



Article

Cyber Threat Intelligence for IoT Using Machine Learning

Shailendra Mishra ¹, Aiman Albarakati ^{2,*} and Sunil Kumar Sharma ^{2,*}

¹ Department of Computer Engineering, College of Computer and Information Sciences, Majmaah University, Majmaah 11952, Saudi Arabia

² Department of Information System, College of Computer and Information Sciences, Majmaah University, Majmaah 11952, Saudi Arabia

* Correspondence: a.albarakati@mu.edu.sa (A.A.); s.sharma@mu.edu.sa (S.K.S.)

Abstract: The Internet of Things (IoT) is a technological revolution that enables human-to-human and machine-to-machine communication for virtual data exchange. The IoT allows us to identify, locate, and access the various things and objects around us using low-cost sensors. The Internet of Things offers many benefits but also raises many issues, especially in terms of privacy and security. Appropriate solutions must be found to these challenges, and privacy and security are top priorities in the IoT. This study identifies possible attacks on different types of networks as well as their countermeasures. This study provides valuable insights to vulnerability researchers and IoT network protection specialists because it teaches them how to avoid problems in real networks by simulating them and developing proactive solutions. IoT anomalies were detected by simulating message queuing telemetry transport (MQTT) over a virtual network. Utilizing DDoS attacks and some machine learning algorithms such as support vector machine (SVM), random forest (RF), k-nearest neighbors (KNN) and logistic regression (LR), as well as an artificial neural network, multilayer perceptron (MLP), naive Bayes (NB) and decision tree (DT) are used to detect and mitigate the attack. The proposed approach uses a dataset of 4998 records and 34 features with 8 classes of network traffic. The classifier RF showed the best performance with 99.94% accuracy. An intrusion detection system using Snort was implemented. The results provided theoretical proof of applicability and feasibility.

Keywords: cyberthreat; IoT security; embedded subsystems; MQTT protocol



Citation: Mishra, S.; Albarakati, A.; Sharma, S.K. Cyber Threat Intelligence for IoT Using Machine Learning. *Processes* **2022**, *10*, 2673. <https://doi.org/10.3390/pr10122673>

Academic Editors: Yo-Ping Huang and Xiong Luo

Received: 1 October 2022

Accepted: 30 November 2022

Published: 12 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of Things (IoT) has revolutionized technology in many areas of life. The Internet of Things model aims to connect people to everything, everywhere, all the time. In general, the Internet of Things is characterized by a three-layer architecture consisting of perception, network, and application layers. To ensure the stability of the Internet of Things, security principles must be applied at each layer [1]. Moreover, the number of vulnerabilities in embedded subsystems increases with technological progress. Therefore, embedded security is an integral part of embedded system design. With a technological revolution that enables human-to-human and machine-to-machine communication, the Internet of Things (IoT) will allow us to develop new online applications and services for all living beings to improve our quality of life. Trust is critical in the context of IoT devices and services. In addition, IoT security devices and networks must be monitored and investigated to prevent damage to system components from posing unacceptable risks and to ensure effective security by analyzing the social behavior and ethical use of IoT technologies [2].

The Internet of Things IoT systems have been found to have vulnerabilities that make them susceptible to various types of attacks. In addition to the risk of losing important information, other security issues such as confidentiality, privacy, and accessibility also pose a threat. By monitoring IoT devices and vulnerable resources, it is possible to determine what types of attacks are likely to occur against low-cost IoT devices [3].

In recent years, the number of IoT devices in our homes and lives has increased significantly. Technology is advancing rapidly, and the number of computers connected to the Internet is increasing. It is expected that in the next few years, this number will multiply and become much larger than it is today. There will be more devices, but they will be different. The fear is that hackers will take advantage of the growth of this technology to launch attacks. They will rely largely on discovered vulnerabilities and inadequate user security settings. This means that not only is this device vulnerable, but so are other devices on the network. Vulnerabilities in IoT systems can be a problem and lead to devices being exposed to many types of attacks, including the risk of denial of service and security issues such as confidentiality, privacy, availability, and vulnerability to attack [4,5].

MQTT is a protocol that can be vulnerable to many forms of cyberattacks and is used in this study to address issues related to IoT and embedded systems [6]. It helps in the transmission of low-bandwidth data connection, authentication, communication, and termination and also the publish/subscribe model [7]. The problem occurs when the MQTT protocol receives messages or requests from nearby nodes in the same area of the network, especially in networks where authentication has not been performed. The most well-known of these attacks is the HELLO flooding attack, which targets an IoT device and floods it with contact requests until the service is discontinued [8]. To enable routing and data transfer between IoT devices, the Cooja IoT simulator was used to simulate HELLO flooding attacks in this paper.

In recent years, machine learning has made significant progress as machine intelligence has evolved from a laboratory curiosity to practical machines in several important applications. IoT device intelligence provides important solutions to new or zero-day attacks, as these devices can be monitored. Using powerful data exploration methods (ML), the “normal” and “abnormal” behavior of IoT devices and components in their environment can be determined [9]. These methods are, therefore, of great importance for transforming the security of IoT systems into a security-based intelligent system and not only for secure communication between devices.

This research is also a step forward in identifying cybersecurity threats in modern technology and identifying vulnerabilities in organizations that deploy technologies and systems. We identify key areas where vulnerabilities may occur within the system to develop a hardened cybersecurity defense methodology, analyze various classifiers used for DDoS attack detection, including support vector machine (SVM), random forest (RF), k-nearest neighbors (KNN), and logistic regression (LR), as well as an artificial neural network, multilayer perceptron (MLP), naive Bayes (NB), and decision tree (DT), and propose a cybersecurity incident response plan to help organizations efficiently and quickly respond to security incidents and also implement intrusion detection systems using Snort to monitor server and system activities in real-time. The key details of the experiment include protocol analysis, network flow analysis, intrusion detection, vulnerability scanners, cyber-attack defense, and return to normality. The experimental results are compared with the existing works [9–12] for validation. The results show that the random forest algorithm (RF) has high accuracy in detecting DDoS attacks compared to existing research work.

The DDoS accuracy detection rate reported in [9–12] using support vector machines (SVM), random forest (RF), k-nearest neighbors (KNN), and logistic regression (LR) classifier artificial neural networks (ANN) is in the range of 63% to 98%. In this research, we investigated whether machine learning techniques, including support vector machines (SVM), random forest (RF), k-nearest neighbors (KNN), logistic regression (LR), naive Bayes (NB) and decision tree (DT) classifiers can be useful tools to support DDoS attack detection. Additionally, an artificial neural network-based approach called Multilayer Perceptron (MLP) has been investigated. These techniques could detect malicious activities and attacks, improve human analysis, and automate repetitive security tasks. The dataset used in this research was collected by Al-Kasassbeh et al. [13]. The results obtained in this research suggest that random forest (RF) is more suitable for anomaly detection using machine learning techniques.

The main objectives of the study are to:

- Identify issues related to the MQTT protocol and possible attacks in different types of networks and their countermeasures using ML;
- Simulate and analyze different network attacks in IoT networks by simulating MQTT over a virtual network, which allows us to analyze and detect anomalies;
- Identification of the best classifier with high accuracy using some types of DDoS attacks and one or two machine learning algorithms.

This paper is divided into six sections. The Section 1 provides the purpose of the study, its background, and detailed information about the topics covered in the study. Through this particular study, questions need to be answered. The Section 2 presents the literature review of related work by analyzing and reviewing the previous research on this topic, especially explaining the main gaps in the study. The Section 3 deals with an overview of the IoT and the MQTT protocol, and the proposed methods to be used in the research process are discussed in Section 4. The Section 5 deals with the experimental setup, analysis, results and discussion, conclusions, and future scope of the work discussed in Section 6.

2. Related Work

With thousands of new devices being connected every day, the wireless IoT infrastructure is growing significantly. This has enabled the introduction of numerous smart applications that are transforming people's lifestyles. However, as security has always played a secondary role in innovation, significant questions have been raised about the security of these infrastructures. The Internet of Things (IoT) integrates almost all objects in the environment to create new digitized services that improve people's lifestyles, whether physically or virtually, via the Internet. Currently, many IoT technologies, including smart wearable devices, connected health services, connected cars, and others, have a direct impact on people's daily activities. This raises many security issues, even as the IoT offers myriad benefits. Addressing these challenges should be a top priority for IoT manufacturers to continue the successful adoption of IoT applications. IoT device owners should ensure that their devices are equipped with appropriate security mechanisms. The number of security threats and cybercrimes has increased significantly with the development of the IoT [14].

Improper device updates, active device monitoring, inadequate and insecure protocols, and user ignorance are some of the challenges facing IoT [15]. Using approaches to the security of IoT components, environments, and systems, existing security solutions, best privacy models, and the need for different layers of IoT applications, the authors in [16] discuss the background of IoT systems and security measures.

In [17], an edge/cloud-based IoT layered model was proposed, implemented, and evaluated. In the lowest layer, Amazon's Web Service (AWS) IoT nodes are represented by virtual machines. A hardware kit for Raspberry Pi 4 with AWS support was used for the middle layer (edge). AWS can be used to run the IoT environment on the upper layer (the cloud). Between these two layers, a security protocol and a critical management session have been established to ensure user privacy. In the proposed cloud/edge-enabled IoT model, security certificates have been implemented to enable data transmission between the layers. It is proposed to use the best security techniques at each layer—edge, cloud, and IoT layer—to close potential security gaps and protect against cybersecurity threats.

Objects and devices in the IoT environment can be addressable, identifiable, and locatable via low-cost sensors. Although the IoT offers countless benefits, there are also many challenges, especially related to privacy and security. Addressing these issues and ensuring protection and privacy for IoT services and products must be a fundamental priority. IoT-related services must be secure, and users must trust the devices. The IoT brings together a significant number of smart devices and components that communicate with each other with minimal human intervention. The security measures used for IoT devices, such as authentication, network security, encryption, access control, and application security, are

ineffective, and their vulnerabilities are deeply rooted. Therefore, existing security methods need to be improved to ensure that the IoT ecosystem is indeed secure [18,19].

