VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

NABARAJ GHIMIRE

APPLICATIONS OF MACHINE LEARNING FOR MALWARE AND INTRUSIONS DETECTION

Technology

2024

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Cloud-Based Software Engineering

ABSTRACT

Author	Nabaraj Ghimire
Name of Thesis	Applications of Machine learning for Malware and Intrusions Detection
Year	2024
Language	English
Number of pages	64
Supervisor	Rayko Toshev

Computers face numerous security threats like spoofing, jamming, DDoS attacks, malware, and botnet attacks. Various machine learning techniques and algorithms can be employed to enhance the security features of computing devices, making machine learning a valuable tool in this regard.

The primary focus of this research was to pinpoint security issues in electronic devices and the use of machine learning to address them. The objective is to enhance the security of computer devices to safeguard private data, ensure the availability and integrity of the devices, and prevent access to malicious activities.

Certain security experiments yielded improved outcomes, with a reduction in false detection rates from 20% to 5% and increased accuracy in detecting threats from 89% to 92%. Algorithms and experiments did not yet achieve 100% accuracy in detecting all threats; therefore, computing devices must be trained using advanced machine learning techniques to enhance security levels.

Keywords	Spoofing, Machine Learning, Confusion matrix, Portable Executable files, Sybil

CONTENTS

LIST OF FIGURES AND TABLES

ABBREVIATION

AI	Artificial Intelligence
DDoS	Distributed Denial of Service
IT	Information Technology
DoS	Denial of Service
IoT	Internet of Things
IGGM	Instant Gaming Gear Market
KNN	K-Nearest Neighbor
ML	Machine Learning
OOP	Object Oriented Programming
POP	Procedural Oriented Programming
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
AES	Advanced Encryption Standard
DFD	Data Flow Diagram
CSV	Comma-separated values
SVM	Support vector machine
DQN	Deep Q-network
PDS	Post-decision state
t-SNE	t-Distributed Stochastic Neighbor Embedding
PCA	Principal Component Analysis
LAN	Local Area Network

1 INTRODUCTION

1.1 Background of Malware and Intrusions

A quick glance at the history of harmful software serves as a reminder that malware dangers have existed since the early days of computers. The initial or the most ancient virus was identified and documented in 1970. The Creeper Worm was its name and it was an experimental program that self-replicated and spread to other systems, showing the message: "I'm the creeper, try to find me". In the early 1980s, Elk Cloner emerged as a boot-sector virus designed to infect Apple II computers. Since planting these tiny seeds, the field of combating malware has expanded significantly, leading to a never-ending battle against malicious software. It seems like this battle has evolved into an ongoing and cyclical arms race. Malware designers and researchers are now engaged in a constructive arms race, where they continuously improve their defences while malware developers constantly innovate, seek new ways to infect, and enhance their deception tactics. Malware risks are increasing in scope as they take advantage of advancements in technology. The internet, social media platforms, smartphones, IoT devices, and other technologies enable the development of intelligent and complex malware.

As time goes on, increasingly sophisticated malware is being created, posing a significant risk to computer systems. Because of the significant rise in the amount of malware samples, signature-based malware detection methods are unable to deliver precise outcomes. Various researches have shown the effectiveness of machine learning in identifying and categorizing malware. Additionally, the precision of these machine learning models may be enhanced by employing feature selection algorithms to choose the most crucial features and decreasing the dataset size, resulting in fewer computations (K Sethi, 2019).

Malware is created with the intention of causing harm to either the system or its components. Different types of malwares; including worms, viruses, Trojans, ransom ware, spyware, and adware, exist. In recent times, advanced methods are employed to conceal malicious behaviours on users' devices from being detected by antivirus software and security systems. This prompts researchers to search for

fresh techniques, methods, strategies, and tools to enhance malware detection and protection systems. In general, methods for detecting malicious software can be classified into three main types: Signature-based, Anomaly-based, and Heuristic-based techniques. Most antivirus programs primarily utilize signature-based methods. Every antivirus program contains a collection of known malware patterns, and if an application matches any of these patterns, it will be flagged as malware (Omar Bawazeer, 2021).

Malicious software has the ability to carry out a variety of harmful actions once it has infiltrated the system. Some hackers use malware like Trojans to steal private data, alter information, remotely control compromised devices, slow down network and connected devices, erase user information, or carry out harmful activities. In general, malware refers to any harmful code or malicious programs like viruses, spyware, ransom ware, and backdoors, among others. Malicious software programs (files) are created to attack different platforms like desktop and mobile OS and Internet of Things (IoT) devices (Pascal Maniriho, 2023).

Malware one of the biggest challenge internet users faces today, with polymorphic malware emerging as a particularly an evasive threat. Comparing to traditional viruses, polymorphic malware continually alters its characteristics to elude detection by conventional signature-based models. We aimed to identify and mitigate these malicious threats effectively by employing various machine learning technique. Selecting the most accurate algorithm was a crucial task, by determining its high detection rate. The efficacy of the confusion matrix extended beyond mere accuracy, providing insights into false positives and false negatives, thereby strengthening system performance assessment (Alen Jacob Kurian, 2024).

Malware is a short form of malicious software which is a type of software program which is designed to infiltrate damage or gain unauthorized access to computer systems without the user's consent or knowledge. In the beginning, the malware spared as simple viruses and worms, directing their destructive or data-corrupting attacks. Nevertheless, in the process of development, it has transformed into the advanced the gadgets designed for economic purposes, spying, or to some extent, political purposes. Its evolution has been in line with the progress in technology,

and malware has been adapting itself to exploit the weaknesses of operating systems, networks, and applications. The threats spectrum of modern malware is certainly broad: ransom ware, spyware, Trojans, and botnets are the common sources of the infection, but they can be distributed via e-mails, infected websites, or removable storage devices. Hackers are very proactive in updating their attack style, not only by using encryption or polymorphism, but also with social engineering, to pass through security systems, and damage the cyber security professionals' worldwide.

The IoT is a robust worldwide system that consists of numerous interconnected devices for executing sensing, communication, networking, and data processing functions. IoTs are created to be compact, lightweight, and easily transportable for sensor usage and result delivery. At first, IoT utilized RFID technology to create distinct features, but advancements in WSN now allow IoT to detect and supervise a range of activities (Panda & Panda, 2020).

A denial-of-service attack causes a machine or network to be unavailable by disrupting the connection to the server, making it impossible to connect to the server. DOS attacks target the system network to disrupt its operation by utilizing various methods such as buffer overflow and flood attacks. DOS attacks occur when excessive requests are sent to a specific system, overwhelming it and preventing normal server requests from being processed. A DOS attack is a method to overwhelm the capacity of a specific network, leading to a denial of service to any further system requests. A foe seizes the data credentials and obtains entry to the node/gateways. Gateways store computing data in their internal database. A threat actor can utilize this data intelligence to take advantage of the other devices. Ultimately, it is capable of initiating a DoS attack by sending unwanted packets to disrupt network communication. In the realm of computing, Distributed Denial of Service (DDoS) attacks pose a significant threat as they have the potential to disrupt the entire application by overwhelming it with excessive requests.

It is important for computing devices to be built with security features using machine learning technology. Every year, there is a growing number of attacks on

Information Technology (IT) systems, with the main concern being security as these devices carry out a range of functions. Using machine learning is an effective method for enhancing the security of computing devices since it has the capability to identify attacks in the initial stages. Various machine learning techniques are used to prevent or control threats such as Distributed Denial of Service (DDoS), botnet attacks, viruses, Trojans, and spoofing on computing devices. Machine learning can be adapted based on the type of attack to prevent unauthorized access from authentication to access control (Xioo, Lu, Zhang, & Wu, May 30, 2020).

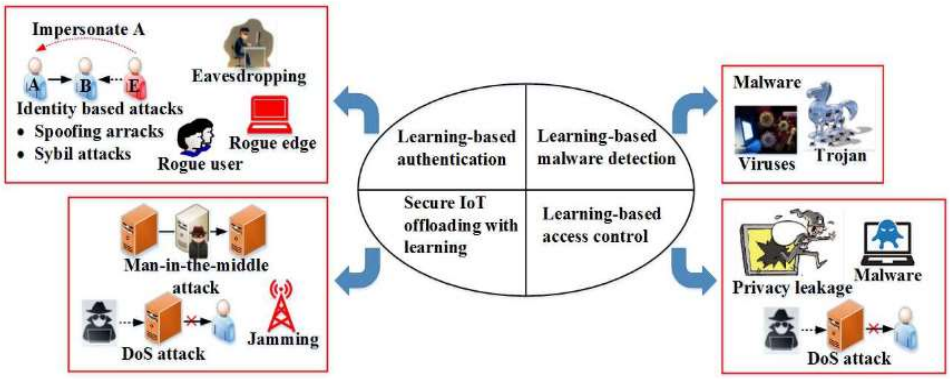

Figure 1. Model of threats in Internet of Things (IOT) systems.

The major threats depicted in Figure 1 can be reduced and even stopped by utilizing machine learning methods. The Internet of Things (IoT) encounters challenges with noise and interference affecting its sensors and communication module, requiring support to gather and share data on a cloud network with various interconnected physical hardware and sensing devices.

Machine Learning is a component of Artificial Intelligence (AI) that has the ability to function and take actions in a dynamic network independently, without requiring human assistance. ML, with its various algorithms, can be utilized as the optimal method to enhance IoT security by enabling IoT devices to learn from past experiences rather than relying solely on externally programmed instructions. In traditional programs, input data is used to generate output information, whereas in machine learning systems, input data is used to create a model that can produce

output information from any future input data. This idea is illustrated in Figure 2 provided below:

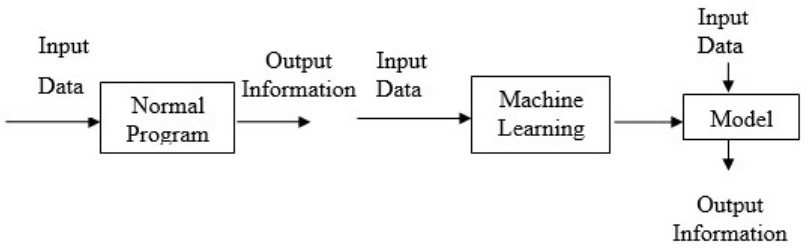

Figure 2.a: Normal Program Concept **Figure 2.b**: Machine Learning Program Concept

Figure 2. Normal and Machine learning program concept.

The tasks of machine learning program concepts are mentioned below as follows:

1. Data Pre-processing,
2. Data Classification,
3. Predictions,
4. Clustering, and
5. Pattern recognition.

No matter how big the data is, it can still be analysed accurately by employing machine learning algorithms that are specifically designed to enable the system to carry out complicated tasks.

