Application Layer-Based Denial-of-Service Attacks Detection Against IoT-CoAP

Sultan Almeghlef *, Abdullah S. AL-Malaise AL-Ghamdi, Muhammed sheer Ramzan, Mahmoud Ragab

Posted Date: 2 May 2023

doi: 10.20944/preprints202305.0070.v1

Keywords: Denial of Service; IoT attacks; CoAP Security; Application layer; DTLS

Article

Application Layer-Based Denial-of-Service Attacks Detection Against IoT-CoAP

Sultan Almeghlef [1,*], Abdullah AL-Malaise AL-Ghamdi [1], Muhammad Sher Ramzan [1] and Mahmoud Ragab [2]

[1] Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; salmeghlef@stu.kau.edu.sa; aalmalaise@kau.edu.sa; msramadan@kau.edu.sa

[2] Information Technology Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; mragab@kau.edu.sa

* Correspondence: salmeghlef@stu.kau.edu.sa

Abstract: Internet of Things (IoT) is a massive network of tiny devices connected internally and to the internet. It is uniquely identified in the network (i.e. dedicated IP) and can share the information with other devices. However, the low power and low resources that distinguish IoT devices render them unsecure and targeted by different kinds of attacks since IoT devices cannot tolerate heavy security models. Also, due to the heavy nature of famous protocols such as HyperText Transport Protocol (HTTP), it is costly to be used with IoT devices, and alternatively, different lightweight protocols are implemented to fit IoT devices. One of the prevailing protocols used over IoT networks is the Constrained Application Protocol (CoAP). Therefore, CoAP is popular, and that makes it targeted by different types of attacks. One of the major attacks that target CoAP is distributed denial of service (DDoS) attacks. DDoS aims to overwhelm the resources of the target and make them unavailable to legitimate users. As a result, different kinds of methods were used to secure CoAP against DDoS attacks such as Datagram Transport Layer Security (DTLS) and Lightweight and Secure Protocol for Wireless Sensor Networks (LSPWSN). However, the existing models suffer from two issues: DTLS is not designed for constrained devices and is considered a heavy protocol. Besides, LSPWSN is working over the network layer, not in the application layer that CoAP works on. In this paper, we build a machine learning model that can detect the DDoS attacks against CoAP with an accuracy of 98%. The CIDAD dataset is extended from ~11000 to 100,000 samples using GANs because it has fewer samples of malware (less than 0.2% of the total dataset). Our model outperforms the existing models that target securing CoAP in the application layer and obtains 93% of accuracy.

Keywords: denial of service; IoT attacks; CoAP security; application layer; DTLS

1. Introduction

IoT is a massive network that interconnects low power and low resources devices rendering them connected or connected to the Internet [1]. These devices are capable of exchanging data with each other without human intervention. The growth of number of IoT devices that will emerge by 2025 is approximately 75 billion [2]. Therefore, IoT will lead the development of a smarter world for the future decades in different fields such as smart homes, smart industries, and smart healthcare [3]. IoT relies on different protocols to exchange information between the devices and the internet. IoT is composed of three layers namely, the perception layer which includes sensors that gathers data from the environment, the network layer which receives data from sensors and processes it accordingly, and finally, the application layer which receives information from the network layer [4]. In terms of communications protocols, there are different protocols in the application layer which include CoAP, Message Queuing Telemetry Transport (MQTT), and Advanced Message Queuing Protocol (AMQP). Indeed, CoAP prevails among other protocols due to its lightness and it fits lower power and lower resource devices, thus, it can be secured using DTLS [5]. CoAP works on the REST model and resources can be addressed from the server and accessed by the clients with the help of the standard

 doi:10.20944/preprints202305.0070.v1

HTTP methods such as (GET, PUT, POST, and DELETE). CoAP prevails over other application protocols due to its simplicity to developers, lightweight in terms of power consumption and communication, mobility, and portability [6]. Therefore, CoAP is targeted by DDoS attacks. The DDoS attack is a collection of infected devices (Zombies) that are controlled by the attacker, and once the attacker aims to launch the attack, he or she commands and controls those Zombies, asking them to overwhelm the victim with a high volume of traffics which results in consuming the victim's resources and make it unavailable to legitimate users. Therefore, securing CoAP against DDoS attacks is significant. DTLS is proposed to secure CoAP, however, DTLS suffers from the communication overhead because it performs six messages for the handshake process which results in consuming the constrained device energy. Moreover, DTLS is not designed for constrained devices [6]. Likewise, LSPWSN is used to secure messages of CoAP, but this system is working in the network layer while CoAP is working in the application layer. Motivated by the assumption that detection of DDoS attacks is more beneficial to be near the victim whereas mitigation is more effective to be near the attacker [7], this works aims to propose a method to detect DDoS attacks against CoAP in the application layer. This study uses CIDAD dataset that contains DDoS attacks (interception, modification, and duplication of CoAP messages). The dataset has ~11000 samples where the malware samples are only 288. As a result, we extend the dataset to 100,000 samples with ~50% for each category (benign and malware) using Generative Adversarial Networks (GANs). In addition, four different ML classification models namely Naïve Bayes, Random Forest, SVC, and Decision Tree since these ML methods showed impressive results in classifying IoT attacks [8]. The proposed model gains an accuracy of 98% with the Decision Tree algorithm which outperform other algorithms. The main contributions of this work are as follows:

1 Extending and balancing the CIDAD dataset using GANs.
2 Focusing on the CoAP level features to ensure securing CoAP in its vicinity.
3 Build a machine learning model that can classify the benign against malware with an accuracy of 98% using the decision tree algorithm.

1.1. IoT Overview

The Internet of Things inspires the existence of tiny object devices so they can send and receive data with little users' intervention [9]. IoT is defined as extending the network connectivity and computing capability to tiny objects such as temperature sensors and blood pressure monitors to allow them to operate independently. The application of the IoT will involve the military, industry, transportation, and even our daily life activities. The significant and powerful data analytic capabilities combined with the revolutionary of IoT devices are promising to change the way we live and behave. As per the anticipation of some research, 75 billion IoT devices will take place in the communication technology environment by 2025 [2]. IoT consists of three layers (as depicted in Figure 1): the perception layer that represents the physical sensors and actuators, the network layer that enables device-to-device and device-to-cloud communication, and the application layer that delivers the services to other devices or humans [2]. IoT architecture can be extended to have two more layers, namely, MAC and adaptation layer resulting in what the so-called five-layer architecture [10]. To connect IoT devices to other devices or the internet, several communication models are proposed to connect IoT devices to another device, to the cloud, and to a gateway. Due to the heterogeneity aspect of IoT devices that will interact with other devices, and different IoT protocols are developed to enable the interactions between IoT devices. Consequently, the Institute of Electrical and Electronics Engineers (IEEE) and the Internet Engineering Task Force (IETF) are developing standardized protocols. Figure 2 depicts the layers and protocols deployed for each layer based on the five-layer architecture of the IoT network. After reviewing the IoT architecture, the protocols related to the IoT will be discussed with more focus on the CoAP protocol. CoAP prevails among other protocols in the application layer because it is a lightweight protocol and fits for constrained devices [5]. However, CoAP and many other protocols are susceptible to DDoS attacks. DDoS attacks target different layers and many protocols rendering the service unavailable for legitimate users. In

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

the next sections, IoT protocols and CoAP architecture will be reviewed, then DDoS attacks against IoT and CoAP specifically will be investigated.

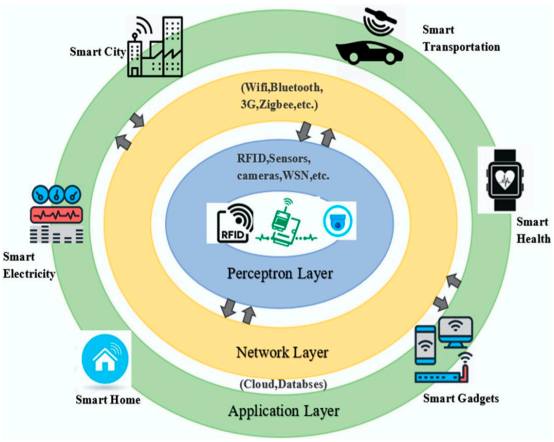


Figure 1. Layer Architecture of IoT network. Adopted from [9].

Application	• CoAP
Network	• RPL
Adaptation	• 6LoWPAN
MAC	• IEEE 802.15.4
Physical	• IEEE 802.15.4

Figure 2. Layers and protocols of IoT network.

1.2. IoT Protocols

1 IEEE 802.15.4 Protocol

Both physical and MAC layers are ruled by IEEE 802.15.4 protocol which emerged in 2003. The physical layer provides some functionalities such as transferring data, detecting the energy of the channel, and indicating the link quality [11], whereas the MAC layer associates, disassociates, acknowledges frame arrival, and validates frame.

2 6LoWPAN protocol

Lower-Power Wireless Personal Area Network 6LoWPAN has emerged in 2007 due to the demand for low-energy IoT protocol [10]. It allows a direct connection to the internet and defines encapsulation and header compression mechanisms. 6LoWPAN is considered a replacement for IPv6. This protocol supports addresses with different lengths, low bandwidth, and costs.

3 Routing – RPL protocol

IETF proposed Routing Protocol Layer (RPL) for Low Power and Lossy Networks (LLNs) that provides IPv6 connectivity to the LLNs [12]. This protocol is used in multiple networking facilities such as automated homes, automated industry, and automated buildings.

4 CoAP Protocol

The Constrained Application Protocol (CoAP) is a proper web transfer protocol for lower resources devices and LLNs [13]. The nodes often have 8-bit microcontrollers and limited RAM and ROM. The protocol supports M2M applications like an automated smart home. Due to the demand for a proper design of a generic web protocol that fits constrained devices, CoAP has emerged.