IoT uses special protocols for communication and information transfer. IoT devices of different structures use these protocols for communication and data exchange. To secure IoT networks, it is necessary to monitor protocol behavior and data traffic. The most important protocols for IoT networks include Wi-Fi, Zigbee, Bluetooth, and MQTT. MQTT is a lightweight protocol for exchanging simple data streams between sensors and applications. The MQTT protocol is the most widely used and is expected to become the de facto standard for IoT communication. It uses the TCP/IP network to enable simple data transmission. A publisher/subscriber model facilitates communication between devices using MQTT, resulting in lightweight message exchange [20].

In the paper [21], the authors discussed three types of attacks (DoS attack, RPL rank attack, and wormhole attack) on sensor networks using the Cooja simulation system running on Contiki OS. The tool contains many types of nodes. In Cooja simulations, “sky node” and client server were used as node types. Multiple nodes can be added, including the source code of the new node. Adding the source code may require a high level of knowledge and experience in dealing with the implemented network. A packet viewer can be used for devices such as temperature sensors, motion sensors, and light sensors. In the case of an attack, a large dataset has already been created by combining normal, network, and dataset traffic. This introduced a new dataset that can be used as a basis for learning-based intrusion detection systems (IDS) designed to attack IoT devices.

A moving target defense (MTD) removes the biggest attack advantage of static computer systems - which is a major advantage held by attackers. In order to analyze this impact, MTD as well as cyber-attack concepts need to be formalized. A theory of cyber-attacks is presented in [22], to support the understanding and analysis of how MTD systems interact with attacks. In order to find interactions between attackers and MTD systems, the relation between attack parameters and MTD configuration parameters is usually exploited.

A distributed denial-of-service (DDoS) attack is a self-imposed mechanism to disrupt normal network traffic by flooding the network to the point where the network or associated resources are significantly slowed or crippled. Many studies [13,23,24] have used Simple Network Management Protocol (SNMP-MIB) data. Sometimes, statistical analysis of Management Information Base (MIB) data is also used. Other approaches have recently used machine learning techniques and artificial neural networks for network attacks and other anomalies. The algorithms used are support vector machine (SVM), random forest (RF), k-nearest neighbors (KNN), and logistic regression (LR).

In [9], researchers analyzed the DDoS detection in the SDN using the proposed algorithm DDAML and compared the result with the KNN and SVM algorithms. Experimental results demonstrated that the proposed algorithms perform better in comparison to other algorithms. In [10], researchers used UNBS-NB 15 and KDD99 Botnet DDoS attack detection using SVM, ANN, naïve Bayes (NB), decision tree (DT), and unsupervised learning (USML) investigated for performance measures (accuracy, sensitivity, and specificity). In [11], researchers used machine learning (SVM, KNN) and neural network (ANN) algorithms to detect DDoS attacks in software-defined networks. Using SVM, 92.6%, KNN, 95.67%, ANN, 91.07%, and NB, 94.48% of DDoS attacks were successfully detected. In [12], researchers used machine learning (SVM) for SDN self-defense systems, and detection accuracy was found to be in the range of 97–98%.

The literature review has shown that there is a need to increase the level of protection in embedded subsystems and IoT devices. Several authors have investigated this using different methods, techniques, and research tools. The physical and technical aspects are also mentioned. Compared to all the research mentioned in the relevant study and after investigating their methods, this research takes the approach of extracting and analyzing data in and from networks of embedded subsystems, which are very valuable for researchers. This has been studied by several authors using different approaches, tactics and research tools. The technical and physical aspects are also discussed.

3. MQTT: Threat Model

The default configuration of MQTT does not support authentication or payload encryption schemes because it is a lightweight, simple, and generic protocol. Wildcard support appears to be a glaring oversight since it creates a flaw that allows an adversary to listen in on any data passing through the broker. The MQTT is not designed with security as a primary concern [6,25]. In order to enable bidirectional communication and remote control of IoT devices, middleware based IoT application protocols are indispensable. MQTT is one of the most widely adopted IoT application protocols. Identifying possible threats is necessary before implementing countermeasures in MQTT-based IoT environments. The MQTT threat model is presented, and the attack against MQTT brokers is evaluated. That protocol used the publish-subscribe model. This paradigm demonstrates how the “broker,” a crucial element in MQTT, separates the “publisher,” a client that publishes messages, from the “subscribers,” which are other clients that receive the messages. This task involves receiving messages from the publisher and sending them to the subscribers. Figure 1 shows the MQTT threat model. As one of the most popular IoT application protocols, Patel, C. et al. [6] outlined the threat model for MQTT and examined the denial of service (DoS) attack encountered when MQTT brokers are attacked. A virtual machine testbed was set up for the investigation of the performance of an MQTT broker server during a DDoS attack.

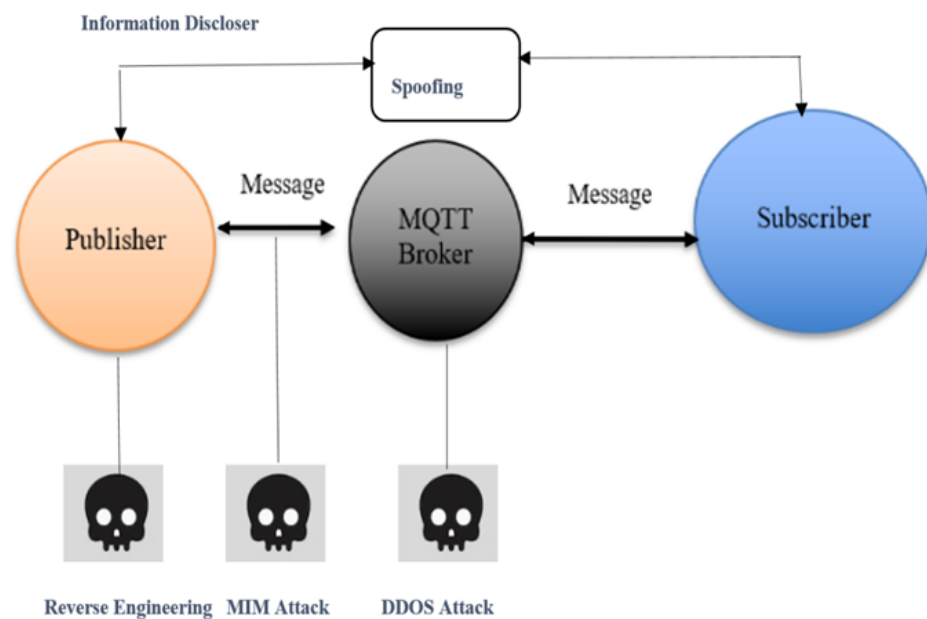


Figure 1. MQTT Threat Model.

The proper identification of possible threats is needed to apply any countermeasure to the MQTT. Formally, attacks are related to the attacker’s knowledge and are known as information parameters. The relation between the attack’s parameters and the MTD configuration parameters is usually exploited to find an interaction between the attackers and the MTD system [22]. The information parameters as a name-value pair for the proposed system can be described as follows.

An information parameter (ψ) can take a value (v) based on its type (n) and can be represented as $\psi = (n, v)$.

1. An assumed information parameter ($\hat{\psi}$) = $\{\psi_1, \psi_2, \psi_2, \dots, \psi_n, \dots\}$ refers to a domain of all possible values (v) assigned to m represented as $m = (\psi.v)$.
2. A composite parameter (ψ) refers to a set of all sub-information parameters given as $\psi = \langle n, \{\psi_1, \psi_2, \dots, \psi_n\} \rangle$.

Attacker: The attacker effectively attacks the publisher by using internal user launching and reverse engineering. In reverse engineering, without having much (if any) knowledge

of the mechanics of how it does so, one attempts to understand how a previously established item, process, system, or piece of software executes a task. It basically involves opening up or dissecting a system in order to learn how it functions in order to duplicate or enhance it. Depending on the system being examined and the technologies being utilized, the information collected through reverse engineering can help with learning how something works, reusing outdated items, doing security checks, and carrying out other activities. Data is manipulated in MIM attacks through eavesdropping. On both the broker and the subscriber, attackers utilize brute force and execute DoS/DDoS attacks. The planner's IP address, server port number, and operating system must all be known to the attackers. Attacker knowledge in the form of information parameters can be used to gain these. A broad grasp of captures as well as specialized knowledge, such as how to use `mmap()`, a POSIX-compliant Unix system function that maps devices or files into memory, are also necessary for the attacker. A memory-mapped file I/O method is used. Since file contents are not immediately read from the disk and initially do not use any physical RAM at all, demand paging is used.

Attack Type: We assume that earlier attacks have provided the attacker with knowledge of the planner's OS system, IP address, and port number. There are several types of assaults, including MIM (man-in-the-middle), brute-force, DoS (denial of accommodation), and DDoS attacks. An attack is defined as the effect on the system information. By monitoring the impact of information change, an attack can easily be identified. An attack (a) is a tuple of information parameters $\Omega = (\Omega_{pre}, \Omega_{post})$ that, when executed, copies the value of ψ_2 into ψ_1 , which is denoted as $(\psi_1.v = \psi_2.v)$. Formally, the execute operation as execute (a) $\Leftrightarrow (a.\psi_1.v = a.\psi_2.v)$. Different types of attack specifications discussed in [22] are shown in Table 1.

Table 1. Types of attack specification.

