UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Cyber Security Curriculum

Anel Abylkassymova

Machine Learning Method For Detecting Botnet Attacks Originated From The IoT Networks

Master's Thesis (24 ECTS)

Supervisor(s): Hayretdin Bahsi, Ph.D.
Sven Nõmm, Ph. D,
Raimundas Matulevicius, Ph. D

Tartu 2022

Machine Learning Method For Detecting Botnet Attacks Originated From The Iot Networks

Abstract: Recently, botnet attacks have become more sophisticated than other malware since they can expand to other devices and cause even more damage. The botnet attacks are large-scale attacks that can compromise IoT devices due to their lack of security measures. An intrusion detection system (IDS) is used to monitor the network traffic and capture the suspicious traffic. The IDS, based on the machine learning approach, has been more utilized by security analysts for IoT botnet detection. This approach applies a machine learning model to enhance the IoT botnet detection process. The botnet attack development causes the demand to improve the botnet detection workflow.

The botnet attacks are multi-stage attacks that harm systems gradually. Thereby, there is a need for early stages of attack detection to prevent malware from expanding. Although some machine learning-based research papers focused only on malware detection, those papers did not consider the structure of IoT botnet attacks, which can also include a multi-stage attack approach. Also, there is a problem with the IoT device type identification. The compromised IoT devices in the IoT environment should be defined to prevent the spread of malware.

Thereby, this thesis is intended to improve the malware detection procedure utilizing different machine learning methods that not only address binary classification problems but also can be applied in early attack stages detection with a categorization of malware with device type and attack stage with the device type. The binary classification models defined the IoT botnet malware such as Mirai, Bashlite, and Torii. The multiclass classification models included: 4 classes of scenarios with three malware types and legitimate traffic, 8 classes of an experiment that differentiated the malware type and device type, and 12 classes of scenarios that distinguished the attack stage, whether it was command and control (C&C) or spread and the device type.

CERCS: **T120 System technology, computer technology**
Keywords: IoT, Botnet, Machine learning, Supervised learning

Masinõppe Meetod Botnet Rünnakute Tuvastamiseks Pärineb Asjade Interneti Võrkudest

Lühikokkuvõte: Viimasel ajal on botivõrgu (botnet) rünnakud muutunud keerukamaks kui muu pahavara, kuna need võivad laieneda teistele seadmetele ja põhjustada rohkem kahju. Botivõrgu rünnakud on suuremahulised rünnakud, mis võivad ohustada IoT-seadmeid nende turvameetmete puudumise tõttu. Sissetungituvastussüsteemi (IDS) kasutatakse võrguliikluse jälgimiseks ja kahtlase liikluse tuvastamiseks. IoT botivõrgu tuvastamiseks on rohkem kasutatud masinõppel põhinevat IDS-i. See lähenemisviis rakendab masinõppe mudelit, et täiustada asjade Interneti botneti tuvastamise protsessi. Botivõrgu rünnakute arendamine nõuab robotivõrgu tuvastamise töövoo täiustamist.

Botivõrgu rünnakud on mitmeastmelised rünnakud, mis kahjustavad süsteeme järk-järgult. Seetõttu on vaja rünnet tuvastada varajases staadiumis, et vältida pahavara laienemist. Kuigi on tehtud masinõppel põhinevaid uurimistöid, mis keskendusid pahavara tuvastamisele, ei võtnud need arvesse asjade Interneti botneti rünnakute struktuuri, mis võib hõlmata ka mitmeastmelise ründe tüüpi. Samuti on probleem IoT-seadme tüübi tuvastamisel. Pahavara leviku tõkestamiseks tuleks määratleda IoT keskkonnas ohustatud IoT-seadmed.

Seejuures on käesoleva lõputöö eesmärk täiustada pahavara tuvastamise protseduuri, kasutades erinevaid masinõppe meetodeid, mis mitte ainult ei lahenda binaarse klassifikatsiooni probleeme, vaid mida saab rakendada ka varajases ründefaasis tuvastamisel, kategoriseerides pahavara vastavalt seadme tüübile ja ründefaasile. Binaarsed klassifikatsioonimudelid määratlesid IoT botneti pahavara, nagu Mirai, Bashlite ja Torii. Mitmeklassilised klassifitseerimismudelid hõlmasid: 4 klassi stsenaariumi kolme pahavaratüübi ja healoomulise liiklusega, 8 klassi katset, mis eristas pahavara tüüpi ja seadme tüüpi, ning 12 klassi stsenaariumi, mis eristas ründe etappi, olgu selleks siis käsk ja kontroll (C&C) või levi ja seadme tüüp.

CERCS: T120 Süsteemitehnoloogia, arvutitehnoloogia
Märksõnad: IoT, Botnet, masinõpe, juhendatud õpe

Contents

List of Figures

List of Tables

1 Introduction

A botnet attack is a form of cyberattack that intends to compromise devices by interconnected computers infected with a malware. The botnet attack is usually under the control of a cyber-criminal. This type of attack also includes data theft, exploitation of sensitive information, and Distributed Denial-of-Service (DDoS) attack [60]. IoT devices might compromised by the large-scale attacks. An intrusion detection system (IDS) applied for the botnet detection [61]. The IDS monitors incoming traffic and detects malicious behavior. The IDS, based on a machine learning approach, learns and detects an attack. It analyzes the traffic and its pattern against the machine learning model. The botnet attacks are multi-stage attacks [62] that spread in the systems gradually. Therefore it is crucial to detect the attack at the early stage and implement the countermeasures for attack prevention. The introduction covers the problem statement of the thesis, its research objective, research questions, research structure, and research scope with limitations.

1.1 Problem Statement

Internet of Things or IoT is a concept of technology development towards automation processes where the human factor will be diminished to the minimum. IoT consists of a network of physical objects that have sensors, software, and other technologies. Those devices interact with each other by sending and receiving packets of data and they are being managed via the internet. According to Statista [1], the quantity of IoT devices around the world will be 38.6 billion by 2025, whereas, for the comparison, in 2018 the number of IoT devices was around 22 billion. The indication that the IoT ecosystem is increasing and changing our daily life is obvious.

With the advancement of technology, IoT devices face some challenges simultaneously. Many of these IoT devices brought a security issue because of their architecture, which, indeed, leads to different types of attacks. For instance, the IoT botnet attacks utilize compromised interconnected IoT devices that the attacker triggers to launch a DDoS attack. Distributed Denial-of-Service (DDoS) attack attempts to flood incoming traffic that originates from various sources simultaneously. The reason for the attacks is explained by lacking security software on the IoT devices, not applying secure development processes, not doing good security administration, and cyber security awareness problems. The IoT botnet attack becomes more advanced and sophisticated, with the raise of IoT devices worldwide the tendency of IoT botnet attacks becomes prominent.

The most common attack is Mirai malware, which originated in 2016. The attack process includes several phases where the vulnerable machines are becoming a part of the botnet which then are being controlled by the attacker’s command and control center. The attacker executes a DDoS attack to take down target destinations such as the servers of the victims.

The botnet attack can utilize thousands and even millions of devices to implement an attack. For botnet attack detection, there are many solutions designed. One of them is the intrusion detection system. It allows to monitor the network and detect suspicious network traffic. The intrusion detection system can catch malicious network activities with help of different techniques. To illustrate, machine learning techniques can be utilized for botnet detection. Nowadays, this technique is developing tremendously because it can not only detect malicious traffic but also prevent botnet attacks. Moreover, with increased interest in IoT devices, there has been a spread of cyber-attacks that could compromise those devices. And to mitigate those botnet attacks the solutions like intrusion detection systems have been used accordingly.

Based on the 2019 SonicWall Cyber Threat Report [2], the IoT device producers failed to apply appropriate security controls to prevent the devices from the remote attacks thereby it caused to the growth of IoT attacks in 2018 by 217.5%. The usage of machine learning techniques is approved by its successful implementation in multiple areas, thus the potential solutions provided by machine learning in the security field are going up in recent years. Analysts at ABI Research [3] reported that utilizing machine learning for security purposes will expand investment in artificial intelligence, big data, and analytics in 2021 by 96 billion dollars, moreover huge tech giants are actively using machine learning to analyze threats against endpoint devices. In comparison with traditional network intrusion detection systems, which in most cases are signature-based making it harder to detect if the network traffic is malicious or not, since it is utilized only to detect known attacks, but it is blind to detect zero-day vulnerabilities. While the machine learning-based detection system is being used as an effective detection mechanism to identify the nature of the traffic pattern and differentiate the variance of the attacks [4][5][6]. Those research papers included various machine learning techniques to detect the botnet in an IoT environment. The machine learning-based approach allows to detect the suspicious activity and prevents the botnet attack by detecting the attack at an early stage.

For example [4], to simplify the botnet detection process, researchers used the feature selection technique to capture the botnet attack pattern. A different set of 55 features chosen from the network traffic have been utilized for early-stage botnet detection. They applied a different range of features to detect the malware. High detection performance was achieved when the number of features reached from 10 to 40, hence the higher features number is not influenced much by Command-and-control traffic detection. Another paper [5] was focused on the dimensionality reduction problem since the traffic consisted of many features that could cause biased accuracy results. Thus, by dimensionality reduction, the accuracy of the machine learning model could lead to better accuracy performance. The decision tree and K-Nearest Neighbor classifiers have been utilized with a set of 2,3 and 10 selected features on the N-BaIoT dataset. Botnet detection results showed that the Decision tree classifier outperformed the K-Nearest Neighbor classifier with an accuracy score of 99%. Alternatively, for early-stage botnet detection [6], researchers selected features based on PCA and Information Gain and applied SVM, Logistic Regression, Multilayer Perceptron, and Random Forest classifiers. Accuracy of 97.8% was achieved based on a Random Forest classifier with 37 selected features. The proposed model demonstrated high-performance metrics in detecting the early attack stage which is Command and control.

Various studies consider other aspects of successful botnet detection. For example, the IoT malware inspection [7] defines which stage of the attack shows behavior patterns. This research also demonstrated that the botnet architecture of malware remains the same

for a while, thus saving the same behavioral patterns. Besides, the architecture of IoT botnet [8] plays a crucial role in the botnet attack detection process. In this research, some IoT botnet architectures have been stated with the impact of DDoS attacks that are usually initiated in the Command and control stage, where the attack starts to infect the device. In addition, another research dedicated to the application of cross-validation strategy on a Decision tree, Random Forest, LDA, and Neural networks classifiers to detect DDoS attacks by classifying malicious and normal data [9]. Currently, the identification and detection of different stages of attack is priority research amongst others [10], because of its crucial influence on the infrastructure. Multiple stages of the attack make it harder to grasp the malicious activity in terms of time limits that could be indeed significant for the IoT environment.

Observed research papers that have been conducted in recent times revealed a research gap that does not cover the detection of early attack stages in the IoT environment. The detection of early-stage botnet attacks is crucial since it allows not only to detect the attack but also to prevent it by applying appropriate countermeasures. In other words, the existing detection measures do not cover the intersection of the malware with IoT device type and attack stage with IoT device type that could enhance the botnet detection measures. These malware category models assumingly can simplify the IoT botnet detection in a medium-size IoT environment.

1.2 Research Objective

The research objective of this thesis is to build machine learning models that will allow conducting binary classification experiments where the outcome will be the malicious and legitimate classification for malware attacks such as Mirai, Bashlite, and Torii. These models include several machine learning techniques that improve the model performance and reduce the probability of biased results.

Another objective is to categorize malware attacks and benign data with given 4 classes, such as Mirai, Bashlite, Torii, and legitimate traffic, malware attacks, and device types that consist of 8 classes and attack stages with device types that include 12 classes utilizing different classifiers.

The motivation for early attack stages detection is to prevent lateral malware movement through IoT devices. In most cases, the IoT environment consists of multiple IoT devices interconnected in the same network. Those IoT devices can be differentiated according to their device type. Furthermore, in the case of medium size of IoT network that includes around 80 IoT devices, it is complicated to track which devices have been infected. However, the categorization of IoT device types with malware (Mirai, Bashlite, or Torii) and early attack stages (Command and Control or Spread) with its corresponding IoT device type could presumably simplify the IoT botnet detection workflow for security analysts. The early stage of an IoT botnet attack, for example, includes Command and Control stage, where the malware is used to compromise IoT devices. In the later step, a spread stage is characterized by spreading the malware to non-compromised devices. The botnet detection at the Command and Control stage and spread stage could detect infected IoT device types and prevent attacks by proceeding with the incident handling procedure.

Therefore, the research objectives demonstrated the aim to apply the machine learning techniques that will be used to implement binary classification to detect the malware and normal traffic, and multiclass classification that detects three malware and benign traffic, the malware with device type and the stage of an attack and its corresponding device type.

1.3 Research Questions

The research aims to develop efficient machine learning models to address botnet detection problems. The research questions for this thesis help to guide the research methodology step by step and follow the botnet detection problem, with a binary use case, and multi-class use case with attacks or malware and their corresponding device type.

Research Question 1(RQ1): What steps does the malware classification procedure cover?

Research Question 2(RQ2): Whether the binary classification model demonstrates high accuracy and results in malware detection problem?

Research Question 3(RQ3): Whether the multiclass classification can help to distinguish malware with device type? Is it applicable for the attack stage with device type use case?

Research Question 4(RQ4): Whether the proposed machine learning-based approach demonstrate high-performance metrics?

1.4 Research Structure

The research includes the following chapters to implement defined research objectives and cover research questions.

- Chapter 2 Preliminaries. The preliminary chapter covers the main vectors of research work, and selected methods and justifies chosen techniques.
- Chapter 3 Research design and research methodology. This section covers the framework of research methods and explains each step of the proposed methods.
- Chapter 4 Discussion. The discussion chapter covers the results discussion and their performance in comparison with existed research work.
- Chapter 5 Conclusion. This chapter provides a summary of the research and an overview of the results. It also covers contributions and further research.

1.5 Research Scope and Limitations

This thesis considers the following scope to proceed with experiments and gather information on proposed research questions.

- Utilization of medium size IoT botnet dataset that includes the early stage attacks and three malware such as Mirai, Bashlite, and Torii
- Implementation of Nested cross-validation technique for binary and multiclass classification models
- Utilizing a different set of features with various classifiers and analyzing its impact on the performance metric
- Using multiclass classification model to detect malware with device type and attack stages with devices

Limitations cover the utilization of the TalTech AI Lab environment to process the calculations required for the machine learning models. The AI Lab environment was provided by the School of Information Technologies at the Tallinn University of Technology.

2 Preliminaries

The immense interest in IoT botnet attack research derives from the popularity of IoT devices. IoT devices and their weak architecture opened a new threat landscape in the cyber security field. IoT device usage led to continued research work and methods recommendation.

The preliminary chapter provides an overview of major concepts and their relation to the research scope: Internet of things, IoT device architecture, botnet architecture, Mirai malware detection methodology, and machine learning. Besides, in the end, the most recent and related literature review is given.

