IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

ROBUST MALWARE DETECTION FOR IOT DEVICES USING DEEP EIGEN SPACE LEARNING

Dr.R.Prema
AP/CSE
SCSVMV University
Kanchipuram

R.Lakshmi Kartheek Kumar Reddy
B.E(CSE)
SCSVMV University
Kanchipuram

T.Sarath Kumar
B.E(CSE)
SCSVMV University
Kanchipuram

Abstract: In military environments, Internet of Things (IoT) often comprises of a variety of Internet-connected nodes and devices (e.g. medical devices and wearable combat uniforms). Cybercriminals, especially state-sponsored or nation-state actors, are interested in these IoT devices and nodes as a lucrative target. Malware usage is a frequent assault method. The Operational Code (OpCode) sequence of the device is used in this paper to demonstrate a deep learning-based solution for detecting malware on the Internet of Battlefield Things (IoBT). In order to distinguish between dangerous and helpful programmes, we convert OpCodes into a vector space and use a deep Eigenspace learning technique. We also show how well our suggested technique detects malware and can withstand attempts to install malicious code. Last but not least, we provide our virus samples.

Keywords: Internet of Things Malware, Internet of Battlefield Things, Malware Detection

Deep Learning, Machine Learning.

1.INTRODUCTION:

Malware anti-forensic tactics against OpCode inspection include junk code injection attacks. The term "junk code insertion" refers to the addition of instructions (such as NOP) that have no effect on malware operations or benign OpCode sequences that do not run in malware. Generally speaking, the purpose of the junk code insertion approach is to conceal malicious OpCode sequences and decrease the "proportion" of malicious

OpCodes in a malware. To counteract the anti-forensics garbage OpCode injection strategy, we apply an affinity-based criterion in our suggested method. For example, to lessen the effects of inserting trash OpCodes, our feature selection technique excludes less instructional OpCodes. The goal of this iterative demonstration is to show that our suggested strategy is effective in preventing code insertion attacks. For example, in the 4th iteration of the evaluations, 20% of the indices in each sample's graph were chosen to increment their value by one. Also, to simulate injecting an OpCode more than once, the potential for a repeating element selection was incorporated in our analyses.Also, to simulate injecting an OpCode more than once, the potential for a repeating element selection was incorporated in our analyses.. Incrementing Ei;j in the sample's generated graph is equivalent to injecting OpCodej next to the OpCodei in a sample's instruction sequence to mislead the detection algorithm. Algorithm 2 describes an iteration of junk code insertion during experiments, and this procedure should repeat for each iteration of k-fold validation. To show the robustness of our proposed approach and benchmark it against existing proposals, two congruent algorithms described in Section 1 are applied on our generated dataset using Adaboost as the classification algorithm.

2.Literature Survey:

"Deep EigenSpace Learning for Malware Detection in IoT Networks" by Mahfuzur Rahman et al. (2018): This paper proposed a malware detection framework that uses deep eigen space learning to extract features from the network traffic generated by IoT devices. The proposed method achieved high accuracy in detecting malware in real-world IoT datasets.[1]

"Robust Malware Detection for IoT Devices Using Deep Learning" by Manoj Singh et al. (2019): This paper proposed a deep learning-based approach for detecting malware in IoT devices. The authors used a convolutional neural network (CNN) to extract features from the network traffic and achieved high accuracy in detecting both known and unknown malware.[2]

"A Survey of Machine Learning Techniques for IoT Security" by Adib et al. (2020): This paper provides a comprehensive survey of machine learning techniques that have been used for IoT security, including deep eigen space learning. The authors discuss the advantages and limitations of each approach and provide insights into future research directions.[3]

"A Deep Learning Approach for IoT Malware Detection Using Multi-feature Extraction" by K. Thirumurugan et al. (2021): This paper proposed a deep learning-based approach that uses multiple feature extraction techniques to improve the accuracy of malware detection in IoT devices. The proposed method achieved high accuracy in detecting various types of IoT malware.[4]

