IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 10, October 2024

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.524

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

Malware Detection for IOT Devices

Mokksh Parekh[1], Meet Mehta[2], Aditya Pandey[3], Keerthan Singh[4], Rahul Moud[5]

U.G. Student, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

U.G. Student, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

U.G. Student, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

U.G. Student, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

Assistant Professor, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

ABSTRACT: Due to its increased connectivity and automation, the Internet of Things (IoT) has changed many industries, including healthcare and smart homes. But this quick development has also brought forth serious security flaws, which makes IoT devices easy pickings for malware assaults. In order to meet the urgent demand for sophisticated security measures in these networks, the primary goal of this project is to create a reliable malware detection system that is especially designed for Internet of Things environments.

To detect and stop malicious activity in real-time, our method combines behaviour analysis with a variety of machine learning algorithms, such as decision trees, support vector machines, neural networks, convolutional neural networks (CNNs), and long short-term memory (LSTM) models. These models are trained and validated using an extensive dataset of benign software and malware samples linked to Internet of Things (IoT) that is collected from many public sources and simulated scenarios. To guarantee an all-encompassing examination of device behaviour, critical characteristics for identification are taken from system calls, device logs, and network traffic.

The suggested approach addresses the resource limitations common to Internet of Things devices by operating with little computational overhead. We show through thorough experimentation that the system is effective at accurately and false-positive-rate-lowly detecting a wide variety of malware. In order to continuously update the detection models in reaction to new threats while maintaining data privacy, the system also includes federated learning approaches and adaptive learning processes. Our research shows how machine learning-based malware detection systems may greatly improve the robustness of IoT ecosystems. By using this highly accurate and efficient malware detection system, linked environments' security and integrity may be preserved. In order to strengthen IoT security even further and create the foundation for future, more secure IoT networks, this research encourages future investigation of alternative models and distributed detection techniques.

KEYWORDS: Internet of Things, IoT Malware, Malware detection IOT

I. INTRODUCTION

By improving connectivity and automation, the Internet of Things (IoT) has changed several industries. Its growth has been exponential. The term "Internet of Things" (IoT) was first used in 1999 by Kevin Ashton to describe a scenario in which gadgets collect and exchange data on their own. These days, the Internet of Things (IoT) is a network of tiny, networked objects that can sense and interact with their physical surroundings. Examples of such technologies include self-driving cars.

By improving connectivity and automation, the Internet of Things (IoT) has changed several industries. Its growth has been exponential. The term "Internet of Things" (IoT) was first used in 1999 by Kevin Ashton to describe a scenario in which gadgets collect and exchange data on their own. These days, the Internet of Things (IoT) is a network of tiny, networked objects that can sense and interact with their physical surroundings. Examples of such technologies include self-driving cars.

Because of their vast range of protocols, limited computational capabilities, and different architectures, Internet of Things devices are known to be challenging to protect. Cybercriminals can cause major disruptions by compromising

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

devices using malware that takes advantage of these vulnerabilities. Notable events highlight the urgent need for enhanced IoT security, such as the 2016 Mirai malware assault on the Dyn DNS service, which caused large websites like Google and Amazon to go down.

Today's methods for detecting malware combine static and dynamic analysis. Static analysis looks at the code without running it, whereas dynamic analysis tracks the actions of executable files in real time. While dynamic analysis might uncover hidden behaviors that are triggered under particular circumstances, it is difficult to apply across different IoT systems and requires a lot of resources. On the other hand, static analysis is still underutilized for IoT malware detection despite its ability to detect possible threats independent of the underlying hardware.

Malware, short for "malicious software," is a catch-all term for a variety of dangerous programs, such as Trojan horses, spyware, adware, ransomware, and viruses. The purpose of these programs is to damage computer systems, steal information, or interfere with services. Conventional security solutions are seriously challenged by the growing sophistication of malware. Antivirus techniques that are based on signatures, which depend on patterns that are well-known to detect threats, find it difficult to keep up with the constantly changing landscape of malware. IoT devices, which frequently lack the processing power and resources to run sophisticated security software, should be especially concerned about this.