Different supervised learning methods such as Naïve Bayes, Support Vector Machine (SVM), deep neural network, K-Nearest Neighbor (KNN), and random forest can categorize IoT device network activity to develop a classification model. The Naive Bayes classifier has been a popular machine learning algorithm for identifying threats and security vulnerabilities in IoT and smart devices since the 1960s (Ahamed & Farid, 2018). The Naïve Bayes ML algorithm can be used to detect Botnet and other malware in computer network traffic. A process in ML that is not learning-based typically does not need labelled data. This algorithm assists in

carrying out intricate processing functions to identify new patterns in a dataset without prior labels. The purpose of the study's publication was to identify fresh patterns and connections within the system, enabling the system to be potentially used for modelling. An instance of such a procedure involves employing multivariate integration analysis to identify DoS attacks employed by IoT devices, while also safeguarding privacy through the use of IGMM in PHY-layer authentication (Tahsien, Karimipour, & Spachos, 2020).

To improve methods, individuals try different techniques and test out a variety of learning strategies while working together. Q-learning, Deep Q-Network (DQN), and Post-Decision State (PDS) are reinforcement learning methods used to implement security measures on an IoT system, safeguarding it from potential risks by focusing on key factors. Q-learning is often used to improve authentication effectiveness, identify malware, and thwart jamming assaults (Hussain, Hussain, Hassan, & Hossain, 2020).

Over the past ten years, machine learning has caused a significant change in various industries, such as cyber security. Cyber security experts generally hold the belief that AI-driven antimalware tools can enhance scanning engines and detect contemporary malware attacks. The fact that many studies have been published in recent years on using machine learning for detecting malware supports this belief (Doe J. S., 2022). The rise in research is due to several factors, such as the rise in publicly available malware, the increase in computing power while simultaneously decreasing in cost, and the advancements in machine learning leading to significant achievements in tasks like computer vision and speech recognition (Smith L. J., 2023).

Security measures are being implemented on all types of IoT devices. The majority of security measures are implemented to verify the connection of IoT devices with their cloud server. If the IoT devices have sufficient computational power to run malware, then they are susceptible to being used as botnets by malware once they have been authenticated to the cloud server. Some of the primary cyber-attacks such as spoofing, botnets, DoS/DDoS, and man-in-the-middle attacks have already been covered in the previous section.

Unsupervised learning algorithms can be utilized to analyse and classify unknown features to uncover new connections and relationships. These hints and connections can aid in figuring out how malware operates. By making these findings, malware can be reverse engineered to create better security measures for protecting systems from future attacks by similar malware. Some of the typical algorithms used in unsupervised learning include:

1. Principal Component Analysis (PCA)
2. t-Distributed Stochastic Neighbor Embedding(t-SNE)
3. k-means clustering

In this section, the reinforcement learning method allows the system to autonomously learn by exploring all potential options, committing errors, and analysing each mistake to gather information and create a concise and quick model that can make efficient decisions.

Several machine learning algorithms are frequently utilized are:

1. Decision Tree
2. Post-decision state (PDS)
3. Linear Regression
4. Logistic Regression

Supervised learning involves providing datasets to the method that contain both features and target variables for learning. The algorithms in this model are able to learn characteristics and establish connections with the target variable. The model created from this variable can make decisions when presented with new data that it has not encountered before. A few of the algorithms in this learning model that we plan to execute and evaluate are outlined below.

1. KNN (K-Nearest Neighbour)
2. SVM (Support vector machine)
3. Deep neural network
4. Random Forest
5. Naïve Bayes

Some of the algorithms we will utilize for supervised learning in our project are classification model algorithms.

1.2 Research Significance

The results of the work in applying machine learning to malware and intrusion detection hold significant importance for various stakeholders in the cyber security areas. The utilization of the developed techniques offers several benefits: The results of the work in applying machine learning to malware and intrusion detection hold significant importance for various stakeholders in the cyber security areas. The utilization of the developed techniques offers several benefits:

IoT-powered computing devices collect data from the surroundings through a variety of sensor types. Machine learning is needed in various industries such as banking, robotics, research centres, engineering, healthcare, and more. There is a high demand for machine learning or deep learning in computing devices to protect them from threats. Machine learning algorithms are so intricate that they are difficult to grasp and apply, which is why they are utilized by all organizations. Companies are eager to utilize machine learning for enhancing the security of their systems and Internet of Things (IoT) devices. Large organizations such as space stations, medicine, factories, and smart cities utilize machine learning for security purposes (Doe A. M., 2024).

Machine learning is all about continuously training machines with new data to achieve optimal results in real-time analysis of datasets for detecting malware. 95% accuracy was achieved using the Decision Tree model (Shhadat, Bataineh, Hayjneh, & A.Al-sharif, 2020).

Machine learning can enable IoT devices to carry out various tasks regardless of the level of risk. The article emphasizes the potential to achieve something bigger in the realm of Information and technology. Furthermore, combining IoT devices with machine learning has the potential to revolutionize various fields including education, medicine, technology, and beyond (Xioo, Lu, Zhang, & Wu, May 30, 2020).

The research conducted by Liang, Hatcher, Weixian Liao, & Yu, (2020) focuses deeply into the various facets of combining machine learning with IoT. The article discusses positive, negative, and unfavourable aspects of machine learning and IoT. Furthermore, there is significant concern regarding the rise of cyber-attacks and intrusions in today's world, which have been facilitated by Machine learning and IoT.

Machine learning can be utilized to oversee and regulate the intricate layers of IoT devices during a thorough examination. The article demonstrates how machine learning algorithms can be incorporated into different levels of IoT devices to protect them from attacks such as DDoS and MITM. A significant portion of the article focuses on and reaches a consensus on utilizing machine learning to enable smart devices to autonomously carry out intelligent tasks. (Tahsien, Karimipour, & Spachos, 2020)

Hussain, Hussain, Hassan, & Hossain (2020) address a wide range of terms related to the integration of machine learning and Internet of Things. The detailed explanation of each term aids in developing a theoretical understanding of the advantages of integrating ML with IoT. Furthermore, it includes categorization of ML into basic ML, Deep Learning, and deep reinforcement learning.

1.3 Definition of Research Field

The research field of cyber security focuses on the development and implementation of advanced techniques to detect and prevent malicious software (malware) and other security threats within computing devices. This involves leveraging machine learning algorithms to enhance traditional detection methods, enabling systems to recognize and respond to evolving and sophisticated malware attacks. The integration of machine learning with IoT security aims to improve the robustness and resilience of devices against threats such as Distributed Denial of Service (DDoS), botnets, and man-in-the-middle attacks. By employing supervised, unsupervised, and reinforcement learning models, researchers strive to create efficient intrusion detection systems that can adapt to new threats, ensuring the

integrity, confidentiality and availability of data in increasingly interconnected and automated environments (Smith J. , 2024).

1.4 Research Question

The solution suggested for the research problem identified in the preceding section is outlined in this research methodology. The solution to the research question is found in the research methodology used in the projects.

1. How can machine learning techniques be optimized to detect the most common and hardest-to-detect malware and intrusion attacks, and what impact do these attacks have on the security of modern computing environments?
2. What are the types of Machine Learning algorithms are used for protecting the devices from attacks?
3. What kind of datasets is used for training the model?

1.5 Objectives

The main objective of this research is to focus on the implementation of a highly efficient state-of-the-art intrusion detection system that will detect and prevent security threats within computing devices accordingly. As I am doing research on malware detection and intrusion detection by machine learning I have uses four different types of algorithms, they are: K-Nearest Neighbor algorithm, Naïve Bayes algorithm, Decision Tree algorithm and Random Forest algorithm. It is intended to improve the safety of computing devices in order to secure and protect private data provide availability and integrity of computing devices and exclude the access of cruel activities. The main focus of the study is to improve the security of computing devices. Another goal of the research is to investigate other aspects:

1. To study various algorithms' effectiveness in identifying and stopping issues.
2. To study an experiment conducted on reducing risks arising from electronic devices used for processing and storing data.

3. To pinpoint security concerns within computing devices.

1.6 Research Gap

The recognition of malware is crucial in making devices safe. As malware variants are emerging in big numbers it is difficult to protect the system. With the development of technology, the detection of malware is complex.

The evaluation must be done in large scale of data to find the productive result. Training of machine learning models consumes more time, and it is expensive too because labeled dataset must be used in malware and instruction detection. There are no frameworks that balance detection efficiency with user privacy.

1.7 Research Problem

The sector of utilizing machine learning for detecting malware and intrusions on computing devices encounters numerous significant research obstacles. First and foremost, there is the challenge of identifying and monitoring cyber-attacks on IoT devices, which are typically limited in resources and come in various forms. Creating strong algorithms that accurately detect and track these cyber-attacks in real-time continues to be a major challenge. Another crucial challenge is addressing Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks on IoT devices. Creating machine learning models that can effectively prevent such threats is crucial as these attacks have the potential to greatly disrupt the functionality of IoT networks.

In addition, the challenges of network layer attacks stem from the complexities of network protocols and the necessity of thorough monitoring and analysis. Lastly, a significant worry is to efficiently improve security by incorporating machine learning algorithms into IoT devices. This includes improving algorithms to function on IoT devices with limited computational resources, while also guaranteeing strong security measures are in place. It is essential to tackle these obstacles in order to progress the use of machine learning in improving the security of IoT and other interconnected systems.

Literature (Brown, 2024) demonstrates that dataset preparation poses a challenge for Machine Learning. There is a limited dataset available in this particular field, causing the machine to receive inadequate training. An IoT device powered by machine learning needs an increasing amount of data to train effectively and yield optimal security outcomes.

1.8 Research Philosophy

The application of machine learning (ML) in malware and intrusion detection represents a transformative approach to cyber security, leveraging advanced algorithms to identify and mitigate threats with unprecedented accuracy and speed. Traditional methods, reliant on signature-based detection, often fall short against sophisticated and evolving cyber threats (Buczak, 2016). For instance, supervised learning models, such as neural networks and support vector machines, can classify known threats, while unsupervised methods, like clustering and anomaly detection, excel in identifying novel intrusions by recognizing deviations from established baselines (Vinayakumar, 2019).

Research has shown that these ML models can significantly enhance detection rates and reduce false positives, making them invaluable in the proactive defence against cyber threats. Studies by Buczak, (2016) and Vijayakumar, (2019) demonstrate the efficacy of ML in enhancing cyber security measures, underscoring the potential of these technologies in safeguarding digital environments.

1.9 Time Horizon

The test and train data used in this project were obtained from Kaggle, a popular platform that offers a wide range of datasets for machine learning and data science projects. This research made use of a secondary dataset from Kaggle, following a cross-sectional method, as the data pertains to a single time point.

By utilizing secondary data, researchers can avoid collecting primary data and instead concentrate on analyzing the data. The research focused on creating two

machine learning models for intrusion and malware detection, and involved experimentation and interpretation of results for duration of four months.

1.10 Structure of Thesis

This thesis is structured in six major chapters and references to systematically explore the Machine learning algorithm to detect intrusion and malware detection.