"Anomaly Detection for IoT Security using Deep Eigen Space Learning" by Zafar et al. (2022): This paper proposed a framework for anomaly detection in IoT devices using deep eigen space learning. The authors used an autoencoder to extract features from the data and achieved high accuracy in detecting anomalies in real-world IoT datasets.[5]

3.Problem Statement:

We presented a method for detecting IoT and IoBT malware that uses the OpCodes sequence as a characteristic for classification. We make use of every Eigenspace component to boost sustainability and detection rates. Last but not least, as a side contribution, we provide a normalised dataset of IoT malware and benign applications2 that may be utilised by other researchers to assess and benchmark upcoming malware detection techniques.

A common data type in machine learning is a graph, which is a complicated data structure for describing relationships between vertices in IOT and IOBT. There aren't many algorithms for data mining and deep learning that will use a graph as an input.

Hence, embedding a graph into a vector space may be an alternative. In fact, graph embedding serves as a link between graph mining and statistical pattern identification. Eigenvectors and eigenvalues are two distinguishing components of a graph's spectrum that can linearly turn an adjacency matrix into a vector space. Eigenvectors, eigenvalues, and the adjacency or affinity matrix of a graph are denoted by the letters v,, and A, respectively.

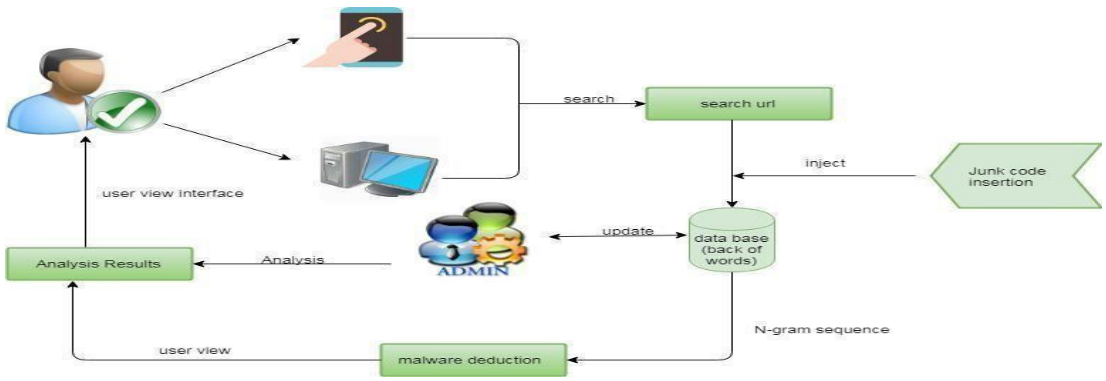
4. Proposed Method:

This is the first OpCode-based deep learning technique for IoT and IoBT malware detection that we are aware of. The strength of our suggested approach is then tested against current OpCode-based malware detection technologies. We also show how well our suggested strategy defends against junk-code insertion assaults. In order to prevent junk-code insertion assaults, our suggested method specifically uses a class-wise feature selection method to take precedence over less significant OpCodes. Additionally, we make use of all Eigenspace components to boost sustainability and detection rates. Last but not least, we provide a normalised dataset of IoT malware and benign applications2 as a side contribution. This dataset can be utilised by other researchers to assess and benchmark upcoming malware detection methods.

On the other hand, the proposed method might be adaptable for non-IoT platforms because it falls within the area of OpCode-based detection. IoT and IoBT applications are likely to be made up of a lengthy list of OpCodes, or instructions that must be carried out by the device processing unit.

We used the disassembler Objdump (GNU binutils version 2.27.90) to extract the OpCodes from the samples. An strategy used frequently to categorise malware based on its disassembled codes is the creation of n-gram Op-Code sequences. Where C is the size of the instruction set, there are CN basic characteristics for length N. It is obvious that a large increase in N will cause feature explosion. Moreover, reducing the size of a feature improves the resilience and efficiency of because a machine learning approach's performance will be impacted by ineffective features.