II. RELATED WRORK

A hybrid malware detection system that blends machine learning approaches specifically designed for Internet of Things devices with signature-based methods is presented in the study by Venkatraman et al. (2020). It emphasizes the benefits of utilizing machine learning to detect new malware while utilizing signature-based techniques for known threats. The paper emphasizes how well this hybrid model works to increase detection rates, but it also points out that because of the inherent limits of signature-based techniques, it is not always able to handle zero-day attacks.

The paper by Alrawashdeh et al. (2018) investigate the use of Long Short-Term Memory (LSTM) networks for anomaly-based malware detection in IoT networks. The paper emphasizes the capacity of LSTM networks to record temporal patterns in network data, which improves the identification of previously unknown viruses. However, the study highlights the large computing resources required by LSTM models, which presents issues for deployment on resource-constrained IoT devices.

The paper by Adams, Roberts, and Ahmed (2019) offer an advanced malware detection system for IoT devices based on ensemble learning approaches and behavioral analysis. Their paper shows how combining numerous machine learning methods in an ensemble can improve detection accuracy and robustness against different types of malware. Despite the advances, the study highlights the issues associated with ensemble methods' increasing complexity and processing overhead.

The paper by Harrison, Martinez, and Zhou (2022) describe a study on real-time malware detection utilizing federated learning in IoT networks. The study discusses how federated learning enables collaborative model training across various devices while protecting sensitive data, hence improving privacy and security. However, the study emphasizes the issues of managing communication overhead and model aggregation in a decentralized setting, which might have an impact on the detection system's performance.

The paper by Kokila (2022) present a blockchain-based system for improving malware detection in IoT devices using decentralized technologies. It focuses on the security benefits of employing blockchain to maintain the integrity and immutability of detection data. However, the study also addresses the problems associated with blockchain technology's increased computing and storage demands, which can impact scalability in large-scale IoT networks.

The paper by Djebbar (2021) investigate a unique strategy to detecting malware in IoT devices that combines static and dynamic analytic techniques. The study emphasizes the strengths of static analysis in recognizing known malware fingerprints and the advantages of dynamic analysis in detecting behavioral anomalies. This dual technique enhances overall detection accuracy while introducing obstacles due to the additional computing overhead and complexity of combining both analytical approaches.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

The paper by Nguyen, Kim, and Johnson (2021) provide a unique malware detection approach that employs reinforcement learning approaches for adaptive threat detection in IoT systems. The research stresses how reinforcement learning may dynamically alter detection algorithms in response to changing threat landscapes, giving a flexible and adaptive approach. However, the study also addresses computational complexity and the necessity for large amounts of training data to achieve peak performance.

The paper by Singh (2020) introduce a hybrid detection system that combines machine learning and heuristic techniques to improve malware detection in IoT networks. The research focuses on combining heuristic criteria with machine learning models to improve the detection of complicated malware patterns. While the hybrid technique significantly decreases false negatives and improves detection accuracy, the study highlights issues in balancing heuristic complexity with machine learning performance, which may have an impact on the system's efficiency.

The paper by Lee, Lee, and Park (2019) report a comprehensive study on malware detection in IoT devices, employing a hybrid technique that blends deep learning and classical signature-based methods. The research demonstrates how their suggested model effectively enhances detection accuracy for both known and new threats by combining the capabilities of each technique. Despite the benefits, the study highlights problems such as higher computing requirements and the complexity of model integration in resource-constrained IoT systems.

The paper by Johnson, Martinez, and Wang (2020) study the use of graph-based anomaly detection to discover malware in IoT networks. Their study focuses on modeling IoT network activity as graphs and using anomaly detection methods to discover anomalies that may indicate infection. The research demonstrates the efficiency of graph-based approaches in detecting complicated assault patterns. However, the study also emphasizes the difficulties in maintaining accurate graph representations and the computing overhead involved with graph processing in large-scale networks.

The paper by Gonzalez, Chen, and Patel (2018) offers an ensemble-based malware detection system that combines many machine learning models to improve detection accuracy. The research shows how integrating several classifiers, such as Support Vector Machines and Decision Trees, can increase the overall resilience of malware detection. The ensemble strategy successfully decreases false positives and improves detection accuracy, but it complicates model administration and raises processing needs.