1.2.1. CoAP Architecture

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

CoAP relies on the client/server model like HTTP with the usage of the request-response model for exchanging messages. The communication between two machines inspires a CoAP to interfere and acts as a client or as a server. A CoAP request is similar to an HTTP request which asks for a resource on a server. The resource is identified by URI. Thus, a response code from the server is sent back with the representation of the resource. So, CoAP architecture includes the message layer and the request-response layer as the main layers. The former is responsible for message delivery over UDP for two nodes and it supports optional reliability. As a result, it is in charge of the redundancy and consistency of any message. The latter avoids having outdated messages or having more copies of a message by holding the code of the requesting and responding.

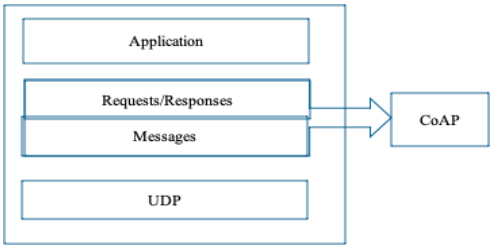


Figure 3. Layers of the CoAP Protocol.

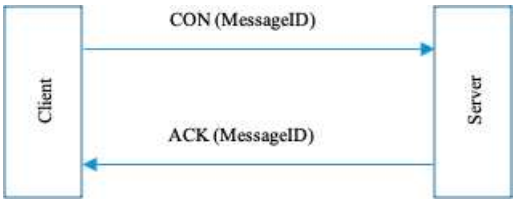
1.2.2. Messaging Model

Messaging model of CoAP is based on sending and receiving messages over UDP for two nodes. CoAP has a 4-byte header, then it has options and payload. Each message has a message ID for duplication check and a reliable connection if desired. Four types of messages are supported by CoAP as follows:

1 Confirmable Message (CON): in this mode, all messages are marked as confirmable messages (reliable mode). The message is resent using a default timeout till receiving an Acknowledgement from the recipient with the same messageID as the sender. If the recipient failed to process the confirmable message, the recipient will send a reset message (RST) instead of (ACK) to reset the communication. Figure 4 shows the confirmable mode between the client and the server.

Figure 4. Confirmable message transmission.

2 Non-Confirmable Message (NON): if reliable delivery is not desired, the message can be sent as a non-confirmable message (unreliable mode). For duplication check purposes, the message is not acknowledged but still has a messageID. Besides, the recipient could send a reset message if it failed to respond to the NON-message. Figure5 depicts the NON-message between the client and the server.



Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

Figure 5. NON-Confirmable message transmission.

1.2.3. Request/Response Model

Request and response semantics which include method/response code is carried by the CoAP message. Some information about the request and response are considered optional such as the URI and payload media type. To check the identity of requests and responses, a token is used to connect responses with their opposite requests independently. Conceptually, Token differs from messageID. There are three types of messages in the request-response model as follows:

Piggybacked message: CON or NON-message carries a request and if instantly enforced, the response carried in a CON message is carried in the resulting ACK message. Figure 6 shows two examples of a basic GET request with a piggyback response, one for success and the other resulting in a 4.04 (Not Found) response. Hence, the code [0x00] is the message ID.

Empty Message: if the server is not able to respond immediately to a request carried on the CON message, an empty Acknowledgement.

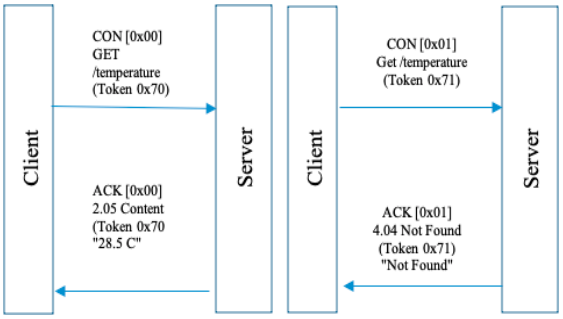


Figure 6. Two GET Requests with Piggybacked Responses.

The message is used to respond to the request so the client can stop retransmitting the request. If the response is ready, the server sends it in a new confirmable message as depicted in Figure 7.

Non-Confirmable Message: if a non-Confirmable mode is used to send a message, then either a new Non-Confirmable or a Confirmable message is used to send the response as illustrated in Figure 8.

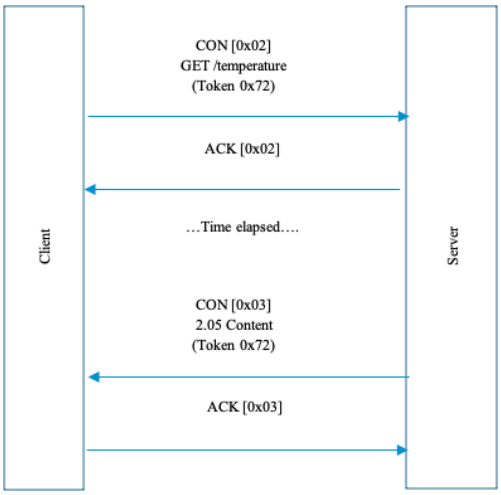


Figure 7. A GET request With separate responses.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

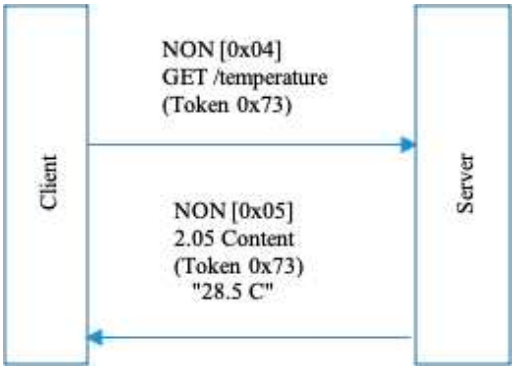


Figure 8. A Non-confirmable message carries Request and Response**.**

1.2.4. Message Format

CoAP sends/receives messages and employs UDP for transportation. The encoding of CoAP messages is a simple binary format. A fixed-size 4-byte format appears in the header of the message, then it is followed by a Token value that is 0-8 bytes long. After the Token value, CoAP Options in Type-Length-Value (TLV) format appear, or a sequence of zeros for non-option is displayed. Finally, an optional payload appears that takes up the rest of the datagram. **Figure** 9 depicts the CoAP message format. The header fields can be elaborated as follows:

(1) Version (V): unsigned integer (2-bit) that represents the CoAP version number. This field takes (01 binary), and other values are reserved for future versions. If the message comes with unknown version numbers, it must be ignored.
(2) Type (T): unsigned integer (2-bit) that represents 0 for Confirmable, 1 for Non-Confirmable, 2 for Acknowledgement, or 3 for reset as illustrated in the previous section.
(3) Token Length (TKL): unsigned integer (4-bit) with a length of 0 to 8 bytes. Lengths 9-15 are specialized for message format errors.
(4) Code: unsigned integer (8-bit) that is divided into the most significant bits (3-bit) and the least significant bit (5-bit). It is represented as "c.dd" ("c" can be 0-7 as a digit for the 3-bit, and "dd" can be two digits in the range from 00 to 31 for the 5-bit). The most significant bits view 0 for a request, 2 for a successful response, 4 for a client error response, or 5 for a server error response. The other most significant bits are reserved. The code 0.00 represents an Empty message as a special case.
(5) MessageID: unsigned integer (16-bit) used for duplicate vetting purposes. It is also used to match Acknowledgement/Reset messages to messages of type Confirmable or Non-Confirmable, respectively.
(6) Token: maybe 0 to 8 bytes based on the length stated in the TKL field. It is used to correlate requests and responses.
(7) Options: can be 0, by another option, or by the payload.
(8) Payload: if exists, it is prefixed by a (0xFF) marker as a benchmark for payload start. To calculate the length of the payload, it is counted from the end of the marker until the UDP datagram end.

Version (V)	Type (T)	Token Length (TKL)	Code	Message ID
Token (if any)				
Options (if any)				
Payload (if any)				

Figure 9. Message header of CoAP**.**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

1.2.5. Method Definitions

Like HTTP, CoAP uses methods (GET, POST, PUT and DELETE) to take any action on a URI resource. CoAP message follows RESTful architecture that makes it fit for constrained devices as a lightweight protocol. A request that carries fault method code results in a 4.05 (Unallowed method) piggyback response. We will briefly elaborate on each method.

GET

The GET method retrieves the information's representation that belongs to the resource identified by the request URI. If the GET method succeeds, a 2.05 (Content) is presented, or a 2.03 (Valid) response code appears.

POST

The POST method functionality is to process the representation retrieved by the request. The origin server performs the main functionality of the POST method depending on the target resource. The result of this action is a creation of a new resource, or an updating process is performed on the target resource. In the case of the creation of a new resource, the response should have a 2.01 (Created) code with the corresponding URI of the created source. However, if POST is processed but the creation of a new source on the server fails, so a 2.04 (Changed) response code is generated. If POST is processed and results in a deletion of a resource, the response should have a 2.02 (Deleted) response code.

PUT

The PUT method functionality is confined to creating or updating the resource identified by the request URI. The enclosed representation format is specified by the media type and content coding provided in the Content-Format option (if it is given). In the case of existing resources at the request URI, the enclosed representation is a modification copy, and a 2.04 (Changed) response code is issued. Otherwise, a new resource should be created by the server and aligned to that URI, and then, a 2.01 (Created) response code is issued. In case of modifying or creating failure, then the error response code is returned.

DELETE

The DELETE method requests to drop the resource that the request URI identifies. If the deletion is a success, a 2.02 (Deleted) is issued. The same message is returned if the resource did not show up before the request.

1.2.6. CoAP URIs

The CoAP uses "coap" and "coaps" URI schemes to identify and locate the resources. A CoAP server hierarchically organizes and governs these resources, and it waits to receive CoAP requests on a specified UDP port number. So, the CoAP methods discussed above are used to access the resources on the CoAP server. Therefore, the "coap" and "coaps" URI schemes are similar to that of "HTTP" and "HTTPS" used with the HTTP protocol.