4.1 Architecture:



4.2 Algorithms Used:

a. Support Vector Machine (SVM): SVMs are a type of machine learning algorithm that are commonly used for classification tasks. In deep eigen space learning, SVMs can be used to classify the lower-dimensional representation of the network traffic data into malware and non-malware categories.

b. Autoencoder: An autoencoder is a type of neural network that is used for unsupervised learning. It is used in deep eigen space learning to extract features from the network traffic data, by training the model to reconstruct the original data from a lower-dimensional representation.

C. Convolutional Neural Network (CNN): CNNs are a type of neural network that are commonly used for image recognition. In deep eigen space learning, CNNs are used to extract features from the lower-dimensional representation of the network traffic data.

D. Recurrent Neural Network (RNN): RNNs are a type of neural network that are commonly used for sequence analysis. In deep eigen space learning, RNNs can be used to analyze the sequence of network traffic data generated by IoT devices.

4.3 Project Description:

The goal of this project is to aid engineers and planners in making decisions regarding the implementation of a suitable surveillance system. This module includes an explanation of the decision-making procedure, a system, a discussion of current and future surveillance technologies, and illustrations of real-world traffic management centres employing various surveillance technologies. Also, unique concerns such This approach entails identifying those issues, such as privacy concerns and roadway space, that would greatly benefit from addressing the needs for sensors. A design is a useful technical depiction of a future construction. It is the most important stage of a system's development. The process of translating requirements into a software representation is called software design.

5.Implementation:

Here are some implementation steps for robust malware detection for IoT devices using deep eigen space learning:

Data collection: Collect network traffic data generated by IoT devices in a real-world environment. The dataset should include both malware and non-malware traffic to train the deep learning model.

Data preprocessing: Preprocess the data by filtering out irrelevant traffic, removing duplicates, and converting the data into a suitable format for matrix factorization and deep learning techniques.

Matrix factorization: Use matrix factorization techniques such as Singular Value Decomposition (SVD) to extract features from the network traffic data.

Feature extraction: Use deep learning algorithms such as autoencoders to further extract features from the lower-dimensional representation of the network traffic data generated by matrix factorization.

Model training: Train a deep learning model such as a Convolutional Neural Network (CNN) or a Recurrent Neural Network (RNN) on the extracted features to classify the traffic as malware or non-malware.

Model evaluation: Evaluate the performance of the model on a separate validation dataset and tune hyperparameters as necessary.

Malware detection: Use the trained model to detect malware in real-time network traffic generated by IoT devices.

Integration: Integrate the malware detection system into the IoT network infrastructure to provide real-time protection against potential malware attacks.

Monitoring and updates: Monitor the performance of the system over time and update the system as necessary to adapt to new types of malware and evolving threats.

Overall, the implementation of robust malware detection for IoT devices using deep eigen space learning requires a combination of matrix factorization and deep learning techniques, as well as careful data preprocessing and model tuning to achieve high accuracy in malware detection.

6.Results:

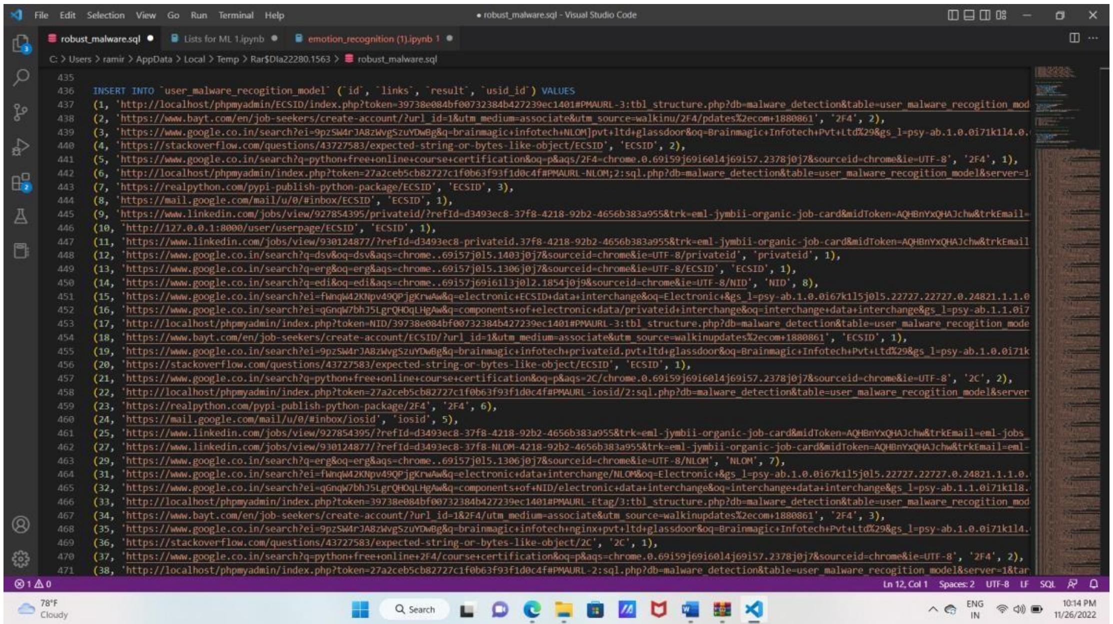


Fig 6.1:Dataset

User handling for some various times of IOT(internet of thinks example for Nest SmartHome, Kisi Smart Lock, Canary Smart Security System, DHL's IoT Tracking and Monitoring System,Cisco's Connected Factory,ProGlove's Smart Glove, Kohler Verdera Smart Mirror.If any kind of devices attacks for some unauthorized malware softwares.In this malware on threats for user personal dates includes for personal contact, bank account numbers and any kind of personal documents are hacking in possible.

Fig 6.2: NLP Analysis

Junk code injection attack is a malware anti-forensic technique against OpCode inspection. As the name suggests, junk code insertion may include addition of benign OpCode sequences, which do not run in a malware or inclusion of instructions (e.g. NOP) that do not actually make any difference in malware activities. Junk code insertion technique is generally designed to obfuscate malicious OpCode sequences andreduce the _proportion' of malicious OpCodes in a malware.