2.1 IoT

IoT (or Internet of things) is multiple devices or different items connected by a network. They interconnect with each other and leverage the transmitted data. IoT devices perform many tasks in various fields. For instance, industrial equipment, healthcare tools, smart bulbs, TV, phones, smart refrigerators, etc. The fundamental IoT architecture includes three tech levels that allow devices to communicate. The first one is the physical or perception layer. The end device works as an interface for the user. It helps to get the information needed. The second is the network layer where devices get information through a cellular, LAN, or satellite network, and special protocols. It also connects devices with the cloud. Third, the application layer is the IoT platform where the data is accumulated and processed (Figure 1).

Nowadays, IoT devices include high-speed microprocessors and other technical characteristics, that allow for the implementation of multiple varieties of attacks on the end devices. In addition to that, the default settings on that device as the default password and username, and open ports are much more appropriate for malicious activities run by attackers [11].

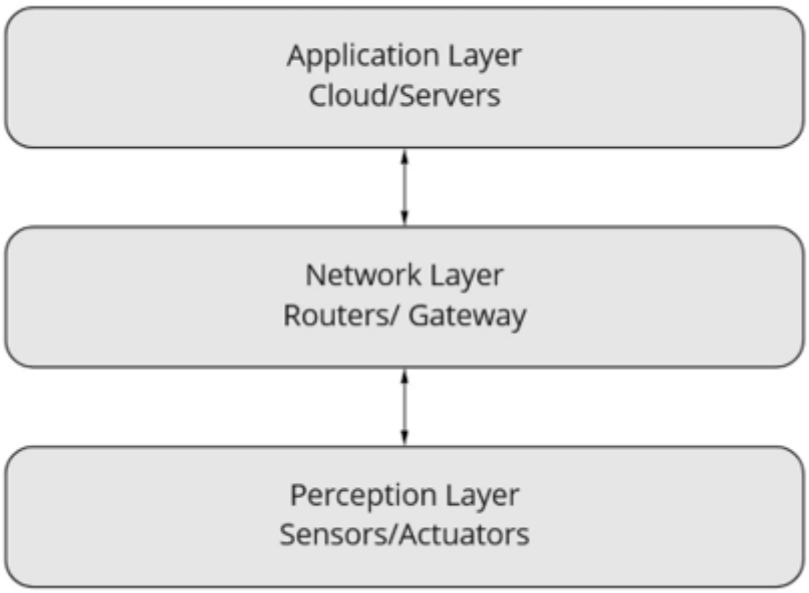

Figure 1. Fundamental IoT Architecture

Many botnet attacks have been used to compromise IoT devices. Those are Mirai, Reaper, Echobot, Emotet, Gamut, and Necrus malware [12]. Existing mechanisms of IoT devices cannot deal with threats efficiently. Data privacy and security concerns will increase tremendously because of an increased quantity of IoT devices. According to Statista [13], more than 75 billion IoT devices will be by 2025. Hence, more IoT devices will be integrated into humans' daily life. And the threat landscape is becoming more prominent, with the numerous IoT devices interconnected with the internet and forming small hubs.

2.2 Botnet Attacks

Botnets are compromised computers connected to a master, which gives the commands and controls the attacks (Figure 2). In the diagram below, the basic botnet architecture is presented. It contains a master that controls the attack process, they can compromise vulnerable machines that ultimately become controllers. Controllers can be divided into two types. One type of controller is responsible to recruit new intruders. While another one provides commands to bots. Whereas the intruders mainly intend to compromise more machines. According to the master's commands, the bots launch the attack.

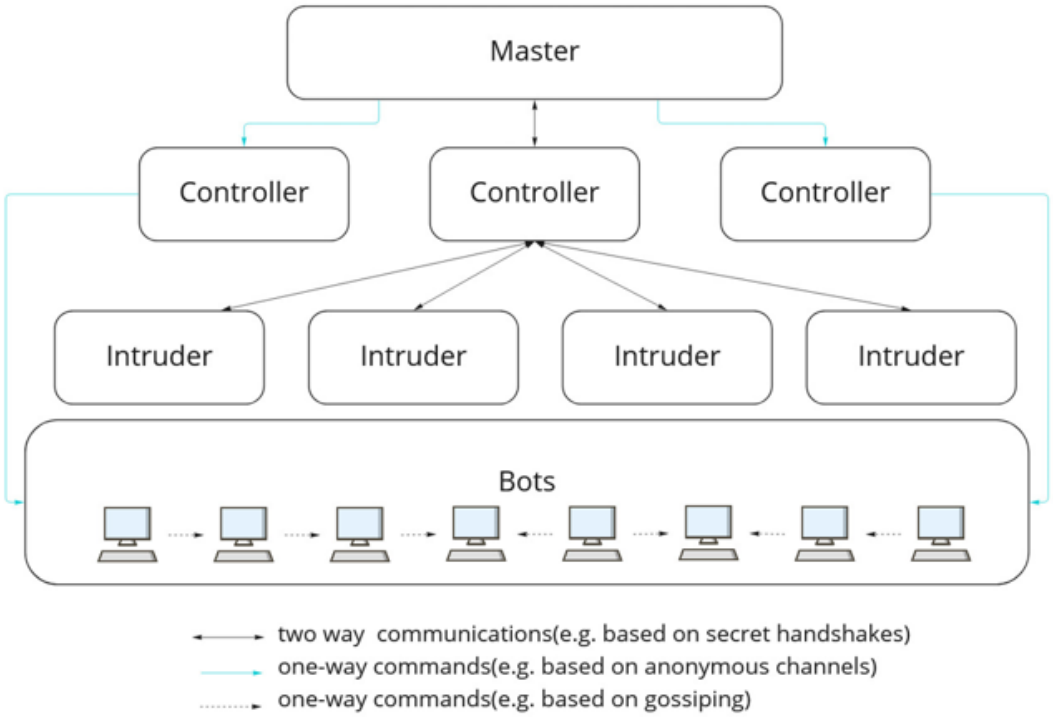

Figure 2. Botnet Architecture

The botnet lifetime cycle consists of formation, C&C, attack, and post-attack phases. The botnet lifecycle begins with compromising vulnerable machines and creating the botnet chain. Then, using Command & Control mechanism, the instruction of attacks is being implemented. Afterward, the attack is launched using bots. With a postattack phase, the master may compromise more bots to rerun the attack [14]. The rapid development of malicious activities led to complex botnet attacks. Especially regarding the attack purpose which has a wide range starting from DDoS [15], cryptocurrency stealing [16], and credential brute-force attacks [17]. Different IoT Malware types are most known nowadays. For instance, the infamous Mirai malware attack in 2016 showed a true power

that from that time continues to increase [18]. The Mirai source code has been published online, where it has served as a source to develop and improve the attack. According to the Security Intelligence report [19], in 2021, the IoT attacks raised almost 500% in comparison with 2019. Moreover, the tendency of Mirai malware that intends to evolve and multiply, shows an emerging variety of the malware, for example, Mozi, Echobot, Zeroshell, Gafgyt, and Loli. Attacks are focused on compromising existing infrastructures and implementing payloads.

2.3 Botnet Detection Methods

With the rise of botnet attacks, botnet detection methods have been introduced to the security market. According to R. B. Auliar et al. [20], the Mirai detection techniques can be divided into two main categories (Figure 3).

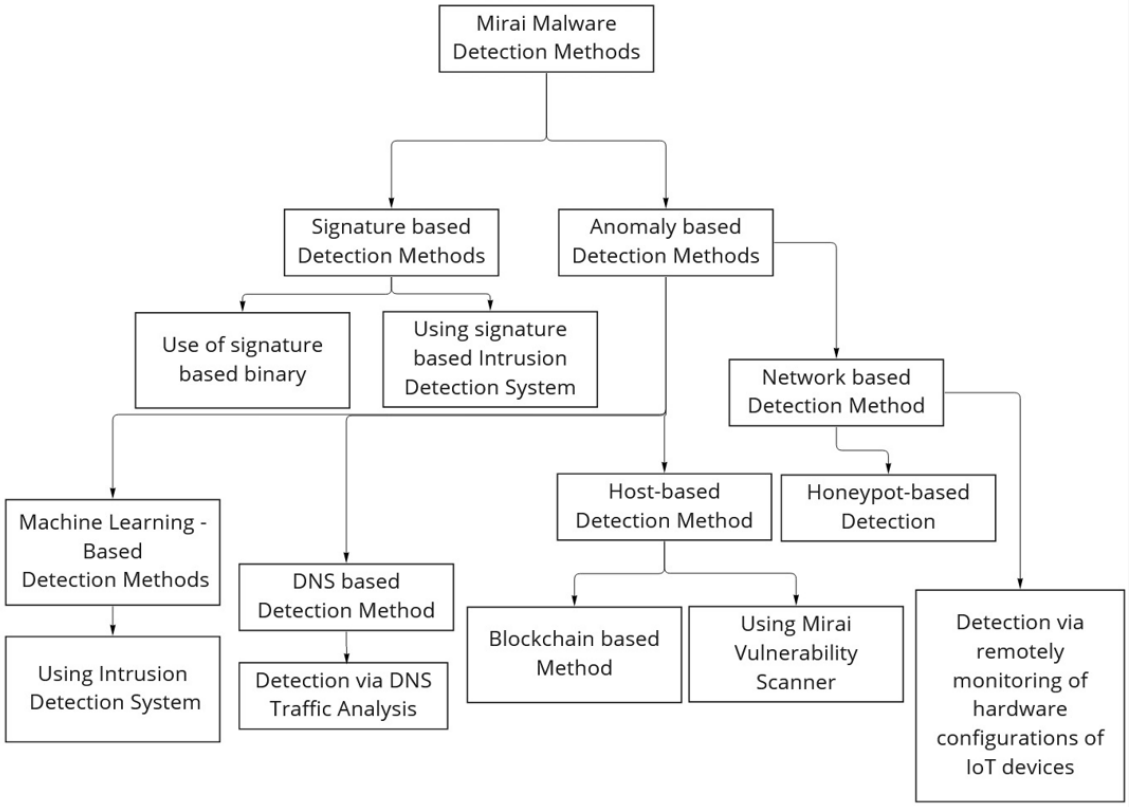

Figure 3. Mirai Malware Detection Methodology

One of Mirai Malware Detection's methodologies is Signature-based detection methods. In this method, the prominent attack patterns are being included in the database, therefore it is possible to identify the known botnets. In the case of intrusion detection systems, the rules are applied to analyze the incoming traffic which helps to recognize the malware. With signature-based binary, the special framework is engaged to detect the Mirai malware, which can define the acceptance criteria for Mirai attacks false positive values. It works with the application of Mirai source code on the signature-based binary [21].

Another type of malware detection method is Anomaly-based techniques. They include configuration of anomaly detection that is based on high traffic volumes, suspicious ports

among others, and unusual system behavior patterns. The detection does not include previous signature investigations and is considered useful for defining new botnet attacks.

This category has some subcategories such as Host-based detection, DNS-based detection, Network-based detection, and Machine Learning based detection methods. The host-based method goes through the inside monitoring of the IoT device system only. It also includes Blockchain-based detection and usage of the Mirai Vulnerability scanner. The blockchain-based technique can spot the Mirai attack, by using an Autonomous system to match the number of packets with the threshold, if the threshold is being overlapped the IP address will be blocked thereby denying the forwarded access [22]. Another method consists of the usage of the Mirai scanner. The scanner helps to get potential entry points for attack, by taking the open ports of the IoT network and simulating them a brute force attack, which is mimicking the Mirai credentials. In case if compromised IoT device is detected, the scanner will report a vulnerable place in the network [23]. DNS-based detection is implemented with the monitoring of DNS traffic that helps to identify the malware. The DNS-based technique can be used on a DNS-based botnet. In this method, the novel DNS-based scheme is applied to Mirai alike malware [24].

Network-based detection is applied via scanning traffic and its configurations. Some of the detection methods can be done with remote monitoring of hardware configurations of IoT devices. For example, the Mirai malware imitation can detect the vulnerabilities of the device and apply security measures to resolve the found issue. Another method is based on the honeypot technique application. As it is widely known honeypots are usually used as a security measure to detect malicious network activity. This method includes the configuration of a honeypot by defining specific packets that have been associated with the Mirai attack. Thereby, the honeypot will be able to capture suspicious activity in the IoT network [25]. Another method that has been quite used in recent times, is Machine Learning-based detection technique. It mainly aims to build an appropriate model that can detect malicious behavior of the network traffic and separate the suspicious and normal traffic. This type of detection helps to identify and prevent a threat to the IoT environment [26]. A traditional intrusion detection system is not always capable to deal with attacks when it comes to IoT devices. The reason for that is cryptographic mechanisms that cannot be embedded into IoT devices [27].

Thus, the machine learning approach is quite intriguing, since with the evolving botnet attacks, there are more techniques are being presented to build efficient machine learning models. They distinguish the malicious traffic by predicting accuracy and anomalies in the network. Automation and application of advanced data processing techniques allow getting the results faster and with high efficiency in comparison with other detection methods. It also allows for to prevention of botnet attacks and compromise of the IoT environment. Taking into consideration all detection aspects and obvious advantages of the Machine learning approach, this paper will be mainly focused on the Machine learning-based detection method.

2.4 Machine Learning

Since machine learning is a branch of Artificial Intelligence, the concept of Artificial Intelligence will be introduced. Artificial Intelligence, according to IBM [28], is a science and engineering of intelligent machines or systems, that can mimic and act as a human mind. The goal of Artificial intelligence is to develop a system that can act rationally to

solve various problems. Whereas Machine learning is about using the data and applied algorithms, to imitate human thinking, and using the accuracy metrics to scale the algorithms [29]. Machine learning algorithms are demonstrated as observing a target function (f) that connects an input value (X) and an output value (Y).

$$Y = f(X) \quad (1)$$

In other words, it represents the problem when it is needed to make predictions in the future (Y) based on some input values (X). In most cases, it is hard to find an appropriate function (f) that manages to find the right solution. It can be explained by error (e) value existence independent of input (X). This error value is inevitable due to insufficient attributes to provide the best alignment from X to Y. This concept explains the problem of learning the function from given data [30].

$$Y = f(X) + e \quad (2)$$

Machine learning classifies supervised and unsupervised approaches with mathematical models behind them (Figure 4). The main difference is underlying in labeled data that help to predict an outcome where another method is not. Supervised learning technique used to train labeled data. It can be used in solving two types of problems. One of them is the classification problem. The algorithm can separate data in specific subgroups like in problems to distinguish dogs and cats. Or in the case of cyber security, it can be used in spam, fraud, or attack detection. Another one is the regression problem, that used to define the dependencies between different variables. Given algorithm can forecast sales revenue or another numerical value.