2. Literature Review

It is a complicated task to secure IoT from a variety of possible attacks [1]. Every layer in IoT architecture namely (Application layer, Network layer, and Infrastructure layer) is susceptible to different kinds of attacks. These attacks need to be identified and then develop security models to beat these attacks and ensure a secure IoT environment. Indeed, attacks show up due to some vulnerabilities present in the IoT environment. A vulnerability is incompetency that the attacker will exploit to penetrate the network. These vulnerabilities will result in threats to the IoT network if

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

ignored, and consequently, it may lead to an attack. This work focuses on vulnerabilities related to DDoS attacks, so other vulnerabilities are out of the scope of this work.

2.1. DDoS attacks in IoT

Due to the huge amount of data flooding into the network, it may result in overwhelming the bandwidth, so the server will deny serving any request and will be inaccessible. This overflow of data is called a DDoS attack in which the legitimate user cannot access the server. Indeed, IoT which is considered a revolution in technology now can be a bane to it because IoT can contribute to DDoS attacks by adopting them negatively to create Botnets [9]. Botnets are defined as the process of infecting a massive number of IoT devices with malware to compromise these devices and make them under the control of an attacker who can trigger these devices to start an attack by dictating them to send massive traffics to the victim and consume the resources of that victim. Consequently, the infected IoT devices are called Bots. Normal IoT devices may be enforced to behave like a bot without the user's awareness. These bots are slaves and controlled by Master Bot Controller. This leads to making these botnets for sale and misuse nowadays. The Master Controller can attain some profits by selling their attack services. As a result, several variants of botnets appear in the market recently. Next, we mention three of the discovered IoT botnets.

A. Mirai

Mirai means the future in Japanese. It injected over 500.000 non-secured IoT devices, thus generating massive traffics to a target server with a flooding speed of 1 Tbps (Terabyte per second). The popularity of Mirai is gained from its irrefutable impact in 2016. It is designed to target Linux-running systems and some IoT devices, resulting in adding them as bots for the malware. Mirai targeted the famous Dyn DNS (French hosting provider) service in 2016, causing the internet inaccessible for major websites such as Twitter, Amazon, and Netflix. After that, IoT devices are exploited by different variants of Mirai and consequently, more DDoS attacks are launched. It can be inferred that using IoT devices as botnets is one of the concerns in network security recently.

B. Wirex

Wirex is a botnet used to trigger a DDoS attack on multiple Content Delivery Networks (CDNs) and content providers. Wirex had infected an enormous number of Android devices having applications that behave as benign applications but were malware. Consequently, Google has banned some applications from the Play Store.

C. Reaper

This botnet is stronger than Mirai since the latter can penetrate devices with default credentials. This malware created a huge number of bots including Cisco routers and other brands by brutely adding them to its botnet. Reaper can exploit other vulnerabilities in IoT devices.

D. Torri

Torri is considered a new botnet nowadays [14]. It can target most of today's recent computers, tablets, and smartphones due to its architectural design that has similar (64-bit), x86, etc.

Taxonomy of DDoS attacks in IoT network

In this section, we will consider the major IoT layers (application layer and infrastructure layer) and review the DDoS attacks that target these layers.

A- Network Layer Attacks: application layer of the IoT network is targeted by different DDoS attacks where the flooding packets seem at the same rate of request (HTTP Get/Post). These attacks are not easy to detect due to the low-rate generation of traffic that seems to be legitimate, but they are behaving maliciously. HTTP flood and DNS service-based attacks are examples of this kind of attack.

B- Infrastructure layer attacks: transport layer or network layer is targeted by different DDoS attacks due to some vulnerabilities rendering them out of service for legitimate users. There are two

 doi:10.20944/preprints202305.0070.v1

types of these attacks: protocol-based and volume-based attacks. These attacks often use reflection and amplification techniques to trigger the attack. In the reflection technique, the attacker tries to make congestion in the victim network by using a spoofed IP address to flood the victim with an unrequested reply reflected from sent requests. Similarly, amplification generates huge replies from small requests which result in consuming the bandwidth of the network. The attacks targeting protocols consume the actual server resources including intermediate communications such as firewalls, Intrusion Detection Systems (IDS), and load balancers. SYN floods, Ping of Death, and smurf are considered samples of these attacks. Volume-based attacks generate massive traffic to the target system causing bandwidth overwhelming. These attacks use both reflection and amplification techniques. UDP/TCP floods and ICMP floods are examples of these kinds of attacks.

2.2. Proposed defense mechanisms for IoT Network

Several works are proposed for inventing defense mechanisms to protect IoT networks from DDoS attacks. All such defense mechanisms for defending IoT against DDoS attacks can be categorized into Software-Defined Networking (SDN) methods, and machine learning-based detection. In this section, several proposed mechanisms will be discussed in detail.

2.2.1. SDN Platform for IoT network security

Some researchers have deployed SDN as a platform to secure IoT networks. SDN is a new networking paradigm, introducing different benefits such as ubiquitous accessibility, managing resources dynamically, flexible programmed interfaces, and extreme authority for the controller which introduces the second generation of networking [15]. SDN consists of a separate control plane and a data plane. The basic operations and control of the network are centralized in the controller or distributed controllers. So, SDN design has enhanced network security because all major network operations are visible to the controllers, thus controllers have the authority to resolve any conflict. Mert Ozcelik et al., (2017) proposed to exploit the extreme functionalities and authorities of the SDN controller in the edge IoT networks such as traffic monitoring and dynamic rule updates to detect and mitigate IoT-based DDoS attacks [16]. Their system builds rules in the switches based on the discovered attack packets to blacklist and deletes the attacks. The authors tested their system in a real-time environment and the results indicate extreme effort in detecting Mirai attacks. Similarly, Yin et al., (2018) proposed a framework for IoT-based SDN that has distributed controller pool [17]. The system named SD-IoT uses cosine similarity of the vectors of the incoming packet and based on that it decides if a DDoS attack is triggered or not. This work employs a threshold-based cosine similarity, if the threshold is exceeded, the system blocks the source of the attack. However, this work is susceptible to large amounts of traffic. Edge computing is also employed to defend IoT against DDoS attacks. Bhardwaj et al., (2018) developed a proactive defense while rendering the edge as the first line to counter DDoS attacks [7]. It is a cloud-based platform that uses ShadowNet employed at the edge nodes of the cloud architecture. Edge nodes will send the incoming packets for check purposes to the ShadowNet web service to assess the packet whether is a benign or a DDoS attack packet. However, this approach is focusing on speed but ignores accuracy because no way to differentiate between the attack and the Flash crowd. Stateful SDN architecture is used to develop an entropy-based solution to detect and mitigate IoT-DDoS attacks (Galeano-Brajones et al., 2020) [18]. The authors claim their results demonstrate the significant role of calculating the correlation of the entropy values for different extracted features in identifying the attacks. Besides, SDN is employed for mitigation by easily updating the switches' flow tables with new entries. However, this system considers only limited types of DDoS attacks. Yang et al., (2019) claim that using the controller to defend the network against DDoS attacks is time-consuming and a waste of resources [19]. Alternatively, they consider the IoT traffic features and exploit the edge computing concept benefits to put the detection and mitigation system into the Open Flow (OF) switches of IoT. This results in a distribution of anomaly detection and avoids overloading the controller. They employ machine learning in the OF switches and gain around 99% precision. The demerit of this work is the lack of handling sophisticated DDoS attacks such as the Crossfire attack.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

2.2.2. Machine Learning and Deep Learning for IoT network security

Several works employed machine learning or deep learning methods to detect and mitigate IoT-DDoS attacks. Median Y. et al., (2018) developed an anomaly-based detection method for the IoT network [20]. The authors extract the behavior for every benign traffic independently and use the deep autoencoders to capture the benign behavior of the IoT traffic. Then, the autoencoder attempts to compress snapshots and if it fails to reconstruct the snapshot, this is an indication of anomalous behavior. They evaluate their method by injecting some IoT devices in the lab with popular malicious botnets (Mirai and BASHTILE) and demonstrate that their method can accurately and instantly detect the attacks launched from infected IoT devices. Ivan et al., (2019) proposed an anomaly-based DDoS attacks detection framework [21]. They classify the IoT traffic based on the belonging class. These classes are categorizing the IoT devices, and the traffic generated from a given class should not deviate from other traffics generated from the same class. To classify new IoT devices, Logistic Regression is employed to affiliate it to a dedicated class. Then, the Adaboost machine learning algorithm is used to measure the deviation of any traffic from normal behavior. Ultimately, any traffic behavior that deviates from the corresponding class will be considered a DDoS attack. Hussain et al., (2019) claim that recent IoT detection models are degraded due to model aging and the outdated dataset used for training [3]. In their work, they convert the network traffic into image form, then they train the state-of-the-art Convolutional Neural Network (CNN) model named ResNet with the new form. The authors evaluate their work using the CICD2019 dataset which contains 11 types of DDoS attacks and gains an accuracy of 99.99% for the binary classification technique. A honeypot-based detection is also used to detect Zero-day attacks. According to Vishwakarma et al., (2019), a honeypot can help to detect new variants of IoT-DDoS attacks [9]. In their work, they lure the attacker by allowing him to invade the protection wall. Thereafter, the honeypot comes into the picture and records the manipulations that the invader tends to do as log files. These log files are then transformed into a tabular format that makes them work as a dataset. Then, they train the machine learning on these log files after being formatted to get useful information about the attack. However, this work is not tested in a real-world environment. Soe Y. et al., (2019) employ Artificial Neural Network (ANN) to detect IoT-DDoS attacks[22]. The authors used the newest released dataset named Bot-IoT dataset. Due to the imbalance in the dataset which contains 1.9 million attack packets and 477 benign packets, they apply a technique called SMOTE (Synthetic Minority Over-sampling Technique) to generate several benign packets equal to the number of DDoS packets. The detection system gains 100% accuracy. Dao N. et al., (2019) proposed MECshield, a framework that employs mobile edge computing to protect the Heterogeneous IoT environment [23]. The framework uses multiple intelligent filters and puts it at the edge of the attack resource/destination. A centralized controller supervises the interactions of the smart filters and propagates the corresponding features of the attack to the smart filters. The authors tested their mechanism using three different datasets (CAIDA, NSL-KDD, and DARPA). The experiment shows impressive results in the detection accuracy and the dilemma of the bottleneck which is a crucial issue in DDoS attacks is resolved by distributing the filters at multiple mobile edge points. Yizhen et al., (2020) proposed a detection system named FlowGuard which operates at the edge servers and is closest to the IoT network [24]. FlowGuard vets any packet passing through the edge server using a flow filter. They evaluated their work with CICDDoS2019 and other self-generated datasets and attained an accuracy of 98.8%.