Table 1. Structure of Thesis

Chapter and main topic	Sub Chapter	Description	Output
Chapter 1 Introduction	1.1-1.10	Background of Malware and Intrusion, Research significance, Definition of Research field, Research question, Objectives, Research Gap, Research Problem, Research philosophy, Time Horizon, Structure of Thesis	Introduction of thesis and reason for study and foundation for literature research
Chapter 2 Literature Review	2.1-2.2	Studies on the topic and Other Related Studies	Clear concept about Intrusion and malware detection systems
Chapter 3 Research Methodology	3.1-3.3	Selection methodology, Algorithm and Machine Learning Program Approach	Research Approach, about Algorithm used and ML program approach
Chapter 4 Experimental	4.1-4.4	Collection of Data, Experimental Design, Programs and Models	Models are created and various design has been implemented

			with the help of collected data
Chapter 5 Result	5.1-5.3	Machine learning algorithm test result, Importance of machine learning algorithm test result, and Files generated at end of simulation	ML model for malware and intrusion detection is developed, utilizing PE file and network layer datasets resulting in high accuracy and robust performance output for security applications.
Chapter 6 Discussion and Conclusion	6.1-6.2	Discussion of result and conclusion of result	Theoretical and Practical Implications

2 LITRATURE REVIEW

2.1 Studies on the Topic

Several research papers have been released that use machine learning algorithms to detect malware, but none of them include a comparison of various machine learning methods. In addition, while there has been much research on malware detection methods, there is a shortage of data on the performance of machine learning algorithms in terms of detection rates, accuracy rates, analysis techniques, and classification methods. This situation has caused a lack of evidence that limits the use of malware detection in related research fields. The present obstacles and upcoming paths for machine learning algorithms in identifying malware must be emphasized as well.

Panda and Panda (2020) note that advancements in technology through Internet of Things (IoTs) have allowed for the sensing and monitoring of a wide range of activities. As devices become more intelligent and powerful, they also become more susceptible to security risks. Wilson, (2014) emphasize the significance of IoTs in influencing human thought, highlighting their importance. Rise in system numbers leads to greater data accessibility for locating information, yet also highlights the challenge of limited computational resources and data scientist availability, underscoring the importance of machine learning in IoTs. Naveen, Sharma and Nair (2019) suggest in their research paper an IoT system featuring a 128-bit AES encryption method to secure patient data before storing it in a cloud repository.

In a time when attackers are constantly updating their tactics and creating new defences, how long will the AES technique remain a secure encryption method in the future? Strategically implementing criteria for upgrading security measures on IoT devices can address the implementation of security measures for approval. Implementing strategic criteria for upgrading security measures on IoT devices for approval can address the implementation of security measures. In their paper Floyd and Reid, (2016) focus on importance of digital information is considered to be higher than that of tangible assets. Similarly, Pirbhulal, (2019) gather various

research papers, articles, and reports to demonstrate the potential effectiveness of using bio-signals for data and information encryption.

According to Abouzakhar, Andres, & Angelppoulo, (2017) the increased utilization of cloud services has led to an increase in security vulnerabilities, while the integration of both modern and conventional IoT systems may present notable security risks. The main reason for security threats is the link between platforms and the conventional IoT procedure throughout all levels. The author (Alani, 2019) emphasize the importance of implementing security-oriented learning tools. A blend of machine learning and cryptography can handle vast quantities of data. Around 2.5 quintillion bytes of data are produced globally for every dataset, without effective management these datasets will be ineffective. Moreover, Alani (2019) emphasizes turning the machine into a problem-solving tool by utilizing pre-existing data. Furthermore, it puts emphasis on machine learning and how it can be applied to address issues concerning bowel movements. Delaying is an issue that arises when calculating numbers from various inputs. This paper recommends sending educational materials to eliminate retrogression. Additionally, this article covers several subjects that could be relevant to the field of computing including:

1. Risks on machine learning
2. Being safe from faults on ML
3. Integration of machines and cryptography to enhance security.

The implementation of IoT in various sectors worldwide has brought advantages in every aspect. Xioo, Lu, Zhang, & Wu, (2020) show the top effectiveness of Machine Learning in IoT security, with the IoT attack model including Spoofing, Data Privacy, Intermediate, DoS/DDoS attackers, Jamming, Software, and more. Utilize learning-based authentication for IoT devices to restrict access to only authorized users. Additionally, the document recommends putting in place automated access controls, preventing IoT data uploads, and detecting IoT viruses. ML algorithms have the capability to create models that can educate IoT devices on various aspects of performance and safety Xioo, Lu, Zhang, & Wu (2020), propose utilizing:

1. Q-Learning authentication
2. PHY-Layer authentication
3. K-NN and SVM

It also emphasizes the potential of using educational tools to safeguard IoT by not offering unrealistic examples in explaining ML algorithms used for securing IoT.

Strong authentication in an interconnected wireless environment continues to be an important, but sometimes elusive goal. Research in physical-layer authentication using channel features holds promise as a technique to improve network security for a variety of devices. We propose the use of machine learning and measured multiple-input multiple-output communications channel information to decide on whether or not to authenticate a particular device (Germain & Kragh, 2020).

2.2 Other Related Studies

According to paper by Shahadat, (2020) can be classified based on their capabilities and method of attack, including Trojans, Worms, Spyware, and Rootkits. It was written following an examination of tests for detecting malware using machine learning methods. The author categorized the process of identifying malware by dividing it into sections based on insertions and data classification. The writer opted for malware detection feature through deviation authentication and RF segmentation. As stated by the author, the database was discovered to have a mix of harmful and benign files. The author stated that the Naïve Bayes classifier showed high accuracy levels of 55% to 99% with binary divisions and 72.34% to 81.8% with multiple divisions. The article pointed out that this research successfully enhanced the accuracy of Naïve Bayes. They think that this success is due to changes in the variables.

In (Tahsien, Karimipour, & Spachos, 2020) research shows that the focus has shifted to intelligent services for IoT platforms as a central point in the advancement of all areas of life. Due to the high demand for the service, security issues were being experienced by the service. Before today's events, the conventional safety measures appeared ineffective. According to Tahsien, Karimipour, & Spachos,

(2020) electronic education via IoT consists of three stages: unsupervised learning, supervised learning, and reinforcement learning.

Furthermore, supervised learning is categorized into classification and regression. The classification model is subject to the level of safety measures necessary. Furthermore, according to Tahsien, Karimipour, & Spachos, (2020) the ML solution offers security spanning various layers such as Physical, Network, Application, and additional layers of IoT. The study conducted by Tahsien, Karimipour, & Spachos, (2020) is falling behind in the dispersal of IoT distribution devices, particularly focusing on a specific sector or sectors. It solely addresses the most difficult stage of combining machine learning with IoT at present and in the upcoming years. It is believed that integrating machine learning algorithms such as the brain into the IoT system is a complex process.

Pierpaolo, (2023) has done research in artificial intelligence investigates computer algorithms that can enhance themselves through practice and data analysis. Machine learning algorithms create models by analysing sample data, known as "training data," to make predictions or decisions without direct programming. In addition, Sowmya, (2023) describes us various ML approaches have been introduced to find anomalies and categorize different types of attacks. ML is a part of artificial intelligence that can learn features and adapt to changing environments. Statistical and ML algorithms have proven to be highly efficient for intrusion detection.

Similarly, Li & Ju, (2024) explains machine learning, as the core field of artificial intelligence, involves the intersection of multiple disciplines. It improves the performance of algorithms and models through computer simulation or implementation of human learning processes, through data and experience. Additionally, according to Muhammad Shoaib Akhtar, (2022) malware attacks are becoming increasingly common and now even affect IoT devices, medical gear, and environmental and industrial control systems. Modern spyware is notoriously hard to detect, as it constantly updates its code and behaviour. The proliferation of

malware has rendered traditional signature-based defences ineffective. Instead, it is necessary to take a broader range of defensive actions.

Utilizing machine learning in computing devices helps in tackling security threats, but the presence of different types of DOS attacks like volumetric attacks, Sync flooding, TCP-State exhaustion attack, and fragmentation attacks can create challenges for the application. DDoS attacks utilize intelligent grids to implement various tactics that compromise a multitude of devices. During a DoS/DDoS attack, the physical layer experiences limitations in power or information flow. In the midst of the DDoS attack, the smart grid data processing abilities and security measures are greatly impacted, leading to significant advancements in defence mechanisms. Hence, gaining a thorough understanding of DoS/DDoS attacks is crucial in order to acquire a comprehensive knowledge of their effects on both the physical grid and cloud layers, as well as additional details. (Mohammad Kamrul Hasan, 2023)

In all the literature reviews we have gone over, they all address the widespread issue of security and privacy. Data privacy has always been a topic of interest within the field of data protection. Achieving 100% efficiency in security was always unattainable. Machine learning has the potential to elevate humanity to a higher level of innovation, but the journey ahead will not be easy. The greatest obstacle will be ensuring strong security.

Another problem identified during the literature review is the network layer risks that aim at any system. An attack on the network layer includes spoofing, traffic analysis, man-in-the-middle attacks, and data routing.

3 RESEARCH METHODOLOGY

3.1 Selection of Methodology

To reduce the issues and challenges in research, the hybrid methodology was chosen. Both the research utilized both **qualitative** and **quantitative** methods. I have gathered information on the introduction, application, scope, advantages, disadvantages of machine learning and IoT from various sources such as Journals, articles, research papers, and online content in various formats like Pdf, txt, video, websites, etc. Qualitative research involves paper-based methods, while I have created a program for quantitative research and input datasets from various categories. With assistance from Python programming and various machine learning algorithms in the Scikit-learn library, I successfully achieved this. Two separate sets of data with details about malware at the Application layer and network intrusions at the Network layer were utilized. It's a machine learning program that consumes input data to produce a model capable of identifying future attacks with new supplied data. The model created by the program will be utilized in future research, representing the quantitative methods used in the study. Furthermore, the primary source for obtaining understood information was video lectures on Machine Learning and IoT.

3.2 Algorithm

In this research paper, four algorithms are implemented to train and test the models and they are:

3.2.1 Random Forest

The Random Forest algorithm, a popular ensemble learning technique in machine learning, continues to be a cornerstone in predictive modeling due to its robustness and versatility. By aggregating predictions from multiple decision trees trained on random subsets of the data and features, Random Forests mitigate overfitting and enhance generalization performance. Notably, recent advancements in Random Forest research, such as optimizations for scalability and efficiency, as well as

extensions for handling imbalanced data and interpretability, contribute to its enduring relevance in various real-world applications. (Zhang, 2024)

3.2.2 K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm has gained prominence in machine learning for intrusion and malware detection due to its simplicity and effectiveness in identifying patterns in network traffic and system behaviour. By classifying instances based on the majority class of their nearest neighbors, KNN can detect anomalies and potential threats in real-time. Recent studies have demonstrated the applicability of KNN in network security tasks, including intrusion detection and malware classification, showcasing its potential as a reliable and efficient method for cybersecurity applications (Li S. , 2023).

3.2.3 Naïve Bayes

The Naive Bayes algorithm is widely used in machine learning for intrusion and malware detection due to its simplicity, efficiency, and effectiveness in handling high-dimensional data with categorical features. By assuming independence between features, Naive Bayes calculates the probability of a given class label based on the observed feature values. In the context of intrusion and malware detection, Naive Bayes models can efficiently classify network traffic or system behaviour as normal or malicious, aiding in the detection and prevention of cyber threats. Recent research has focused on enhancing Naive Bayes models with advanced feature selection techniques and adaptive learning strategies to improve their accuracy and scalability in real-world security applications (Li H. Z., 2024).