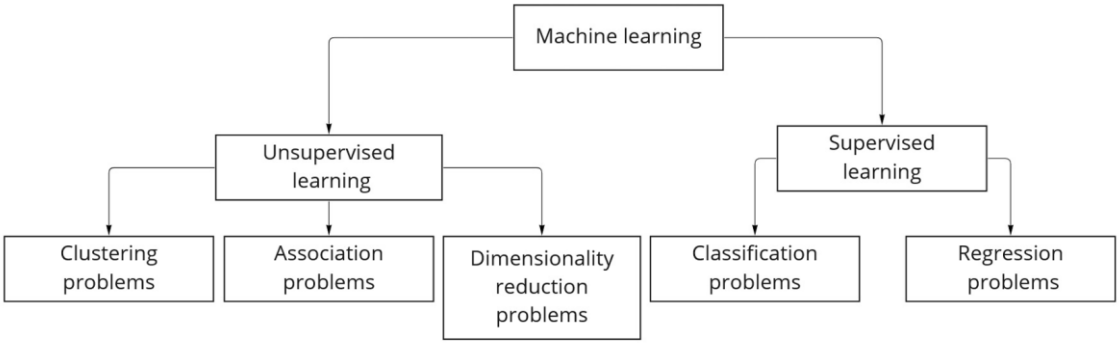

Figure 4. Machine Learning Techniques

Unsupervised learning algorithms use inputs and group, cluster, or reduce dimensionality with those data points. For instance, a clustering algorithm identifies patterns and structures of data points according to their similarities or differences. It is applied for image processing, customer segmentation, recommendation engine, and search result clustering. Another method is the association algorithm based on observing frequently occurring patterns and correlations. The goal is to define the associations of items that occur together rather than random samples, which boost decision-making in different fields. As an example, this technique helps to detect fraudulent activities. It implements rule matching and generates those rules from dataset changes. Dimensionality reduction is a learning method, that reduces the number of data input to achieve an appropriate size of features and maintains the data integrity [31]. Since the application of machine learning techniques is increasing, many problems are not falling into the previous two categories.

Some data consists of a mixture of labeled and unlabeled data. To illustrate, it involves a small portion of labeled samples and a large portion of unlabeled samples. The data is used for a model where it should make predictions for new samples. This type of problem is categorized under semi-supervised machine learning. It is a learning problem that do both apply supervised and unsupervised learning techniques.

To apply machine learning to the classification problem, and to decide whether the traffic is malicious or not, is required to use classification algorithms. In other words, classification is a task that helps to assign a class label to input samples that are being processed in a machine learning problem. Many classification types are applied for different domain problems. Classification types and their algorithms are provided in the table below (Table 1). There is a binary and multi-class classification that will be discussed further. Binary classification can predict one of two classes, e.g., malicious, or benign traffic. While multi-class classification refers to the prediction of one of more than two classes, e.g., Mirai, Bashlite, Tori, or legitimate classes. There is also another category of classification which is multi-label and imbalanced classification. The first one involves distinguishing one or more classes for each sample, e.g., attack name and attack phase, like Mirai C&C or Mirai spread and Bashlite C&C or Bashlite spread. Another one is when the distribution of samples across the classes is not equal.

To get the class label, the training dataset is used by the model, consequently, defining the best mapping from input samples to class labels. In binary classification, the class labels include string values, e.g., attack or normal traffic, and those values to be calculated with a modeling algorithm should be mapped to numeric values. For that, unique integer is assigned to each class label, e.g., 'attack' = 1, 'normal' = 0. This procedure is called label encoding, which is quite crucial and useful for further experiments.

Table 1. Application of Classification Algorithms

	Binary classification	Multi-class classification	Multi-label classification	Imbalanced classification
Classification algorithms	Logistic Regression	K-Nearest Neighbors	Decision Trees	Logistic Regression
	K-Nearest Neighbors	Decision Trees	Random Forests	Decision Trees
	Decision Trees	Naïve Bayes	Gradient Boosting	Support Vector Machines
	Support Vector Machine	Random Forest		
	Naïve Bayes	Gradient Boosting		

As experiments that were implemented within this paper scope refer to supervised learning, thus some classification algorithms will be presented below. Classification algorithms that are used for predicting the class have different calculation methods for each domain problem [32].

Logistic Regression predicts the probability of the class label as a function of the independent variables. The objective of Logistic regression is to find the best parameters θ, for $h_\theta(x) = \sigma(\theta^T x)$, that the model best predicts the class of each case.

K- Nearest Neighbors identify the ‘k’ nearest points to classify a test sample. The algorithm predicts the class with the highest frequency among the k neighbors. This method is useful for low-dimensional problems since it is computationally inefficient for high-dimensional data.

The decision Tree determines the prediction of fixed feature order by mapping the input sample with various outcomes. If/then the rule is being applied in the algorithm to build each branch, where the outcome is calculated from the training dataset. The algorithm builds classification models in the form of a tree.

Support Vector Machine finds the center of the wildest strip with a maximal margin that helps to separate classes and classify them according to the separating hyperplane.

A naïve Bayes classification algorithm is an extension of the Bayes theorem that determines a probability of a hypothesis with prior knowledge and depends on the conditional probability. The algorithm predicts the class of the dataset by assuming each feature’s independence. Thus, each feature contributes to the class identification independently.

Random Forest learns many trees and takes prediction out of them so that the algorithm predicts the final output. The greater the number of trees in the algorithm the higher accuracy, it also solves the overfitting problem.

Gradient Boosting introduces a weak learner to compensate for the shortcomings of existing weak learners. The objective of this algorithm is to minimize the loss function, in classification problems e.g., log-likelihood, by adding weak learners using gradient descent.

It is imperative to state, that some of those classifiers are specifically focused on binary classification and do not support more than two classes, e.g., Logistic Regression and Support Vector Machine. Thus, those classifiers are not considered when further multi-class and multi-labeled experiments were implemented.

2.5 Literature Review

The application of machine learning models to detect various attacks is rising, because of the computational efficiency and high accuracy of the models. In the following section, some related studies will be discussed. These are IoT botnet detection, Machine learning techniques (dimensionality reduction, machine learning classifiers), multistage attacks, and IoT fingerprinting concepts.

It has been proposed various methods in IoT botnet detection, for example, the comparison of machine learning and deep learning application on different IoT datasets (N-BaIoT, IoT-23, Kitsune, MedBIoT) [33], where they applied chi-square feature selection on two classes of data (malicious and benign), applied diverse models such as Naïve Bayes, K-Nearest Neighbor, Random Forest and Long Short-Term Memory on train data and got high accuracy results with Random forest classifier in the shortest time on test data.

Another approach was conducted by Maudoux et al. [34], where the model building began with dividing packets into smaller packets with the use of the Weka tool, continuing with four Decision Tree models, and processing those with AdaBoost meta classifier, finishing with a combined Forest model based on Stacking method. This multi-step method allows determining C&C exchanges even in real-time that were deducted from the high precision

of the given traffic. Nithya et al. [35] showed in their research, that the application of Principal Component Analysis is much valuable for feature selection and thus higher performance metrics were obtained with algorithms like Random Forest, Decision Tree, and Support Vector Machine. This study illustrated on the IoT-23 dataset that dimensionality reduction can influence the model accuracy, especially in the case of usage of Decision Tree and Random Forest classifiers.

In the research work of Mehra et al. [36], the different Feature sets were applied including 36 Static features (e.g., headers, size of functions, etc. these were collected from ELF files and information from the symbol table), 13 Dynamic Features (e.g., numbers of system call like 'OPEN', 'CLOSE', etc.), 14 Network Features (e.g., Destination IPs, number of Packets, 'ARP', 'DNS' etc.). With several scenarios, those sets were used to train Logistic Regression, SVM, KNN, and Random Forest, models. The results showed the importance of 28 features among all of them, like Program headers, Section headers that go under Static set, in case of dynamic features those are Read, Connect, and Socket, and network features are Number of Destination IPs and packets. Another comprehensive analysis by Tikekar et al. [37] showed that many studies that use a machine learning-based approach to detect botnets use a similar logical path, but what is important to mention is that essential features play a crucial role in malware detection. This tendency shows that it is necessary to develop new machine learning workflows that could help in botnet mitigation techniques.

It is known that security mitigation can be applied on the device level as well as the network level. Device-level security measures are often failed to be implemented since not all vendors provide full-cycle service therefore the risk still exists. Where network-level solution allows access to IoT network and exploits potential attacks. Besides, when it comes to detection of the attack, it depends on the type of attack: single-stage or multi-stage. Examples of single-stage attacks are backdoor, fuzzes, exploits, denial of service (DoS), shellcode, and worms. Whereas multiple-stage attacks can be distributed denial of service (DDoS), malware, and exploit kits. The difference between them underlies the implementation and time of conducting the attack. A single-stage attack is relatively easier to detect rather than a multiple-stage attack. However, the drawback which should be considered is low interpretability on multistage attacks pattern behavior and computationally expensive calculations [38].

Another study that should be considered is related to the IoT fingerprinting concept. It helps to identify the device based on its behavioral characteristics. In terms of IoT botnet detection, IoT fingerprinting could help to identify the device in a multiclass classification problem. IoT fingerprinting concept IoT fingerprinting is quite challenging nowadays because of the large amount of IoT devices, their protocols, and control interfaces. Usually, the identity and type of IoT device are retracted using a specific query. This technique is often used to get remote identification of the device. However, in the case of a compromised device, this response can be cloned or modified by the attacker. Thereby, the device behavior illustrated in the network traffic can be detected by extracted features. In other words, aggregated features are used in the model to identify the device.

To detect a specific device in an IoT environment, it is needed to perceive some packets and compare them with previous behavioral patterns. Those behavioral patterns are recorded using the packet header feature and payload features. Moreover, the statistical models for IoT fingerprinting behavior can be challenging especially in the case of a multi-class dataset [39]. Therefore, machine learning classifiers can be applied alternatively. The classifiers are a valuable tool that can distinguish specific IoT devices. Usually, it is common that one defining feature of a device allows for classifying it against

other devices since machine learning techniques work well in such classification problems [39]. Zhang et al [40] in their work developed a three-step approach to identifying IoT devices. This method included network traffic capture, detection of IoT fingerprints, and classification. The methodology started with a selected specific time window to extract data and then detect fingerprints required for IoT device identification. The result showed high accuracy in testing data. However, only one machine learning classification technique was considered in this set of experiments, which is the K-Nearest Number classifier.

2.6 Summary

With the rise of IoT devices, a new security landscape has evolved. Due to poor architecture and weak security measures of IoT devices, all attacks, the botnet attacks have been developed at a more advanced level by infecting not only one IoT device but rather the chain of IoT devices. Many IoT botnet detection methods have been developed in recent times. Machine learning-based detection method has different ways to classify malicious traffic. One of them is supervised machine learning models. Models use labeled datasets, those train models, which can classify data. Some classification algorithms like Random Forest, Decision Tree, and K Nearest Neighbors utilize for binary and multiclass classification problems. The recent research papers demonstrated high demand for machine learning-based methods for botnet detection use cases [33][34][35][36][37] [38][39][40].

3 Research Design and Research Methodology

The research design consists of a data preparation step, where dataset structure, data points distribution, and data sampling for experiments are discussed. Then feature selection step covers filter, wrapper, and embedded methods, their advantages, and disadvantages. The third step contains a modeling process that utilizes scaler, nested cross-validation, classifiers, and performance metrics. And the last step concludes with selected methods that are used in binary and multiclass classification models.

3.1 Data Structure

For the IoT botnet detection problem, existing models that are trained on the non-IoT dataset cannot be utilized on the IoT dataset as they will demonstrate high false-positive and high false-negative accuracy results [41]. Hence, the nature of the dataset takes a prominent place in IoT botnet detection. In this research, the MedBIoT dataset will be utilized [42]. In comparison to N-BaIoT (2018), Bot-IoT (2018), and IoT-23(2020) datasets, the MedBIoT dataset contains a medium-sized IoT network infrastructure with 83 devices and 100 features (Table 2).

Features based on 4 categories and their corresponding 5-time windows as 100 ms,500 ms,1.5 s, 10 s, and 1 min. The IoT devices include 2 physical switches, 1 physical light bulb, 20 virtual locks, 20 virtual fans, 20 virtual switches, and 20 virtual light bulbs. Time, protocol, TCP stream, stream size, source, destination IP, MAC addresses, TCP raw message, and response code structured in the network traffic. MedBIoT dataset contains extracted features from the raw PCAP files. Traffic aggregated by Source MAC-IP(MI), Source and destination host IP (HH), Source and destination host and port (HpHp), and Source and destination host traffic jitter (HH_jit).

Table 2. Feature Categories

Category	**Features**
Host-MAC&IP	Packet count, mean, and variance
Channel	Packet count, mean, variance, magnitude, radius, covariance, and correlation
Network Jitter	Packet count, mean, and variance of packet jitter in channel
Socket	Packet count, mean, variance, magnitude, radius, covariance, and correlation

The dataset includes spread and C&C attack stages that relate to infection, spreading the malware, and communication in the target network. The recorded network traffic contains packets with 30% malicious and 70% benign traffic, with 17 845 567 packets (Figure 5). It also includes three types of botnet malware Mirai, Bashlite, and Torii. Compromised IoT devices with malware are 40 devices with Bashlite, 25 with Mirai, and 12 with Torii. The legitimate traffic of 83 devices is presented in the dataset.

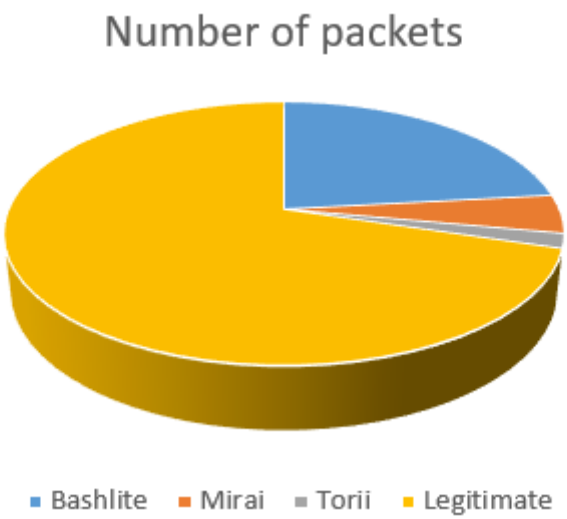

Figure 5. The proportion of Network Traffic

Different combinations of classes are aggregated into new datasets. Then composed datasets test the efficiency of the Machine learning pipeline in binary and multi-class classification problems. In the first round of experiments where the main goal was to differentiate malicious and benign behavior, the datasets were combined with each type of malware and corresponding non-infected traffic separately and its arrangement of all botnet attack network traffic. Those are Mirai and legitimate, Bashlite and legitimate, Torii and legitimate traffic. Another experiment applied to Mirai, Bashlite, Torii, and legitimate traffic.