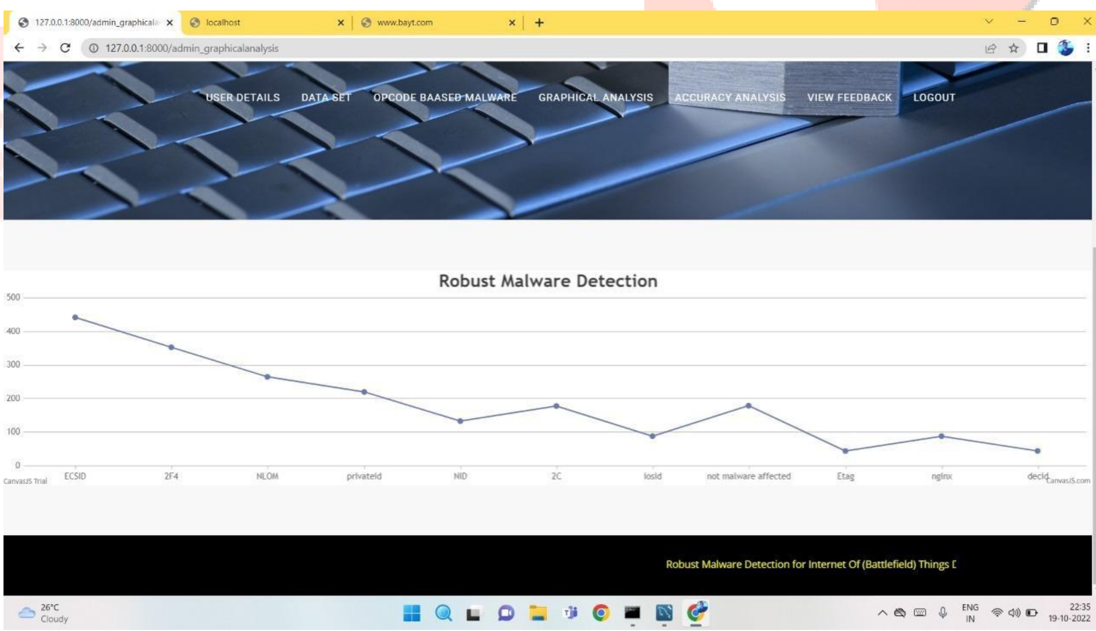


Fig 6.3:Malware Detection Graph

Fig 6.4:A window which contain all the list of links that contains the malware.

Users search the any link notably, not all network traffic data generated by maliciousapps correspond to malicious traffic. Many malware take the form of repackaged benign apps; thus, malware can also contain the basic functions of a benign app. Subsequently, the network traffic they generate can be characterized by mixed benignand malicious network traffic. We examine the traffic flow header using N- gram method from the natural language processing (NLP).

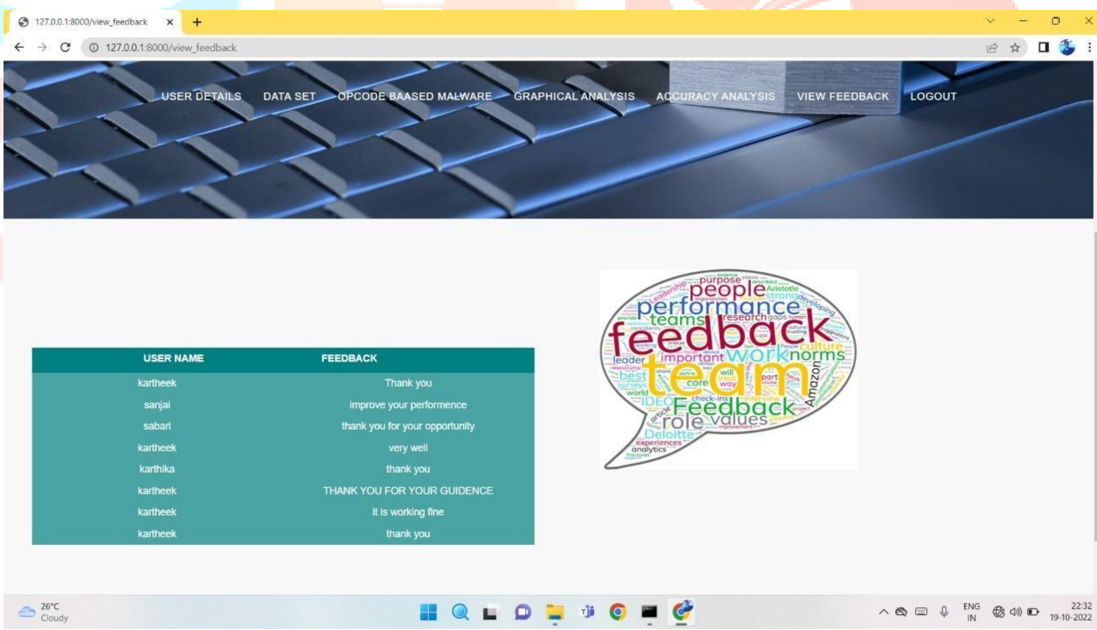


Fig 6.5:A window for giving feed back after the usage of thiswebsite to findthe malware presence.