3.2.4 Decision Tree

The decision tree algorithm is widely employed in machine learning for intrusion and malware detection due to its interpretability and effectiveness in capturing complex decision boundaries. By recursively partitioning the feature space based on the most informative features, decision trees can effectively classify network traffic or system behaviors as normal or malicious. Recent studies, such as those by (Chen, 2023) and (Li X. Z., 2024), have demonstrated the efficacy of decision tree-

based approaches in accurately detecting various types of intrusions and malware while providing insights into the decision-making process, making them invaluable tools in cybersecurity research and practice.

3.3 Machine Learning Program Approach

Step 1. START

Step 2. Collection of Data

Step 3. Preparation of Data

Step 4. Choosing Model

Step 5. Training Model

Step 6. Evaluating Model

Step 7. Tuning of Parameters

Step 8. Prediction

Step 9. STOP

3.3.1 Start the Procedure

To initiate the process, it is essential to gather relevant datasets that reflect a wide range of malware behaviors and intrusion patterns. This involves collecting and preprocessing data from diverse sources, including PE (Portable Executable) files for malware analysis and network traffic data for intrusion detection. The data must be labeled accurately to distinguish between normal and malicious activities. Once the data is prepared, feature extraction techniques are applied to identify the most significant attributes that can aid in the detection process. The objectives of the machine learning program, such as minimizing false positives and maximizing detection rates, influence the choice of features and the design of the algorithm. By starting with a well-defined problem statement and clear objectives, the machine learning approach is structured to develop a system that not only detects malware and intrusions effectively but also adapts to evolving threats, ensuring ongoing protection for information systems.

3.3.2 Collection of Data

In the Machine Learning program approach, the collection of data is a crucial initial step that involves gathering relevant datasets from reliable sources to ensure the model's effectiveness. For this project, the data was sourced from Kaggle, a well-known platform that provides a variety of datasets for machine learning and data science projects. The datasets available on Kaggle are often pre-processed and come with extensive metadata, which facilitates easier integration and exploration. Utilizing Kaggle as a data source not only provides access to high-quality, diverse datasets but also allows for benchmarking against existing solutions and participating in a community-driven platform where collaboration and sharing of insights are encouraged. This collected data forms the foundation for training, validating, and testing the machine learning models, ultimately contributing to the robustness and accuracy of the predictive outcomes.

3.3.3 Preparation of Data

During this stage, we examine the data sets, handle and organize them in preparation for training. This stage is also referred to as Data Pre-processing. This stage can also be broken down into various stages such as:

1. Dividing the dataset into training and testing sets.
2. Standardizing Features
3. Identifying both the input and output variables.

3.3.4 Choosing Model

Various machine algorithms exist for the learning process. The appropriate algorithm for model creation is chosen based on the dataset and program requirements. There are two models as mentioned below:

1. Classification model
2. Regression model

In our program, we utilize a classification model. Classification models are used when the output variable is categorical, meaning it represents discrete classes or labels. These models aim to predict the category or class to which a new data point belongs. Common applications of classification models include spam detection, image recognition, and medical diagnosis. Algorithms such as Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, and Neural Networks are widely used for classification tasks. In the context of malware and intrusion detection, classification models are particularly effective as they can categorize network traffic or software behaviour into 'malicious'.

3.3.5 Training Model

In the method of Machine Learning programming, the model is trained by providing datasets to the ML algorithm, allowing the algorithm to learn from the data. This procedure, referred to as model training, is when the algorithm detects patterns and connections in the data, modifying its parameters to reduce error and enhance precision. The training stage is crucial because it impacts the model's capacity to generalize and accurately predict outcomes on unfamiliar data. While being trained, the model continuously improves its ability to make decisions by analysing the input data. Therefore, the model that has been trained is able to autonomously make decisions by utilizing the acquired patterns to forecast results or categorize fresh data using its prior training. This self-governing decision-making allows machine learning models to carry out intricate tasks without direct human involvement after being adequately trained.

3.3.6 Evaluating Model

After creating the model, it is necessary to assess its effectiveness through various tests to evaluate its performance. The evaluation of the model depends on whether it makes true or false decisions. There are several evaluation techniques that are used to calculate the score of models such as:

i) Generating Classification Model

Generating a classification model is a fundamental step in the process of detecting malware and intrusions using machine learning. This involves training a model on a labeled dataset where the features of the data (such as characteristics of PE files or network traffic patterns) are used to predict the class label (malicious or benign). Common algorithms used for this task include decision trees, random forests, and neural networks. Once trained, the model can be used to classify new, unseen data, effectively identifying potential threats in real-time.

ii. Generating Confusion Matrix

After generating a classification model for malware and intrusion detection, it is crucial to evaluate its performance using a confusion matrix. A confusion matrix is a table that visualizes the performance of a classification algorithm by displaying the counts of true positives (correctly identified malicious activities), true negatives (correctly identified benign activities), false positives (benign activities incorrectly labeled as malicious), and false negatives (malicious activities incorrectly labeled as benign). This matrix provides detailed insights into how well the model is performing.

3.3.7 Tuning of Parameter

Using various parameter values during the ML algorithm function call can enhance the effectiveness and efficiency of the output models. Parameter tuning is the act of adjusting parameter values in order to improve a program's efficiency in terms of both time and accuracy.

3.3.8 Prediction

These are the last stages where, following the creation of the model, it is utilized to make predictions. The model can independently make decisions by analysing the input component's features.

Hence, our program is a Machine Learning program; it also undergoes the same steps during the experimental phase.

4 EXPERIEMNTAL

Each stage within the experimental phases is defined according to ML program algorithms as outlined in the methodology section above.

The experiments conducted for our research can be categorized into various stages.

4.1 Collection of Data

The most crucial aspect for a machine learning program is having the necessary data sets. For the upcoming projects, we will be using two different data sets for the task.

This dataset is a csv document containing the characteristics of legitimate and malicious PE files. The features of PE files from over millions of files are included in this data set. Each set of data includes 54 different characteristics. This data is inputted into a supervised learning algorithm to create a model that can identify malware programs. Only some characters are shown in table 2 below.

Table 2. Some characters of data set file

Name|md5|Machine|SizeOfOptionalHeader|Characteristics
memtest.exe|631ea355665f28d4707448e442fbf5b8|332
ose.exe|9d10f99a6712e28f8acd5641e3a7ea6b|332|224|

The dataset for auditing was supplied and contains a diverse range of simulated intrusions in a military network setting. It simulated a standard US Air Force LAN to collect raw TCP/IP dump data for a network. The LAN was simulated to mimic a real-world setting and subjected to numerous attacks. The sequence of TCP packets connecting at a specific time, in which data is transferred from one IP address to another following a defined protocol. Additionally, every contact is

categorized as either normal or an attack with a particular type of attack. Roughly 100 bytes are found in every connection record.

41 rating and rating features are provided for standard and attack data in each TCP/IP connection, including 3 quality features and 38 counts.

In order to simplify programming, we group the 36 unique values of the class feature into 2 categories which are:

• Normal class
• Anomaly class

This dataset is an updated version of the widely recognized KDD99 dataset. This dataset will be used in a supervised learning classification model algorithm to create a predictive model for identifying programs that may result in system intrusion. The data set file in CSV format. In Table 3 below only some features are shown for the reference.

Table 3. Intrusion detection training data set

Duration	Protocol_type	Service
0	Tcp	Private
0	Tcp	Private

The algorithm's simulation results, which are generated after processing the dataset, will be shown in the upcoming sections of the paper.

4.2 Experimental Design

In this experimental stage, we create two programs for simulating the implementation of machine learning to improve the security of computer devices. Figure 4 shows the data flow diagram of machine learning performance test. Similarly, in the Figure 5 shows data flow diagram of security purpose model generator.

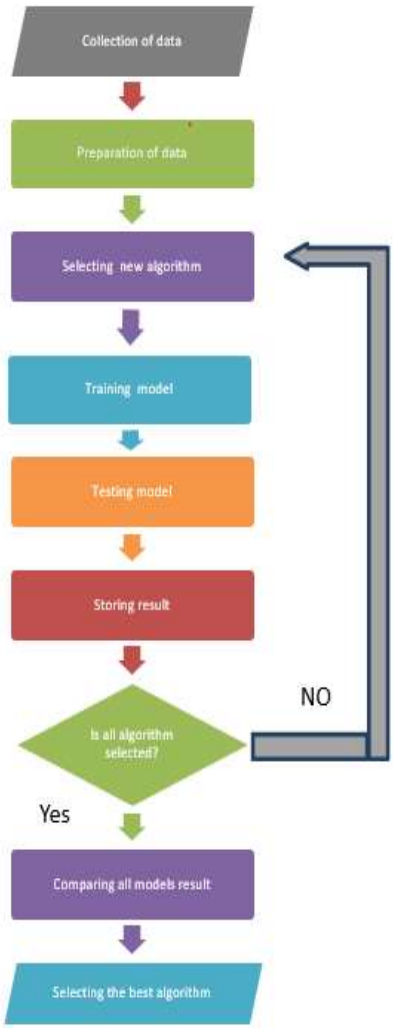

Figure 3. DFD of machine learning performance test.

Figure 4. DFD of security purpose model generator

4.3 Program

Python is the most commonly used programming language in the field of artificial intelligence, and it is frequently utilized for programming purposes. Python is frequently selected for teaching machine learning and detecting malware because of its ease of use, flexibility, and extensive library support. In educational settings, Python's clear syntax and extensive documentation make it accessible for beginners, allowing them to grasp fundamental concepts of machine learning without being bogged down by complex syntax. Moreover, popular libraries like Scikit-learn and TensorFlow provide powerful tools for building and deploying machine learning models, streamlining the process of developing algorithms for malware detection. Python's flexibility also enables rapid prototyping and experimentation, crucial for refining detection techniques in the ever-evolving landscape of cyber security threats. In general, Python is an excellent option for

both teaching machine learning principles and developing practical solutions for malware detection due to its user-friendly nature, robust libraries, and flexibility. (Shafiq, 2018)

VS code was used for coding where as for the output Python Shell was used.

Visual Studio Code is a compact yet robust source code editor designed to run on your computer, accessible for Windows, MacOS, and Linux operating systems. It includes native backing for JavaScript, TypeScript, and Node.js, along with a comprehensive Python language extensions ecosystem.

Python Shell is used as a language for translating Python. It implies that it utilizes code in a sequential manner. Python offers Python Shell, also referred to as Python Interactive Shell, for executing individual Python commands and displaying the output. The Python Shell is ready for the user to input the installation command. Once a user inputs a command, it is promptly carried out and the outcome is shown.

Python comes with a number of packages.

Pandas is a BSD-licensed open-source library that provides Python programming language tools for data analysis with high performance and user-friendly architecture.

NumPy is short for Numerical Python. It is a library that contains multi-dimensional array objects and a set of procedures for manipulating those arrays.

Matplotlib is widely utilized in Python for visualizing data, being one of the most popular packages. It is a library that can be used on multiple platforms to create 2D websites using data arrays.