Type	Ω_{pre}	Ω_{post}
ϕ_1	$\psi_{d1}.ip \neq \psi^x_{d1}.ip$	$\langle \psi^x_{d1}.ip, \psi_{d1}.ip \rangle$
ϕ_2	$\psi^x_{d1}.ip = \psi_{d1}.ip \wedge \psi^x_{d1}.port \neq \psi_{d1}.port$	$\langle \psi^x_{d1}.port, \psi_{d1}.port \rangle$
ϕ_3	$\psi^x_{d1}.ip = \psi_{d1}.ip \wedge \psi^x_{d1}.port = \psi_{d1}.port \wedge \psi^x_{d1}.os \neq \psi_{d1}.os$	$\langle \psi^x_{d1}.os, \psi_{d1}.os \rangle$
ϕ_4	$\psi^x_{d1}.ip = \psi_{d1}.ip \wedge \psi^x_{d1}.port = \psi_{d1}.port \wedge \psi^x_{d1}.os = \psi_{d1}.os \wedge \psi^x_{d1}.vul \neq \psi_{d1}.vul$	$\langle \psi^x_{d1}.vul, \psi_{d1}.vul \rangle$
ϕ_5	$\psi^x_{d1}.ip = \psi_{d1}.ip \wedge \psi^x_{d1}.port = \psi_{d1}.port \wedge \psi^x_{d1}.os = \psi_{d1}.os \wedge \psi^x_{d1}.vul = \psi_{d1}.vul$	$\langle \psi_{d1}.exa, \psi^x_{d1}.exa \rangle, \langle \psi^x_{d1}.exa, \psi^x_{d1}.exa \rangle$
ϕ_6	$\psi^x_{d1}.ip = \psi_{d1}.ip \wedge \psi^x_{d1}.port = \psi_{d1}.port \wedge \psi^x_{d1}.exa = \psi_{d1}.exa \wedge \psi^x_{d1}.root \neq \psi_{d1}.root$	$\langle \psi^x_{d1}.root, \psi_{d1}.root \rangle$

The Table 1. Shows that types of attack specification on the network. In the proposed system, it is assumed that x has the goal to exploit the target system privileges, and it can do it by following a sequence of different types of attacks, $(\varphi = \varphi_1, \varphi_2, \dots, \varphi_5)$, where

- φ_1 —captures the IP address of the target;
- φ_2 —captures the port number of an application;
- φ_3 —captures the features of the OS;
- φ_4 —exploits the dynamicity of the application;
- φ_5 —deploys an agent on the target to exploit.

4. Research Methodology

The methodology used for this research is quantitative and qualitative. In our reviews of the previous research, this study served to collect and analyze information related to cyber issues and include threats to IoT networks that have taken the communication protocols on which IoT devices depend, as well as detection measures using techniques based on the simulation of Internet of Things networks. The suggested methodology of this research work is shown in Figure 2.

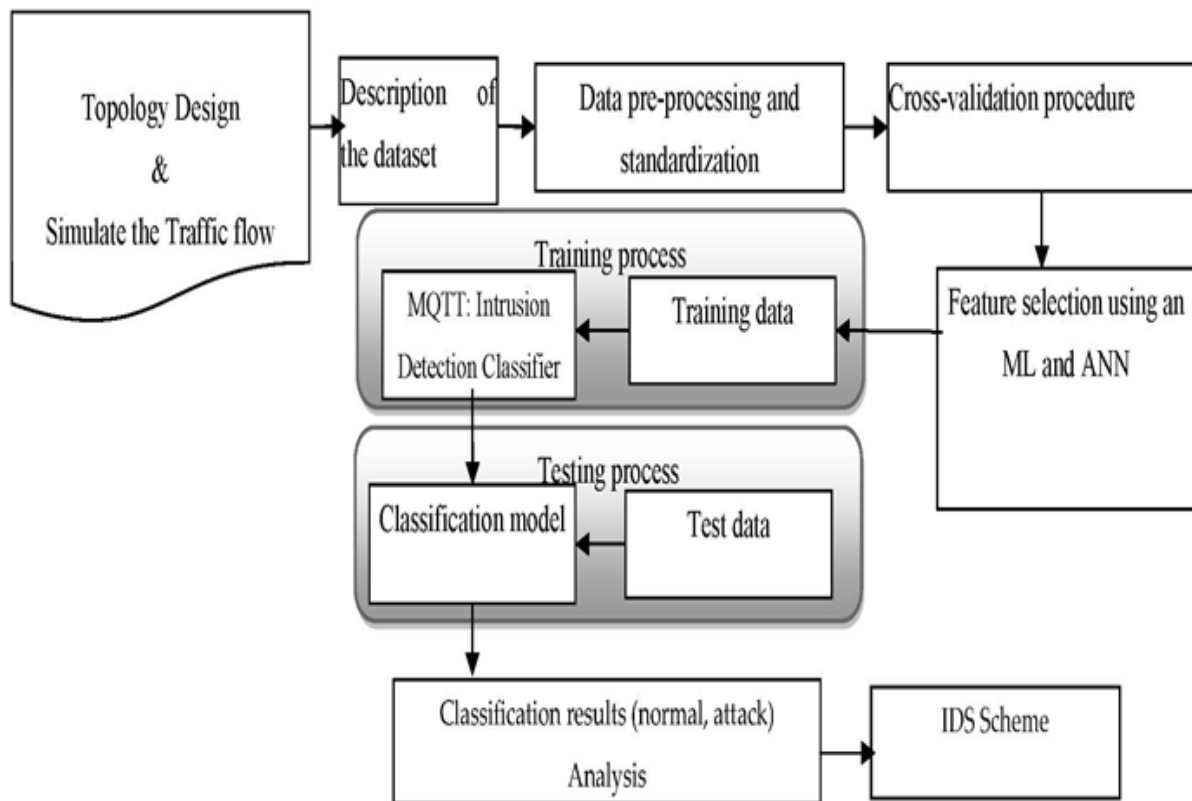


Figure 2. The suggested methodology of this research work.

The research process is as follows:

Step 1: Build a virtual network, specifically a virtual network created to monitor the normal or abnormal behaviour of the network.

Step 2: Simulate the traffic flow.

- In the first scenario, the legitimate nodes were distributed, and normal communication was established between them. We collected data using the features available in the tool, such as temperature, battery voltage, packet consumption, and loss rate.
- In the second scenario, an attacking node was created using the flooding attack, which attacks the neighboring nodes.

The subsequently extracted data can be useful to assist IDS in the early detection of abnormal behavior attack communications in the Internet of Things.

Step 3: Classify the traffic and extract the feature.

Step 4: Conduct packet analysis.

Step 5: Train the network for attack detection and mitigation.

Step 6: Conduct a classification and performance analysis.

Step 7: Execute the IDS scheme.

4.1. Intrusion Detection Classifier

Below are the steps to obtain the dataset and elements of the IoT environment. In this case, an attack on the server was launched from another computer. The CSV data were generated by recording, dissecting, and tagging all traffic through the router. The steps to create the attack dataset are shown in Figure 3.

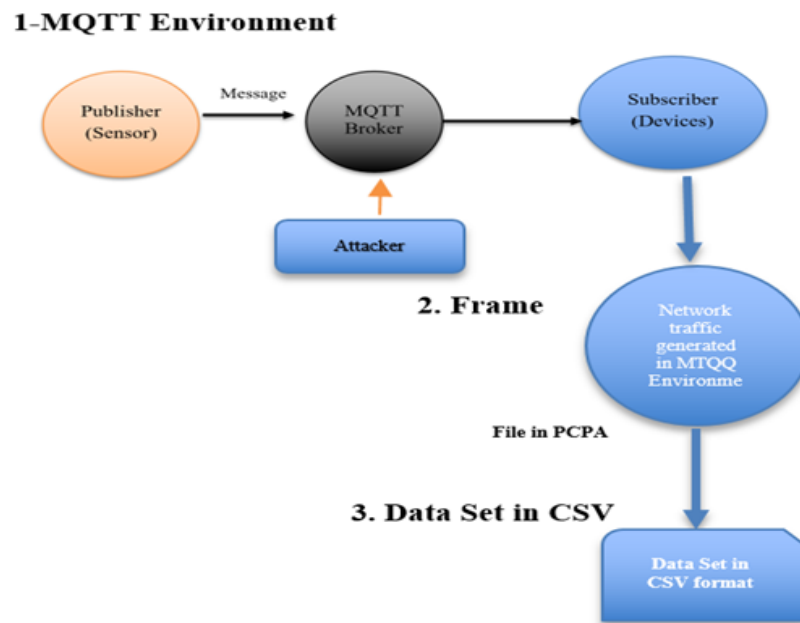


Figure 3. Steps to create the attack dataset.

To address the security issues of MQTT, one-classification techniques have been introduced using a classifier approach.

4.1.1. Description of the Dataset and Preprocessing of the Data

An MQTT brute-force attack, as well as network scanning attacks, are included in the dataset. The training dataset [13] contains 4998 records with 8 classes and 34 attributes. Data preprocessing steps were performed before the classification task. We removed the redundant columns with a high correlation of 0.9. Missing values in the data were treated with mean imputation, and scaling of the characteristics was also performed [26,27].

4.1.2. Cross-Validation Procedure

Cross-validation procedures (CV) [28] were used to evaluate the predictive models. Two cross-validation procedures were investigated: k-fold and leave-one-out. In the k-fold method, 5-fold cross-validation was performed with 80% of the data as the training set and 20% as the test set without replacement.

4.1.3. Artificial Neural Network (ANN)

One type of feed-forward ANN is the multilayer perceptron (MLP) class. It consists of an input channel, a hidden channel, and an output channel. MLP is based on backpropagation for supervised learning training. MLP can distinguish data that are not linearly separable. Rectified linear unit (ReLU) is used as the activation function for the input and hidden layers, and adaptive moment estimation (Adam) [29] was used as an optimizer.

4.1.4. Performance Measure

Each model's performance was evaluated using measurements such as sensitivity, specificity, and accuracy. For accuracy, completeness, and balance between precision and recall, other parameters such as precision, recall, and F1 score are required [30].

5. Experimental Setup and Analysis

We assumed that the attacker has started to launch cascading attacks on the broker to receive messages sent to and from the devices connected to it and that it can be exploited and corrupted. Since the broker used in IoT devices is designed to receive messages from any sending device in its vicinity, and the attacker can consume and corrupt IoT devices without

revealing its command, they easily fall into the trap of this type of attack. Therefore, there are operating systems and simulation tools that help us create virtual networks, observe the network and monitor the anomalies that may occur in its behavior. Taking these factors into account helps us investigate the cause of this anomaly, find out what is behind it, and find a solution. This is done before the anomaly occurs in a real network.