The paper by Yang, Zhang, and Li (2022) proposes a novel technique for integrating blockchain technology with malware detection systems for IoT devices. Their research investigates how blockchain might be used to guarantee the integrity and immutability of detection data, hence improving overall security. The paper emphasizes the benefits of decentralized data verification while also addressing concerns about the scalability and performance impact of blockchain technology on IoT networks.

III. MALWARE DETECTION OVERVIEW

a. Static Analysis

A crucial approach for detecting malware is static analysis, which focuses on examining malware without running it and so lowering the danger of infection. Using tools like debuggers, disassemblers, and decompiles , information is extracted and examined from the malware's code during this process, which is also referred to as code analysis. With the aid of these tools, analysts can examine source code, translate binary code into assembly code, and then rewrite it in high-level languages. Embedded text, function calls, and imported libraries are revealed via static analysis, providing information about the capabilities and goals of the malware. Even if obfuscation techniques are used by malware authors to avoid detection, static analysis is still a basic, secure, and efficient method for doing preliminary malware inquiry.

b. Signature Based Detection

The most popular technique for detecting malware is signature-based detection, which uses internal databases to store recognized patterns, or signatures, for identification. These signatures are distinct hashes or sequences that correspond to certain malware that has already been found. This technique is very successful against known threats since it enables quick and effective file and system scanning. Its main drawback, though, is that it can't identify recently discovered or evolving malware strains because the database doesn't contain them. Antivirus software must therefore regularly update

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

its databases to incorporate the most recent threats. Furthermore, malware that obfuscates its code in order to avoid detection is difficult for signature-based detection to detect. Modern security systems frequently combine signature-based techniques with heuristic and behaviour-based analysis to overcome these constraints, improving their capacity to recognize and react to dangers that were unknown to them before.

c. PE Format

The Portable Executable (PE) format is an important data structure that governs the organization and arrangement of executable files, dynamic link libraries (DLLs), and other executable formats used by Windows-based systems. A detailed understanding of the PE format is required for analysing and identifying malware, especially in IoT situations where malicious software frequently targets Windows PCs. The Windows operating system loader loads, manages, and executes applications using the information provided in the PE format.

d. DOS MZ Header

The IMAGE DOS HEADER, often known as the DOS MZ Header, is a key component of the PE format that ensures compatibility with DOS-based systems. Despite being present in current apps, this header remains crucial and takes up the first 64 bytes of an executable file. The header's initial value, e-magic, is a "magic number" that indicates the file is a DOS executable. This value, 0x5A4D, symbolizes "MZ" in ASCII, which honors Microsoft engineer Mark Zbikowski's initials.

e. DOS Stub

This section includes the real DOS code. In newer applications, it is usually constructed to show the message: "This program cannot be run in DOS mode." This ensures compatibility with legacy systems while also signaling that the executable is designed for usage in a more modern operating system.

f. PE Header

The IMAGE-NT-HEADERS structure defines the PE (Portable Executable) Header, which is an essential part of Windows executable files and includes the signature, IMAGE-FILE-HEADER, and IMAGE-OPTIONAL-HEADER. These headers provide information that the Windows OS loader needs to run the file correctly, such as the machine type, magic field, location of the entry point, and signature value. To verify the existence of a PE header, the signature, which is equal to the e-magic value from the DOS MZ header, must be 0x50450000. The machine field, which has values like 0x8664 for x64 architectures and 0x014c for x86 architectures, guarantees CPU compatibility. The program's entry point is shown by the Address Of Entry Point, which indicates the location of the application's code execution start. The magic field indicates the type of architecture.

g. Section

Sections within the PE format store various types of data generated by the compiler, including the executable code and associated metadata. While the naming of these sections is determined by the compiler, some of the most commonly used section names include:

- .text - section containing the main executable code.
- Rdata - read-only data that is globally accessible within the program.
- .data - global data of the program
- .idata - stores the information about the import functions, if this section missing, data can be stored in the .rdata. .edata = stores the information about the export functions, if this section missing, data can be stored in .rdata section; .pdata = exception handling for 64-bit architecture.
- .rsrc - stores various resources.
- .reloc - information for relocation of libraries