2.2.3. Cloud Platform for IoT network security

Different works have also deployed cloud computing platforms to secure IoT networks. Due to the lack of protection of the IoT network from an insider attack on the SDN platform, cloud-based approaches have emerged. In SDN, edge, and gateway routers are exploited for the deployment of the detection system, and thus cannot handle attacks coming from a single device during the M2M communication (Conti et al., 2018) [15]. Weiqi et al., (2018) define the infrastructure-as-a-service (IaaS) which offers the network as a service, storage, and CPU for tenants, then the tenants outsource their infrastructure to Amazon and Microsoft Azure [25], for example. These outsourced infrastructures consist of virtual machines and are allocated to a certain tenant called Tenant Network (TN). The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

cloud administrator has full privileges on the TN but has no control over the physical infrastructure. For this reason, the authors proposed a TNGuard platform that keeps the administrator authorities at low levels. This would result in degraded privileges for the administrator and should result in a secured tenant IoT network. The authors evaluate their work using: 1) the time the system spends to boot, 2) the average time for response and 3) the rate of the cross-zone communication. All metrics show TNGuard is sufficient for most of the applications. However, this work lacks dynamic integrity verification. Djouani et al., (2018) proposed a framework to integrate SDN and cloud computing to secure IoT networks [26]. They separate the control plane and the data plane by putting the SDN controllers in the cloud and putting the data plane over the nodes and the gateways. The lack of resources in IoT is compensated by delegating the security mechanism to the cloud. This method is not tested in a real environment. Moreover, Fog computing integrated with SDN and blockchain is proposed by Muthanna et al., (2019) to secure IoT networks [27]. The authors leverage the benefits of Fog computing such as minimizing the latency in communication. Also, in the case of data offloading, Fog computing provides a path to face this issue. Moreover, Fog computing renders new services more efficient. So, the authors come up with a layer of distributed edge computing-based Fog nodes and locate it in the middle between IoT central cloud and different IoT nodes. They evaluate their work and claim that their method is efficient in terms of computing resource utilization and latency. Table 1 summarizes the IoT defense mechanisms for DDoS attacks.

Table 1. Summary of IoT defense mechanisms.

Research Objective	Methodology Used	Results	Limitations
Employ SDN controller in the edge IoT networks	Builds rules in the switches based on the discovered attack packets to blacklist and delete them	Effective to detect Mirai-based attacks	N/A
Design framework for Software-Defined Internet of Things (SD-IoT) (2018)	SD-IoT uses cosine similarity of the vectors of the incoming packet and based on that it decides if a DDoS the attack is triggered or not.	Detect and block the attack at the source	Susceptible to large amounts of traffic
Edge computing-based cloud platform (2018)	Cloud ShadowNet web service will receive the packets from edge nodes to vet them	It performs 10x faster in detecting UDP flood attacks than existing models	Focus on speed but ignore accuracy since no method to differentiate between attack and Flash crowd
Stateful SDN architecture is used to develop an entropy-based detection for IoT-attack (2020)	calculating the correlation of the entropy values for different extracted features to identify the attacks	Entropy-based with SDN can mitigate the attack by adding entries to the flow table of switches	Considers only limited types of DDoS attacks
Edge computing based SDN (2020)	Intelligence is employed in the edge devices (OF switches) instead of the controller	99% precision rate for detecting the attack at the OF switches	Susceptible to sophisticated DDoS attacks like Crossfire attack
Anomaly-based detection to detect IoT traffic attacks	Deep Autoencoders are used to learn the behavior of normal IoT traffic, and then detect the anomalies if reconstruction is failed	N-BaIoT succeeded in detecting every single attack with 100% TPR	Deployment in a real-world scenario is costly
Detect DDoS based on the traffic behavior (2019)	Anomaly-based detection that measures the deviation of any traffic compared to legitimate traffic from the same class	Adaboost can effectively detect anomalies in the traffic behavior	No significant criterion to classify IoT devices
Use state-of-the-art deep learning techniques to classify DDoS attacks	Transform IoT traffic to image form and train ResNet over the new image format	ResNet gains 99.99% accuracy for binary classification	N/A
Honeypot-based detection framework (2019)	Luring the attacker to invade the IoT protection wall, then log all the activities and gain useful information about the new attacks	New variants of attacks are detected	Not tested in the real environment
Apply ANN to classify benign packets and DDoS packets (2019)	Constructing a simple ANN network with a single layer to classify DDoS attacks and benign packets	The detection results reached up to 100%	N/A
Employ Mobile Edge Computing (MEC) to safeguard the IoT environment	MECshield is proposed that uses a smart filter distributed over MEC to mitigate DDoS attack	Better detection accuracy and DDoS bottleneck is resolved	N/A
Detect IoT-DDoS at the edge servers (2020)	FlowGuard is employed at the edge server to vet the traffic passing through the server	Identification of Long-short-term memory gains an accuracy of 98.8%	N/A
Securing tenant IoT network (2018)	TNGuard to degrades the tenant network administrator to ensure the security	Several metrics used show sufficient work with most of the applications	Lack of dynamic integrity verification

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

2.3. CoAP Security Overview

This section introduces the DTLS binding for CoAP. Indeed, CoAP is equipped with the security demands such as keying materials and an access control list during the provisioning phase, or what so-called RawPublicKey mode. After RawPublicKey mode finishes, the IoT device should be in one of the following security modes:

A- NoSec Mode: No security protocol is engaged (DTLS disabled). Alternatively, assuming lower-layer protocol will implement the security mechanism; thus, messages are transferred with no security.

B- PresharedKey Mode: enabled DTLS. This mode has a list of pre-shared keys and there is a list of nodes that are assumed to engage in the communication for every key. For instance, in a significant scenario, every node has its key if it engages in CoAP communication. Contrarily, if a specific pre-shared key is shared with two entities or more, the entities are authenticated as a group by that key and would not be considered as specific peers anymore.

C- RawPublicKey Mode: enabled DTLS. This mode is adopted by the devices that required authentication, which uses the asymmetric key for each device to help to identify and communicate with these devices without a certificate.

D- Certificate Mode: DTLS is enabled. In this mode, an asymmetric key pair with X.509 certificate is reserved for a given device. The certificate is validated by what is called Certification Authority (CA).

2.3.1. Proposed defense mechanisms for Securing CoAP against DDoS Attacks

DTLS for CoAP Security

Some researchers proposed that DTLS be implemented in CoAP for security purposes. Maleh et al., (2016) mentioned that Datagram Transport Layer (DTLS) handshake suffers from spoofing IP address DDoS attacks [28]. To face this issue, the DTLS handshake is expanded with a cookie exchange technique. The capability and threshold for receiving packets must be declared with the IP address to the server, then, the server reserves resources for new communication. Because of the costly energy of this technique, they moved it to Proxy AC Server with no energy constraints. Their method reduces the resource occupancy DTLS ROM by 23% compared to standard DTLS. Haroon A. et al. (2017) proposed an enhancement to DTLS to make it resistant to DDoS attacks [29]. They claim that their method can reduce the overhead of handshaking time, packet size, and energy consumption compared to other works. The authors' system named E-lithe relies on Trusted Third Party (TTP) to reduce the overhead on the server-side. Compared to Lithe and other works, E-lithe outperforms others in terms of running time and reduced packet size. Later, this work was enhanced by Shruti et al., (2018) who claimed that their work outperforms E-Lithe [30]. The authors' work essentially focuses on the comparison between the payload of benign and malicious packets, defining a threshold, if the threshold is exceeded, then the source IP is blocked. They evaluate their work based on the malicious packet delivery ratio and legitimate packet drop ratio which outperforms the work done by Haroon et al.

SDN for CoAP Security

Alzahrani et al. (2020) implemented a software-defined networking scheme to authorize the messages over the CoAP protocol [31]. They argue that distributed approach in which IoT devices employ powerful gateways attached to them may be insufficient and making the access control decisions accomplished by the controller renders the security of CoAP messages more efficient and can help to avoid DDoS attacks.

Machine Learning for CoAP Security

Machine Learning is also proposed to detect and mitigate DDoS attacks against CoAP. Granjal et al., (2018) developed a framework that employs a threshold to mitigate DDoS attacks [32]. They

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

define a limit for messages of CoAP, and after the limit is exceeded, extra messages will be dropped. The authors enhanced their work and proposed an anomaly-based detection system to protect the 6LoWPAN and CoAP protocol from DDoS attacks [33]. SVM is used as an ML-Classifier and gains an accuracy of 93%. However, the system generates a high false-positive rate of around 20%. Doshi et al., (2018) developed a machine learning pipeline that is employed on middleboxes such as routers or firewalls to detect IoT DDoS attacks [34]. They claim that IoT traffic is distinct from other traffics coming from other internet-connected devices because IoT traffic is repetitive and often communicates with a small finite of endpoints. After testing this method, it gains an accuracy of 99% using RF, KNN, and Neural Networks.

Other Methods for CoAP Security

Other works propose different methods to cope with DDoS attacks against CoAP. Anirudh et al. (2017) propose a honeypot to lure the attacker and log his information for future verification or block purposes.

Table 2. Summary of IoT-CoAP defense mechanisms.