For this purpose, we created a virtual environment by installing VMware as a virtual operating system (ContikiOS) and then used the IoT simulation tool Cooja, which has many features to monitor network traffic and analyze the details that appear in it, which we relied on in this study. This experiment was also conducted for some types of attacks on IoT, but the type of HELLO flooding attack was not so much in focus. This is one of the network attacks where links or nodes become unavailable by generating a large amount of traffic. This can exhaust all network resources. Such attacks can be carried out by both internal and foreign attackers. To carry out a HELLO flooding attack, prompt messages are used.

This is also the contribution of the author, who has conducted many studies to define the problem and formulate a hypothesis to find a solution. Wireless Sensor Nodes (WSN) OS is an open-source operating system for an event-driven kernel underlying this lightweight and compact operating system. Preemptive multitasking at the system level is possible with this OS. Typical Contiki OS configurations require 40 kilobytes for ROM and 2 kilobytes for RAM. Prototype threads, preemptive multithreading, TCP/IP networking, and IPv6 are included in a full Contiki installation, as are an Internet browser and private web server, and various other utilities such as a screen saver and virtual network computations. Contiki has two types of communication stacks (Rime and uIP). uIP is a small communication stack of TCP/IP RFC-CONFORME, which simplifies Internet communication. Rime is a communication stack that has a low-power radio. It is said to be lightweight. It provides a set of basic communication options. Contiki OS Architecture 4.3.2.1. The attacker sends DODAG Information Solicitation (DIS) messages to neighboring nodes, which must reset their trickling timer, or sends unicast DIS messages to each node, which must respond with a DODAG Information (DIO) Object message to perform flooding attacks at the transport layer. Table 2 shows the simulation parameters.

Table 2. Simulation parameters.

Simulation Parameters	Value
Simulation time	3 Min
Simulation area	200 × 200 m
Mac protocol	IEEE 802.11
Number of motes	12
Number of mote types	3
Radio medium	UDGM: Distance Loss
Transmission range	50 m
Interference range	100 m
Number of sources	11
Number of destinations	1

5.1. Test Results

There were two scenarios for this simulation; the first was to evaluate the power consumption and the number of packet losses. The second was to detect anomalies in the WSN due to DoS attacks (HELLO flood attacks) to detect the hacker among many clients. With some skill in using the tools, we can achieve this simulation result. We performed this simulation process based on three criteria [31,32]. These criteria were battery voltage, average power consumption, and packet loss over time.

5.1.1. Scenario 1: Normal State of the Network (Power Consumption and Packet Loss)

Node 1 was a sink node (green color) that acted as a border router, and the other nodes were transmitting nodes that acted as normal sensors. Note that nodes (2, 4, and 5) were all

within the range of node 12. Figure 4 shows the normal state of the network, and Figure 5 shows the average power consumption. Node 1 and the nodes in yellow have the same roles as before. Node 12 became a malicious node that performed a flooding attack and directly affected nodes 2, 4, and 5. Figure 6 shows the steps to create the attack dataset.

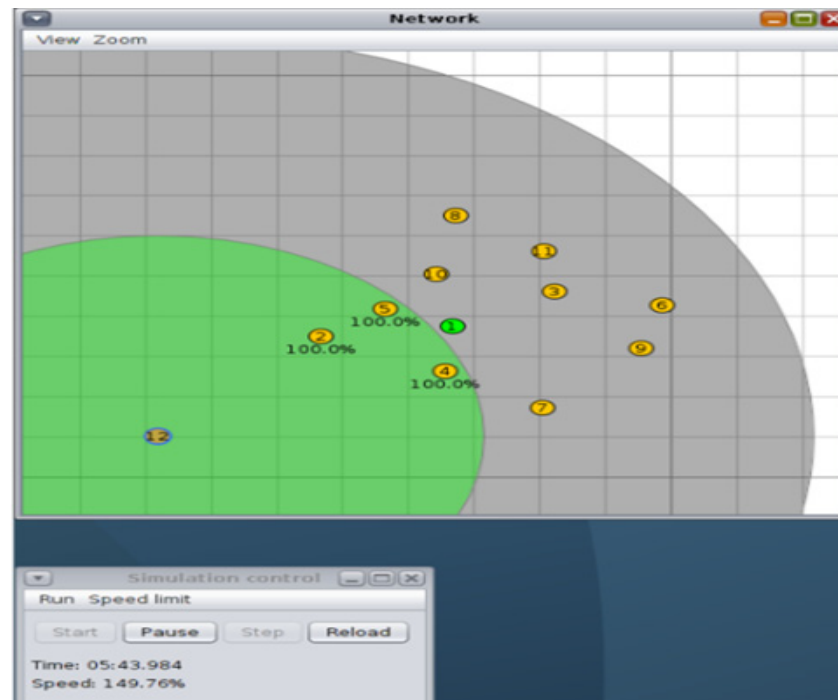


Figure 4. The normal state of the network.

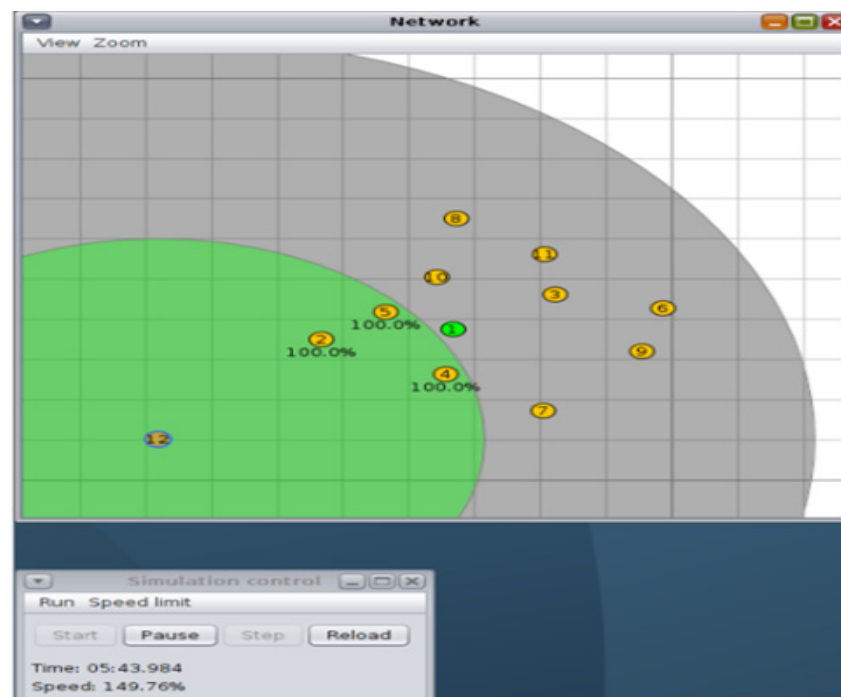


Figure 5. The normal state of the network (average power consumption).



Figure 6. Abnormal state of the network.

5.1.2. Scenario 2: Network under Attack (Attacks on WSNs to Detect Anomalies)

Figure 6 shows the network under attack, and Figure 7 shows the average power consumption during the attack. All sender nodes (2 to 12) consumed almost the same current, which is at a low level of about 1.2 mW. If we compare nodes (2, 4, 5) and 12 (attacker) with other nodes, we see that they have a much higher power consumption of about 30 mW. The power consumption of the other nodes is also higher than before and is about 2.5 mW or even higher. For node 12 (attacker), we found that radio transmission accounts for a large part of the power consumption since it is constantly sending messages to nodes 2, 4, and 5. For nodes 2, 4, and 5, the simulation shows that listening accounts for a large portion of the power consumption since they constantly receive requests from node 12.

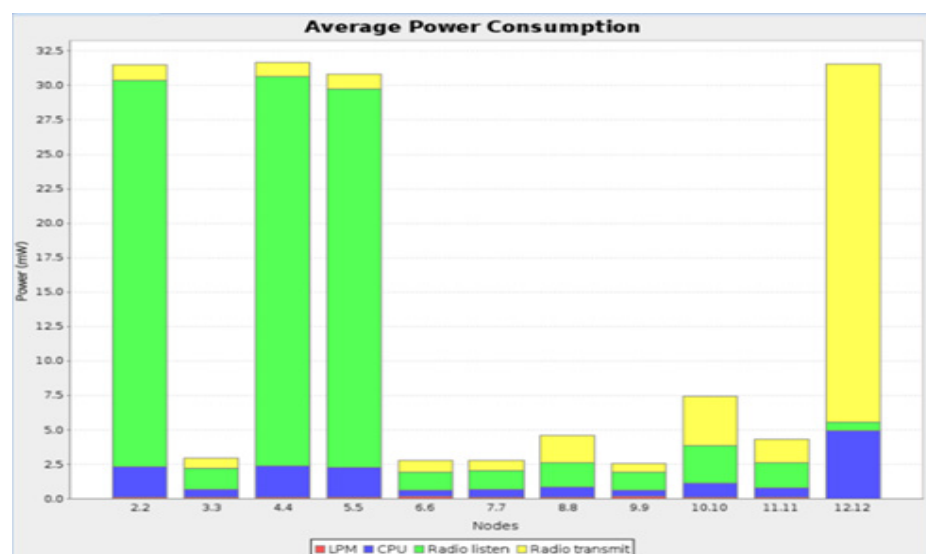


Figure 7. Average power consumption during the attack.

As a result of monitoring the network battery and voltage during the attack simulation, the effects of the attack are shown in Figure 8. Figure 9 shows the lost packets over time, and Figure 10 shows the state of the packets during the attack.