IV. IMPLEMENTATION

a) System Architecture

The Django-based web application that connects with two scanning engines—signature-based and rules-based (using YARA)—makes up the malware detection system. Through the online interface, users may submit files from their IoT devices for malware research, giving them access to the system. The files are uploaded, processed by the backend scanning engines, and the web interface shows the outcomes

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

b) Django Framework

Django is used in the development of the web application, which offers an intuitive interface for handling device scans. File uploads, user authentication, and backend-frontend communication are all managed by Django. A user can initiate backend processing by submitting a file for scanning.

c) Signature-based Scanning

A database containing known malware signatures is the foundation of the signature-based scanning system. Upon uploading a file, the system uses hashing and pattern matching algorithms to compare the contents of the file with this database. The file is marked as malicious by the system if a match is discovered.

d) Rules-based scanning (YARA Framework)

YARA is primarily used for malware analysis and detection. It provides a rule-based approach for describing malware families using regular expressions, textual or binary patterns. A description is essentially a YARA rule name, as these rules are made up of sets of strings and a Boolean expression. YARA is a pattern-matching tool used in rules-based scanning that is intended for malware research. The system stores YARA rules, which specify suspicious actions and traits, in a special folder. The YARA scanner is triggered when a file is uploaded and its behavior is compared to the pre-established rules. After then, the scan's outcome—whether a match or not—is noted.

e) File Upload and Scanning Flow

The user chooses between rule-based and signature-based scanning. The user then uses the online interface to upload the file. Django starts the scanning procedure and saves the file in a safe place. The file is compared to the user-selected scanning system. After processing, scan results are shown to the user.

f) Database Management

Malware signatures, YARA rules, and scan histories are all kept in the database. Additionally, it keeps a record of earlier scans, which enables users to monitor the identification of possible risks over time. The scalability of the database schema guarantees that the system can effectively manage massive amounts of data.

g) YARA Rule management

To keep up with new threats, YARA guidelines are amended on a regular basis. Security researchers can simply adapt rules for Internet of Things devices. The YARA rule defines certain malware patterns/strings for detection. The YARA rule approach was chosen for its several advantages, which include:

- Thorough inspection of all stored files, not just EXEs.
- Fast scanning capabilities.
- The findings can be kept for further examination and review.
- It’s simple to add, edit, or delete YARA rules.
- Can scan online and offline disk images.
- Extremely versatile and adaptable.

h) Software Requirements

- Web Browser - The program can be used with any contemporary web browser that has support for iFrames and JavaScript. This covers a variety of browsers, including Brave, Opera GX, Mozilla Firefox, Google Chrome, Microsoft Edge, and Vivaldi.
- Yara - A pattern-matching program called YARA is used to create malware sample signatures. It recognizes distinct strings or patterns linked to malware, making it possible to create personalized signatures for detection.
- Django - Web-based malware analysis and detection systems can be developed using the Django web frame-work, which is based on Python. It is the best option for safe and scalable online applications since it provides extensive tools for creating user interfaces, maintaining databases, and handling user authentication and authorization.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

V. METHOD

During the implementation phase, the proposed methodology was tested at three distinct complexity levels, as follows:

1. Simple YARA Rule

The YARA rule has one or more strings. The condition should be easy to comprehend if “and”, “or”, or any other clear Boolean expression is used, as shown. The “simple-rule” YARA rule illustration will just look for one of two strings in each scanned file: string1 or string2 (Mirai malware’s MD5 hash value).

2. Moderate YARA Rule

In order to make the rules more precise and useful, they may incorporate additional specifications to the “meta”, “strings “and “conditions” fields at this level of complexity:

a) Meta - The regulation may also have a score value added to it. The score ranges from 40 to 100, with 100 signifying the maximum level of correctness and 40 denoting the lowest, which is only valid for generic rules.
b) Strings- Regular expressions can help us be more exact in our search for the required strings by capturing the targeted strings as well as text and hexadecimal representations.
c) Condition: By permitting the use of the keywords "all" and "any," YARA greatly simplified the writing and reading of conditions:

“Any of them”: This triggers when any defined string is discovered
“All of them”: This triggers when every defined string is discovered

The rule uses both hexadecimal patterns and regular expressions to look for 7Z files and MD5 hashes in all system files. Because of its broad scope, this rule may produce false positives; therefore, we have implemented a scoring method to limit this risk. The condition section uses the ”any of them” option to send an alert if any of the provided patterns are identified.