Research Objective	Methodology Used	Results	Limitations
Detect and mitigate DDoS attacks against CoAP(2016)	DTLS handshake is expanded with a cookie exchange technique to check the authentication of a message.	Low computation time by delegating all handshakes to a third party	The assumption of the third party (Proxy AC server) is trustworthy for all the time.
Secure DTLS for IoT(2017)	Trusted Third Party (TTP) is used to avoid DoS attacks and reduce overhead on the server-side.	Energy consumption, reduced packet size, and reduced running time outperforms similar works	DTLS is computationally heavy for IoT devices.
Detect DDoS against IoT (2018)	Compare the payload of the benign and malicious packet, define a threshold and if exceeded, the source IP is blocked	Evaluate the average for both delivering of packets and dropping of legitimate packets	Focuses on packet payload feature only
Securing CoAP messages (2020)	The SDN-based approach is developed to authorize the messages of CoAP	Decrease overhead to the controller and CoAP responses become faster	N/A
Prevention framework from intrusion and DoS attack (2018)	Relies on the threshold by limiting the incoming request to a fixed number and if exceeded, the source re- quest is blocked.	Fair energy and memory consumption when running the proposed system.	Susceptible to spoofing at- tack
Protect 6LoWPAN and CoAP from DoS attack (2018)	Employ an anomaly-based detection system to protect CoAP from DoS attack.	SVM classifies the anomalies with an accuracy of 93%	The high false-positive rate of 20%
Detect IoT DDoS attack (2018)	Machine learning DDoS detection framework	RF, KNN, and Neural Networks gain about 99% accuracy	Some IoT devices have different regular patterns but not distinct patterns
Deploy a honeypot to detect DDoS attacks (2017)	Deploy a honeypot with two phases, first to log the anomalies and second to verify or block the client	60% efficiency when a honeypot is implemented	Anomaly-based detection may result in high false-positive
Defend IoT against DDoS while maintaining benign traffic (2018)	Leverage the fast retransmit and flow control mechanism of TCP to retransmit benign packets at the fastest rate and malicious packets at a harmless rate.	Compared to D-WARD, FR-WARD performs better in retransmission, duration, and energy consumption.	Susceptible to other kinds of attacks
Test CoAP MITM attack which results in spoofing, sniffing, and DoS attack (2019)	Set up a client/server architecture to check if the communication between the two devices using CoAP is secure	Burp suite tool is used to intercept the communication between the client and the server	Susceptible to sniffing attack

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

3. Materials and Methods

To secure CoAP in its vicinity, we focus on finding the dataset that contains CoAP-level packets only. As stated in section 1.1, our target is to secure the CoAP in the application layer level. The CIDAD dataset is available on the Github.com [38] is mainly generated to attack the CoAP in its vicinity. However, this dataset has fewer samples of the attacks and ~10,000 samples of the benign packets. Machine learning algorithms always are greedy for decent samples of data to learn. In the next section, we elaborate on how to extend the dataset to gain ~100,000 samples of DoS and benign packets.

3.1. Dataset Collection

The CIDAD dataset is mainly targeting the CoAP with three different DDoS attacks: duplication, interception, and modification of the CoAP message with total malware samples of 288 only. Interception means intercepting stochastically sent packets before reaching the destination, whereas duplication is changing the content of the CoAP message, and modification is increasing the number of tokens. The rest of the ~10,000 packets are benign packets. This poses an imbalance in the dataset since only 0.02% of the dataset is malware. So, we extend the dataset to 100,000 samples, of which ~50% for benign and ~50% are for malware. We used to use Generative Adversarial Network (GAN) to extend the 288 malware samples to ~50,000 samples. The generated samples are compared with the original data to ensure the similarity between them by training each and calculating the RMSE. Our finding is surprising that the RMSE is ~0.005 for original data whereas the generated data is ~0.09. This indicates a coherent similarity between the original malware and the generated ones. On the other hand, for the benign samples, we do the same and find that RMSE for the original samples and the generated ones is ~0.03. Figure 10. Illustrate the general structure for GANs. Figure 11. Shows the distribution of the collected dataset.

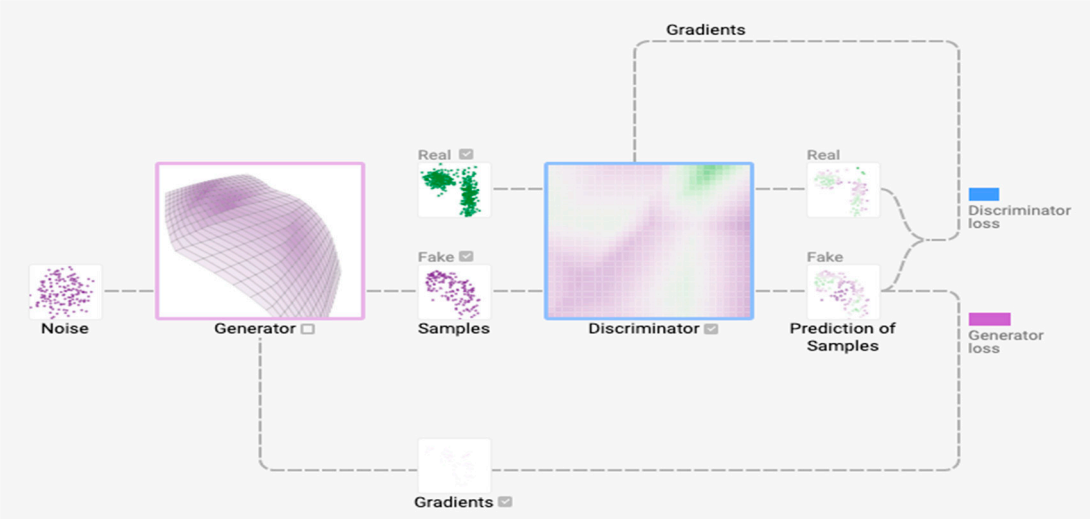


Figure 10. GAN Architecture [40].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

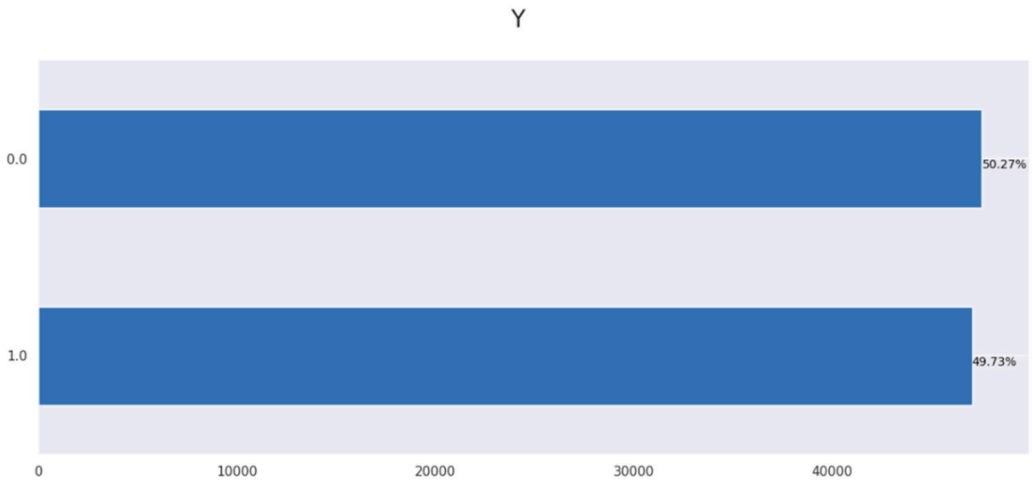


Figure 12. The distribution of the collected dataset (0 for benign, 1 for malware).

3.2. Feature Extraction

CoAP has a total of 86 features that can be extracted from the pcap file. However, most of them have a huge number of missing values due to optionality in the communication and the GANs cannot generate fake data for empty values of the features. Only 10 features have decent values in the dataset. So, we only focus on these features as depicted in **Figure** 13. Then, we use the Pearson Correlation method formula (1) to get the relevant features to the label.

$$r_{xy} = \sum_{i=1}^{n} \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \qquad (1)$$

Where r_{xy} is the Pearson correlation between two features x and y, n represents the total number of samples, x_i ***and*** y_i are the individual sample points indexed with *i*, $\bar{x}$ is the sample mean, and the same for $\bar{y}$.

We focus only on the features that have ±0.30 correlation to the label. **Figure** 13. shows the correlation between the features and the label.

Figure 13. The Pearson Correlation between the features and the label.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

The Pearson Correlation for all the features is less than the target value ±0.30. Therefore, we use other methods which represent the statistical methods: Lasso Regression and One-Way ANOVA test. Lasso Regression is a regression analysis that deepens the accuracy and interpretability by performing regularization alongside the variable selection as shown in formula (2). On the other hand, One-Way ANOVA checks the significant independence of two or more samples, where the P-value decides a rejection for the null hypothesis of samples equality if the score is less than 0.05 as shown in formula (3).

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2+\lambda\sum_{j=0}^{p}|w_j| \qquad (2)$$

Where M for the total number of samples, P for features, and w is the slope.

$$\boldsymbol{F}=\frac{\boldsymbol{MSB}}{\boldsymbol{MSW}} \qquad (\mathbf{3})$$

Where *F* represents the ANOVA coefficient, *MSB*: Mean sum of squares between the samples, and *MSW*: Mean sum of squares within samples.

We assume that any feature that is recommended by the two methods (ANOVA and Lasso) is correlated to the label and will be used for the training phase of the model. After calculating the Lasso and ANOVA methods, we find a total of six features that are recommended by both methods as depicted in **Figure** 14.

The recommended features are coap.mid, coap.opt.disc, coap.opt.location_query,coap.opt.uri_host, coap.retransmitted, and coap.token. Table 3. Shows the description for each feature. The features that are recommended by the ANOVA only are ignored.