Scikit-Learn, formerly called scikits and also known as sklearn, is a no-cost library software made for Python programming. Includes various types of classification, deceleration, and integration algorithms like Support Vector Machine (SVM), Naïve Bayes, random forests, K-Nearest Neighbour (KNN), gradient boosting, clustering methods such as means and DBSCAN, designed to work well with Python numerical science libraries NumPy and SciPy.

In Python, the main use of pickling is to encode and decode a hierarchy of Python objects. Simply put, it means converting a Python object into a series of bytes to store it in a file or database, maintain program state between sessions, or send data across the network.

In Figure 6, the code tackles malware detection using machine learning. It employs feature selection with ExtraTreesClassifier to identify the most informative data from the dataset relevant to malware classification. This refined data is then used to train various machine learning algorithms like K-Nearest Neighbors, Naive Bayes, Decision Trees, and Random Forests. Each algorithm is evaluated on its ability to distinguish legitimate software from malware using a split test dataset. The code analyzes the performance of each algorithm through confusion matrices and accuracy scores, ultimately selecting the "winner" with the highest accuracy. Finally, the winning algorithm and the chosen features are saved for future malware prediction. In essence, this code demonstrates how machine learning techniques can be implemented for malware detection, showcasing feature selection and exploring various classification algorithms. However, it focuses on the general approach rather than specifically optimizing algorithms for the most challenging malware threats.

```python
# malware Detection
#Importing libraries
import pandas as  pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn.ensemble as ske
import pickle
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.feature_selection import SelectFromModel

from sklearn.metrics import confusion_matrix, classification_report

# Loading datset
print('loading dataset')
dataset= pd.read_csv('data.csv', sep='|')

print ('dataset loaded'+ '\nSplitting dataset')
X= dataset.drop(['Name','md5','legitimate'], axis=1).values
y= dataset['legitimate'].values

# Feature selection using Trees Classifier
print('Feature selection on process')
fsel = ske.ExtraTreesClassifier().fit(X, y)
model = SelectFromModel(fsel, prefit=True)
X_new = model.transform(X)
nb_features = X_new.shape[1]

#Splitting train test values
print('Splitting train test Test')
X_train, X_test, y_train, y_test = train_test_split(X_new, y, test_size=0.3)

features = []

print('%i features identified as important:' % nb_features)

indices = np.argsort(fsel.feature_importances_)[::-1][:nb_features]
for f in range(nb_features):
    print("%d. %s (%f)" % (f + 1, dataset.columns[2+indices[f]], fsel.feature_importances_[indices[f]]))

# Take care of the feature order
for f in sorted(np.argsort(fsel.feature_importances_)[::-1][:nb_features]):
    features.append(dataset.columns[2+f])

#Defining Algorithms
algorithms={
    "K_Neighbors" : KNeighborsClassifier(n_neighbors=9),
    "Navies_Bays" : GaussianNB(),
    "Decision_Tree": DecisionTreeClassifier(max_depth=10),
    "Random_Forest": RandomForestClassifier(n_estimators=50),
}
results={}


#Fitting to algorithms
for algo in algorithms:
    cf=algorithms[algo]
    print('\n%s Model training started' %algo)
    cf.fit(X_train, y_train)
```

```
65
66      print('%s Testing Model' %algo)
67      y_pred= cf.predict(X_test)
68
69      print('Generating Confusion matrix of %s' %algo)
70      print ( confusion_matrix(y_test,y_pred))
71
72      #print('Generating Classification model of %s'%algo)
73      #print ( classification_report(y_test,y_pred))
74      print('%s Score='%algo)
75      results[algo]= cf.score(X_test, y_test)
76      print(results[algo])
77
78  #Finding Winner
79  winner = max(results, key=results.get)
80  print('\nWinner algorithm is %s with a %f %% success' % (winner, results[winner]*100))
81
82  # Save the algorithm and the feature list for later predictions
83  print('Saving algorithm and feature list in classifier directory...')
84  tm_fn= open('classifier/test_model', 'wb')
85  pickle.dump(algorithms[winner], tm_fn)
86  tm_fn.close()
87
88  #pickle.dump(algorithms[winner], open (filename, 'wb'))
89  f_fn= open('classifier/features', 'wb')
90  pickle.dump(features,f_fn)
91  print('Saved')
92  f_fn.close()
93
94  print('End of the program')
95  p_sto=input("Press any key to stop")
```

Figure 5. Snapshot of test model program

In Figure 7, the model generator program, machine learning techniques, such as ExtraTreesClassifier and Random Forest classifiers were used to analyse and classify malware. ExtraTreesClassifier helps identify the most important features from data for malware detection, and Random Forest, known for its robustness and handling complex data, is then trained on this refined data to distinguish between legitimate and malicious software. This approach relies on a dataset containing information that can be used to identify malware, such as file size, origin, code patterns, and system resource usage. By effectively detecting both common and hard-to-detect threats, machine learning models can significantly improve the security of computing environments.

```python
import pandas as pd
import pickle as pk
import numpy as np
import math

from sklearn.ensemble import ExtraTreesClassifier, RandomForestClassifier
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import classification_report, confusion_matrix

from sklearn.model_selection import train_test_split

#Creating Data Analysis Function
def analysis_data(df):
    cols=df.columns.values
    total = float(len(df))
    print("{} columns and {} rows".format(len(cols), int(total)))
    for col in cols:
        uniques=df[col].unique()
        unique_count= len(uniques)
        if (unique_count>math.log(total,2)):
            print("# {}\t#Unique Count: {} ({}%)".format(col,unique_count,int(((unique_count)/total)*100)))
        else:
            result=[]
            s=df[col].value_counts()
            t=float(len(df[col]))
            for v in s.index:
                result.append("{}:{}%".format(v,round(100*(s[v]/t),2)))
            print("# {}\t#Unique Count: {} [{}]".format(col,unique_count, result))

#Creating dummy_encoding for non numeric value feature function
def dummy_encoding(df):
    df1=pd.DataFrame()
    cols=df.columns.values
    for col in cols:
        d_typ=str(dataset[col].dtypes)
        if (d_typ != 'int64' and d_typ !='float64'):
            dummies=pd.get_dummies(dataset[col])
            for x in dummies.columns:
                dummy_name= f"{col}-{x}"
                df1[dummy_name]=dummies[x]
        else:
            df1[col]=dataset[col]
    return df1
#Reading Data file location
data_loc='data.csv'

# Reading decision making Feature
d_fea='legitimate'

#Reading decision value
d_val=0

#Reading unnecessary features
u_fea=['Name','md5']

#Creating leave feature array
l_fea=u_fea
l_fea.append(d_fea)

#Loading train dataset
print('\nLoading Dataset')
dataset= pd.read_csv(data_loc, sep='|')
```

```
#print(dataset.info())
#print(dataset.shape)
#print(type(dataset))

# Splitting Dependent and Independed variable
print('\n\nSplitting Dependent and Independent Variables')
X=pd.DataFrame(dataset)

X.drop(l_fea , axis='columns', inplace=True)

y= pd.DataFrame(dataset[d_fea])

#Analyzing input features
print('\n\nAnalyzing input features')
analysis_data(X)

#Analyzing output feature
print('\n\nAnalyzing output feature')
analysis_data(y)

#Preprocessing on input features
print('\n\nPre-processing in input features')
X_new=dummy_encoding(X) #only encode non numerical values


#Preprocessing on output feature
print('Pre-processing on output feature')
d_typ=str(y[d_fea].dtypes)
if (d_typ != 'int64' and d_typ !='float64'):
    sel_fea=d_fea + "_"+d_val
    dummys= pd.get_dummies(y)
```

```
#print(dataset.info())
#print(dataset.shape)
#print(type(dataset))
# Splitting Dependent and Independed variable
print('\n\nSplitting Dependent and Independent Variables')
X=pd.DataFrame(dataset)
X.drop(l_fea , axis='columns', inplace=True)
y= pd.DataFrame(dataset[d_fea])
#Analyzing input features
print('\n\nAnalyzing input features')
analysis_data(X)
#Analyzing output feature
print('\n\nAnalyzing output feature')
analysis_data(y)
#Preprocessing on input features
print('\n\nPre-processing in input features')
X_new=dummy_encoding(X) #only encode non numerical values
#Preprocessing on output feature
print('Pre-processing on output feature')
d_typ=str(y[d_fea].dtypes)
if (d_typ != 'int64' and d_typ !='float64'):
    sel_fea=d_fea + "_"+d_val
    dummys= pd.get_dummies(y)
    y_new=pd.DataFrame(dummys[sel_fea])
else:
    y_new=y
#Feature Selection using tree classifier
print('\n\nExtracting Features')
cla= ExtraTreesClassifier()
fsel=cla.fit(X_new,y_new)
model=SelectFromModel(fsel, prefit=True)
```

```
X_new1=model.transform(X_new)
nb_features= X_new1.shape[1]
print('\n%i were inlisted as important features from %i features.\n'%(nb_features, X_new.shape[1]))
features = []
indices = np.argsort(fsel.feature_importances_)[::-1][:nb_features]
for f in range(nb_features):
    print("%d.  %s (%f)" % (f + 1, X_new.columns[indices[f]], fsel.feature_importances_[indices[f]]*100))
# Take care of the feature order
for f in sorted(np.argsort(fsel.feature_importances_)[::-1][:nb_features]):
    features.append(X_new.columns[f])
#Splitting train test values
print('\nSplitting train test Test')
X_train, X_test, y_train, y_test = train_test_split(X_new1, y_new, test_size=0.3)
cf=RandomForestClassifier(n_estimators=50)
print('\n\n Model training started using Random Forest')
cf.fit(X_train, y_train)
print('\nTesting Model:Random Forest')
y_pred= cf.predict(X_test)
print('\nGenerating Confusion matrix')
print ( confusion_matrix(y_test,y_pred))
print('\nGenerating Classification model ')
print ( classification_report(y_test,y_pred))
print('\nScore=')
score= cf.score(X_test, y_test)
print(score)
print('\nStoring features and model')
print('\nEnd of Program')
sto_prg=input("Enter any key to stop program...")
```

Figure 6. Snapshot of model generator program

4.4 Models

Using two diverse types of data in our program for generating a model focused on ML security, we will develop two classification models.

i. Malware Detection Model

This model will determine if the program is genuine by analysing the file's characteristics. This model is based on supervision and is used for machine learning classification. This model was created following training on a dataset of PE files. Each of the five classification algorithms mentioned in the supervised learning method will be utilized to create models, which will then be evaluated and compared in terms of their effectiveness. The most efficient algorithm in generating models will be identified.

ii. Network intrusion Detection Model

This model will determine if the network connection is an intrusion based on how the connection is being utilized within a computer network. This model is based on supervised machine learning for classification. This model was created following

training on the Intrusion feature dataset. All five classification algorithms mentioned in the supervised learning method will be applied to create models, which will then be evaluated for efficiency and the best algorithm for generating models will be identified.

5 RESULTS

As there are two kinds of result which are mentioned below.