In the second round, the objective was to implement malware category models that can identify attack stages and compromised devices as well as malware types and infected devices. One of the datasets contains 8 classes. Another one includes 12 classes. They are represented in Table 3.

Table 3. Name Of Classes In 8 And 12 Classes Models

Model	Name of classes
8 classes model	Mirai and switch, Mirai and lock, Mirai and light, Mirai and fan, Bashlite and switch, Bashlite and lock, Bashlite and light, Bashlite and fan categories
12 classes model	spread and switch, spread and lock, spread and light, spread and fan, C&C and switch, C&C and lock, C&C and light, C&C and fan, legitimate and switch, legitimate and lock, legitimate and light, legitimate and light, legitimate and fan

Randomly selected 3000 instances from each file assigned to each label. Hence, for binary and multiclass classification, it will be 6000 instances with malware and benign traffic, 12000 instances for 3 botnet attacks and legitimate traffic, 24000 instances for 8 classes model, and 36000 instances for 12 classes classification model. For the feature selection step, selected data points should have the same proportion in the dataset. It allows avoiding being unbalanced on the training dataset in terms of input data. The original dataset has a different quantity of rows in each file. The sampling technique is applied to balance and solve the bias problem.

3.2 Feature Selection

Feature selection is one of the crucial steps to perform a machine learning classification problem. The MedBIoT has 100 features and relates as high dimensional dataset. High-dimensional dataset requires enormous computational resources for data acquisition and network traffic analysis, which demonstrates the IoT environment problem. However, the main problem is that irrelevant features can affect classification accuracy, in other words, it's the curse of dimensionality. These reasons make it difficult for cyber security specialists to interpret traffic since irrelevant features lead to poor model quality. Therefore, they are unable to respond with appropriate incident handling measures on time. It is imperative to select an appropriate set of features. For that, feature selection methods will be utilized to select the most suitable features corresponding with their class labels [32]. This is the goal of all the feature selection techniques. Overall, feature selection is needed to find the most relevant features, exclude features not having an impact on classification or remove noise features, and reduce the dimensionality of the model. Some of the feature selection techniques learn the impact of each feature on the machine learning model and calculate results. Feature selection includes filter, wrapper, and embedded methods from the left to right sequence (Figure 6).

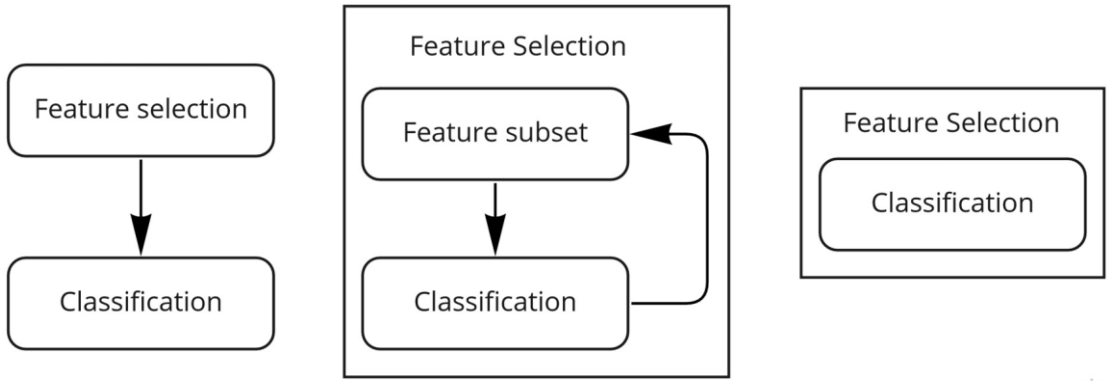

Figure 6. Feature Selection

3.2.1 Filter Method

This method selects a subset of features. A statistical approach like the distance between classes is the basis of the filter method. Scores calculated for each feature are ranked and then applied to the dataset. After that, the dataset is filtered by irrelevant features elimination. This method is univariate and considers features independently or it can not consider dependencies between the variables. There are several methods: chi-square, Gini index, entropy, information gain, and Fisher score.

3.2.2 Wrapper Method

The method iteratively refines a current set of features by selecting features that have been combined and evaluated, then compared to other combinations. Evaluation of combined features and relevant scores assignment requires a machine learning model. After the modeling part, the performance of the model can be checked. The calculation process might be complex in different ways. One of the approaches includes heuristics, like forwarding and backward passes that accumulate or eliminate features. Another method takes the best-first methodology, e.g., the random hill-climbing algorithm. There is also a recursive feature elimination algorithm. The wrapped method can be more

accurate than filtering methods. Nevertheless, it requires computationally expensive data manipulation techniques.

3.2.3 Embedded Method

The embedded method inspects various training iterations and ranks them according to the importance of each feature. It selects features during model building. The regularization technique is a common embedded feature selection strategy. Regularization uses extra coefficients to optimize a machine learning model that biases the accuracy with few coefficients. Examples include LASSO, Elastic Net, and Ridge Regression.

3.2.4 Selected Feature Selection Method for Research

In this research, the wrapper method could show some challenges due to utilizing enormous computational resources. The embedded method requires processing a variety of iterations which is also computational demanding. Consequently, the filter method has been utilized. It is needed to state that filter techniques perform the feature selection independently, reduce model complexity, enhance computational efficiency, and reduce overfitting error. There are many ways to implement the filter method on the dataset. For instance, correlation-based feature selection [43], ReliefF feature selection algorithm [44], Pearson's correlation coefficient [45] [46]. However, the Fisher score metric is one of the most widely used by researchers due to its outperforming results [47]. Hence, the Fisher score will be used for the feature selection step. Additionally, the Fisher score technique can be used on categorical data. It also computes the distribution of its frequency. This technique calculates each feature independently following Fisher's criteria. The formula is represented by a ratio of the average interclass to the average intraclass separation. Whereas the sum of the product of class proportion given by p_i with the squared difference of the class's mean μ_j and the feature's mean μ, whereas i is the class, i = 1,2,3 …N, that corresponds n-th feature, and divided by sum the product of class proportion is multiplied by variance σ_i^2 (3)

$$F = \frac{\sum_{i=1}^{N} p_i(\mu - \mu_i)}{\sum_{i=1}^{N} p_i \sigma_i^2} \tag{3}$$

The dataset includes categorical data as class labels. For computational efficiency, class labels transform into numerical values. In the first round of experiments, the datasets were combined and sampled. Then, it is needed to calculate Fisher scores and assign them to their classes. Fisher score is applied on Mirai malicious data and benign data. And feature selector discriminates it within the binary classification. The same procedure was applied for Bashlite and Torii malware. A feature selector such as the Fisher score is applied for classification problems [48]. Fisher score Moreover, the Fisher score coding part is implemented with Python. It allowed controlling each iteration. The top 20 features with the highest value results were selected and presented in descending order. In the experiments quantity of features is diversified. A combination of 2,3,5,10 and 20 features affect machine learning models. These top 20 selected features are demonstrated below for binary (Table 4 and Table 5) and multi-label class models (Table 6 and Table 7). Then, the features undergo testing with selected classifiers.

Table 4. Selected Features Of Mirai And Bashlite Malware For Binary Classification

	Mirai		Bashlite	
	Feature name	Score	Feature name	Score
1	HH_jit_5_mean	2.66	HH_jit_0.01_weight	14.53
2	HH_jit_3_mean	2.56	HH_jit_0.1_weight	0.86
3	HH_jit_1_mean	2.2	HpHp_3_weight_0	0.68
4	HH_jit_0.1_mean	1.63	HpHp_5_weight_0	0.68
5	HH_jit_0.01_mean	1.28	HpHp_1_weight_0	0.68
6	HpHp_0.01_radius_0_1	0.47	HH_0.1_magnitude_0_1	0.67
7	HH_0.01_radius_0_1	0.38	HH_1_magnitude_0_1	0.66
8	HH_0.01_magnitude_0_1	0.38	HH_jit_1_weight	0.65
9	HpHp_0.1_radius_0_1	0.36	HpHp_0.1_weight_0	0.65
10	HH_0.1_magnitude_0_1	0.33	HH_0.01_magnitude_0_1	0.64
11	HH_0.1_radius_0_1	0.33	HH_3_magnitude_0_1	0.64
12	HH_1_magnitude_0_1	0.3	HH_jit_3_weight	0.64
13	HH_3_magnitude_0_1	0.3	HH_5_magnitude_0_1	0.63
14	HH_5_magnitude_0_1	0.3	HpHp_0.01_weight_0	0.63
15	HpHp_0.01_magnitude_0_1	0.29	HH_jit_5_weight	0.62
16	HpHp_0.1_magnitude_0_1	0.29	HpHp_1_magnitude_0_1	0.6
17	HpHp_5_magnitude_0_1	0.29	HpHp_3_magnitude_0_1	0.59
18	HpHp_3_magnitude_0_1	0.29	HpHp_0.1_magnitude_0_1	0.59
19	HpHp_1_magnitude_0_1	0.29	HpHp_5_magnitude_0_1	0.59
20	HpHp_1_radius_0_1	0.27	HpHp_0.01_magnitude_0_1	0.59

Table 5. Selected Features Of Torii Malware For Binary Classification

	Torii	
	Feature name	Score
1	HpHp_0.1_pcc_0_1	1.39
2	HH_0.01_magnitude_0_1	0.51
3	HH_0.1_magnitude_0_1	0.51
4	HpHp_0.01_magnitude_0_1	0.51
5	HpHp_0.1_magnitude_0_1	0.51
6	HH_1_magnitude_0_1	0.5
7	HpHp_1_magnitude_0_1	0.5
8	HH_3_magnitude_0_1	0.5
9	HpHp_3_magnitude_0_1	0.5
10	HH_5_magnitude_0_1	0.49
11	HpHp_5_magnitude_0_1	0.49
12	HH_5_radius_0_1	0.43
13	HpHp_5_radius_0_1	0.43
14	HpHp_3_radius_0_1	0.43
15	HH_3_radius_0_1	0.43
16	HpHp_1_radius_0_1	0.42
17	HH_1_radius_0_1	0.42
18	HH_0.1_radius_0_1	0.4
19	HpHp_0.1_radius_0_1	0.4
20	HH_0.01_radius_0_1	0.38

In terms of binary selection, it is possible to reveal patterns in different attacks. In the Mirai attack, features such as Source and destination host IP (HH) source and destination host and port (HpHp) type with magnitude are dominated. In Bashlite, Source and destination host and port (HpHp) and Source and destination host IP (HH) types with weight. In Torii, destination host and port (HpHp) and Source and destination host IP (HH) with magnitude. The correlation of scores between different types of attacks is various. For instance, in the Mirai attack, the top scores are around 0.25, while in Bashlite the average score will be higher since the first score shows outperforming value. And in the Torii attack, the average value is higher than 0.5. Nevertheless, selected features are patterns that can identify attack types. The selected features were also utilized to analyze multiclass attacks and categorize their behavior.

Table 6. Selected Features For 4 Class Model Classification

	4 classes	
	Feature name	Score
1	HH_jit_0.01_weight	0.91
2	HH_0.01_weight_0	0.83
3	MI_dir_0.01_weight	0.83
4	HpHp_0.01_weight_0	0.72
5	HH_jit_0.1_weight	0.71
6	HH_0.01_magnitude_0_1	0.69
7	HH_0.1_magnitude_0_1	0.67
8	HH_1_magnitude_0_1	0.64
9	HH_3_magnitude_0_1	0.62
10	HH_5_magnitude_0_1	0.61
11	HpHp_0.01_magnitude_0_1	0.59
12	HpHp_0.1_magnitude_0_1	0.58
13	HpHp_1_magnitude_0_1	0.58
14	HpHp_3_magnitude_0_1	0.58
15	HpHp_5_magnitude_0_1	0.58
16	HpHp_0.01_radius_0_1	0.55
17	HpHp_0.1_radius_0_1	0.52
18	HpHp_1_radius_0_1	0.47
19	HpHp_0.1_weight_0	0.45
20	HpHp_3_radius_0_1	0.44

In the multi-class classification experiments, selected features are differentiated with various patterns. Selected features are the main characteristics of malware types such as Mirai, Bashlite, or Torii, attack stages like command and control or spread with devices, and malware with devices detection (Table 6 and Table 7).

Table 7. Selected Features For 8 And 12 Classes Model Classification

	8 classes		12 classes	
	Feature name	Score	Feature name	Score
1	HH_0.01_weight_0	0.9	HH_jit_0.01_weight	3.37
2	MI_dir_0.01_weight	0.9	HH_0.01_radius_0_1	0.89
3	HH_0.01_magnitude_0_1	0.27	HH_0.1_radius_0_1	0.73
4	HH_0.1_magnitude_0_1	0.24	HH_1_radius_0_1	0.6

5	HH_0.1_weight_0	0.23	HpHp_0.01_radius_0_1	0.57
6	MI_dir_0.1_weight	0.23	HH_3_radius_0_1	0.56
7	HH_1_magnitude_0_1	0.23	HH_5_radius_0_1	0.54
8	HH_3_magnitude_0_1	0.23	HpHp_0.1_radius_0_1	0.49
9	HH_5_magnitude_0_1	0.23	MI_dir_0.01_weight	0.45
10	HH_0.01_radius_0_1	0.22	HH_0.01_weight_0	0.45
11	HpHp_0.01_magnitude_0_1	0.22	HH_0.01_magnitude_0_1	0.45
12	HpHp_5_magnitude_0_1	0.22	HpHp_1_radius_0_1	0.42
13	HpHp_3_magnitude_0_1	0.22	HH_0.1_magnitude_0_1	0.41
14	HpHp_1_magnitude_0_1	0.22	HH_1_magnitude_0_1	0.4
15	HpHp_0.1_magnitude_0_1	0.22	HH_3_magnitude_0_1	0.4
16	HH_jit_0.01_weight	0.21	HH_5_magnitude_0_1	0.4
17	HpHp_5_weight_0	0.18	HpHp_3_radius_0_1	0.39
18	HH_0.1_radius_0_1	0.18	HpHp_0.01_magnitude_0_1	0.39
19	HH_jit_3_weight	0.18	HpHp_0.1_magnitude_0_1	0.39
20	HH_jit_1_weight	0.17	HpHp_5_magnitude_0_1	0.38

For example, based on the given dataset, features for the 4 classes model consist of 6 Source and destination host IP (HH), 2 Source and destination host traffic jitter (HH_jit), 11 Source and destination host and port (HpHp), and 1 Source MAC-IP(MI). The main features for the 8 classes model are 9 Source and destination host IP (HH), 6 Source and destination host and port (HpHp), 2 Source MAC-IP(MI), and 3 Source and destination host traffic jitter (HH_jit). Features for the 12 classes model include 11 Source and destination host IP (HH), 7 Source and destination host and port (HpHp), 1 Source and destination host traffic jitter (HH_jit), and 1 Source MAC-IP(MI). The feature category distribution is shown in Table 8. From the table below it could be seen that in 4 classes model Source and destination host and port (HpHp), in 8 classes model Source and destination host IP(HH), and 12 classes model Source and destination host IP(HH) category features are prominent in each model respectively.