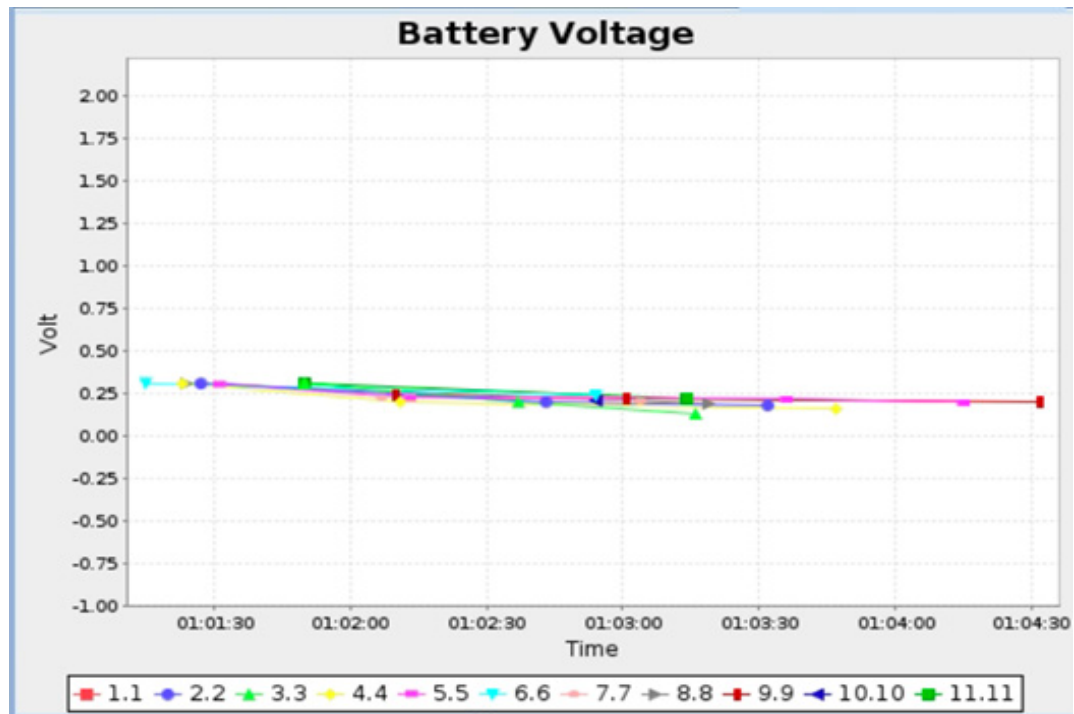


Figure 8. The anomaly on battery voltage.

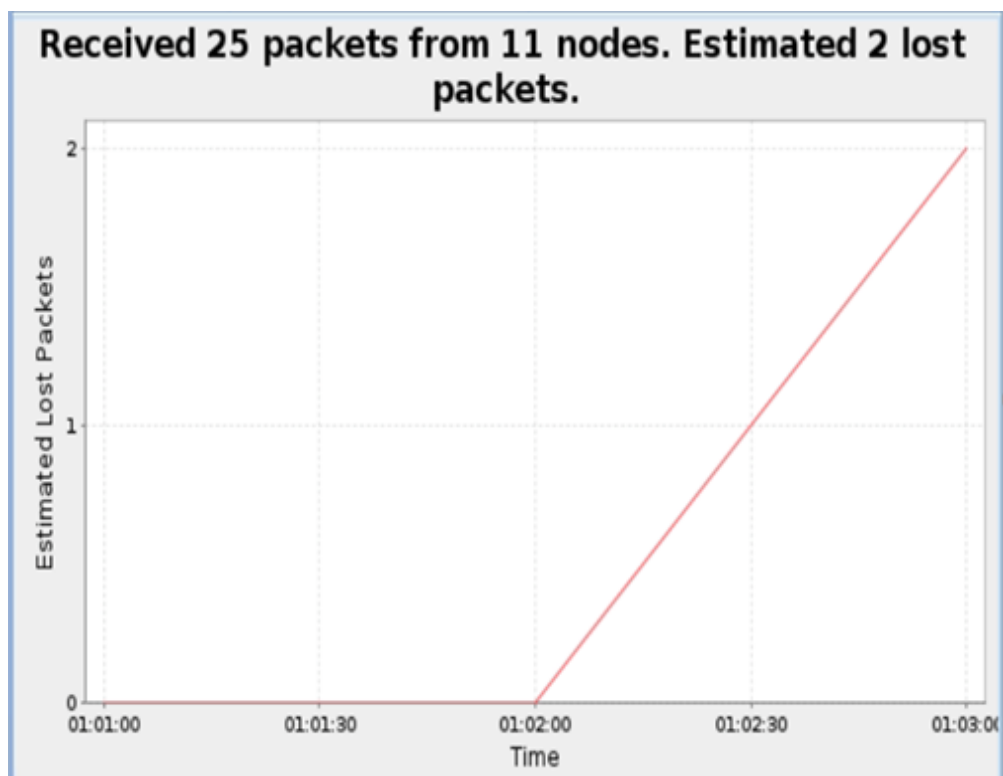


Figure 9. Packets lost overtime.

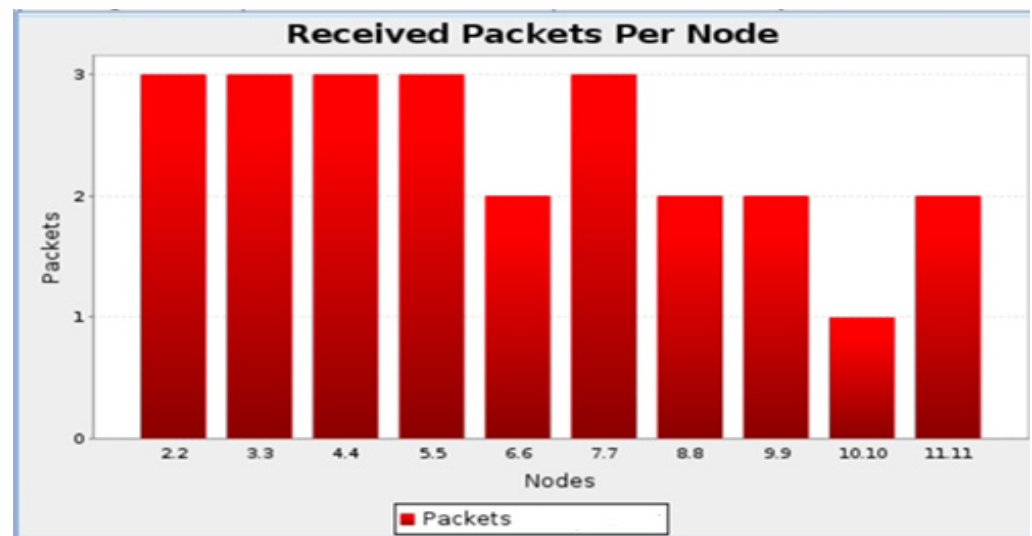


Figure 10. Packet states under the attack.

Node information under normal and attack conditions is shown in Tables 3 and 4, respectively. Through Contiki, we can determine how much time was spent in each location in the following states. Corresponding energy consumption (CPU power, LPM power, transmission power, and listening power) is high during the attack, as shown in Table 4.

Table 3. Node information during normal conditions.

Node	Received	Dups	Lost	Hops	Rtmetric	Ext	Churn	Beacon Interval	Reboot	CPU Power	Low Power Mode (LPM) Power
1.1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0 min, 08 s	0.00	0.00	0
2.2	5.00	0.00	0.00	1.00	521.00	16.00	0.00	2 min, 32 s	0.00	2.01	0.103
3.3	4.00	0.00	0.00	1.00	560.00	16.00	0.00	0 min, 08 s	0.00	0.54	0.147
4.4	4.00	0.00	0.00	1.00	512.00	16.00	0.00	0 min, 08 s	0.00	2.02	0.102
5.5	4.00	0.00	0.00	1.00	512.00	16.00	0.00	0 min, 08 s	0.00	2.00	0.103
6.6	2.00	0.00	0.00	1.00	1005.00	16.00	0.00	1 min, 21 s	0.00	0.50	0.148
7.7	3.00	0.00	0.00	1.00	1039.00	16.00	0.00	1 min, 38 s	0.00	0.50	0.149
8.8	2.00	0.00	1.00	2.00	1498.00	32.00	0.00	0 min, 24 s	0.00	0.91	0.136
9.9	3.00	0.00	0.00	1.00	1048.00	16.00	0.00	2 min, 32 s	0.00	0.55	0.147
10.1	2.00	0.00	1.00	2.00	1936.00	32.00	0.00	2 min, 43 s	0.00	1.04	0.132
11.11	2.00	0.00	1.00	1.50	1389.00	24.00	1.00	0 min, 48 s	0.00	0.72	0.142
Avg	2.82	0.00	0.27	1.14	910.91	18.18	0.09	1 min, 14 s	0.00	1.08	0.119

Table 4. Node information during attack conditions.

Node	CPU Power	LMP Power	Listening Power	Transmission Power	Power	On-time	Listen Duty Cycle	Transmit Duty Cycle
1.1	0.458	0.075	0.000	0.000	0.000		0.000	0.000
2.2	2.018	0.903	25.323	1.021	28.400	1 min	42.205	1.923
3.3	1.543	0.847	1.638	0.707	3.035	0 min	2.73	1.331
4.4	2.219	0.512	24.816	0.891	27.800	1 min	41.359	1.679
5.5	2.802	0.623	25.537	0.971	28.600	1 min	42.562	1.829
6.6	1.500	0.343	1.513	1.028	3.190	0 min	2.522	1.936
7.7	1.495	0.547	1.417	0.709	2.769	0 min	2.362	1.335
8.8	1.506	0.438	2.371	3.500	6.913	0 min	3.951	6.591
9.9	0.946	0.344	1.721	1.179	3.593	0 min	2.869	2.220
10.1	1.639	0.635	2.843	3.812	7.827	0 min	4.739	7.179
11.11	1.823	0.444	2.167	2.209	5.241	0 min	3.612	4.160
Avg	1.631	0.519	8.122	1.457	10.670	0 min	13.537	2.744

Compared to other tools (such as Wireshark and Omnet+), Cooja is easier to use and better suited for simulations as an operating system. To conclude the empirical analysis, it is important to reiterate the importance of attack detection to protect IoT networks from attackers. Any cybercriminal will try to gain access to an IoT device to harm its network by exploiting the loopholes that keep appearing in embedded subsystems and IoT networks. From the related study, it can be seen that while conducting this research work after a comprehensive study, the authors reveal various trends that affect these different capabilities. This might be a small problem for researchers, as most of them have no experience in programming. Therefore, this research takes the approach of using digital forensic tools to detect network anomalies and collect data and variables. The results are compared to determine whether the network traffic is normal. Then, the culprit in the hypothetical scenario that exploited the network is identified.