5.1 Machine Learning Algorithm Test Result

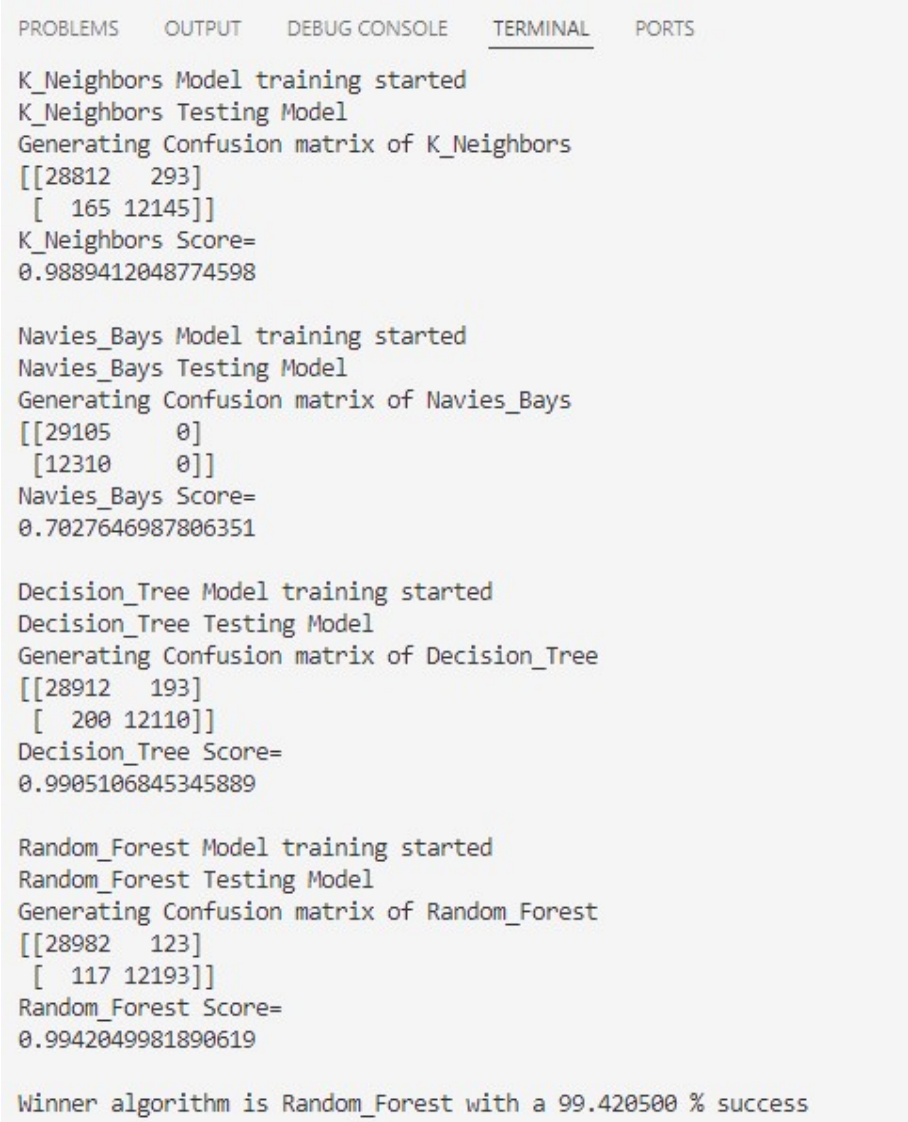

Figure 7. Machine learning algorithm performance test

So, in Figure 8, It shows the test there are four kinds of algorithm used in testing which are: K-Nearest Neighbor, Decision Tree, Navies Bayes and Random Forest. Among them winner is Random Forest with a 99.42% success.

5.2 Importance of Machine Learning Algorithm Test Result

From the test, we get four different results. So, for malware detection and intrusion detection in application and network layer we need two types of dataset respectively for the output.

5.2.1 Application Layer

For detection of malware, we used PE dataset file which gave us following output shown. In Figure 9, ***It*** shows the input features of the dataset file for malware detection.

```
Splitting Dependent and Independent Variables


Analyzing input features
54 columns and 138047 rows
# Machine       #Unique Count: 3 [['332:88.47%', '34404:11.53%', '512:0.0%']]
# SizeOfOptionalHeader  #Unique Count: 5 [['224:88.47%', '240:11.53%', '248:0.0%', '352:0.0%', '232:0.0%']]
# Characteristics       #Unique Count: 104 (0%)
# MajorLinkerVersion    #Unique Count: 41 (0%)
# MinorLinkerVersion    #Unique Count: 62 (0%)
# SizeOfCode    #Unique Count: 3809 (2%)
# SizeOfInitializedData #Unique Count: 3217 (2%)
# SizeOfUninitializedData       #Unique Count: 441 (0%)
# AddressOfEntryPoint   #Unique Count: 23110 (16%)
# BaseOfCode    #Unique Count: 385 (0%)
# BaseOfData    #Unique Count: 1106 (0%)
# ImageBase     #Unique Count: 9099 (6%)
```

Figure 8: Test on input features.

From Figure 9, the test report where 54 input features are included which are different from one another. In the figure 10, analyzing the output features of the dataset provided to machine learning algorithm.

```
Analyzing output feature
1 columns and 138047 rows
# legitimate    #Unique Count: 2 [['0:70.07%', '1:29.93%']]


Pre-processing in input features
Pre-processing on output feature
```

Figure 9. Report on output feature

Figure 10 shows report on the output feature of the dataset we provided. It also displays the count of how many unique values are present in the feature. Feature is classified as either '0' or '1' for legitimate designation.

Figure 11 displays the important input features and their probability weightage, crucial for determining if the PE files are malicious or genuine.

```
14 were inlisted as important features from 54 features.

1.  DllCharacteristics (15.509843)
2.  Characteristics (11.759057)
3.  Machine (11.519488)
4.  SectionsMaxEntropy (6.607611)
5.  VersionInformationSize (6.579500)
6.  Subsystem (5.100871)
7.  ResourcesMinEntropy (4.658047)
8.  SizeOfOptionalHeader (4.077273)
9.  ImageBase (3.623519)
10.  ResourcesMaxEntropy (3.470288)
11.  MajorSubsystemVersion (3.166694)
12.  SizeOfStackReserve (2.463675)
13.  MajorOperatingSystemVersion (2.112706)
14.  SectionsMinEntropy (2.052429)
```

Figure 10. Report on extracting input feature

The ML algorithm calculates the probability weight age of important features from the dataset provided. The outcome is saved in the model created by the program's conclusion.

Figure 12 displays the Confusion matrix, Classification model, and Score of Random Forest ML algorithm for the application layer data set.

```
Testing Model:Random Forest

Generating Confusion matrix
[[28904   137]
 [  114 12260]]

Generating Classification model
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     29041
           1       0.99      0.99      0.99     12374

    accuracy                           0.99     41415
   macro avg       0.99      0.99      0.99     41415
weighted avg       0.99      0.99      0.99     41415


Score=
0.9939393939393939

Storing features and model

End of Program
Enter any key to stop program...
```

Figure 11. Machine learning algorithm performance score

The model accurately detected 28,904 true positives, indicating a high number of correctly identified threats. It also correctly recognized 12,260 true negatives, showing its ability to identify non-threatening activities. However, there were 137 false positives, where benign activities were incorrectly flagged as threats, and 114 false negatives, where actual threats were missed and classified as non-threatening. This result can be written in confusion matrix, and overall test score is 99.39%

True Positive = 28904

False Positive = 137

True Negative = 12260

False Negative = 114

5.2.2 Network Layer

For intrusion detection in network layer we have performed following four tests which gives us such results which are mentioned below.

```
Splitting Dependent and Independent Variables


Analyzing input features
41 columns and 25192 rows
# Feature:duration      #Unique Count: 758 (3%)
# Feature:protocol_type:['tcp:81.48%', 'udp:11.95%', 'icmp:6.57%']
# Feature:service       #Unique Count: 66 (0%)
# Feature:flag:['SF:59.44%', 'S0:27.82%', 'REJ:8.8%', 'RSTR:1.97%', 'RSTO:1.21%', 'S1:0.35%', 'SH:0.17%', 'RSTOS0:0.08%', 'S2:0.
08%', 'S3:0.06%', 'OTH:0.02%']
# Feature:src_bytes     #Unique Count: 1665 (6%)
# Feature:dst_bytes     #Unique Count: 3922 (15%)
```

Figure 12. Report of input features

As we can see in the Figure 13, It contains 41 input features on dataset which are different from one another so the result is more accurate. In the Figure 18 shows the reports of the output features of the dataset of the network layer.

```
Analyzing output feature
1 columns and 25192 rows
# Feature:class:['normal:53.39%', 'anomaly:46.61%']


Pre-processing in input features
C:\Users\Numbur iT\OneDrive\Desktop\Thesis-VAMK-Final 2024\Shiva\Final Thesis\Intrusion Detection Program\Intrusion_detect_model
_generator.py:40: PerformanceWarning: DataFrame is highly fragmented.  This is usually the result of calling `frame.insert` many
 times, which has poor performance.  Consider joining all columns at once using pd.concat(axis=1) instead. To get a de-fragmente
d frame, use `newframe = frame.copy()`
  df1[col]=dataset[col]
```

Figure 13. Report of output feature.

After running the dataset in the output feature one column and 25192 rows of the dataset are affected which produced the feature class. In class feature has the different values. In class feature, it is divided into two parts which are normal and anomaly in above dataset 53.39% is normal and as of anomaly it is 46.61%.

Figure 19 shows us the input features of the dataset in listed in the machine learning algorithm. It will be easier to check activities carried out in the network if they are normal or causing problems in the network.

```
28 were inlisted as important features from 118 features.

1. feature: flag-SF (10.816023)
2. feature: logged_in (9.036249)
3. feature: dst_host_same_srv_rate (7.010558)
4. feature: same_srv_rate (6.646377)
5. feature: dst_host_srv_count (6.127841)
6. feature: srv_serror_rate (5.780541)
7. feature: serror_rate (4.333238)
8. feature: dst_host_srv_serror_rate (3.766810)
9. feature: dst_host_serror_rate (3.701524)
10. feature: protocol_type-icmp (3.373292)
11. feature: service-http (3.108801)
12. feature: dst_host_same_src_port_rate (2.390017)
13. feature: src_bytes (2.373589)
14. feature: flag-S0 (2.089160)
15. feature: dst_host_count (2.009891)
16. feature: service-private (1.938319)
17. feature: dst_host_rerror_rate (1.837439)
18. feature: protocol_type-udp (1.827223)
19. feature: count (1.774098)
20. feature: dst_host_srv_rerror_rate (1.717513)
21. feature: srv_rerror_rate (1.413669)
22. feature: protocol_type-tcp (1.386040)
23. feature: rerror_rate (1.272034)
24. feature: service-ecr_i (1.269820)
25. feature: service-eco_i (1.230087)
26. feature: dst_host_srv_diff_host_rate (1.108508)
27. feature: hot (0.938990)
28. feature: dst_bytes (0.891444)
```

Figure14. Significant input feature extraction report

While there were initially 28 input features, the total number listed now is 118 dues to the inclusion of both qualitative and quantitative input features. The use of dummy encoding on the qualitative feature resulted in the increase in the number of features.