Table 8. Category Distribution For Each Classification Formulation

4 classes		8 classes		12 classes	
Feature Category	Quantity	Feature Category	Quantity	Feature Category	Quantity
Source and destination host IP (HH)	6	Source and destination host IP (HH)	9	Source and destination host IP (HH)	11
Source and destination host and port (HpHp)	11	Source and destination host and port (HpHp)	6	Source and destination host and port (HpHp)	7
Source and destination host traffic jitter (HH_jit)	2	Source and destination host traffic jitter (HH_jit)	3	Source and destination host traffic jitter (HH_jit)	1
Source MAC-IP(MI)	1	Source MAC-IP(MI)	2	Source MAC-IP(MI)	1

Selected features can be analyzed through the scatter plot. It shows the data points distribution of 8 class and 12 class models. The scatter plot showing the distribution of the observation points of those models is in Figure 7 and Figure 8.

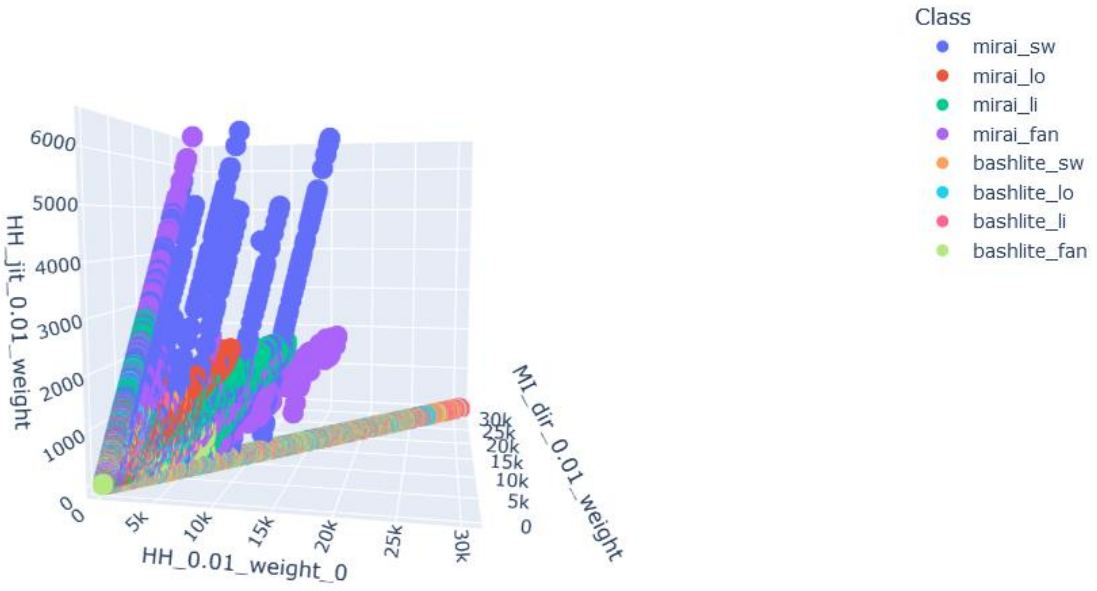

Figure 7. Scatter Plot For Top 3 Features In 8 Class Experiment

The three-dimensional scatter plots demonstrate the distribution of data points based on the top 3 selected features for 3 dimensions. In the case of the 8 classes model, 3000 instances were selected from each file. In total, 24000 instances. The 12 classes model consists of 36000 instances. The scatter plot for the 8 classes as well as for the 12 classes model represents high density, therefore it is challenging to divide the classes with classifiers. It will be tested with practical experiments. The modeling step introduces model structures and evaluation.

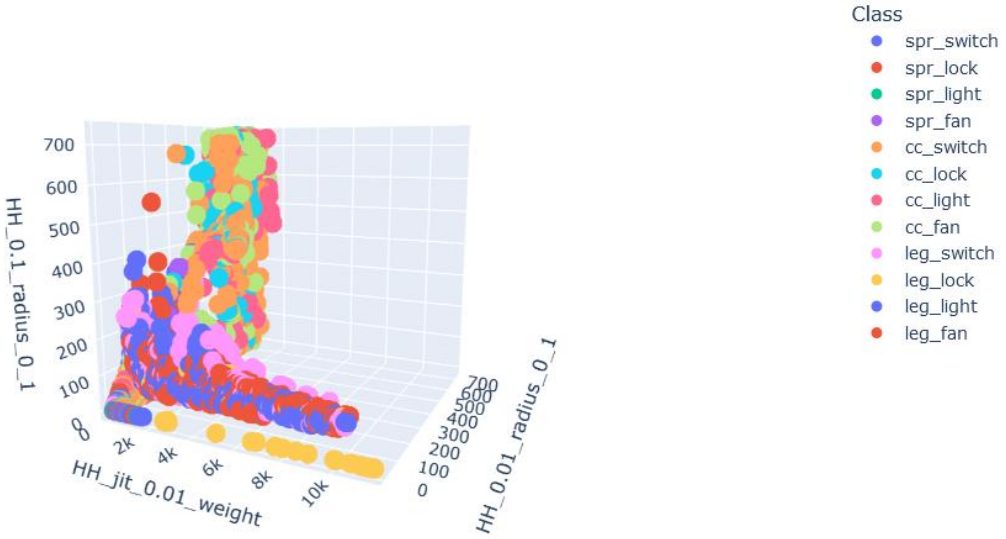

Figure 8. Scatter Plot For Top 3 Features In 12 Classes Experiment

3.3 Modeling

The proposed detection process includes several steps. Scikit-learn library in Python is utilized to implement machine learning workflow. The detection procedure automated several steps: scaler, modeling, and scoring. The scaler helps to balance the data. Modeling needs data classification with a nested cross-validation procedure, and the next step includes evaluating model performance.

3.3.1 Scaling

MinMaxScaler and StandardScaler, are scaling techniques used on the sample dataset. The main goal of using scaling techniques is to balance the data points by shrinking the range of values. MinMaxScaler rescales the dataset in the range from 0 to 1. Whereas StandardScaler removes the mean and scales the data to unit variance. This technique allows scaling measures [49].

3.3.2 Nested Cross-validation

Nested Cross-validation is a procedure for evaluating the performance of machine learning models. There is a difference between k-fold cross-validation and nested cross-validation procedures. The k-fold cross validation evaluates the model performance by making predictions on a dataset that is not used during the training step. However, when the same cross-validation procedure and dataset are simultaneously applied to tune and select a model, the evaluation of such a model can be biased. To tackle the biased model problem, it is needed to nest the hyperparameter optimization process with the model selection procedure [50]. In nested cross-validation, there are two loops presented.First is the outer loop, where the training of each fold with optimal parameters and averaging of each fold's test error. The second is the inner loop when the tuning hyperparameter process is executed. Usually, the number of folds in the outer loop is k=10 and a smaller value goes for the inner loop k=3 or k=5 [51]. Initially, the experiments were executed on the training and testing part with a ratio of 70% and 30% respectively. But those results showed worse performance compared to the nested cross-validation procedure. After configuring the Nested cross-validation procedure with enumerated folds, the following step is utilizing specific machine learning classifiers like Logistic Regression, Decision tree, KNN, and Random Forest.

3.3.3 Classifiers

After configuring the Nested cross-validation procedure with enumerated folds, the following step is utilized specific machine learning classifiers like Logistic Regression, Decision tree, KNN, and Random Forest. Nevertheless, considering multi-label datasets with high dimension attributes, different classifiers were utilized. However, better performance was shown only with Random Forest, KNN, and Decision tree algorithms for the binary and multiclass classification problems. These classifiers will be explained further. It is imperative to mention that all classifiers were applied with help of the sci-kit-learn library for Python, which is available online [52].

3.3.3.1 Decision Tree

A decision tree classifier is utilized for the classification problem, which helps identify which set an object belongs to. It is structured as a flowchart starting from a root node, then it goes to decision nodes following the leaf nodes (Figure 9).

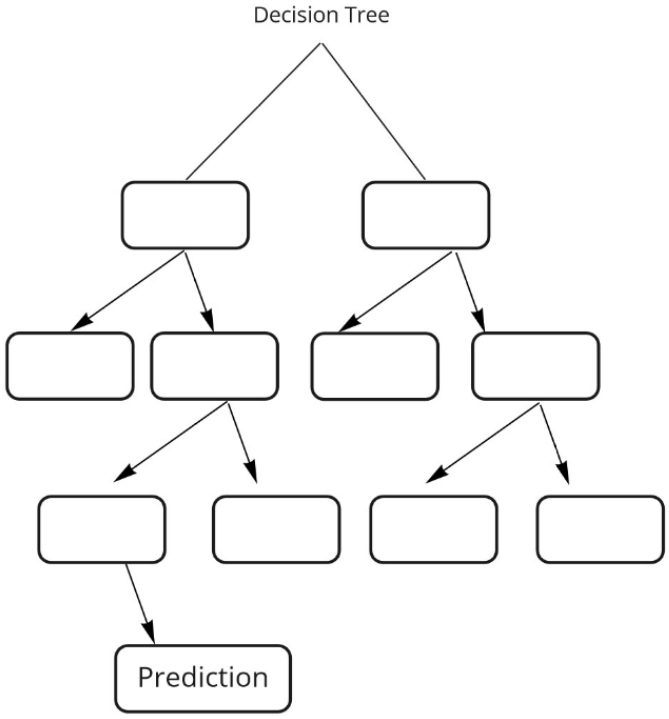

Figure 9. Decision Tree Classifier

This structure of Random Forest is composed of many Decision Trees, though the Decision Tree needs less calculation than mentioned algorithm. The decision of splitting is based on metrics, that have been discussed earlier, which are the Gini index and Information Gain. The decision tree is applied in the set of experiments because it can handle many features and it is easy to interpret, yet the algorithm also tends to overfit, and a little training data is utilized for leaf nodes [53].

3.3.3.2 K Nearest Neighbors

K Nearest Neighbors algorithm can be used for classification problems. The algorithm can adjust the model by exposing it to new data points. The KNN algorithm analyses the behavior of the nearest data points and classifies them respectively. It assumes that data points exist close to each other. In other words, the data points have similar classes. Thus, it is easier to find a category, that the data point relates to (Figure 10).

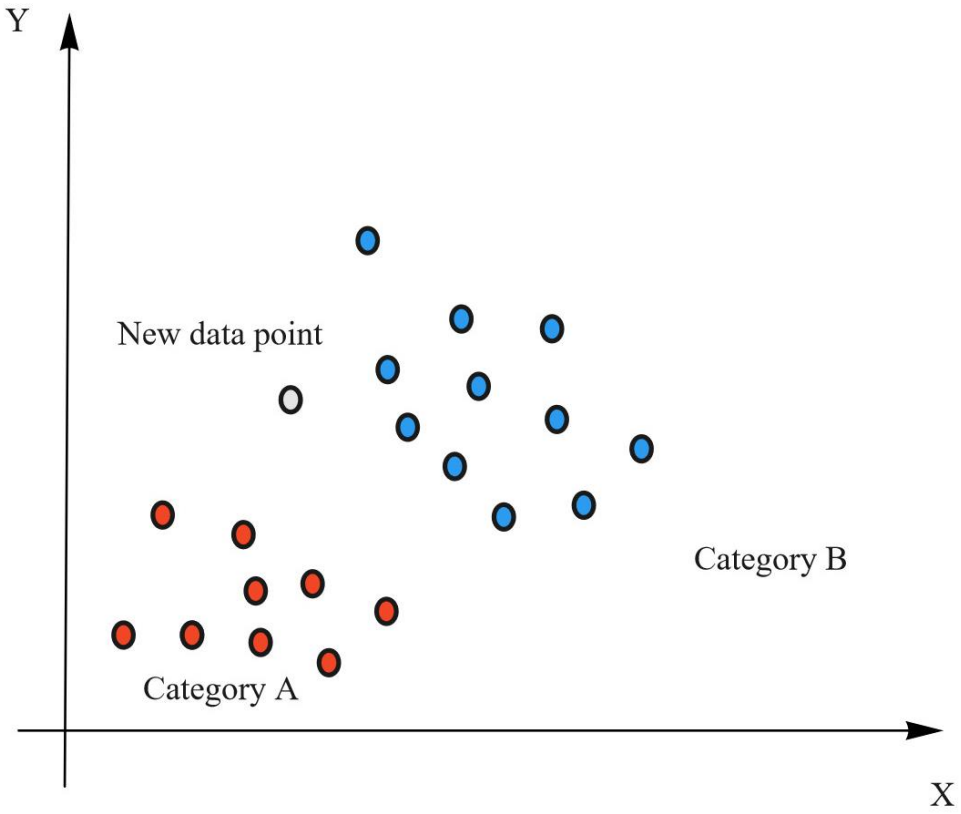

Figure 10. K Nearest Neighbor Classifier

The distance calculated between training data points and new data points. Then sort out the computed distance in ascending order and choose the top K results. And the last step is to assign a class based on the previous calculations to the test data point [54]. It is also important to choose the K value, which in the case of binary classification is not applicable. But in the case of multiple class models, K should be assigned according to the domain problem knowledge. This classifier can be applied to the multi-class problem as it is presented in the dataset. With the range of different numbers of features, it showed different accuracy results and time fitting on the experiment performance. The size of the sample dataset utilized for the set of experiments is relatively small. So, it is required less time to compute the results. Random Forest classifier has an advantage like its ability to deal with a huge amount of data, good performance on the imbalanced dataset, and minimize error due to decision tree ensemble. The Ensemble technique implies the combination of different models. The disadvantage of this algorithm is that it does not suit high-dimensional datasets. Thus, dimension reduction is a must. However, the advantage of the KNN algorithm is that it is easy to understand and interpret. KNN is a constantly evolving model because the modification of the hyperparameter simplifies the task for the rest of the parameters.

3.3.3.3 Random Forest

The random forest classifier creates a set of decision trees from a randomly selected subset of the training set. It collects selected votes to make the final prediction (Figure 11). For calculation, the algorithm uses inputs from all the decision trees, predicts the output data, or in other words, using sample training data, the algorithm creates a variety of subsets that are used for majority voting. The node where the algorithm starts with a

root node. Then root node divides into decision nodes. If further splitting is not applicable, it is called a leaf node. The decision to split the feature depends on the following metrics.