5.2. Classifier Approach: Dataset Description and Processing

The dataset used in this study was collected from Al-Kasassbeh et al. [13]. The computed results showed that the method of the random forest (RF) algorithm is better for anomaly detection using machine learning techniques.

Procedure:

1. The first step is to describe the dataset and how it was preprocessed.
2. In the second step, the different models are cross-validated to see which model is most predictive.
3. The third step is to process the data using machine learning algorithms.
4. In the fourth step, each model is evaluated for performance.

A dataset collected by Al-Kasassbeh et al. [13] was used. The dataset contains 4998 records with 8 classes and 34 attributes, as shown in Table 5. Data for the variables were collected and divided into the Interface, IP, TCP, and ICMP groups. The 34 attributes were collected during attack testing, in which the server (victim) was subjected to various sorts of attacks. The information gain ratio for each feature was used to rate each characteristic included in the data, making it possible to distinguish between features that are necessary and those that are not.

Data preprocessing steps were performed before the classification task. We removed the redundant columns with a high correlation of 0.9. Missing values in the data were handled using mean imputation. Scaling of the characteristics was also performed. A standard scalar was used to transform the data to have a mean of zero with a standard deviation of one [26]. The proper selection of SNMP-MIB variables is crucial to detecting anomalies on networks because no one variable can capture all anomalies. To detect anomalies more accurately, we focused on using effective variables. Router devices were used to collect MIB variables. A total of 34 MIB variables were selected from five MIB groups: IP, TCP, UDP, and ICMP (variables collected from specific router interfaces). A counter 32 is a non-negative four-byte integer that is continuously incremented from 0 to 232, and wraps back to 0 when it reaches its maximum value. As a result of a comprehensive investigation, we selected these variables among other MIB variables in the groups because they are more affected by attack traffic and are continuously updated based on the incoming and outgoing traffic over the network; therefore, they are more effective in detecting attacks.

By demonstrating the identification of as many of the most prevalent and contemporary attacks that can occur on various network layers as is practical, we demonstrate the strength and usefulness of SNMP-MIB data in network anomaly detection (network layer, transport layer and application layer). Using categorization techniques, we are currently testing the SNMP-MIB data in tests. In the first method, we divided the MIB variables into five categories with 34 attributes (Interface, IP, ICMP, TCP, and UDP), with a number of MIB variables belonging to each category. Then, each MIB group was subjected to the classification algorithms separately in order to demonstrate how each group is impacted by attacks and, ultimately, to identify the group or groups that are most successful at spotting anomalies. According to the preliminary findings of this strategy, each classifier

performs differently across the MIB groups, with a range of accuracy rates for the employed classifiers between high and low.

Table 5. All dataset attributes.

Ranked	Attribute Name	Description
0	ifInOctets	Total number of octets received.
1	ipOutDiscards	IPv4 output datagrams that were rejected despite no issues hindering their delivery.
2	icmpOutDestUnreachs	The volume of messages sent to an ICMP destination that is unreachable.
3	ipInDiscards	Quantity of IPv4 datagrams received as input but deleted due to no issues found.
4	ifInDiscards	The number of incoming packets that were rejected despite there being no issues preventing their delivery to a higher layer.
5	ifoutDiscards	Dropping outgoing packets even when there were no problems to prevent them.
6	icmpOutMsgs	Total number of ICMP messages sent by this entity.
7	udpNoPorts	Total number of UDP datagrams received for destinations with no applications.
8	udpInErrors	Amount of UDP datagrams received but unable to be sent due to lack of applications at the destination.
9	ifOutUcastPkts	Total number of packets sent by higher-level protocols without being directed at a multicast or broadcast address.
10	ipInAddrErrors	The number of input datagrams that were rejected because their IPv4 address in their IPv4 header was incorrect and could not be received.
11	tcpEstabResets	TCP connections that have moved directly from an ESTABLISHED or CLOSE-WAIT state to a CLOSED state.
12	tcpInSegs	Segment count, including misdirected segments.
13	tcpOutSegs	Segments sent, excluding those solely retransmitted octets, including those on active connections.
14	tcpPassiveOpens	The number of times a TCP connection has moved directly into the SYN state.
15	ipForwDatagrams	The number of input datagrams that were intended for an IPv4 destination other than this entity, for which a route-finding effort was made.
16	ipInReceives	Datagrams received from interfaces, including any delivered incorrectly.
17	tcpActiveOpens	TCP connections that have moved directly from the CLOSED to the SYN-SENT state.
18	tcpRetransSegs	TCP segments that contain one or more previously sent octets or the number of segments sent repeatedly.
19	ifInUcastPkts	A measure of how many packets this sublayer sent to a higher (sub-)layer that were not addressed to a multicast or broadcast address at this sub-layer.
20	tcpOutRsts	RST-tagged TCP segments sent.
21	icmpInEchos	ICMP Echo (request) messages received.
22	icmpOutEchoReps	The number of ICMP Echo Reply messages sent.
23	icmpInMsgs	The total number of ICMP messages received by the entity.
24	ifOutNUcastPkts	Packets that higher-level protocols requested to be sent to multicast or broadcast addresses at this sublayer.
25	icmpInDestUnreachs	The number of ICMP Destination Unreachable messages that were received.
26	ipOutNoRoutes	In IPv4, the number of datagrams dropped due to a route not being determined.
27	ifOutOctets	Transmitted octets, including framing characters.
28	ifInNUcastPkts	Counts the number of packets sent to a higher layer that do not address a multicast or broadcast address.
29	udpInDatagrams	Count of UDP datagrams sent to UDP clients.
30	ipInDelivers	IPv4 input datagrams transmitted successfully (including ICMP).
31	udpOutDatagrams	The total no. of UDP datagrams sent by this object.
32	ipOutRequests	The total no. of IPv4 datagrams that local IPv4 user protocols, including ICMP, sent to IPv4 in requests for transmission.
33	tcpCurrEstab	The number of TCP connections whose status is either ESTABLISHED or CLOSE-WAIT at the moment.

The cross-validation (CV) method was used to evaluate the models. The K-fold method and the leave-one-out method were examined for cross-validation. An analysis of K-fold cross-validation was conducted using 80% of the data as the training set and 20% as the testing set without replacing 80% of the training data with 20% of the testing data. The original sample's observation was used as testing data for the leave-one-out cross-validation of the K-fold ($k = 5$) method, and the remaining observations were used as training data. As a result, every observation in the sample was used as testing data once. Table 6 shows the collected sample.

Supervised machine learning algorithms can best be understood through the lens of the bias-variance trade-off. Some popular examples of supervised machine learning algorithms are linear regression (LR) for regression problems, random forest (RF) for classification and regression problems, support vector machines (SVM) for classification problems, and k-nearest neighbors (KNN) for both regression and classification. Data are predicted into

discrete class labels through the classification process. Alternatively, regression creates a model that predicts continuous quantities. In this research, we investigated support vector machine (SVM), random forest (RF), k-nearest neighbors (KNN), and logistic regression (LR) classifiers for support DDoS attack detection.

Table 6. Collected sample.

Network Date (Traffic)	Total Number of Records
Complete set data	4998
Normal	1190
DDoS Attack	3808

Additionally, an artificial neural network-based approach called multilayer perceptron (MLP) has been investigated. These techniques could detect malicious activities and attacks, improve human analysis, and automate repetitive security tasks. The implementation was done using Python and related libraries such as Scikit-learn, Pandas, Numpy, TensorFlow, and Keras [33]. Before the classification task, the data were preprocessed. The redundant columns with a high correlation of 0.9 were removed. Missing values in the data were handled by mean imputation. The results show that the random forest (RF) algorithm has high accuracy in detecting DDoS attacks. Moreover, the performance using a multilayer perception (MLP) is generally ideal and very similar to RF. This work provides a robust and efficient approach to predict DDoS attacks from the dataset SNMP MIB. Collected sample records are shown in Table 6. The calculated values for sensitivity, specificity, accuracy, precision, recall and F1-measure for different classes with different algorithms are shown in Table 7.

Table 7. Detection evaluation results.

Method	Sensitivity%	Specificity%	Accuracy %	Precision %	Recall %	F1-Measure%
SVM	96.27	99.65	99.41	97.63	97.82	97.72
RF	99.92	99.84	99.94	99.87	99.83	99.85
KNN	98.71	99.82	99.73	98.71	98.75	98.73
LR	96.71	99.67	99.43	97.74	97.73	97.72
MLP	99.92	99.25	99.47	99.73	98.76	99.24
NB	98.80	95.14	96.96	95.86	95.82	95.65
DT	94.32%	94.05%	94.12%	94.42%	94.57%	94.09%

High sensitivity, specificity, and accuracy are all hallmarks of a good test. The results of our classification performance with traditional machine learning algorithms and multilayer perceptrons (MLP) and USML. We compare the model performance of machine learning algorithms (SVM, RF, KNN, LR) and the results obtained with the multilayer perceptron (MLP).

We compared several traditional machine learning algorithms using all the features from the SNMP-MIB dataset. SNMP-MIB is used to detect patterns of DDoS attacks. Machine learning algorithms, including support vector machine (SVM), random forest (RF), k-nearest neighbors (KNN) and logistic regression (LR), and an artificial neural network, multilayer perceptron (MLP), naive Bayes (NB) and decision tree (DT) are used to classify the dataset. Random forest (RF) is the best classifier with the highest accuracy for detecting DDoS attacks when traditional machine learning algorithms and MLP are used in the experimental analysis. Machine learning algorithms were evaluated based on sensitivity, specificity, accuracy, precision, recall and F1 score. The results presented in this section are based on the use of a 5-fold CV and hyper-parameter tuning with a grid search. RF has high accuracy in detecting DDoS attacks. For validation, these algorithms are used in a

binary classification test, and their performance is statistically measured and compared with the existing literature (Table 8) [9–12].

Table 8. Results compared to similar studies.

