limitations, countermeasures such as data augmentation, ensemble methods, regularization, adversarial training, and explainable AI (XAI) can be employed. These approaches aim to enhance model performance and reliability in malware detection. Table IV summarizes the observed drawbacks and advantages of the algorithms used in this study.

TABLE IV
ALGORITHMS CONS AND PROS

SVM	Cons: High computational cost and sensitivity to noise in data. Pros: Robust to overfitting
KNN	Cons: High memory usage and computationally expensive during prediction. Pros: Easily adaptable to new data
XG-Boost	Cons: Prone to overfitting and requires careful hyperparameter tuning. Pros: Handles missing data well.
RF	Cons: Slow on large datasets and lacks interpretability with many trees. Pros: Provides feature importance
GBM	Cons: More prone to overfitting, longer training times, and requires meticulous hyperparameter tuning. Pros: improves model performance through iterative boosting.
CNN	Cons: Require large datasets and struggle with capturing long-range dependencies in sequences. Pros: Ability to capture local patterns through convolutional layers.
RNN	Cons: Require large datasets. Pros: Suitable for sequential data

V. Conclusion And Future Work

In this study, we employed deep learning-based malware detection and evaluated its effectiveness using various malware sample datasets. This section summarizes the key findings and discusses potential future research to enhance the accuracy and reliability of deep learning-based malware detection.

The project's primary objective was to utilize extracted API call sequences from our created dataset and apply deep learning and machine learning algorithms for malware detection. We employed various machine learning algorithms, including SVM, KNN, RF, GBC, and XGBoost, achieving high accuracy despite their limited reputation for effectiveness on such data. Additionally, we compared and evaluated the performance of two deep learning algorithms, CNN and RNN, to validate the effectiveness of deep learning techniques. Both algorithms demonstrated high accuracy, with RNN outperforming CNN, reaching an accuracy of 99%.

Future work will focus on expanding the dataset to include a wider variety of malware for multiclass classification problems. Additionally, we will address the limitations mentioned in the discussion section, aiming to improve the model's robustness and adaptability to new threats.

Funding

This article did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of Interest

Authors declare that they have no conflict of interest.

References

[1] M. Ali, S. Shiaeles, N. Clarke, and D. Kontogeorgis, "A proactive malicious software identification approach for digital forensic examiners," Journal of Information Security and Applications, vol. 47, pp. 139–155, Aug. 2019, doi:

https://doi.org/10.1016/j.jisa.2019.04.013.

[2] C. Li, Q. Lv, N. Li, Y. Wang, D. Sun, and Y. Qiao, "A novel deep framework for dynamic malware detection based on API sequence intrinsic features," Computers & Security, vol. 116, p. 102686, May 2022, doi: https://doi.org/10.1016/j.cose.2022.102686.

[3] AV-TEST, "Malware Statistics & Trends Report," Av-test.org, Apr. 25, 2019. [Online]. Available: https://www.av-test.org/en/statistics/malware/. [Accessed: 20-May-2022].

[4] M. Sahin and S. Bahtiyar, "A Survey on Malware Detection with Deep Learning," in Proc. 13th Int. Conf. Security of Information and Networks, Nov. 2020, doi: https://doi.org/10.1145/3433174.3433609.

[5] A. Amira, A. Derhab, E. B. Karbab, and O. Nouali, "A survey of malware analysis using community detection algorithms," ACM Computing Surveys, vol. 56, no. 2, pp. 1–29, Sep. 2023, doi: https://doi.org/10.1145/3610223.

[6] S. Talukder and Z. Talukder, "A Survey on Malware Detection and Analysis Tools," Int. J. Network Security & Its Applications, vol. 12, no. 2, pp. 37–57, Mar. 2020, doi: https://doi.org/10.5121/ijnsa.2020.12203.

[7] M. Y. Wong, M. Landen, M. Antonakakis, D. M. Blough, E. M. Redmiles, and M. Ahamad, "An Inside Look into the Practice of Malware Analysis," in Proc. 2021 ACM SIGSAC Conf. Computer and Communications Security, Nov. 2021, doi: https://doi.org/10.1145/3460120.3484759.

[8] B. Yadav and S. Tokekar, "Deep Learning in Malware Identification and Classification," in Malware Analysis Using Artificial Intelligence and Deep Learning, M. Stamp, M. Alazab, and A. Shalaginov, Eds. Cham: Springer, 2021, doi: https://doi.org/10.1007/978-3-030-62582-5_6.

[9] R. Patil and W. Deng, "Malware Analysis using Machine Learning and Deep Learning techniques," in Proc. 2020 SoutheastCon, Mar. 2020, doi: https://doi.org/10.1109/southeastcon44009.2020.9368268.

[10] J. Singh and J. Singh, "Detection of malicious software by analyzing the behavioral artifacts using machine learning algorithms," Information and Software Technology, vol. 121, p. 106273, May 2020, doi: https://doi.org/10.1016/j.infsof.2020.106273.

[11] A. M. Lajevardi, S. Parsa, and M. J. Amiri, "Markhor: malware detection using fuzzy similarity of system call dependency sequences," J. Computer Virology and Hacking Techniques, Apr. 2021, doi: https://doi.org/10.1007/s11416-021-00383-1.

[12] F. O. Catak, A. F. Yazı, O. Elezaj, and J. Ahmed, "Deep learning based Sequential model for malware analysis using Windows exe API Calls," PeerJ Computer Science, vol. 6, p. e285, Jul. 2020, doi: https://doi.org/10.7717/peerj-cs.285.

[13] S. Li, Q. Zhou, R. Zhou, and Q. Lv, "Intelligent malware detection based on graph convolutional network," The Journal of Supercomputing, Aug. 2021, doi: https://doi.org/10.1007/s11227-021-04020-y.

[14] "MalwareBazaar | Malware sample exchange," bazaar.abuse.ch. [Online]. Available: https://bazaar.abuse.ch/. [Accessed: 10-Feb-2023].

[15] T. Copyright, "Latest entries - The Portable Freeware Collection," www.portablefreeware.com. [Online]. Available: https://www.portablefreeware.com/. [Accessed: 10-Feb-2023].

[16] J. Palša et al., "MLMD—A Malware-Detecting Antivirus Tool Based on the XGBoost Machine Learning Algorithm," Applied Sciences, vol. 12, no. 13, p. 6672, Jul. 2022, doi: https://doi.org/10.3390/app12136672.

[17] H. S. Gill, O. I. Khalaf, Y. Alotaibi, S. Alghamdi, and F. Alassery, "Multi-Model CNN-RNN-LSTM Based Fruit Recognition and Classification," Intelligent Automation & Soft Computing, vol. 33, no. 1, pp. 637–650, 2022, doi: https://doi.org/10.32604/iasc.2022.022589.

[18] A. Oliveira, "Malware Analysis Datasets: API Call Sequences," www.kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/ang3loliveira/malware-analysis-datasets-api-call-sequences. [Accessed: 20-Jan-2023].