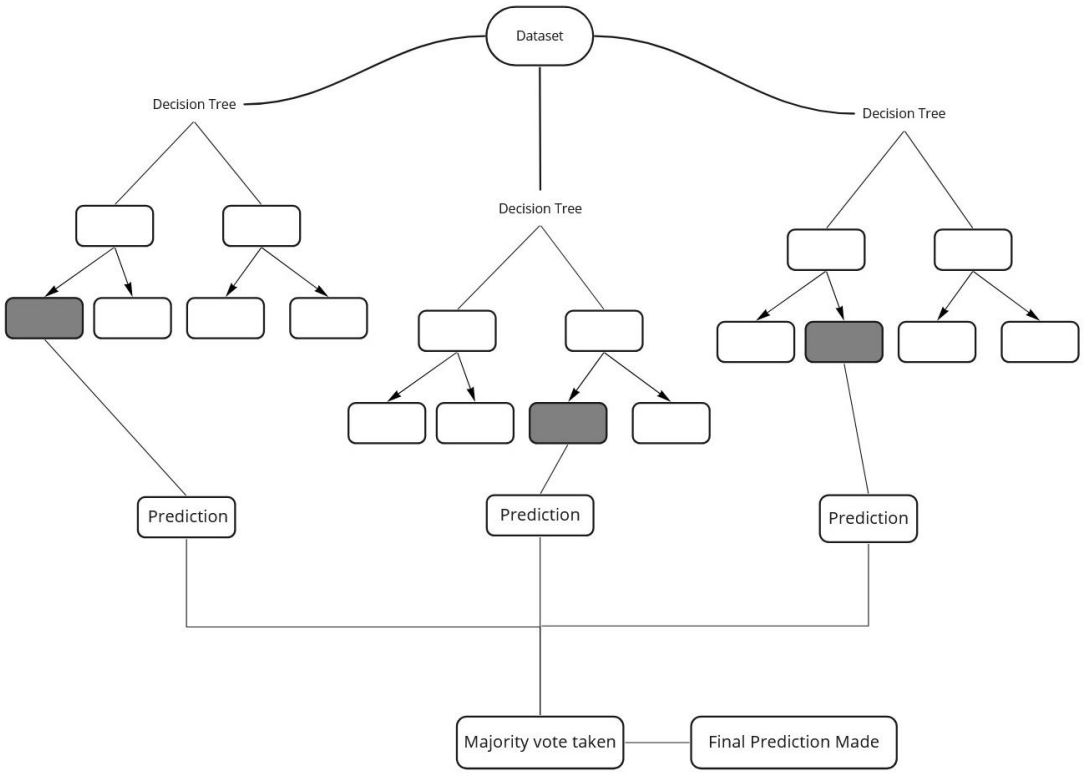

Figure 11. Principle Of Random Forest Classifier

Gini Index or Gini impurity is a metric calculated by subtracting the sum of the squared probabilities of each class from one. Gini index has values from 0 to 1, where 0 is the identification of purity and 1 shows a random distribution of elements among all classes. 0.5 or the middle value of the Gini index defines equal distribution.

The information Gain metric calculates the quality of a split. It is gained by subtracting distribution entropy after the split from distribution entropy before the split. The largest information gain is equal to the smallest entropy. Entropy is the degree to which a system has no pattern.

Some disadvantages should be stated as well. For instance, the Random Forest classifier is highly complex when compared to the Decision Tree algorithm which decisions are based on the path of tree nodes, and computationally slower. However, strong performance, handling of a huge amount of data, and processing of different data types such as binary, continuous, and categorical compensate for it [55].

3.3.4 Performance Metrics

When a classifier applied on the dataset, it is needed to evaluate performance of the machine learning algorithm. Accuracy is the ratio of correctly classified points to the total number of predictions(4). The accuracy value ranges from 0 to 1. This metric is computed based on true positive (TP), true negative (TN), false positive (FP), and false-negative (FN). For the multi-class classification problem, the accuracy metric has been selected as a performance metric since the dataset contains multilabel data. And since the labels are balanced, accuracy metric will be able to define the model performance. The performance

evaluation is based on the following datasets: binary classification with Mirai, Bashlite, Torii malware, multiclass classification with 4 classes(malware type and benign traffic), 8 classes(malware type and device type), 12 classes(attack stage and device type).

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

3.4 Summary

First, the datasets have been sampled and divided. The Fisher score method is applied for feature selection. Scaler utilized for the dataset is the Standard Scaler, since it showed better performance on the classifier step. Standard Scaler helped to normalize the data points. After that, classification, based on nested cross-validation, has been implemented. The classifiers are Decision Tree and Random Forest, which are tree-based models. The KNN classifier has been applied to categorize different classes as well.
The first round of experiments tests the efficiency of the proposed procedure to detect malicious and benign traffic. The second round of experiments tackles multi-class classification problems. In the four classes model, classifiers differentiated malware type and legitimate traffic. The classifiers have been utilized to detect the IoT botnet malware types with corresponding attack stages, in other words, Mirai and Bashlite with spread and C&C attack stages. Likewise, classifiers helped to distinguish malware and device types. Then for the models' evaluation purpose, the accuracy was selected as a performance metric.

4 Results and Discussion

This chapter describes the proposed procedure based on the machine learning approach. It also provides results and discusses the malware detection problems: binary and multiclass use cases. Furthermore, it includes revealed classification problems and suggested solutions. The discussion chapter also observes the defined research questions and gives answers to those.

4.1 Proposed Malware Classification Procedure

RQ1: What steps does the malware classification procedure cover?
The proposed machine learning-based approach for malware detection includes the following procedure. The calculation process starts with the configuration of utilized libraries that are available online. The first step was to accumulate different datasets from given CSV files [57] according to their malware attack and attack stage. For that, pandas and glob libraries helped concatenate data and assign their respective classes. This data manipulation technique is used to accumulate required datasets with various classes. The sampling size defined 6000 instances for binary and 12000, 24000, and 36000 instances for multiclass models. These data points were chosen randomly. The sampling was done by selecting an equal number of rows from each class. Also, sampling helped to balance the input data points and reduce the bias classification in the later step. Then for the dimension reduction, it is needed to apply the feature selection technique. In the given dataset use case, it is the Fisher score. Furthermore, the Fisher score was calculated with Python, where the simple function helps to track and debug each step of the Fisher score if needed. The top 20 scores were extracted and assigned to their classes and printed in descending order.

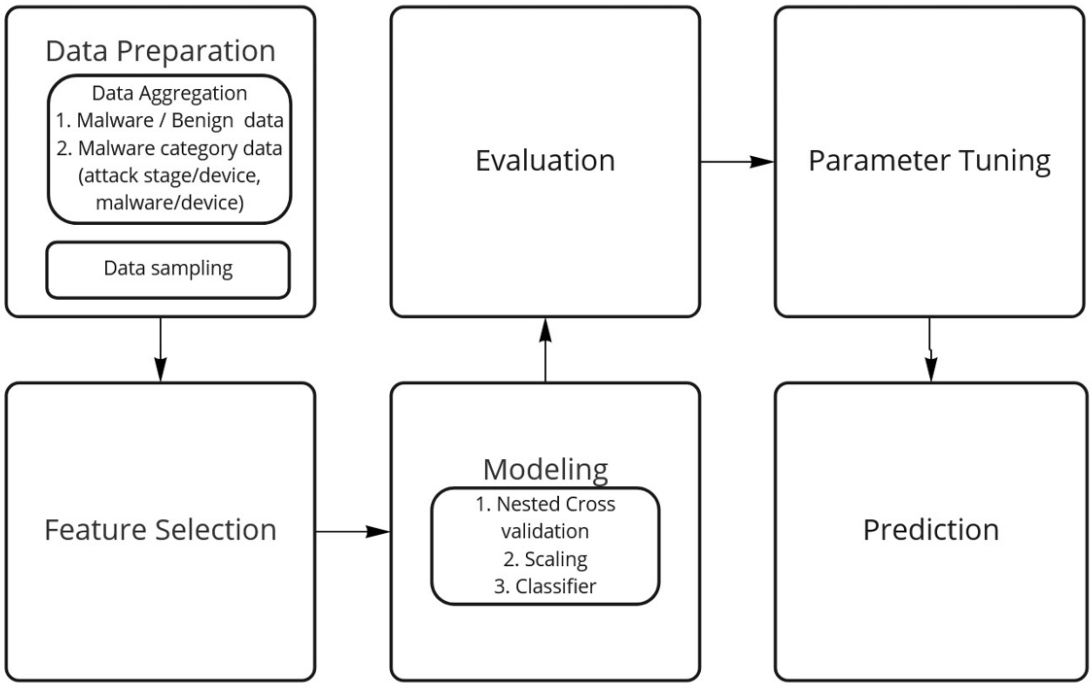

Figure 12. Proposed Malware Classification Procedure

In the modeling part, three different functions allowed to calculate values inside the loop. These functions are dependent on each other. Its dependency can exactly replicate the nested cross-validation loop. In experiments, the nested cross-validation helped to reduce the biased model problem. For outer and inner loops, the K-Fold technique was applied, because it is suitable for multi-class models. Moreover, scaler and number of folds were specified inside functions. That allowed calculation of the classifiers for each fold. In addition, the classifiers like Random Forest, K-Nearest Neighbor, and Decision Tree were inserted inside the experiment function. Then according to the performance metric, some of the parameters were modified to optimize the calculation process. And at the end of the procedure, it was possible to classify malicious traffic based on required classes. The proposed malware classification procedure is illustrated in Figure 12.

4.2 Binary Classification Experiments

RQ2: Whether the binary classification model demonstrates high accuracy and results in malware detection problem?

There are different malware use cases: Mirai, Bashlite, and Torii. The goal of binary classification experiments is to differentiate malicious traffic from benign traffic.

In one of the steps, the feature selection technique helped detect the top 20 features out of 100. Selected features utilized on the sample data size consisting of 6000 instances. With a nested cross-validation loop, the datasets were trained with 3 classifiers: Random Forest, K-Nearest Neighbors, and Decision Tree. A variety of features influences accuracy results. In this analysis, the results based on the Random Forest classifier would demonstrate the influence of features on accuracy. Random Forest classifier demonstrated itself as an efficient algorithm because of its performance compared with KNN and Decision Tree. Nevertheless, results based on KNN and Decision Tree classifiers can be found in Table 9, Table 10, and Table 11. The binary classification allowed to distinguish Mirai, Bashlite, and Torii malware. The results below were calculated with a set of scenarios such as a various number of features and classification models. The selected range of features: 2, 3, 5, 10, and 20 demonstrated different results. The binary classification was implemented with different classifiers: Random Forest, K Nearest Neighbors, and Decision Tree.

Table 9. Binary Classification With Mirai Malware

Mirai malware			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.991	0.993	0.988
3	0.991	0.993	0.988
5	0.993	0.995	0.99
10	0.995	0.996	0.99
20	0.996	0.998	0.993

Table 10. Binary Classification With Bashlite Malware

Bashlite malware			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.963	0.965	0.946
3	0.965	0.955	0.955
5	0.966	0.955	0.948
10	0.978	0.973	0.963
20	0.976	0.966	0.965

Table 11. Binary Classification With Torii Malware

Torii malware			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.998	0.995	0.998
3	1	0.995	0.998
5	0.998	0.995	0.998
10	1	0.995	0.998
20	1	0.995	1

Since the different number of features utilized for the binary experiments, it was imperative to select the best number of features to demonstrate high accuracy results. Selected features helped define malware behavior with a range of features (Figure 13).

For instance, Figure 13 demonstrated that the proposed methodology works well with the Torii attack and its benign data. From the given figure, it is possible to conclude, that number of features does not influence too much to Torii malware detection. The best results for 3, 10, and 20 features are 100%, and the second accuracy result of 0.998 goes with 2 and 5 features. As for the Mirai malware detection, the best accuracy score of 0.996 has been detected with the top 20 features, whereas the second accuracy performance with the top 10 features is equal to 0.995.

Fewer accuracy scores showed usage of top 5 features with 0.993, and top 2,3 features with 0.991. In Bashlite malware detection, the proposed model showed the highest accuracy of 0.978 with a set of top 10 selected features. The second result is calculated

with the top 20 features: 0.976. And the lower performance metric goes with 5, 3, and 2 features since their accuracy scores are 0.966, 0.965, and 0.963 respectively.

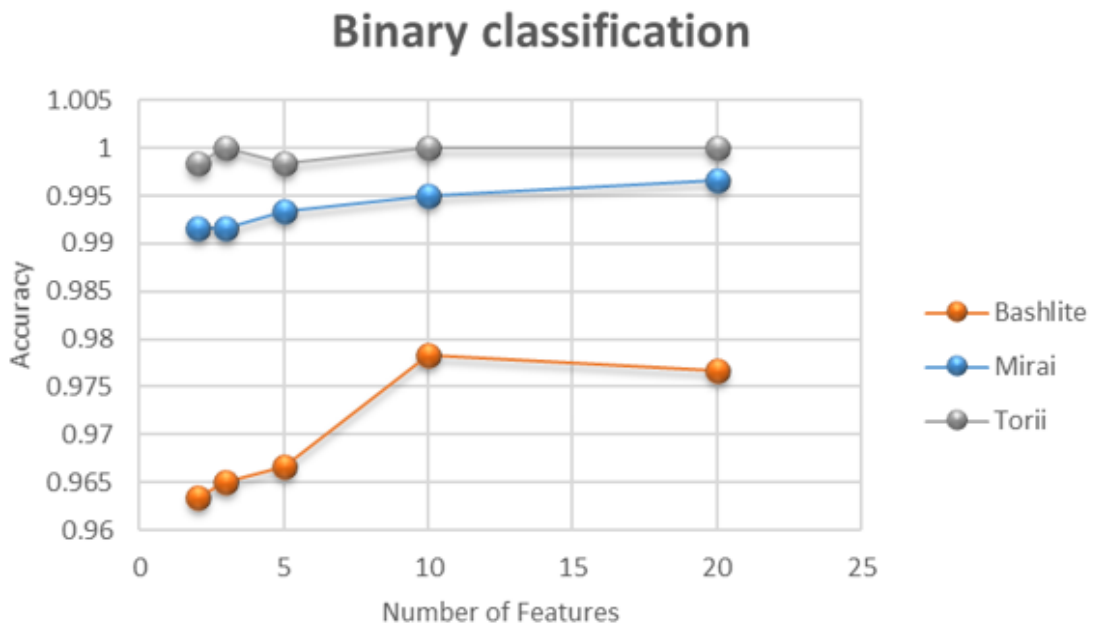

Figure 13. Accuracy For Binary Classification Based On The Proposed Procedure

Although the selected machine learning model demonstrated good performance metrics, there was a point to compare the results of experiments with previous research papers [33] [58]. The binary classification experiment is based on the same MedBIoT dataset, that has been utilized earlier. Gandhi et al. [33] used chi-square feature selection and algorithms including selected KNN, Decision Tree, and Random Forest, however, the accuracy showed lower results with 95.1%, 95.3%, and 95.3% respectively. Whereas Guerra-Manzanares et al. [58] used a cross-validation technique and applied KNN, Decision Tree and Random Forest classifiers, the results of those are 90.2% 93.1%, and 95.3% respectively.