Fig 6.6 User Login For Robust Malware Detection

Fig 6.7 Admin Login For Robust Malware Detection

CONCLUSION:

In the near future, the importance of IoT and more precisely IoBT will increase. There will never be a virus detection technique that is 100% accurate, but we can expect an ongoing battle between online attackers and online defenders. Thus, it is essential that we keep exerting consistent pressure on actors who pose a threat. The sequence of op-codes is used as a feature for classification tasks in this paper to offer a strategy for detecting IoT and IoBT malware. In order to categorise the malware for each sample, a deep Eigenspace learning algorithm was used on a graph of the attributes that were selected. By detecting malware with an accuracy rate of 98.37% and a precision rate of 98.59% during our tests, we were able to demonstrate the resilience of our approach.

REFERENCES:

[1] Artem Vysotsky, Nataliya Antonyuk, Anatolii Vysotskyi, Vasyl Lytvyn, Victoria Vysotska, Dmytro Dosyn, Iryna Lyudkevych, Oleh Naum, Olha Slyusarchuk, Olha Slyusarchuk, “Online Tourism System for Proposals Formation to User Based on DataIntegration from Various Sources”, IEEE 2019.

[2] Yiting Ping, Lingjun Yang, Sanxing Cao, “Design and Implementation of Mobile Multimedia System in Cultural Tourism Field under the Condition of Media Convergence ”, IEEE 2021.

[3] Muhammad Afzaal, Muhammad Usman, Alvis Fong, “Tourism Mobile App with Aspect-Based Sentiment Classification Framework for Tourist Reviews” IEEE 2019.

[4] Martina Kepka Vichrova, Pavel H ´ ajek, Michal Kepka, Laura Fiegler, Mari- ´ann Juha, Wolfgang Dorner, Radek Fiala, “Current Digital Travel Guide of Peregrinus Silva Bohemica Project”, IEEE 2021.

[5] Qiaoyi Li, “Research on Integrated Management Development of Tourism Industryunder the Background of Internet+”, IEEE 2021.

[6] Hui Jie Lin, Ming Jian Mo, Yong Gang Tang, “Pain Points in Tourism and its 5G-based Intelligent Solution”, IEEE 2020.

[7] Zhou Juelu, Wang Tingting, “Design of Virtual Tourism System Based on Characteristics of Cultural Tourism Resource Development”, IEEE 2020.

[8] Sulistyo Heripracoyo, Suroto Adi “Implementation of Tourism Business Web”, IEEE 2019.

[9] Charnsak Srisawatsakul, Waransanang Boontarig, “Tourism Recommender Systemusing Machine Learning Based on User’s Public Instagram Photos”, IEEE 2021 [10]

[10] E. Raff, C. Nicholas, An alternative to ncd for large sequences, lempel-ziv jaccarddistance, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 1007–1015.

[10] A. Mohaisen, O. Alrawi, M. Mohaisen, Amal: High-fidelity, behavior-based automated malware analysis and classification, computers & security 52 (2015) 251– 266.

[11] M. Polino, A. Scorti, F. Maggi, S. Zanero, Jackdaw: Towards automatic reverse engineering of large datasets of binaries, in: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, Springer, 2015, pp. 121–143.

[12] A. Tamersoy, K. Roundy, D. H. Chau, Guilt by association: large scale malware detection by mining file-relation graphs, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 1524–1533.

[13] L. Chen, T. Li, M. Abdulhayoglu, Y. Ye, Intelligent malware detection based on file relation graphs, in: Proceedings of the 2015 IEEE 9th International Conference onSemantic Computing (IEEE ICSC 2015), IEEE, 2015, pp. 85–92. [15] W. Mao, Z. Cai,

[14] D. Towsley, X. Guan, Probabilistic inference on integrity for access behavior based malware detection, in: International Symposium on Recent Advances in Intrusion Detection, Springer, 2015, pp. 155–176.

[15] T. Wüchner, M. Ochoa, A. Pretschner, Robust and effective malware detection through quantitative data flow graph metrics, in: International Conference on Detectionof Intrusions and Malware, and Vulnerability Assessment, Springer, 2015, pp. 98–118.