Dataset Used	An Algorithmic Approach to Machine Learning	Accuracy Percentage
[9] Dataset	SVM	98.5
	KNN	98.5
[10] KDD99 Dataset	SVM	91.55
	ANN	97.44
	USML	98.08
[10] UNBS-NB 15 Dataset	SVM	84.32
	ANN	63.97
	USML	94.78
[11] Dataset	SVM	92.11
	KNN	95.67
	ANN	91.07
	NB	94.48
[12] Dataset	SVM	98.52
Proposed System	SVM	99.41
	RF	99.94
	KNN	99.73
	LR	99.43
	MLP	99.47
	NB	96.96
	DT	94.12

According to the results, the RF algorithm proved to be very accurate in detecting DDoS attacks. The random forest (RF) algorithm is an ensemble algorithm that contains multiple decision tree algorithms. Moreover, the performance using multilayer perceptions (MLP) is generally ideal and very similar to RF. This work provides a robust and efficient approach for predicting DDoS attacks from the SNMP MIB dataset.

5.3. Intrusion Detection Schemes

The details of the experiment given in this section provide a clear overview of the response framework and its associated benefits. Key details of the experiment include log analysis, net flow analyzers, intrusion detection, and mitigation. We implement an intrusion detection system using Snort. A snort is an open-source software that can run in three different modes, i.e., packet capture, packet sniffer mode, packet logger, etc. In packet logger, these packets are written to the disk, or we can run it in intrusion detection mode, using the rule sets available in Snort and IDS (compares packets with rule base) [34]. Snort is on a network, so it listens for traffic coming over the network. Therefore, it is a network-based intrusion detection system. Generally, a network-based intrusion detection system is deployed at a single point of entry into a network. They use simple rules, which are signatures for detection. Snort rules, including malicious traffic, exploit, scan, FTP, telnet, DOS, DDOS, etc., are enabled in Snort. Snort rules are either site-specific policies or are required in most environments to avoid false positives.

The more rules that need to be matched, the slower the IDS, and the more packets are dropped. Snort has three main uses. It can be used as a pure packet sniffer such as

tcpdump, a packet logger used to debug network traffic. Snort logs packets in tcpdump in binary format and names them by their IP address. In packet capture mode, Snort received 142 packets, analyzed 70 packets (49.2%), and discarded 0 (0%), 63 UDP packets, 0 TCP packets, 2 ARP packets, and 6 fragmented packets. In packet logging mode, Snort analyzed 17 packets (47.2%), dropped 0 (0%), logged 17, and issued 0 alarms. In alert mode, Snort analyzed 4 out of 4 packets and discarded 0 (0%). In sniffer mode, Snort analyzed 14 packets and discarded 0 (0%) (Table 9). Figure 11 shows the analysis of packets in the different Snort modes. Network throughput increases with average packet arrival (packets/time slot) and maximum buffer size. This reflects the effectiveness of the rules applied to ensure that as many packets as possible successfully arrive at their destination.

Table 9. Packet analysis in different Snort modes.

Snort Mode	No. of Packets	Protocol by Breakdown									
		TCP	UDP	ICMP	ARP	EAPOL	IPv6	Ethloop	IPX	FRAG	Other
Packet Capture	71	0	63	0	2	0	0	0	0	6	0
	%	0	90	0	2.87	0	0	0	0	8.57	0
Logging Mode	17	0	17	0	0	0	0	0	0	0	0
	%	0	100	0	0	0	0	0	0	0	0
Alert mode	4	2	0	0	2	0	0	0	0	0	0
	%	50	0	0	50	0	0	0	0	0	0
Sniffer Mode	14	0	11	0	3	0	0	0	0	0	0
	%	0	78.5	0	21.5	0	0	0	0	0	0

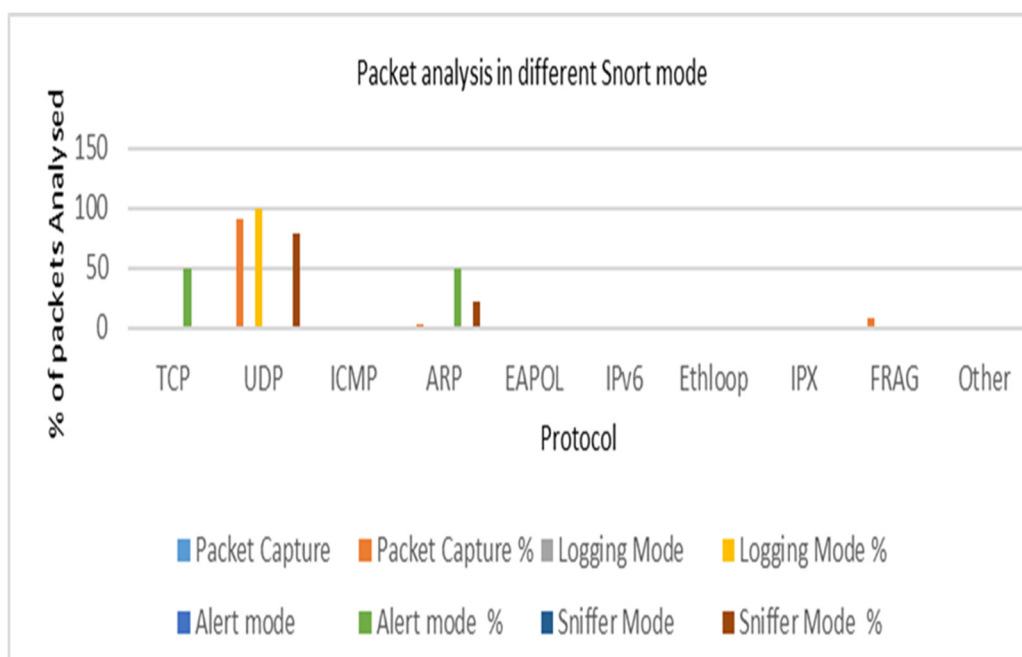


Figure 11. Packet analysis in different Snort modes.

The attack detection rate decreases with the number of packets or nodes, bandwidth consumption increases with the number of nodes, and throughput increases with the number of nodes. When Snort runs in packet logger mode and collects each packet, it arranges the packets in a directory. When running in IDS mode, it uses the rules available in the snort.conf file that specify suspicious network activity and sends an alert if the rules match the actual activity. Network traffic is analyzed using sFlow- RT. It is used for bandwidth analysis, network traffic analysis, and network performance monitoring. As seen in Figure 12, a higher peak indicates flood traffic from random IP addresses. Malicious traffic that saturated the victim was reduced after the network was trained. Lower peaks after 00.24.50 s indicate that mitigation was performed quickly and successfully.



Figure 12. Attack and mitigation flow IO graph.

5.4. Discussion

This research is conducted in a virtual environment where a virtual attack is carried out, and then it uses the situation to collect data and measure the extent of the benefits obtained based on criteria that the previous researchers did not analyze. Considering the relevant study and the findings from the empirical analysis, the most important finding is the weak protection of systems in many IoT devices. The Cooja tool revealed anomalies that could not have been observed without using the tool. Thus, it is an excellent tool for this type of testing. The tool was found to provide several auxiliary methods for the simultaneous comparison of the collected data. The theoretical significance of the results of this study lies in the fact that they will help identify a body of knowledge related to cybersecurity issues associated with networks and embedded subsystems of the IoT. Consequently, the results of this study should be of interest to future researchers studying IoT issues and how to appropriately address them.

The results of this study are relevant to vulnerability researchers and IoT network protection specialists because they guide how to avoid problems that may occur in real networks by first simulating them and then developing proactive solutions to them. In addition to avoiding short-term problems, there are also long-term solutions. It was stated that the IoT has become a material and moral part of our lives, and the weak protection in it may become a real threat to our lives, so it is necessary to search and investigate the areas of its security as much as possible after identifying a few in the relevant study related to the attack that was implemented, which is the Flooding attack, compared to other types of attacks. For the quantitative portion, the environment is configured to default in a preset scenario where M2M traffic monitoring using MQTT is attacked twice, once in normal mode and again after a default attack is run on it and is set up to measure the impact of malicious activity. The regulators and research experts use the findings of this study to detect vulnerabilities for IoT/embedded subsystems in a systematic manner so that the application process can be carried out successfully.

The performance of the prominent machine learning algorithm used in binary is evaluated based on sensitivity, specificity, accuracy, recall, precision, and F1-measure. In this study, we compared the model performance of machine learning algorithms (SVM, RF, ANN, LR) with results obtained with multi-layer perceptron (MLP). The results are based on the use of a five-way CV with a grid search and hyper-parameter tuning. The intrusion detection schemes implemented with Snort include protocol analysis, network flow analysis, intrusion detection, cyber-attack mitigation, and returning to normal.

Every study has its limitations, but this one has so far succeeded in identifying a flooding attack. However, it was utilizing a technology that had its limits, as while analyzing the virtual network, it took several minutes before the anomaly was discovered, which might have had major repercussions and losses if it had occurred. While simulation has been very helpful in gathering data and identifying abnormalities, there are still many other routes to explore to enhance this research and allow for an additional examination of pertinent papers.

6. Conclusions

This study proposes a practical framework and guidance for vulnerability investigation and discovery in IoT networks/embedded systems using Contiki OS and data extraction and analysis with Cooja. The analysis finds some attacks related to the implemented attack, namely flood attacks, compared to other types of attacks that identify the Internet of Things as an integral part of our lives, both material and moral. For the quantitative analysis, a predefined scenario is used in which the monitoring of M2M traffic over MQTT is exploited once in normal mode and once after a standard attack has been carried out to evaluate the impact of the malicious behavior. Machine-to-machine transmission in IoT uses a network analysis system that allows an authorized person to examine the network traffic. In IoT, the behavior of message traffic is monitored and recorded using the simulation method. As a result of the simulation, flood attack records of RPL triage attacks were obtained. This has implications for network status, brightness, speed, power, and battery consumption. A larger dataset was created by combining traffic, attack, and normal network data. This created a new dataset that can be used to detect and explore vulnerabilities in IoT devices based on intrusion detection systems (IDS). Flooding attack detection was the main objective of this research. However, a tool with limitations was used because, in the case of virtual network analysis, it took several minutes to detect the anomaly, which could lead to serious losses. In further studies, the dataset will be extended to include attack periods, the packet and increment types of attacks, and the creation of characteristics of the recorded data.