4.3 Multiclass Classification Experiments

RQ3: Whether the multiclass classification can help to distinguish malware with device type? Is it applicable for the attack stage with device type use case?

The multiclass classification experiments show different results depending on the number of classes, the results of the Decision tree model are given in Figure 14. Among Random Forest, KNN, and Decision Tree classifiers, the best results have been achieved with the Decision Tree algorithm. Therefore, in the multiclass classification problem, the results of the Decision Tree scenario are discussed. Results of KNN and Random Forest classifiers can be found in Table 12, Table 13, and Table 14.

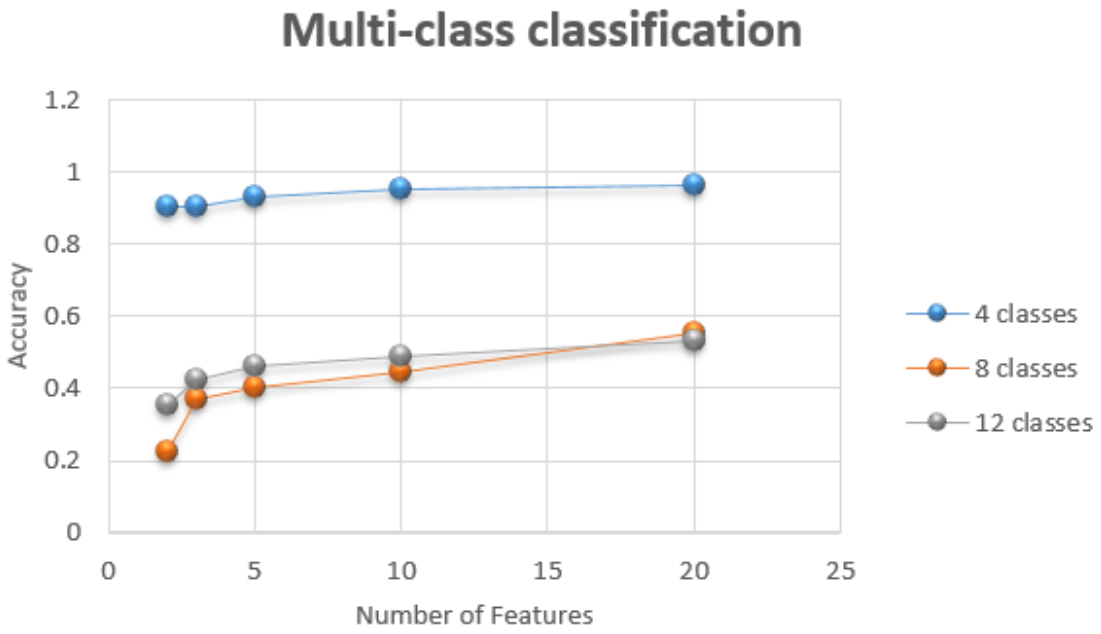

Figure 14. Accuracy For Multiclass Classification Based On The Proposed Procedure

Multiclass classification allowed to distinguish 4 classes (Mirai, Bashlite, Torii, and Legitimate), 8 classes (malware and devices: Mirai and switches, Mirai and locks, Mirai and light, Mirai and fan, Bashlite and switches, Bashlite and locks, Bashlite and light, Bashlite and fan) and 12 classes (attack and devices: C&C and switches, C&C and locks, C&C and light, C&C and fan, spread and switches, spread and locks, spread and light, spread and fan, legitimate and switches, legitimate and locks, legitimate and light, legitimate and fan).

Table 12. Multiclass Classification With 4 Classes

4 classes			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.909	0.885	0.903
3	0.908	0.888	0.907
5	0.939	0.916	0.930
10	0.958	0.937	0.953
20	0.960	0.951	0.962

In the case of 4 classes case with Mirai, Bashlite, Torii, and legitimate traffic, the accuracy metric is the highest with calculating the top 20 features with 0.962 and the top 10 features with 0.953. In another experiment, where the purpose was to classify attacks and devices, the best accuracy equals 0.555 with 20 features. The second result showed a 0.445 accuracy score with 10 features, and 0.402 with 5 features. Whereas the top 2 and 3 features showed the lowest scores 0.226 and 0.368 among other classes.

Table 13. Multiclass Classification With 8 Classes

8 classes			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.226	0.052	0.226
3	0.307	0.186	0.368
5	0.321	0.221	0.402
10	0.338	0.232	0.445
20	0.391	0.280	0.555

Table 14. Multiclass Classification With 12 Classes

12 classes			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.274	0.206	0.353
3	0.342	0.258	0.425
5	0.411	0.266	0.464
10	0.416	0.264	0.491
20	0.412	0.249	0.530

With 12 classes of categorization of malware and device, the accuracy metric showed relatively lower results with top 10 and 20 features of 0.491 and 0.530. Accuracies of utilized top 5,3,2 features are equal to 0.464, 0.425 and 0.353. There is also another interesting fact that among all three experiments of 4, 8, and 12 classes, a set of 10 and 20 features demonstrated better performance results. It confirms the heterogeneity of the given dataset with its features. The multiclass classification shows better results utilizing 20 features in 4, 8, and 12 class models.

In terms of the multiclass classification approach, some papers aimed to detect different categories of attacks. For instance, H. Chunduri et al. [59] utilized two IoT botnet datasets IoT-23 and MedBIoT, and categorized them into three classes: C&C (early attack stage), Malicious and Benign. They used 12 features to build Random Forest, SVM, and KNN classifiers. Experiments based on the MedBioT dataset showed an accuracy of 99.8% with Random Forest, 94.7% with SVM, and 96.1% with KNN. Hasan et al. [60] categorized attacks and anomalies with Logistic Regression 98.3%, SVM 98.2%, Decision Tree 99.4%, Random Forest 99.4%, and Artificial Neural Network 99.4% accuracy. The classes of the dataset such as Denial of Service, Data type proving, Malicious Control, Malicious Operation, Scan, Spying, Wrong Setup, and normal traffic have been used for classification. It is needed to mention that this research was based on virtual IoT environment data(IoT-23). Whereas the MedBIoT dataset has partially virtual environmental data. In mentioned works, there wasn't any focus on IoT devices or IoT device type detection problems. Thus, in the current work, experiments that classified attacks and device types demonstrated low accuracy results. Besides, the top three selected features for 8 and 12 classes experiments (Figure 7, Figure 8) showed high density which makes it harder to classify attacks with device type and attack stage with the device type. It was proved with implemented experiments and their moderate accuracy results.

4.4 Classification Problems and Suggested Solutions

RQ4: Whether the proposed machine learning-based approach demonstrate high-performance metrics?

In the case of binary classification, that proposed methodology gives high-performance metrics with a Random Forest classifier (Table 9). Binary classification experiments helped to identify malicious traffic of Mirai, Bashlite, and Torii malware. The accuracy of these results is 0.993,0.965 and 1. These models showed prominent efficiency. In an experiment with 4 classes scenario, the classification problem was in Mirai, Bashlite, Torii, and legitimate traffic detection. The best result of 0.962 is achieved with the Decision Tree classifier. In 8 classes scenario with malware and devices categorization, the accuracy is 0.555. Whereas in 12 classes scenario, with attack stage and devices categorization, the accuracy is 0.530. These results also showed that the application of the Decision Tree classifier gave a better performance in comparison with other classifiers.

Table 15. Accuracy For Classification Models Based On 20 Features

Classifiers	Binary classification accuracy			Multiclass classification accuracy		
	Mirai	Bashlite	Torii	4 classes	8 classes	12 classes
KNN	0.996	0.966	0.995	0.956	0.28	0.249
Decision Tree	0.993	0.965	1	0.962	0.555	0.530
Random Forest	0.996	0.976	1	0.96	0.391	0.412

The goal of the 8 classes scenario was to detect the malware type and devices, that showed which attack is it, i.e., Mirai or Bashlite, and its combination with the type of IoT devices, e.g. switches, locks, lights, or fans. The features that have been selected using the Fisher score method are presented in Table 4. The variety of features in each loop of experiments influenced the accuracy of models. Three classifiers that have been applied with various quantities of features helped to define the most efficient combination. The best one for 8 classes experiment is the Decision tree classifier.

Many experiments have been done to improve the efficiency before the given result. For instance, 12 classes of multiclass model experiments showed 61% of accuracy, however, the input data utilized was not balanced, so the results were higher. Nevertheless, after balancing the dataset, the result showed a lower result of 0.555. These results showed a realistic setting since the multiclass model has not been biased due to its input data. Another example included changing the number of instances. In the prior version of input data, 6000 instances have been utilized overall. The result showed 0.51 accuracy, while with 12000 instances 0.55 accuracy. The result did not change significantly.

The second model included 12 classes, that categorized attack stages and corresponded IoT device types. For example, spread and switch, spread and lock, spread and light, spread and fan, C&C and lock, C&C and light, C&C and fan, C&C and light, C&C and fan. It also included legitimate data and device types like legitimate and switch, legitimate and lock, legitimate and light, legitimate and fan. The results showed moderate accuracy with the Decision Tree classifier of 0.530. Although, some data alteration has been applied to track the performance difference. For instance, increasing the input instances from 6000 to 12000 showed an even lower score of 0.51 with 10 features. 12 classes

scenario detects the early attack stage, which potentially can simplify the pre-attack process management.

There also have been implemented additional experiments. The experiments could improve the accuracy of 8 classes and 12 classes scenarios. The main difference with prior experiments is increasing the number of instances (10000) from each class. In the case of the 8 classes scenario, it is 80000 instances. The results of the experiments are presented in Table 10. The best accuracy score of 0.618 has been achieved with a combination of 20 features and the Decision Tree classifier. This result is higher in comparison with previous results of 0.555 based on 24000 instances. However, the time required for calculation became more expensive.

Table 16. Accuracy Of 8 Classes Model Based On 80000 Instances

8 classes			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.22825	0.052875	0.227375
3	0.319375	0.19675	0.384625
5	0.379375	0.2495	0.442375
10	0.404	0.2785	0.478375
20	0.49	0.352	0.618625

In 12 classes scenario, the input data increased from 36000 to 120000 instances. The highest result is 0.609 (Table 11). The result combined 20 features and a Decision Tree classifier. The accuracy in this experiment is higher than in the experiment with 36000 instances. The accuracy of the 12 classes model was 0.530. As in the 8 classes experiment, the accuracy improved, but at the same time, the calculation time also increased prominently.

Table 17. Accuracy Of 12 Classes Model Based On 120000 Instances

12 classes			
Number of Features	Accuracy scores		
	RF	KNN	DT
2	0.282583	0.239417	0.3535
3	0.406417	0.30475	0.479833
5	0.501417	0.348083	0.532667
10	0.500167	0.338	0.557167
20	0.501667	0.317833	0.60983

The binary classification experiments showed outstanding results and helped to distinguish malware types: Mirai, Bashlite, and Torii. The multiclass model of 4 classes, that covered three botnets and legitimate traffic demonstrated high accuracy of 0.962. The proposed machine learning procedure revealed features' impurity, that led to low accuracy results in 8 classes and 12 classes scenarios. And a different number of instances helped increase the accuracy, however, led to computationally expensive costs.

In comparison with previous research [58], the main approach of multiclass models is distinguishing malware types and legitimate traffic i.e., malware categorization, while not taking into consideration the attack categorization. Those attacks are also more focused on defining the malware type rather have a combination of attacks with device type and

stage of the attack with the device type. This observation of research papers showed that researchers do not cover a machine learning approach for malware/ attack stages and device categorization.

4.5 Summary

The proposed malware classification procedure includes data preparation with additional steps, feature selection, modeling with nested cross-validation, scaling and classifiers, evaluation of the model, parameter tuning if needed, and final prediction results. This procedure covered in the research question 1 (RQ1). The research question 2 (RQ2) covers the binary classification experiments that were aimed to distinguish IoT botnet malwares such as Mirai, Bashlite, and Torii with respect to their legitimate traffic.The research question 3 (RQ3) includes the multiclass classification experiments that were aimed to detect the malware with device type. In addition, it contained the experiment regarding the attack stage and device type classification. The research question 4 (RQ4) covered accuracy performance of the proposed procedure.

Binary classification experiments demonstrated high accuracy results. The best accuracy has been achieved with the Random Forest classifier and the top 10 selected features(Table 9, Table 10, Table 11). The previous research works demonstrated lower results[33] [58]. Multiclass classification experiments included the following scenario. The four classes scenario was based on the three malware types (Mirai, Bashlite, Torii) and their respective legitimate traffic (Table 12). The 8 classes experiment differentiated the malware and device type: Mirai or Bashlite, and its combination with the type of IoT devices, e.g. switches, locks, lights, or fans(Table 13). And 12 classes model distinguished the attack stage (whether it was C&C or spread) and the device type (Table 14). The best result for all three scenarios was achieved with a combination of the Decision Tree classifier and top 20 features (Table 15). There were also some additional experiments regarding the 8 classes and 12 classes experiments. The instances have been increased by 10000 instances from each class. Results based on increased instances are shown in Table 16 and Table 17. Compared to previous research[58], the experiments were focused not only on the malware categorization, but also on the attack categorization.

5 Conclusion

This thesis was intended to enhance the malware detection procedure with different machine learning methods that not only address binary classification problems but also can be applied in early attack stages detection with a categorization of malware with device type and attack stage with the device type. The binary classification models defined the IoT botnet malware such as Mirai, Bashlite, and Torii. The multiclass classification models included: 4 classes scenario with three malware types and legitimate traffic, 8 classes experiment differentiated the malware type and device type, and 12 classes scenario distinguished the attack stage, whether it was command and control (C&C) or spread and the device type.

5.1 Limitations

In the current work, computational resources have been utilized to calculate the results of experiments. For the calculation purposes, the environment had the following node with AMD Threadripper 3960X 24-Core/48-thread Processor, 128 GB of memory, NVidia 3090 GPU with 24 GB of graphics memory. The computational resources have been provided by the School of Information Technologies at the Tallinn University of Technology.

5.2 Contribution

The proposed procedure improved the performance of malware detection in a binary classification problem. It included 2,3,5,10 and 20 features, that have been selected with the Fisher score method. Afterward, the procedure includes a nested cross-validation loop, that resolves an overfitting problem. It also applied Decision tree, KNN, and Random Forest classifiers, which showed high accuracy scores in binary classification experiments. In the Mirai scenario, the best result (0.998) is based on the combination of 20 features and the KNN classifier. In the Bashlite experiment, the result (0.978) was calculated with 10 features and a Random Forest classifier. In Torii, the accuracy is 1. It is calculated with a combination of 3, 10, and 20 features, and a Random Tree classifier. Based on the proposed procedure, the multiclass model, with three botnet attacks and benign data, has demonstrated high accuracy results.
The 8 classes and 12 class scenarios are addressed to enhance IoT fingerprinting solutions based on the machine learning method. The proposed scenarios consist of two approaches. The first one included malware and device type classification i.e., 8 classes scenario. However, it didn’t reveal a high accuracy performance (0.555). The second model contained the attack stage and device type (accuracy 0.530); the proposed procedure identifies the early-stage attacks. The detection of early-stage attacks could improve the attack detection workflow for security analysts. After that, they could proceed with the incident handling process.