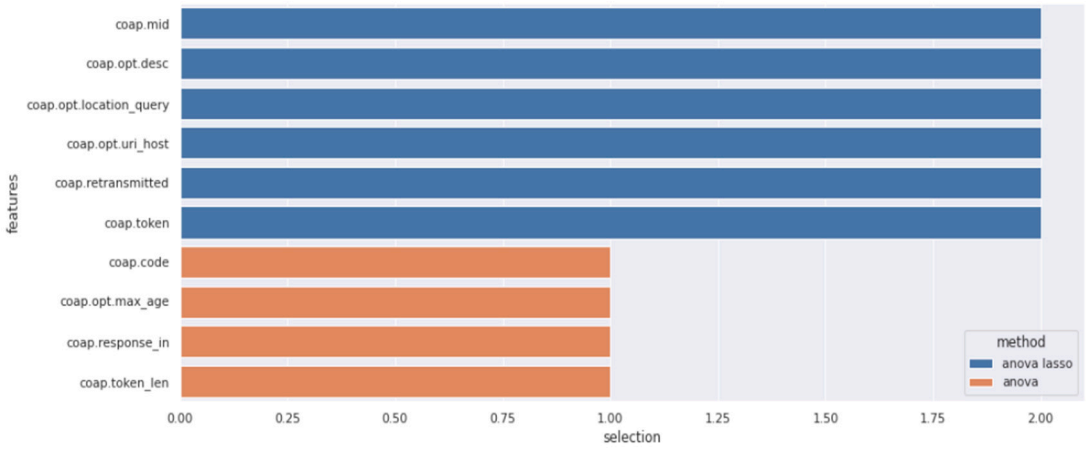


Figure 14. Lasso Regression and ANOVA analysis for CoAP features.

Table 3. CoAP Features Description.

Feature	Description	Type
coap.mid	*Message ID*	*Unsigned integer (2 bytes)*
coap.opt.desc	*Opt Desc*	*Character string*
coap.opt.location_query	*Location-Query*	*Character string*
coap.opt.uri_host	*Uri-Host*	*Character string*
coap.retransmitted	*Retransmitted*	*Label*
coap.code	*Code*	*Unsigned integer (1 byte)*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

3.3. Model Training

We choose to test four machine learning classifiers (LinearSVC, Naïve Byes, Random Forest, and Decision Tree). We split the dataset into 70% for training and 30% for testing. In this section, we analyze the performance of each classifier. The model is depicted in **Figure** 15.

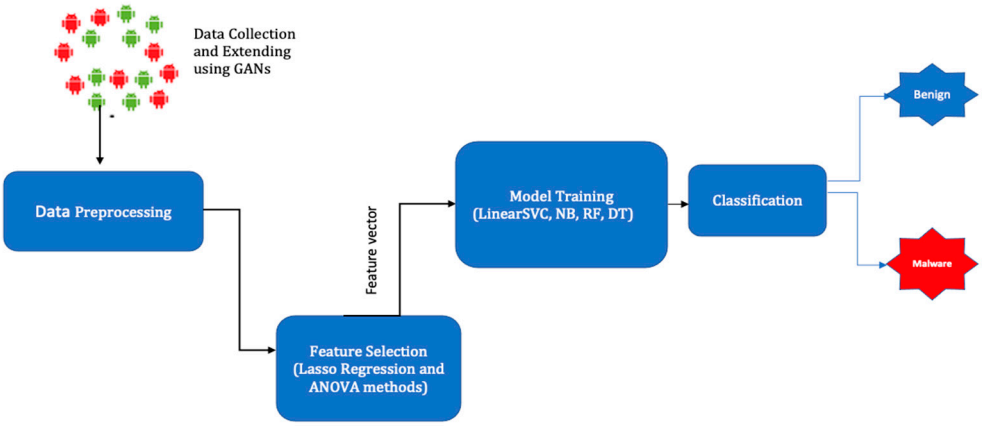


Figure 15. Proposed Model.

4. Results

4.1. LinearSVC

LinearSVC performs worse with an accuracy of 59%. The confusion matrix and Receiver Operating Characteristic (ROC) show a lot of false positives and false negatives (**Figure** 16 and **Figure** 17). There are 7843 benign packets but mistakenly classified as malware. In addition, 4387 malware packets were wrongly classified as benign. It can be inferred that the false positives are higher than the false negatives using the LinearSVC algorithm.

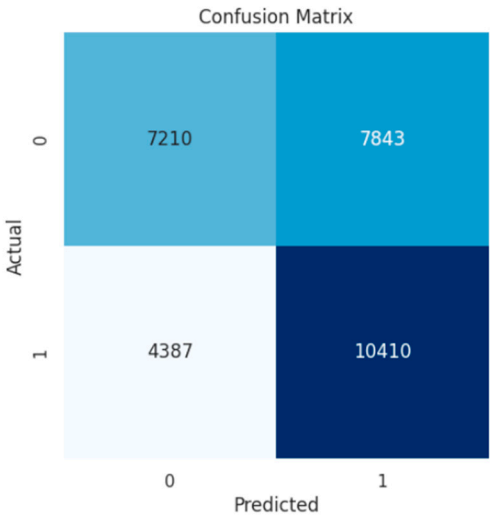


Figure 16. Confusion Matrix for LinearSVC.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

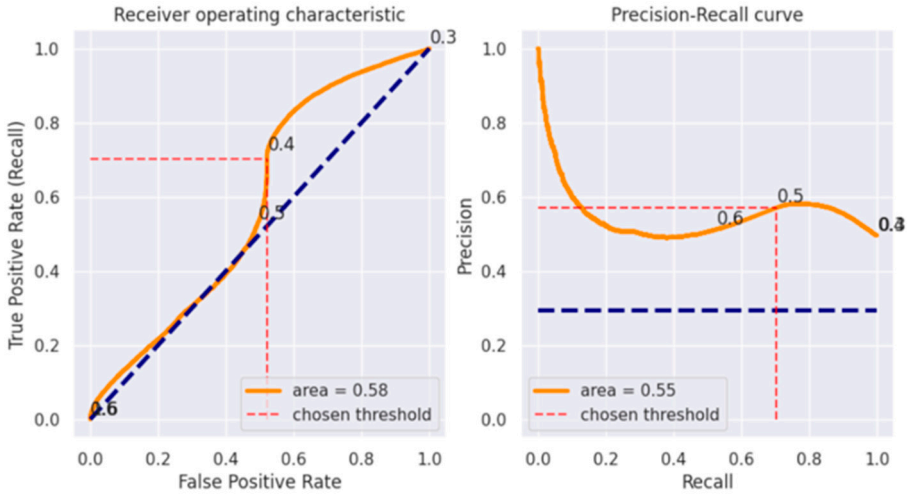


Figure 17. ROC curve for LinearSVC.

4.2. Random Forest

The Random Forest Classifier shows better results than LinearSVC with an accuracy of 92%. However, the false positive rate is fair with total sample of 662 as shown in **Figure** 18. Contrarily, the false negative is higher with 1651 samples. **Figure** 19. Shows the ROC curve for the RF algorithm.

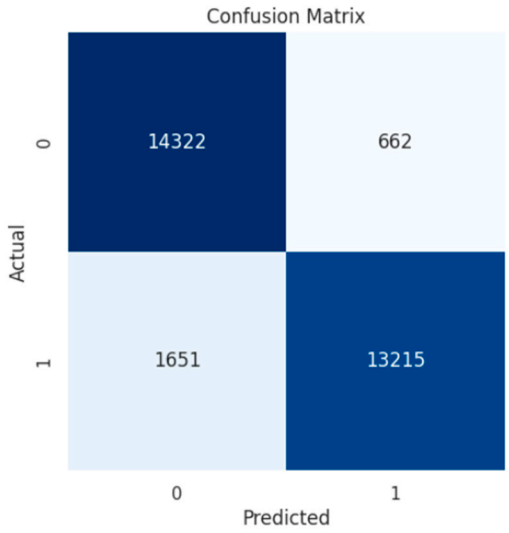


Figure 18. Confusion matrix for RF algorithm.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

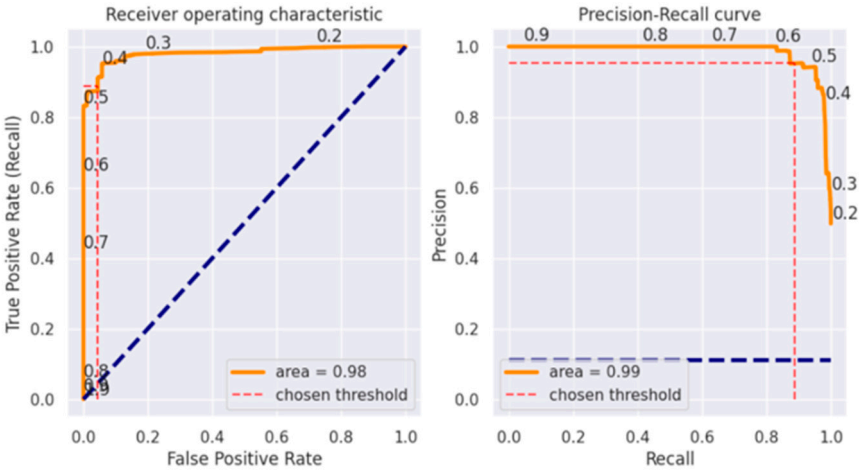


Figure 19. ROC curve for RF algorithm.

4.3. Decision Tree

This algorithm performs well with an accuracy of 98%. The number of false positives is 347 while the false negative rate is 302. The ROC curve infers the optimism of the decision tree algorithm.

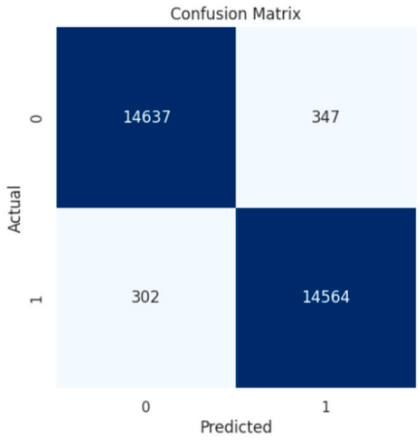


Figure 20. Confusion matrix for DT algorithm.