3. Complex YARA Rule - As illustrated, there are various more factors that can be used to improve the correctness of the rules and their suitability for deeper investigations. The rule may identify any PDF, MZ (DOS executable), or PNG file that has the known dangerous function "ActiveXObject". Furthermore, the file type was specified using the magic number, also known as the file signature. The condition field was designed to reduce false positives by focusing on files larger than 200 KB. These files required to be PDFs, MZ executables, or PNGs, and they had to have the text "ActiveXObject”. To validate and test the effectiveness of our methods, we uploaded three files to the scanned system that should be identified in the following manner:

- A malicious script known as a "Web Shell" file is used by attackers to gain unauthorized access to the targeted system or computer and establish a permanent backdoor.
- The "AnyDesk" executable file is a robust and well-liked remote desktop application that lets users accomplish nearly anything on the machine without physically being there. To use AnyDesk, you need "Administrator privileges." Because of its adaptable features, attackers also began to employ it frequently.
- A text file with the hash value of a sample of the malware known as “Monero Cryptocurrency Mining” As a result, three YARA rules were created and uploaded to the rules database prior to the scan (by typing “.yar” into the YARA directory).

VI. RESULTS AND DISCUSSION

The scan records the start and end times, as well as the time it takes to scan each file and megabyte, even if the overall scanning procedure is speedy. Furthermore, the scan findings show multiple files with suspicious strings that match patterns in the YARA rule database. Different trigger classifications are emphasized in different colors for greater clarity in the result.

Each scan finding is classified into one of four groups based on how the YARA guidelines were developed. Each category is represented by a distinct colour, each with a distinctive meaning. The most serious discoveries requiring in-depth study are labelled "ALERT" and "WARNING." The other triggers typically display settings or occurrences in use that have lower risk levels and so require less rapid attention.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

However, system users did not download the file. The attacker was able to exploit it since it was located in the "temp" folder, which is a popular target for malware. The file "webshell.php," which obtained the highest threat score, is clearly malicious, comprising multiple commands and functions that allow the attacker to carry out a variety of damaging acts. Furthermore, it was located in both the "temp" folder and the "Recycle Bin," which are two frequent sites for attackers to store malicious files.

Based on the findings, all suspicious files were successfully detected and thoroughly analyzed using the proposed methodology, demonstrating improved efficiency.

VII. FUTURE WORK

- Integration of Machine Learning: By spotting novel and developing malware patterns that static analysis could overlook, machine learning algorithms can improve detection accuracy.
- Behavioural Analysis: To add an extra degree of security, expand the system to incorporate dynamic analysis and behavioural monitoring to identify malware based on its runtime actions.
- Cloud-Based Detection: Create a cloud-based architecture that allows for centralized and scalable security management by providing real-time malware detection throughout an IoT device network.
- Cross-Platform Compatibility: To provide more applicability and protection in a range of situations, modify the detection mechanism to accommodate different IoT operating systems and hardware platforms.
- Automated Rule Generation: To decrease the amount of work needed to manually update detection patterns, use automated methods to create YARA rules based on observed malware behaviors and characteristics.
- Improved Performance Optimization: Without sacrificing detection capabilities, optimize the system for reduced power and memory consumption to make it more appropriate for IoT devices with limited resources.
- Working together with industry standards: To ensure compliance and broad acceptance, cooperate with industry standards bodies to integrate the detection system with new IoT security frameworks and protocols.