This study also provides DDoS detection in a network with supervised machine learning. Several traditional machine learning algorithms have been implemented, all using features from the SNMP MIB dataset. The results show that the random forest (RF) algorithm has high (99.94%) accuracy in detecting DDoS attacks. Moreover, the performance using multilayer perception (MLP) is generally ideal and very similar to RF. The intrusion detection schemes implemented with Snort include protocol analysis, network flow analysis, intrusion detection, cyber-attack mitigation, and returning to normal. The datasets for traffic, attacks, and typical network activity were combined to create a more extensive dataset. As a result, a fresh dataset was amassed that might serve as a source for learning and vulnerability detection based on IoT device assault intrusion detection systems (IDS). Attack times, packet and increment types of attacks, and the creation of recorded data aspects can all be added to the dataset in future research to make it richer. sFlow-RT helps in examining the actual traffic on the network; it identifies any intrusion. Future work will focus on Blockchain-based security and cyber threat cognitive intelligence to provide accurate, easy-to-use, and actionable common vulnerabilities and exposures (CVE) information.

Author Contributions: Conceptualization, S.M. methodology, S.M. software, A.A.; validation, A.A.; formal analysis, S.K.S.; investigation, S.M.; resources, S.M.; data curation, S.M.; writing—original draft preparation, S.M.; writing—review and editing, S.M.; visualization, S.M. supervision, S.K.S.; project administration, A.A.; funding acquisition, S.M. Author have read and agreed to the published version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: Sunil Kumar Sharma extends their appreciation to the deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through project number (IFP-2020-95).

Data Availability Statement: Not applicable.

Acknowledgments: The authors sincerely acknowledge the support from Majmaah University, Saudi Arabia, for this research.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

References

1. Fersi, G. Fog computing and Internet of Things in one building block: A survey and an overview of interacting technologies. *Clust. Comput.* **2021**, *24*, 2757–2787. [\[CrossRef\]](#)
2. Nehme, E.; Sibai, R.; Bou, A.; Taylor, A.R. Demerjian. Converged AI, IoT, and blockchain technologies: A conceptual ethics framework. *AI Ethics* **2021**, *2*, 1–15.
3. Torres, N.; Pinto, P.; Lopes, S.I. Security vulnerabilities in LPWANs—An attack vector analysis for the IoT ecosystem. *Appl. Sci.* **2021**, *11*, 3176. [\[CrossRef\]](#)
4. Arfi, W.B.; Nasr, I.B.; Khvatova, T.; Zaied, Y.B. Understanding acceptance of eHealthcare by IoT natives and IoT immigrants: An integrated model of UTAUT, perceived risk, and financial cost. *Technol. Forecast. Soc. Chang.* **2021**, *163*, 120437. [\[CrossRef\]](#)
5. Zhou, W.; Jia, Y.; Peng, A.; Zhang, Y.; Liu, P. The effect of IoT new features on security and privacy: New threats, existing solutions, and challenges yet to be solved. *IEEE Internet Things J.* **2018**, *6*, 1606–1616. [\[CrossRef\]](#)
6. Patel, C.; Doshi, N. A novel MQTT security framework in a generic IoT model. *Procedia Comput. Sci.* **2020**, *171*, 1399–1408. [\[CrossRef\]](#)
7. Rango, F.D.; Potrino, G.; Tropea, M.; Fazio, P. Energy-aware dynamic Internet of Things security system based on Elliptic Curve Cryptography and Message Queue Telemetry Transport protocol for mitigating Replay attacks. *Pervasive Mob. Comput.* **2020**, *61*, 101105. [\[CrossRef\]](#)
8. Cakir, S.; Toklu, S.; Yalcin, N. RPL attack detection and prevention in the Internet of Things networks using a GRU based deep learning. *IEEE Access* **2020**, *8*, 183678–183689. [\[CrossRef\]](#)
9. Mishra, S. Network Traffic Analysis Using Machine Learning Techniques in IoT Networks. *Int. J. Softw. Innov.* **2021**, *9*, 1–17. [\[CrossRef\]](#)
10. Tuan, T.A.; Long, H.V.; Kumar, R.; Priyadarshini, I.; Son, N.T.K. Performance evaluation of botnet DDoS attack detection using machine learning. *Evol. Intell.* **2019**, *13*, 1–12. [\[CrossRef\]](#)
11. Polat, H.; Polat, O.; Cetin, A. Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models. *Sustainability* **2020**, *12*, 1035. [\[CrossRef\]](#)
12. Mishra, S.; Sharma, S.K.; Alowaidi, M.A. Multilayer self-defense system to protect enterprise cloud. *Comput. Mater. Contin.* **2021**, *66*, 71–85. [\[CrossRef\]](#)
13. Al-kasassbeh, M.; Al-Naymat, G.; Al-Hawari, E. Towards generating realistic SNMP-MIB dataset for network anomaly detection. *Int. J. Comput. Sci. Inf. Secur.* **2016**, *14*, 1162.
14. Servida, F.; Casey, E. IoT forensic challenges and opportunities for digital traces. *Digit. Investig.* **2019**, *28*, 22–29. [\[CrossRef\]](#)
15. Ali, B.; Awad, A.I. Cyber and physical security vulnerability assessment for IoT-based smart homes. *Sensors* **2018**, *18*, 817. [\[CrossRef\]](#)
16. Cui, L.; Xie, G.; Qu, Y.; Gao, L.; Yang, Y. Security and privacy in smart cities: Challenges and opportunities. *IEEE Access* **2018**, *6*, 46134–46145. [\[CrossRef\]](#)
17. Li, Y.; Orgerie, A.C.; Rodero, I.; Amersho, B.L.; Parashar, M. End-to-end energy models for Edge Cloud-based IoT platforms: Application to data stream analysis in IoT. *Future Gener. Comput. Syst.* **2018**, *87*, 667–678. [\[CrossRef\]](#)
18. Behrad, S.; Bertin, E.; Tuffin, S.; Crespi, N. A new scalable authentication and access control mechanism for 5G-based IoT. *Future Gener. Comput. Syst.* **2020**, *108*, 46–61. [\[CrossRef\]](#)
19. Alshunaifi, S.Y.; Mishra, S.; AlShehri, M. Cyber-Attack Detection and Mitigation Using SVM for 5G Network. *Intell. Autom. Soft Comput.* **2022**, *31*, 13–28. [\[CrossRef\]](#)
20. Dinculeana, D.; Cheng, X. Vulnerabilities and limitations of MQTT protocol used between IoT devices. *Appl. Sci.* **2019**, *5*, 848. [\[CrossRef\]](#)
21. Bhosale, S.D.; Sonavane, S.S. A real-time intrusion detection system for wormhole attack in the RPL-based Internet of Things. *Procedia Manuf.* **2019**, *32*, 840–847. [\[CrossRef\]](#)
22. Zhuang, R.; Bardas, A.G.; DeLoach, S.A.; Ou, X. A theory of cyber attacks: A step towards analyzing MTD systems. In Proceedings of the Second ACM Workshop on Moving Target Defense, Denver, CO, USA, 12 October 2015; pp. 11–20.
23. Yu, J.; Lee, H.; Kim, M.S.; Park, D. Traffic flooding attack detection with SNMP MIB using SVM. *Comput. Commun.* **2008**, *31*, 4212–4219. [\[CrossRef\]](#)
24. Al-Naymat, G.; Al-Kasassbeh, M.; Al-Harwari, E. Using machine learning methods for detecting network anomalies within SNMP-MIB dataset. *Int. J. Wirel. Mob. Comput.* **2018**, *15*, 67–76. [\[CrossRef\]](#)
25. Hue, A.; Sharma, G.; Dricot, M.J. Privacy-Enhanced MQTT Protocol for Massive IoT. *Electronics* **2021**, *11*, 70. [\[CrossRef\]](#)
26. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.

27. Aledhari, M.; Razzak, R.; Parizi, R.M. Machine learning for network application security: Empirical evaluation and optimization. *Comput. Electr. Eng.* **2021**, *91*, 107052. [[CrossRef](#)]
28. Xu, G.; Qian, Y.; Hu, R.Q. Data-driven network intelligence for anomaly detection. *IEEE Netw.* **2019**, *33*, 88–95. [[CrossRef](#)]
29. Ibor, A.E.; Oladeji, F.A.; Okunoye, O.B.; Uwadia, C.O. Novel adaptive cyber-attack prediction model using an enhanced genetic algorithm and deep learning (AdacDeep). *Inf. Secur. J. A Glob. Perspect.* **2021**, *31*, 1–20.
30. Ghorri, K.M.; Imran, M.; Nawaz, A.; Abbasi, R.A.; Ullah, A.; Szathmary, A.I.L. Performance analysis of machine learning classifiers for non-technical loss detection. *J. Ambient. Intell. Humaniz. Comput.* **2020**, 1–16. [[CrossRef](#)]
31. Tsai, R.G.; Tsai, P.H.; Shih, G.R.; Tu, J. RPL Based Emergency Routing Protocol for Smart Buildings. *IEEE Access* **2022**, *10*, 18445–18455. [[CrossRef](#)]
32. Sahay, R.; Geethakumari, G.; Mitra, B. Mitigating the worst parent attack in RPL based internet of things. *Clust. Comput.* **2022**, *25*, 1303–1320. [[CrossRef](#)]
33. Yuan, B.; Wang, J.; Liu, D.; Guo, W.; Wu, P.; Bao, X. Byte-level malware classification based on markov images and deep learning. *Comput. Secur.* **2020**, *92*, 101740. [[CrossRef](#)]
34. Nykvist, C.; Larsson, M.; Sodhro, A.H.; Gurtov, A. A lightweight portable intrusion detection communication system for auditing applications. *Int. J. Commun. Syst.* **2020**, *33*, e4327. [[CrossRef](#)]