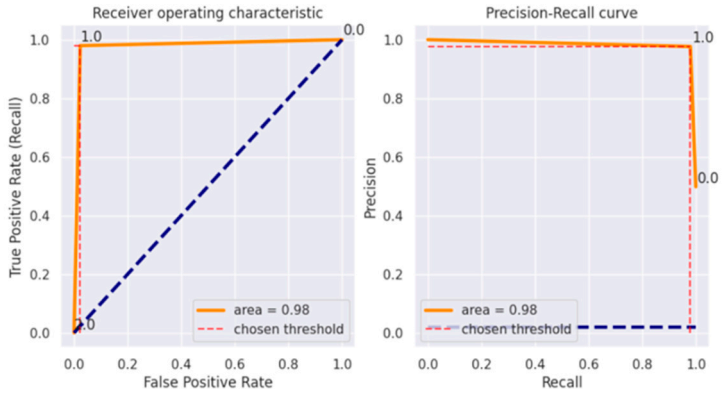


Figure 21. ROC curve for DT.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

4.4. Naïve Byes

Naïve byes perform worse on our data and gain an accuracy of 63%. The false negative rate is higher than the false positives. The former is wrongly classified by the model with total samples of 6415 and 4612 for the latter. **Figures** 22 and 23 show the confusion matrix and ROC curve for Naïve Byes performance.

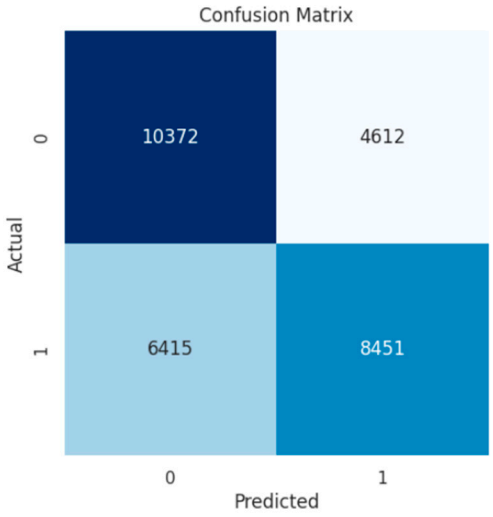


Figure 22. Confusion matrix for Naïve Byes.

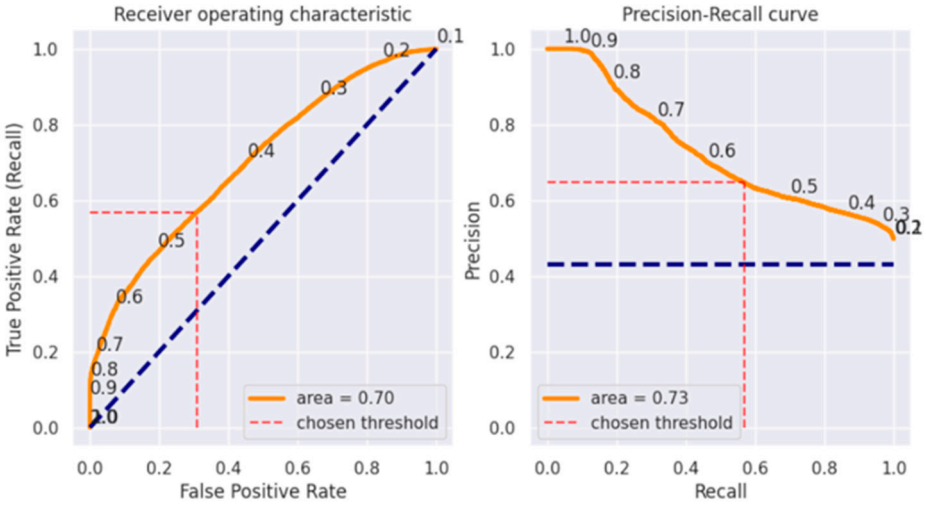


Figure 23. ROC curve for Naïve Byes.

From the results above, it can be inferred that the decision tree algorithm beats other algorithms in all the metrics used to calculate the performance of the selected algorithms. In the field of security, it is important to reduce the false negative to the lowest value due to the impact of single false negative packet can cause huge damage to the network. That what the decision tree algorithm performs in our proposed model since only ~300 were misclassified as benign for a total of 30,000 samples used for testing. **Figure** 24 shows the calibration plot between the selected classifiers. It plots the DT performance with a significant similarity to the perfectly calibrated rather than other algorithms. The mean predicted probability is plotted for each algorithm in **Figure** 25. The decision tree algorithm shows more impressive results than other algorithms. Table 4. Represents the performance of each algorithm in terms of accuracy, precision, recall, and F1-score.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 May 2023 doi:10.20944/preprints202305.0070.v1

Table 4. Metrics for selected algorithms.

Model	Accuracy	Precision	Recall	F1-Score
LinearSVC	59%	62%	48%	54%
Decision Tree	98%	98%	98%	98%
Random Forest	92%	90%	96%	93%
Naïve Byes	63%	62%	69%	65%

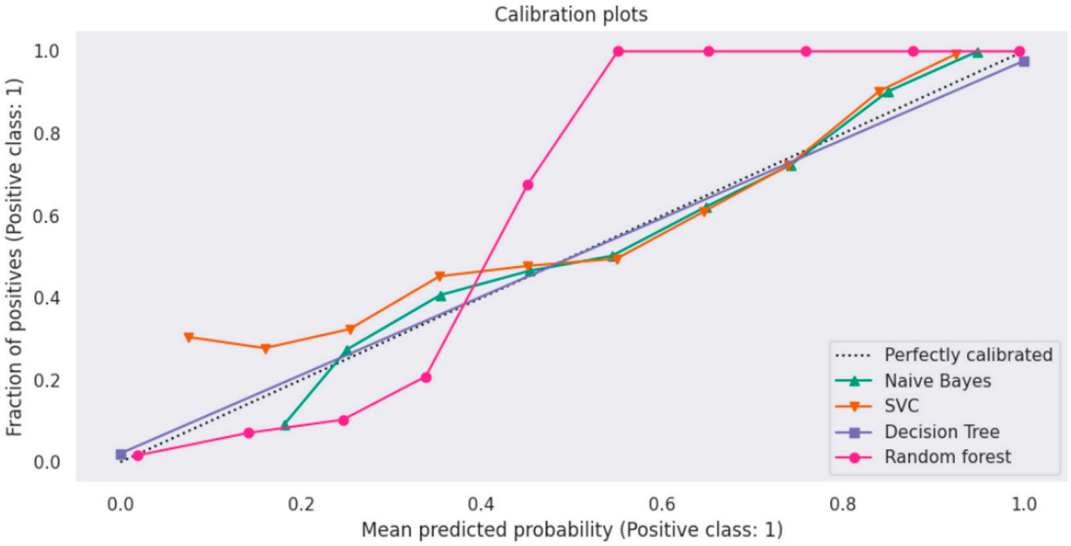


Figure 24. Calibration plot for the selected Classifiers.

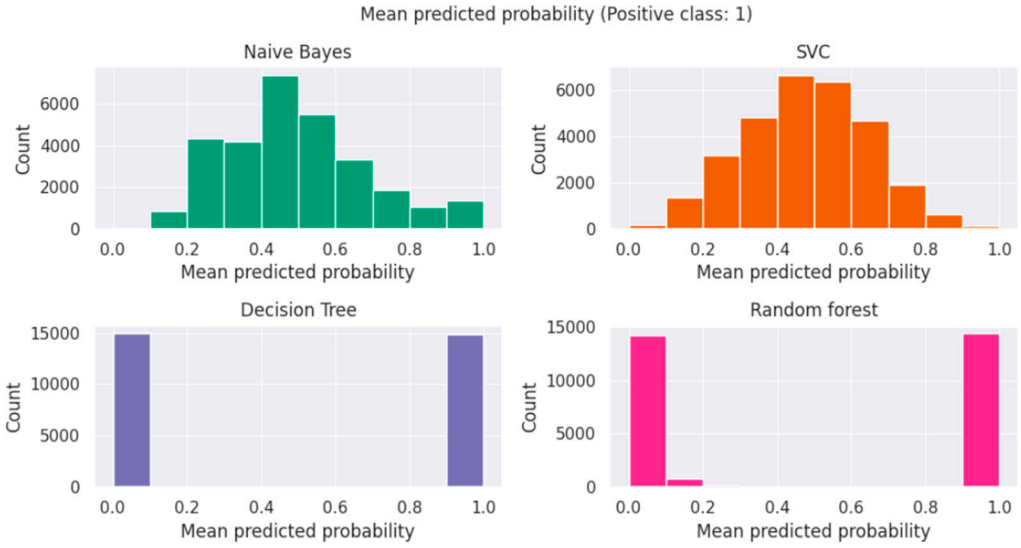


Figure 25. Mean Predicted Probability for the selected classifiers.

5. Discussion

The CoAP protocol suffers from the lack of research that states how to manage and implement the security of CoAP [6]. Several works employed DTLS to secure CoAP protocol from different threats including DDoS attacks. However, DTLS is not specially designed for constrained devices [6] due to the heaviness that renders it unsuitable for constrained devices that have lower power and lower energy. Moreover, DTLS is a cryptography-based protocol, and cryptography cannot detect attackers who use legal keys but act maliciously, thus DDoS attacks that come from legitimate IPs may go undetected [32]. Consequently, the demand for new methods to secure CoAP against DDoS attacks is highly required since DTLS cannot protect CoAP from some internal and external attacks

[33]. On the other hand, LSPWSN is also employed to secure CoAP messages. However, this protocol is working in the network layer, but not the application layer that CoAP works over. It is significant to have the detection system near the victim, while it is beneficial to mitigate the attack near the attack source [7]. So, this work outperforms DTLS and LSPWSN in terms of defending CoAP in its vicinity while DTLS and LSPWSN protect CoAP by vetting the packets in the down layers of IoT architecture, namely, the transport and network layers. Motivated by the assumption that it is beneficial to have the detection system near to the victim [7], this work is developing a model that can secure CoAP in its vicinity (the application layer). Another work [33] builds an anomaly-based detection method in the application layer to secure CoAP from DDoS attacks. However, this work collects the features from the protocols that work over the down layers (IEEE 802.15.4 features, 6LoWPAN features, IPv6 features, and CoAP features) and gains an accuracy of 93%. The proposed method outperforms their work since we gain accuracy of 98% accuracy while focusing on the CoAP-level features only.