5.3 Future Work

In terms of performance, it showed low accuracy results for malware with devices and attack stage with devices categories. These multiclass experiments proved that IoT

fingerprinting with a machine learning approach is complicated and showed that it needs the application of some advanced techniques. One of them is feature engineering, which helps to select, transform, extract, and combine to generate the desired features. While in this work only the feature selection method has been utilized. Another one is the Deep Learning approach, which utilizes the layered structure of the algorithm, for desired accuracy improvement. Whereas in the current work only Machine Learning techniques had been implemented.

References

[1] Vailshery, L. S. (2021). *IoT connected devices worldwide 2030*. Statista. https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology/

[2] Gatlan, S. (2019). *IoT Attacks Escalating with a 217.5% Increase in Volume*. BleepingComputer. https://www.bleepingcomputer.com/news/security/iot-attacks-escalating-with-a-2175-percent-increase-in-volume/

[3] Creamer, L. (2018). *5 Top Machine Learning Use Cases for Security*. Maureen Data Systems. https://www.mdsny.com/5-top-machine-learning-use-cases-for-security/

[4] Feng, Y., Akiyama, H., Lu, L., & Sakurai, K. (2018). Feature Selection for Machine Learning-Based Early Detection of Distributed Cyber Attacks. *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech).* https://doi.org/10.1109/dasc/picom/datacom/cyberscitec.2018.00040

[5] Bahsi, H., Nomm, S., & La Torre, F. B. (2018). Dimensionality Reduction for Machine Learning Based IoT Botnet Detection. *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. https://doi.org/10.1109/icarcv.2018.8581205

[6] Muhammad, A., Asad, M., & Javed, A. R. (2020). Robust Early Stage Botnet Detection using Machine Learning. *2020 International Conference on Cyber Warfare and Security (ICCWS)*. https://doi.org/10.1109/iccws48432.2020.9292395

[7] Jovanovic, E. D., & Vuletic, P. V. (2019). Analysis and Characterization of IoT Malware Command and Control Communication. *2019 27th Telecommunications Forum (TELFOR).* https://doi.org/10.1109/telfor48224.2019.8971194

[8] Mendes, L. D., Aloi, J., & Pimenta, T. C. (2019). Analysis of IoT Botnet Architectures and Recent Defense Proposals. *2019 31st International Conference on Microelectronics (ICM)*. https://doi.org/10.1109/icm48031.2019.9021715

[9] Jeronimo, D. (2018). Detection of Botnet activity via Machine Learning. *Instituto Superior Tecnico*.

[10] Sudheera, K. L. K., Divakaran, D. M., Singh, R. P., & Gurusamy, M. (2021). ADEPT: Detection and Identification of Correlated Attack Stages in IoT Networks. *IEEE Internet of Things Journal*, *8*(8), 6591–6607. https://doi.org/10.1109/jiot.2021.3055937

[11] Mehra, M., Paranjape, J. N., & Ribeiro, V. J. (2021). Improving ML Detection of IoT Botnets using Comprehensive Data and Feature Sets. *2021 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. https://doi.org/10.1109/comsnets51098.2021.9352943

[12] Wazid, M., Das, A. K., Rodrigues, J. J. P. C., Shetty, S., & Park, Y. (2019). IoMT Malware Detection Approaches: Analysis and Research Challenges. *IEEE Access*, *7*, 182459–182476. https://doi.org/10.1109/access.2019.2960412

[13] Statista Research Department. (2021). *Internet of Things - number of connected devices worldwide 2015–2025*. Statista. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[14] Leonard, J., Xu, S., & Sandhu, R. (2009). A Framework for Understanding Botnets. *2009 International Conference on Availability, Reliability and Security*. https://doi.org/10.1109/ares.2009.65

[15]Ashford, W. (2016). *More IoT botnets connected to DDoS attacks*. ComputerWeekly.Com. https://www.computerweekly.com/news/450303601/More-IoT-botnets-connected-to-DDoS-attacks

[16] MSN. (2021). *Botnet steals half a million dollars in cryptocurrency from victims*. https://www.msn.com/en-us/money/other/botnet-steals-half-a-million-dollars-in-cryptocurrency-from-victims/ar-AARSOKB

[17] Cimpanu, C. (2019). *IoT botnet used in YouTube ad fraud scheme*. ZDNet. https://www.zdnet.com/article/iot-botnet-used-in-youtube-ad-fraud-scheme/

[18] Sutherland, L. (2017). *The Weaponization of IoT: Rise of the Thingbots*. Security Intelligence. https://securityintelligence.com/the-weaponization-of-iot-rise-of-the-thingbots/

[19] McMillen, D. (2021). *Internet of Threats: IoT Botnets Drive Surge in Network Attacks. Security Intelligence.* https://securityintelligence.com/posts/internet-of-threats-iot-botnets-network-attacks/

[20] Auliar, R. B., & Bekaroo, G. (2021). Security in IoT-based Smart Homes: A Taxonomy Study of Detection Methods of Mirai Malware and Countermeasures. *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. https://doi.org/10.1109/iceccme52200.2021.9590841

[21] Jaramillo, L. E. S. (2018). Malware Detection and Mitigation Techniques: Lessons Learned from Mirai DDOS Attack. *Journal of Information Systems Engineering & Management, 3*(3). https://doi.org/10.20897/jisem/2655

[22] Ahmed, Z., Danish, S. M., Qureshi, H. K., & Lestas, M. (2019). Protecting IoTs from Mirai Botnet Attacks Using Blockchains. *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. https://doi.org/10.1109/camad.2019.8858484

[23] Vysakh, S., & Binu, P. K. (2020). IoT based Mirai Vulnerability Scanner Prototype. *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*. https://doi.org/10.1109/icssit48917.2020.9214099

[24] Dwyer, O. P., Marnerides, A. K., Giotsas, V., & Mursch, T. (2019). Profiling IoT-Based Botnet Traffic Using DNS. *2019 IEEE Global Communications Conference (GLOBECOM)*. https://doi.org/10.1109/globecom38437.2019.9014300

[25] Semic, H., & Mrdovic, S. (2017). IoT honeypot: A multi-component solution for handling manual and Mirai-based attacks. *2017 25th Telecommunication Forum (TELFOR)*. https://doi.org/10.1109/telfor.2017.8249458

[26] Hussain, F., Abbas, S. G., Pires, I. M., Tanveer, S., Fayyaz, U. U., Garcia, N. M., Shah, G. A., & Shahzad, F. (2021). A Two-Fold Machine Learning Approach to Prevent and Detect IoT Botnet Attacks. *IEEE Access*, *9*, 163412–163430. https://doi.org/10.1109/access.2021.3131014

[27] Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., & Sakurai, K. (2020). Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture. *Sensors*, *20*(16), 4372. https://doi.org/10.3390/s20164372

[28] Education,I.C.(2021). *Artificial Intelligence (AI)*. IBM. https://www.ibm.com/cloud/learn/what-is-artificial-intelligence

[29] Education,I.C.(2021). *Machine Learning*. IBM. https://www.ibm.com/cloud/learn/machine-learning

[30] Brownlee, J. (2019). *How Machine Learning Algorithms Work (they learn a mapping of input to output)*. Machine Learning Mastery. https://machinelearningmastery.com/how-machine-learning-algorithms-work/

[31] *Supervised vs. Unsupervised Learning: What's the Difference?* (2021). IBM. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

[32] Aggarwal, C. C. (2015). *Data Mining: The Textbook* (2015th ed.). Springer.

[33] Gandhi, R., & Li, Y. (2021). Comparing Machine Learning and Deep Learning for IoT Botnet Detection. *2021 IEEE International Conference on Smart Computing (SMARTCOMP)*. https://doi.org/10.1109/smartcomp52413.2021.00053

[34] Maudoux, C., Boumerdassi, S., Barcello, A., & Renault, E. (2021). Combined Forest: a New Supervised Approach for a Machine-Learning-based Botnets Detection. *2021 IEEE Global Communications Conference (GLOBECOM)*. https://doi.org/10.1109/globecom46510.2021.9685261

[35] Nanthiya, D., Keerthika, P., Gopal, S., Kayalvizhi, S., Raja, T., & Priya, R. S. (2021). SVM Based DDoS Attack Detection in IoT Using Iot-23 Botnet Dataset. *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*. https://doi.org/10.1109/i-pact52855.2021.9696569

[36] Mehra, M., Paranjape, J. N., & Ribeiro, V. J. (2021). Improving ML Detection of IoT Botnets using Comprehensive Data and Feature Sets. *2021 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. https://doi.org/10.1109/comsnets51098.2021.9352943

[37] Tikekar, P. C., Sherekar, S. S., & Thakre, V. M. (2021). Features Representation of Botnet Detection Using Machine Learning Approaches. *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)*. https://doi.org/10.1109/iccica52458.2021.9697320

[38] Shin, J., Choi, S. H., Liu, P., & Choi, Y. H. (2019). Unsupervised multi-stage attack detection framework without details on single-stage attacks. *Future Generation Computer Systems*, *100*, 811–825. https://doi.org/10.1016/j.future.2019.05.032

[39] Bezawada, B., Bachani, M., Peterson, J., Shirazi, H., Ray, I., & Ray, I. (2018). Behavioral Fingerprinting of IoT Devices. *Proceedings of the 2018 Workshop on Attacks and Solutions in Hardware Security*. https://doi.org/10.1145/3266444.3266452

[40] Zhang, L., Gong, L., & Qian, H. (2020). An Effiective IoT Device Identification Using Machine Learning Algorithm. *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. https://doi.org/10.1109/iccc51575.2020.9345256

[41] Sharma, S., Zavarsky, P., & Butakov, S. (2020). Machine Learning based Intrusion Detection System for Web-Based Attacks. 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS). https://doi.org/10.1109/bigdatasecurity-hpsc-ids49724.2020.00048

[42] Ahmad, R., & Alsmadi, I. (2021). Machine learning approaches to IoT security: A systematic literature review. *Internet of Things*, *14*, 100365. https://doi.org/10.1016/j.iot.2021.100365

[43] Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., & Sakurai, K. (2020). Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture. *Sensors*, *20*(16), 4372. https://doi.org/10.3390/s20164372

[44] Zhang, L., Gong, L., & Qian, H. (2020). An Effiective IoT Device Identification Using Machine Learning Algorithm. *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. https://doi.org/10.1109/iccc51575.2020.9345256

[45] Guerra-Manzanares, A., Bahsi, H., & Nomm, S. (2019). Hybrid Feature Selection Models for Machine Learning Based Botnet Detection in IoT Networks. *2019 International Conference on Cyberworlds (CW)*. https://doi.org/10.1109/cw.2019.00059

[46] Wiyono, R. T., & Cahyani, N. D. W. (2020). Performance Analysis of Decision Tree C4.5 as a Classification Technique to Conduct Network Forensics for Botnet Activities in Internet of Things. *2020 International Conference on Data Science and Its Applications (ICoDSA)*. https://doi.org/10.1109/icodsa50139.2020.9212932

[47] Gu, Q. (2011). *Generalized Fisher Score for Feature Selection | Semantic Scholar*. Semantic Scolar. https://www.semanticscholar.org/paper/Generalized-Fisher-Score-for-Feature-Selection-Gu-Li/464c737376f34c5d38d92940f711abd94d37e84f

[48] Tsuda, K., Kawanabe, M., & Müller, K. R. (2003). *Clustering with the Fisher Score*. Research Gate.https://www.researchgate.net/publication/2880442_Clustering_with_the_Fisher_Score/citation/download

[49] *Compare the effect of different scalers on data with outliers*. (2022). Scikit-Learn. https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html

[50] Brownlee, J. (2021). *Nested Cross-Validation for Machine Learning with Python*. Machine Learning Mastery. https://machinelearningmastery.com/nested-cross-validation-for-machine-learning-with-python/

[51] Castilla, J. A. (2021). *Nested Cross-Validation | Guide to Nested Cross Validation*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/03/a-step-by-step-guide-to-nested-cross-validation/

[52]*Supervised learning*. (2022). Scikit-Learn. https://scikit-learn.org/stable/supervised_learning.html#supervised-learning

[53] Amrutha, K. (2022). *Decision Tree Machine Learning Algorithm*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/decision-tree-machine-learning-algorithm/

[54]Varun, N. (2022). *Introduction to KNN Algorithms*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/introduction-to-knn-algorithms/

[55] Sruthi, E. R. (2021). *Random Forest | Introduction to Random Forest Algorithm*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

[56] Guerra-Manzanares, A., Medina-Galindo, J., Bahsi, H., & Nõmm, S. (2022). *MedBIoT Data Set*. TalTech. https://cs.taltech.ee/research/data/medbiot/

[57] Guerra-Manzanares, A., Medina-Galindo, J., Bahsi, H., & Nõmm, S. (2020). MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network. *Proceedings of the 6th International Conference on Information Systems Security and Privacy*. https://doi.org/10.5220/0009187802070218

[58] Chunduri, H., Gireesh Kumar, T., & Charan, P. V. S. (2021). A Multi Class Classification for Detection of IoT Botnet Malware. *Communications in Computer and Information Science*, 17–29. https://doi.org/10.1007/978-3-030-76776-1_2

[59] Hasan, M., Islam, M. M., Zarif, M. I. I., & Hashem, M. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. *Internet of Things*, *7*, 100059. https://doi.org/10.1016/j.iot.2019.100059

[60] Greenlee, M. (2022). *What Is a Botnet Attack? A Guide for Security Professionals*. Security Intelligence. https://securityintelligence.com/articles/what-is-botnet-attack/

[61] Aziz, A., & Siddiqi, M. A. (2021). *Network Intrusion Detection Techniques using Machine Learning*. GISPP - Global InfoSec Pakistani Professionals. https://www.gispp.org/2021/01/25/network-intrusion-detection-techniques-using-machine-learning/

[62] Ellis, L. (2022). *Everything You Need to Know About Botnet Attacks*. Abusix. https://abusix.com/resources/botnets/everything-you-need-to-know-about-botnet-attacks/

License

Non-exclusive license to reproduce thesis and make thesis public
I,
Anel Abylkassymova

1. herewith grant the University of Tartu a free permit (non-exclusive license) to reproduce, for preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,
   title of thesis: Machine Learning Method for detecting botnet attacks originated from the IoT networks,
   supervised by: Hayretdin Bahsi, Ph.D, Sven Nõmm, Ph.D, Raimundas Matulevicius, Ph.D

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons license CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified on p. 1 and 2.

4. I certify that granting the non-exclusive license does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Anel Abylkassymova
16.05.2022