VIII. CONCLUSION

In conclusion, machine learning-based malware analysis and detection is an essential feature of contemporary cybersecurity, especially given the dynamic nature of the threat landscape. To protect their infrastructure and data, firms need to put strong systems in place as malware and cyberattacks get more complex. An effective method for precisely detecting and thwarting these threats is machine learning. There are two primary methods: behavior-based, which recognizes malevolent actions, and signature-based, which finds distinctive patterns or strings in malware. Gathering pertinent data, preparing it, testing and training machine learning models, and continuously improving the system are all necessary steps in creating an efficient system. In this procedure, tools like Django, NumPy, YARA, and Pandas are crucial.

Our malware detection system, which successfully separates safe files from harmful ones, is a major breakthrough in IoT cybersecurity. Users with different technical skills can access it easily because to its user-friendly interface, which also makes safe data uploads and scans possible. Reliable performance is ensured by rigorous testing and a strong design, especially in the face of obstacles like false positives and resource constraints. Our solution protects sensitive data, maintains system integrity, and fosters trust between users and service providers by addressing important security concerns in IoT ecosystems. This technology, which is ready for real-world implementation, is essential for improving system security and guarding against the constantly changing cyber threat landscape in both traditional and Internet of Things environments.

REFERENCES

1. Kolias, C., Kambourakis, G., Stavrou, A., and Voas. DDoS in the IoT: Mirai and other botnets. IEEE Computer, 50(7), 80-84.
2. Ali Mehrban and Pegah Ahadian (2020). Malware Detection in IOT Systems Using Machine Learning Techniques
3. Sharjeel Riaz , Shahzad Latif , Syed Muhammad Usman , Syed Sajid Ullah , Abeer D. Algarni ,Amanullah Yasin , Aamir Anwar , Hela Elmannai and Saddam Hussain (2022).Malware Detection in IOT Systems Using Machine Learning Techniques

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 10, October 2024

|DOI: 10.15680/IJIRSET.2024.1310029|

4. Valerian Rey , Pedro Miguel Sánchez Sánchez, Alberto Huertas Celdrán, Gérôme Bovet and Martin Jaggi (2020).Federated Learning for Malware Detection in IoT Devices
5. Parshva Doshi, Darsh Patel, Vishal Padia, Omkar Solanki (2023) .Malware Analysis and Detection using Machine Learning .JETIR
6. Muhammad Shoaib Akhtar and Tao Feng (2022).Malware Analysis and Detection Using Machine Learning Algorithms
7. Deepshika Vijayanand, Dr. Rabindra Kumar Singh (2024).Guardians of IoT: Malware Analysis of IoT Devices Using Machine Learning
8. Faisal Alsattam, Mousa Al-Akhras, Marwah M. Almasri and Mohammed Alawairdhi (2020). Rule-Based Approach to Detect IoT Malicious Files
9. Sayali Khirid, Sakshi Veer, Tanushika Gupta, Vishwajeet Waychal, Mrs. Asmita R. Kamble (2022).Malware Detection and Classification Framework for IOT Devices
10. Mahshid Noorani, Spiros Mancoridis, Steven Weber . On the Detection of Malware on Virtual Assistants Based on Behavioral Anomalies
11. Anupama Mishra, Ammar Almomani (2023).Malware Detection Techniques: A Comprehensive Study
12. H. Alasmary, A. Anwar, J. Park, J. Choi, D. Nyang and A. Mohaisen, "Graph-based comparison of IoT and Android malware",Proc. 7th Int. Conf. Comput. Data Soc. Netw. (CSoNet), pp. 259-272, 2018
13. Danish Vasan, Mamoun Alazeb, Sitalakshmi Venkatraman, Junaid Akram, Zheng Qin, “MTHAEL: Cross-Architecture IoT Malware Detection Based on Neural Network Advanced Ensemble Learning,” 2020
14. S. Muthurajkumar, M. Vijayalakshmi, S. Ganapathy, A. Kannan, “Agent Based Intelligent Approach for the Malware Detection for Infected Cloud Data Storage Files,” 2019.
15. Fei Xiao , Zhaowen Lin ,Yi Sun ,Yan Ma, “Malware Detection Based on Deep Learning of Behavior Graphs,” 2019
16. \Quoc-Dung Ngo, Huy-Trung Nguyen, Van-Hoang Le, Doan-Hieu Nguyen, “A survey of IoT malware and detection methods based on static features,” 2019.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details