6. Conclusions

The proposed model defines a method to secure the CoAP in its vicinity from several DDoS attacks (duplication, modification, and intercepting of CoAP messages). The dataset used in this work is extended from ~11000 samples to 100,000 samples because it lacks the sufficient number of malware samples using GANs. Implementing a detection method in the same layer with the protocols is significant work that fulfills the assumption of the effectiveness of developing the detection system with protocol in its vicinity. So, this work discusses the different attacks that target IoT networks globally and the attacks that target CoAP specifically. The proposed model can detect DDoS attacks with an accuracy of 98% with the decision tree algorithm. In future work, deploying and implementing the proposed model is encouraged. Moreover, combining the dataset with other attacks such as enumeration and amplification attacks will result in a coherent dataset for testing with new methods.

Author Contributions: This research is done by the corresponding author Mr. Sultan Almeghlef as a requirement for PhD degree. The PhD candidate must publish in high impact journals to obtain the PhD. The supervisors are supportive and leaders for this research. Prof. Dr. Abdullah AL-Ghamdi is the leader for this research as a supervisor for the corresponding author, while Prof. Dr. Muhammad Ramzan and Prof. Dr. Mahmoud Ragab are co-supervisors for the corresponding author. The supervisors always guide and comment to the correspondence in every step of this research.

Funding: This research is completely funded by the corresponding author as a requirement to obtain the PhD degree.

Data Availability Statement: The dataset is publicly available at https://www.kaggle.com/datasets/salmeghlef/ddos-coap-dataset-cidad.

Acknowledgments: I am thankful for every support and direction from the supervisors to produce this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vishwakarma, R. and Jain, A. K. [2020], 'A survey of DDoS attacking techniques and defense mechanisms in the IoT network', Telecommunication systems 73(1), 3–25.
2. Syed, N. F. [2020], 'IoT-MQTT based denial of service attack modeling and detection.
3. Hussain, F., Abbas, S. G., Husnain, M., Fayyaz, U. U., Shahzad, F. and Shah, G. A. [2020], 'IoT dos and DDoS attack detection using resnet', arXiv preprint arXiv:2012.01971 .
4. Deng, L., Li, D., Yao, X., Cox, D. and Wang, H. [2019], 'Mobile network intrusion detection for IoT system based on transfer learning algorithm', Cluster Computing 22(4), 9889–9904.
5. Iglesias-Urkia, M., Orive, A., Urbieta, A. and Casado-Mansilla, D. [2019], 'Analysis of coap implementations for the industrial internet of things: a survey, Journal of Ambient Intelligence and Humanized Computing 10(7), 2505–2518.
6. Alhaidari, F. A. and Alqahtani, E. J. [2020], 'Securing communication between fog computing and IoT using constrained application protocol (coap): A survey, Journal of Communications 15(1), 14–30.
7. Bhardwaj, K., Miranda, J. C. and Gavrilovska, A. [2018], Towards IoT-DDoS prevention using edge computing, in '{USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 18)'.

8. Shafiq, M., Tian, Z., Sun, Y., Du, X. and Guizani, M. [2020], 'Selection of effective machine learning algorithm and bot-IoT attacks traffic identification for the internet of things in the smart city, Future Generation Computer Systems 107, 433–442.
9. Vishwakarma, R. and Jain, A. K. [2019], A honeypot with machine learning-based detection framework for defending IoT based botnet DDoS attacks, in '2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)', IEEE, pp. 1019–1024.
10. Rahman, R. A. and Shah, B. [2016], Security analysis of IoT protocols: A focus in coap, in '2016 3rd MEC international conference on big data and smart city (ICBDSC)', IEEE, pp. 1–7.
11. Mohamadi, M., Djamaa, B. and Senouci, M. R. [2020], 'Industrial internet of things over IEEE 802.15. 4 tsch networks: design and challenges', International Journal of Internet Technology and Secured Transactions 10(1-2), 61–80.
12. Musaddiq, A., Zikria, Y. B., Kim, S. W. et al. [2020], 'Routing protocol for low-power and lossy networks for heterogeneous traffic network', EURASIP Journal on Wireless Communications and Networking 2020(1), 1–23.
13. Shelby, Z., Hartke, K. and Bormann, C. [2014], 'The constrained application protocol (coap)'.
14. Munshi, A., Alqarni, N. A. and Almalki, N. A. [2020], DDoS attack on IoT devices, in '2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)', IEEE, pp. 1–5.
15. Conti, M., Kaliyar, P. and Lal, C. [2019], 'Censor: Cloud-enabled secure IoT architecture over SDN paradigm', Concurrency and Computation: Practice and Experience 31(8), e4978.
16. Özçelik, M., Chalabianloo, N. and Gür, G. [2017], Software-defined edge defense against IoT-based DDoS, in '2017 IEEE International Conference on Computer and Information Technology (CIT)', IEEE, pp. 308–313.
17. Yin, D., Zhang, L. and Yang, K. [2018], 'A DDoS attack detection and mitigation with software-defined internet of things framework', IEEE Access 6, 24694–24705.
18. Galeano-Brajones, J., Carmona-Murillo, J., Valenzuela-Valdés, J. F. and Luna-Valero, F. [2020], 'Detection and mitigation of dos and DDoS attacks in IoT-based stateful Sdn: An experimental approach', Sensors 20(3), 816.
19. Yang, Y., Wang, J., Zhai, B. and Liu, J. [2019], IoT-based DDoS attack detection and mitigation using the edge of sdn, in 'International Symposium on Cyberspace Safety and Security', Springer, pp. 3–17.
20. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D. and Elovici, Y. [2018], 'N-baIoT—network-based detection of IoT botnet attacks using deep autoencoders', IEEE Pervasive Computing 17(3), 12–22.
21. Cvitić, I., Peraković, D., Periša, M. and Botica, M. [2019], 'Novel approach for detection of IoT generated DDoS traffic', Wireless Networks pp. 1–14.
22. Soe, Y. N., Santosa, P. I. and Hartanto, R. [2019], DDoS attack detection based on simple ann with smote for IoT environment, in '2019 Fourth International Conference on Informatics and Computing (ICIC)', IEEE, pp. 1–5.
23. Dao, N.-N., Phan, T. V., Kim, J., Bauschert, T., Cho, S. et al. [2017], 'Securing heterogeneous IoT with intelligent DDoS attack behavior learning', arXiv preprint arXiv:1711.06041 .
24. Jia, Y., Zhong, F., Alrawais, A., Gong, B. and Cheng, X. [2020], 'Flowguard: an intelligent edge defense mechanism against IoT DDoS attacks', IEEE Internet of Things Journal 7(10), 9552–9562.
25. Dai, W., Wan, P., Qiang, W., Yang, L. T., Zou, D., Jin, H., Xu, S. and Huang, Z. [2018], 'Tnguard: Securing IoT oriented tenant networks based on sdn', IEEE Internet of Things Journal 5(3), 1411–1423.
26. Djouani, R., Djouani, K., Boutekkouk, F. and Sahbi, R. [2018], A security proposal for IoT integrated with sdn and cloud, in '2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM)', IEEE, pp. 1–5.
27. Muthanna, A., A Ateya, A., Khakimov, A., Gudkova, I., Abuarqoub, A., Samouylov, K. and Koucheryavy, A. [2019], 'Secure and reliable IoT networks using fog computing with software-defined networking and blockchain', Journal of Sensor and Actuator Networks 8(1), 15.
28. Maleh, Y., Ezzati, A. and Belaissaoui, M. [2016], An enhanced dtls protocol for internet of things applications, in '2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)', IEEE, pp. 168–173.
29. Haroon, A., Akram, S., Shah, M. A. and Wahid, A. [2017], E-lithe: A lightweight secure dtls for IoT, in '2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)', IEEE, pp. 1–5.
30. Kajwadkar, S. and Jain, V. K. [2018], A novel algorithm for dos and DDoS attack detection in internet of things, in '2018 Conference on Information and Communication Technology (CICT)', IEEE, pp. 1–4.
31. Alzahrani, B. and Fotiou, N. [2020], 'Enhancing internet of things security using software-defined networking', Journal of Systems Architecture 110, 101779.
32. Granjal, J. and Pedroso, A. [2018], 'An intrusion detection and prevention framework for internet-integrated coap wsn', Security and Communication Networks 2018.

33. Granjal, J., Silva, J. M. and Lourenço, N. [2018], 'Intrusion detection and prevention in coap wireless sensor networks using anomaly detection', Sensors 18(8), 2445.
34. Doshi, R., Apthorpe, N. and Feamster, N. [2018], Machine learning ddos detection for consumer internet of things devices, in '2018 IEEE Security and Privacy Workshops (SPW)', IEEE, pp. 29– 35.
35. Anirudh, M., Thileeban, S. A. and Nallathambi, D. J. [2017], Use of honeypots for mitigating dos attacks targeted on IoT networks, in '2017 International conference on computer, communication and signal processing (ICCCSP)', IEEE, pp. 1–4.
36. Mergendahl, S., Sisodia, D., Li, J. and Cam, H. [2018], Fr-ward: Fast retransmit as a wary but ample response to distributed denial-of-service attacks from the internet of things, in '2018 27th International Conference on Computer Communication and Networks (ICCCN)', IEEE, pp. 1–9.
37. Arvind, S. and Narayanan, V. A. [2019], An overview of security in coap: Attack and analysis, in '2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)', IEEE, pp. 655–660.
38. Vigoya, L., Fernandez, D., Carneiro, V., & Cacheda, F. (8135, January 01). Cidad.pcap · Dad-Repository/CIDAD@A109B87. Retrieved April 14, 2023, from https://github.com/dad-repository/cidad/commit/a109b8706174af5d6b1cb06f6afac5fe0ce2b28e
39. Kahng, M., Thorat, N., Chau, D. H., Viégas, F. B., & Wattenberg, M. (2018). Gan lab: Understanding complex deep generative models using interactive visual experimentation. IEEE transactions on visualization and computer graphics, 25(1), 310-320.
40. Vigoya, L., Fernandez, D., Carneiro, V., & Cacheda, F. (2020). Annotated dataset for anomaly detection in a data center with IoT sensors. Sensors, 20(13), 3745.