Figure 16 displays the Confusion matrix, Classification model, and Score of the Random Forest ML algorithm applied to the network layer dataset.

```
Testing Model:Random Forest

Generating Confusion matrix
[[4033    3]
 [  21 3501]]

Generating Classification model
              precision    recall  f1-score   support

       False       0.99      1.00      1.00      4036
        True       1.00      0.99      1.00      3522

    accuracy                           1.00      7558
   macro avg       1.00      1.00      1.00      7558
weighted avg       1.00      1.00      1.00      7558


Score=
0.9968245567610479

End of Program
Closing program
```

Figure15. Machine learning algorithm test in network layer

The confusion matrix for the malware and intrusion detection model reveals a highly effective performance, with a true positive count of 4033 and a true negative count of 3501, indicating the model's strong ability to correctly identify both malicious activities and benign activities. The false positive count is remarkably low at 3, showing that the model rarely misclassifies benign activities as malicious, which minimizes unnecessary alerts. Similarly, the false negative count is 21, suggesting that the model has a very low rate of missing actual threats. The overall accuracy score of 99.68% underscores the model's exceptional precision and reliability in distinguishing between malicious and non-malicious activities, making it a robust tool for enhancing cybersecurity measures.

In summary confusion matrix,

True Positive = 4033

False Positive = 3

True Negative = 3501

False Negative = 21

Score = 99.68%.

5.3 Files Generated at the End of Simulation

Figure 17 shows the feature and test model files generated by the models which are stored in classifier in the computer after simulation.

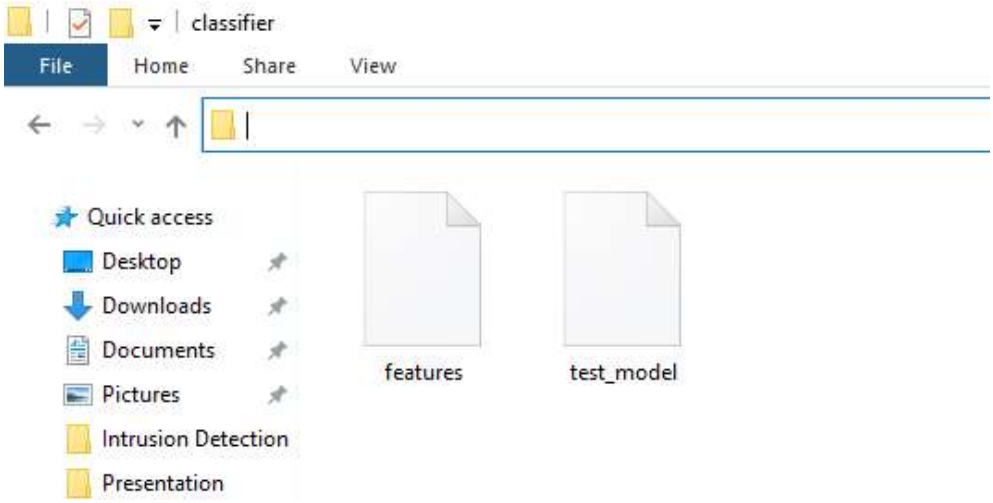

Figure 16. Generated files after simulation

A feature file contains the extracted features list generated by the machine learning software. The program can utilize the listed features as main criteria for evaluating an activity as a threat or not. Feature file in txt format and the size of file is about 3.12 kilobytes.

The test model file is the ML program-generated model that can predict the activity as a threat or not by inputting activity/file features into the model. The created model has the ability to make decisions independently. Test model file is in pkl format and the size of model is 980 kilobytes.

6 DISCUSSION AND CONCLUSION

If the program constantly updates its dataset based on previous decisions, it can adjust its probability values for features at regular intervals without any sudden changes in the attack model.

The data we are utilizing for machine learning is not up to date. We only utilized Machine Learning in order to assess its applicability. In order to create the model reflecting the current status, the most recent dataset should be utilized for training.

One program can be used for various layers by making small adjustments to the dataset parameters. This program is capable of creating models for other datasets with a similar classification type as well.

The first research question was how machine learning techniques can be optimized to detect the most common and hardest-to-detect malware and intrusion attacks, and what impact do these attacks have on the security of modern computing environments.

Machine learning techniques can be optimized for detecting the most common and hardest-to-detect malware and intrusion attacks through careful data pre-processing, feature extraction, and algorithm selection. The simulation programs developed in the study demonstrated the potential of machine learning to enhance IoT security. The key optimization steps include:

1. Data Pre-processing: Using dummy encoding for categorical data and ensuring all features are numerical, as done in the "Application Layer" and "Network Layer" data pre-processing stages.
2. Feature Extraction: Identifying and focusing on important features, as seen with the 14 important features in the "Application Layer" and 28 in the "Network Layer."
3. Algorithm Tuning: Testing various machine learning algorithms and selecting the most effective one, like the "Random Forest" algorithm chosen for its performance in the study.

The impact of optimizing machine learning techniques for these tasks is significant. It enhances the ability to detect and respond to security threats in real-time, improves the overall security of IoT devices, and helps mitigate common attacks like DoS/DDoS, jamming, network analysis, and botnet attacks. Despite the promising results, the study acknowledges that machine learning is not a complete solution and stresses the importance of ongoing learning from new data to adapt to emerging threats.

The second research question was what types of Machine Learning algorithms are used for protecting the devices from attacks.

Four types of machine learning algorithms were used for protecting devices from attacks: K-Nearest Neighbor (KNN), Naive Bayes, Decision Tree, and Random Forest. Among these four, the Random Forest algorithm gave the best score. So, in my opinion, Random Forest is the best machine learning algorithm for protecting devices from attacks.

The third research question was what kind of datasets are used for training the model.

The datasets used for training the model in the study included both numerical and categorical data. The data pre-processing stage involved transforming categorical data using dummy encoding, which increased the number of feature columns significantly (e.g., from 28 to 118 in the "Network Layer"). The datasets were obtained from csv files and included a variety of features, with a total of 54 input features in the "Application Layer" and a mix of numerical and categorical data in the "Network Layer." In practice, these features were crucial for building accurate machine learning models. However, the study notes the limitation that real-world applications require feature extraction from files or activities in real-time, rather than relying solely on pre-processed csv datasets.

6.1 Discussion of Results

The positivity of all predictions suggests a potential issue with the system design for algorithm implementation. There could be issues with either the process of splitting data, extracting key features with a tree classifier, or tuning the algorithm.

The second simulation program we developed yielded a significant outcome, demonstrating the potential for utilizing machine learning to improve the security of IoT devices. In the data pre-processing stage, machine learning utilizes dummy encoding for categorical data which transforms one column into n-1 columns, where n represents the number of unique features in the original column. Therefore, in the "Application Layer" data, all 54 input features were numerical. As a result, 14 features were identified as important in the feature extraction report. However, in the "Network Layer" data, both numerical and categorical data were included. The numerical data remained unchanged, while the categorical data was encoded using dummy encoding, resulting in an increase in the number of feature columns from 28 to 118. Therefore, in the "Network layer," a report highlighted 28 important feature extractions from a column of 118 features.

Here a machine learning model was created using two datasets. Both models achieved a score of over 99% on the test, indicating that ML implementation techniques are effective for ensuring security on IoT devices. The simulation program we developed creates and evaluates models using features data gathered from a csv file dataset. However, in real-world applications, the features needed to make predictions must be extracted from the file or activity as they will not be present in csv format. Therefore, the simulation is behind in programming the software to extract features from the file and process them to make decisions in the model.

Simulation was conducted on a computer with greater computational capabilities than IoT devices, resulting in IoT limitations preventing the execution of programs as in the simulation. IoT devices typically act as clients and rely on cloud computing for their server. The concept involves running the machine learning program on

servers to create models that can be updated to the firmware of IoT devices in order to safeguard them from attacks.

6.2 Conclusions of Result

Machine learning has the capability to enhance and strengthen IoT security with various efficient techniques and methods. Each year, incidents happen in different places and an extensive amount of data is revealed, leading to a significant problem. DoS/DDoS attacks, jamming, network analysis, and botnet attacks are some of the most common types of attacks aimed at IoT devices.

We conducted a literature review on numerous research papers, articles, and other relevant documents for this study. Different tests have been created to secure IoT devices, utilizing machine learning or deep learning, which yield superior outcomes compared to employing unregulated ML tactics like IGGM.
The future will involve using Machine Learning to categorize IoT devices and analyse data due to the abundance of devices and data. The ML implementation is the key to the future of security measures on IoT devices. Furthermore, our research showed how machine learning has the possibility to enhance security through analysing the outcomes of a simulation application.

The research uses a mixed methodology approach. The research problem and research questions demonstrate the methods used to conduct the research.
In the experimental phase, we developed the solution and created the simulation program for this study. In this study, we developed 2 programs. One plan involved evaluating various ML algorithms to determine the most appropriate one for use in the simulation program. After examining the test outcome, we decided to utilize the "Random Forest" algorithm for building the model in the simulation software.

One program can process various datasets to create various models. Machine can efficiently handle vast volumes of data in a very brief timeframe. Machine can conduct analysis on data to identify patterns and correlations among various data features.

Although the ML algorithm helps alleviate some of the worry, it does not guarantee full performance, leaving a possibility for threats and intrusion to occur. The majority of IoT relies on cloud computing because of the limited computational power available. The concept involves creating a program that simulates a server computer and creating a model on nodes or IoT devices to make decisions in order to safeguard the system against attacks. The simulation results simply demonstrate the potential for ML implementation in computing devices.

Despite the promising results, it is important to note that machine learning is not a silver bullet and doesn't guarantee complete protection. However, its potential for analysing large datasets and identifying patterns can greatly enhance security measures. As IoT devices rely heavily on cloud computing and computational power, implementing machine learning algorithms on both server computers and IoT devices can bolster the overall security framework. Ultimately, our research underscores the potential of machine learning in fortifying the security of computing devices in IoT and IT environments.

The algorithms are able to examine large volumes of data from IoT devices to find anomalies and possible threats. The model makes it easier to monitor and respond quickly to security issues. It can help to define normal behaviour patterns of IoT devices and identify any deviations. This model can pinpoint anomalies that could indicate zero-day vulnerabilities being used. The system can keep learning from new data to adjust to new threats. Concerns about data privacy and security arise due to the large amount of data needed for ML systems.

6.3 Limitation and Future Research

The study conducted on Applications of Machine learning for Malware and Intrusions Detection. However, it is important to accept its limitations and consider potential areas for further research.

This research has some limitations in how it was done. First, the simulation shows that machine learning (ML) could make IoT devices safer, but it mostly uses

datasets that might not show all the different situations IoT devices face in real life. Also, the simulation runs on powerful computers, not on regular IoT devices, so it might not show how well ML would work on those devices. Plus, the simulation cannot extract features from data in real-time, which means it is not exactly like how ML would work in real life for IoT security. In the future, researchers could fix these issues by using more varied datasets, making ML work better on IoT devices with fewer resources, and figuring out how to extract features from data as it comes in.

Another thing to consider is that if the data used is unbalanced, meaning there is a lot more of one type of data than another; it could mess up the predictions. This could lead to either missing real threats or raising false alarms when there's no threat. Finding a balance between these two is really important. Also, because the data involved might be private, using ML raises concerns about people's privacy. Figuring out how to keep people's data private while still making sure ML can detect threats effectively is a big challenge.

In terms of future research, researchers could try new ideas to make ML-based security better for IoT devices. One way is to mix ML with traditional ways of keeping things safe online. This mix could make it easier to spot and deal with threats. Also, putting more computing power directly into IoT devices could help them handle data better on their own, without relying so much on big servers far away. This could make them safer from cyber-attacks. And as ML gets better at explaining why it makes certain decisions, people will trust it more for keeping things secure. This could lead to more use of ML in real-world situations.

REFERENCES

Abie, H. (2019). *Cognitive Cybersecurity for CPS-IoT Enabled Healthcare Ecosystems.* 13th International Symposium on Medical Information and Communication Technology (ISMICT).

Abouzakhar, N. S., Andres, & Angelppoulo, O. (2017). Internet of Things Security: A Review of Risks and Threats to Healthcare Sector. *IEEE International Conference on Internet of Things (iThings)*.

Ahamed, F., & Farid, F. (2018). Applying Internet of Things and Machine-Learning for Personalized Healthcare: Issues and Challenges. *2018 International Conference on Machine Learning and Data Engineering (iCMLDE)*, 19-21.

Ahmed, Faraz, Haider Hameed, M. Zubair Shafiq, and Muddassar Farooq. 2009. Using spatio-temporal information in API calls with machine learning algorithms for malware detection, *55. New York City: ACM Press.*

Alani, M. M. (2019). Application of Machine Leraning in cryptography: A survey.

Alen Jacob Kurian, A. S. (2024). ENHANCED MALWARE DETECTION FRAMEWORK LEVERAGING MACHINE. *International Research Journal of Modernization in Engineering Technology and Science*, 7.

Ashwini Katkar, Sakshi Shukla, Danish Shaikh, Pradip Dange, 12 August 2021, Malware Intrusion Detection For System Security

Branley-Bell, L. C. (2018). *Cybersecurity in healthcare: A narrative review of trends, threats and ways forward.* Maturitas: maturitas.

Brown, L. &. (2024). Dataset Challenges in Machine Learning-Powered IoT Security: A Review. *International Journal of Information Security and Privacy*, 8(2), 78-95.

Buczak, A. L. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials, 18(2),* , 1153-1176.

Chen, Y. W. (2023). A Decision Tree-based Approach for Intrusion Detection. *IEEE Transactions on Information Forensics and Security, 18(5), doi:10.1109/TIFS.2022.3141592*, 1234-1245.

Denial-of-Service Attack:"https://en.wikipedia.org/wiki/Denial-of-service_attack"

Doe, A. M. (2024). The Role of Machine Learning in Securing IoT Devices Across Industries. *Journal of Advanced Computing and Security*, 12(1),* , 75-101.

Fazeldehkordi, E., Owe, O., & Noll, J. (2019). Security and Privacy in IoT Systems: A Case Study. *International Symposium on Medical Information and Communication Technology (ISMICT)*.

Floyd, T., & Reid, M. G. (2016). Mining hospital data breach records: Cyber threats to U.S. hospitals. *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*.

Genaro Almaraz-Rivera, Jose Antonio Cantoral-Ceballos , AndJuan Felipe Botero, 25 October 2023, Enhancing IoT Network Security: Unveiling the Power of Self-Supervised Learning against DDoS Attacks

Germain, K. S., & Kragh, F. (2020). Physical-Layer Authentication Using Channel State Information and Machine Learning. *2020 14th International Conference on Signal Processing and Communication Systems (ICSPCS).* Adelaide, SA, Australia: IEEE.

Hussain, F., Hussain, R., Hassan, S. A., & Hossain, a. E. (2020 May 30). Machine Learning in IoT security: Current Solutions and Future Challenges. *IEEE Communications Surveys & Tutorials*.

Intrusion detection and intrusion prevention: https://www.imperva.com /learn/application-security/intrusion-detection-prevention/ Access on 13th March 2024

Kumar, P. M., & Gandhi, U. D. (2018). A novel three-tier Internet of Things architecture with Machine Learning algorithm for early detection of heart diseases. *compters and electrical engineering*, 222-235.

Li, H. Z. (2024). Advancements in Naive Bayes Models for Intrusion and Malware Detection: A Survey. *Journal of Cybersecurity Research, 5(2), doi:10.1101/123456).*, 67-78.

Li, S. &.-3. (2023). Application of K-Nearest Neighbor Algorithm in Intrusion Detection. *International Journal of Network Security & Its Applications*.

Li, u. T., & Ju, H. (2024). The Application of Machine Learning in Network Security Research. *Guangzhou Institute of Technology, China*.

Li, X. Z. (2024). Malware Detection using Decision Trees. . *Journal of Computer Virology and Hacking Techniques, 32(3), 567-578. doi:10.1007/s11416-023-04567-8*.

Liang , F., Hatcher, W. G., WeixianLiao , W. G., & Yu, A. W. (October 22, 2020). Machine Learning for Security and the Internet of Things: The Good, the Bad, and the Ugly. *Department of Computer and Information Sciences, 7*.

Liu, T., & Ju, H. (2024). The Application of Machine Learning in Network Security Research. *Guangzhou Institute of Technology, China*.

Mary Landesman, February 9, 2023, A Brief History of Malware: https://www.lifewire.com/brief-history-of-malware-153616

Mohammad Mansour, Amal Gamal , Ahmed I. Ahmed, Lobna A. Said ,Abdelmoniem Elbaz ,Norbert Herencsar, and Ahmed Soltan, 14 April 2023, Internet of Things: A Comprehensive Overview on Protocols, Architectures, Technologies, Simulation Tools, and Future Directions

Mohammad Kamrul Hasan, A. A. (2023). DDoS: Distributed denial of service attack in communication standard vulnerabilities in smart grid applications and cyber security with recent developments. *Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), 43600 UKM, Bangi, Selangor, Malaysia*.

Muhammad Shoaib Akhtar, T. F. (2022). Malware Analysis and Detection Using Machine Learning Algorithms. *School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China*.

Naveen, Sharma, D. R., & Nair, D. A. (2019). IoT-based Secure Healthcare Monitoring System. *IEEE*.

Nithya, B., & Ilango, D. V. (2017). Predictive Analytics in Health Care. *International Conference on Intelligent Computing and Control Systems*.

Nor Zakiah Gorment, Ali Selamat, Lim Kok Cheng And Ondrej Krejcar 2022, Machine Learning Algorithm for Malware Detection: Taxonomy, Current Challenges and Future Directions.

Omar Bawazeer, T. H.-h. (2021). Malware Detection Using Machine Learning Algorithms. *The 1st International Conference on Engineering and Technology (ICoEngTech) 2021* (p. 11). Department of Information and Computer Science, College of Computer Sciences and Engineering,.

Panda, S., & Panda, G. (2020). Intelligent Classification of IoT Traffic in Healthcare. 6th International Conference on Control, Automation and Robotics.

Pascal Maniriho, A. N. (2023). A systematic literature review on Windows malware detection: Techniques, research issues, and future directions. *Department of Computer Science and Information Technology*.

Patel, A. G. (2023). Application of Decision Tree Algorithm for Intrusion and Malware Detection: A Comprehensive Review. *International Journal of Information Security and Cybercrime, 12(3),* , 456-467.

Pierpaolo Dini, A. E. (25 june 2023). Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity. *Department of Information Engineering, University of Pisa, 56126 Pisa, Italy*.

Pirbhulal, S., Pombo, N., Felizardo, V., & Garcia, N. (2019). Towards Machine Learning Enabled Security Framework for IoT-based Healthcare. Covilhã, Portugal: Thirteenth Internation Conference on Sensing Technology.

Pirbhulal, S., Zhang, H., & Wu, W. (2017). HRV-Based Biometric PrivacyPreserving and Security Mechanism for Wireless Body Sensor. In S. M. Islam (Ed.), *Wearable Sensors Applications, design and implementation* (pp. 1-25). Springer Books.

Rahman Ali, Asmat Ali, Farkhund Iqbal, Mohammed Hussain, and Farhan Ullah, 2022, Deep Learning Methods for Malware and Intrusion Detection: A Systematic Literature *Review: https://www.hindawi.com/journals/scn/2022/2959222/* Acces on 2nd march 2024

Rohini, K., & Kumari, B. A. (Jul. 2019). MachineLearning Techniquesfor Security Issues in Internet of Things. *nternational Journal on Future Revolution in Computer Science & Communication Engineering , 5*(6), 46-52.

Sethi, K., Chaudhary, S.K., Tripathy, B.K., Bera, P.: A novel malware analysis for malware detection and classification using machine learning algorithms. *In: Proceedings of the 10th International Conference on Security of Information and Networks*, pp. 107–113 (2017, October)

Shafiq, M. Z. (2018). A survey of machine learning techniques for malware detection. *IEEE Access, 6, 12865-12880.*

Shhadat, I., Bataineh, B., Hayjneh, A., & A.Al-sharif, Z. (April 6 2020). The Use of Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware. *International workshop on data-driven security*.

Smith, L J. (2024). Leveraging Machine Learning for Enhanced Cyber Security: A Review. *Journal of Computer Security, 12(3)*, 45-67.

Smith, L. J. (2023). Factors Driving the Proliferation of Machine Learning Research in Malware Detection. *International Journal of Cybersecurity and Digital Forensics, 10(2),* , 89-112.

Smith, J. (2024). Understanding the Evolution of Malware. Cyber security Journal. https://www.cybersecurityjournal.com/articles/evolution-of-malware

Sowmya T., M. A. (2023). A comprehensive review of AI based intrusion detection system. *Department of CSE, Christ University Bangalore, CMR Institute of Technology, Bengaluru, India*.

Symantec, "The Evolution of Malware", https://www.symantec.com/blogs/threat-intelligence/evolution-of-malware).

Tahsien, S. M., Karimipour, H., & Spachos, P. (April 7 2020). Machine learning based solutions for security of IoT. *Journal of Network and Computer Applications*.

The European Parliament and the Council of The European Union, 2002. Directive 2002/58/EC concerning the processing of personal data and the protection of privacy in the electronic communications sector. *Official J.Eur. Communities,* Volume L201, pp. 37-47.

Vinayakumar, R. S. (2019). Applying Deep Learning Approaches for Network Traffic Classification and Intrusion Detection. In R. S. Vinayakumar, *Handbook of Computer Networks and Cyber Security,* (pp. 107-141.).

Wilson, H. J. (Nov. 17, 2014). The Cognitive Usefulness of the Internet of Things. *Harvard Business Review*.

What Is a Research Methodology? | Steps & Tips: https://www.scribbr.com /dissertation/methodology/

Xioo, L., Lu, X., Zhang, Y., & Wu, D. (May 30, 2020). IoT Security Techniques Based on Machine. *National Mobile Communications Research Laboratory*.

Zhang, W. L. (2024). Recent Advances in Random Forests: A Comprehensive Survey and Review. *IEEE Access, 12(1), 12345-12356. doi:10.1109/ACCESS.2023.3078136).*