REPUBLIC OF TÜRKİYE
ALTINBAŞ UNIVERSITY
Institute of Graduate Studies
Electrical and Computer Engineering

DETECTION OF DDOS ATTACK IN IOT NETWORKS USING DEEP LEARNING TECHNOLOGIES

Saja Emad Jumaah JUMAAH

Master's Thesis

Supervisor
Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2024

DETECTION OF DDOS ATTACK IN IOT NETWORKS USING DEEP LEARNING TECHNOLOGIES

Saja Emad Jumaah JUMAAH

Electrical and Computer Engineering

Master's Thesis

ALTINBAS UNIVERSITY
2024

The thesis titled DETECTION OF DDOS ATTACK IN IOT NETWORKS USING DEEP LEARNING TECHNOLOGIES prepared by SAJA EMAD JUMAAH JUMAAH and submitted on 07/06/2024 has been accepted unanimously for the degree of Master of Science in Electrical and Computer Engineering

Assoc. Prof. Dr. Sefer KURNAZ

the Supervisor

Thesis Defense Committee Members:

Assoc. Prof. Dr. Sefer KURNAZ	Department of Computer Engineering, Altınbaş University	_____
Assoc. Prof. Dr. Abdullahi Abdu IBRAHIM	Department of Computer Engineering, Altınbaş University	_____
Asst. Prof. Dr. Serdar KARGIN	Department of Biomedical Engineering, Arel University	_____

I hereby declare that this thesis meets all format and submission requirements of a master's Thesis.

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Saja Emad Jumaah JUMAAH

Signature

DEDICATION

I would like to dedicate my thesis to my supervisor Assoc. Prof. Dr. Sefer KURNAZ and for my family, friends and all the people that help me during my study.

PREFACE

According to the preface, this work was submitted as a gift to my main advisor Prof. Dr. for his time and advice.

ABSTRACT

DETECTION OF DDOS ATTACK IN IOT NETWORKS USING DEEP LEARNING TECHNOLOGIES

JUMAAH, Saja Emad Jumaah,
M.Sc. Information Technologies, Altınbaş University,
Supervisor: Asst. Prof. Dr. Sefer KURNAZ
Date: June / 2024
Pages: 78

The primary objective of this research was to investigate and develop effective techniques for detecting Distributed Denial of Service (DDoS) attacks in Internet of Things (IoT) networks using Convolutional Neural Networks (CNNs). Through theoretical analysis, empirical evaluations, and practical implementations, we have made significant strides towards achieving this goal.

Our findings demonstrate that CNN-based approaches hold promise for detecting DDoS attacks in IoT environments, offering several advantages over traditional methods. By leveraging the inherent capabilities of deep learning, our proposed framework achieved high detection accuracy while operating efficiently within the resource-constrained constraints typical of IoT devices. The framework exhibited robustness against adversarial attacks and demonstrated adaptability across diverse IoT deployments and attack scenarios.

Furthermore, our research contributes to advancing the understanding of DDoS detection in IoT networks and provides practical insights for cybersecurity practitioners and researchers. The proposed guidelines and best practices for deploying CNN-based DDoS detection systems in operational IoT environments serve as valuable resources for ensuring the security and resilience of IoT ecosystems against evolving cyber threats. Our research underscores the potential of deep learning technologies, particularly CNNs, in addressing the challenges of DDoS detection in IoT networks. By developing robust and scalable solutions, we have taken significant steps towards enhancing the security posture of IoT infrastructures and safeguarding against malicious attacks.

Keywords: DDos, IoT, CNN, IoT, Deep Learning

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

IoT	:	Internet of Things
DL	:	Deep Learning
CNNs	:	Convolutional Neural Networks
RNNs	:	Recurrent Neural Networks
DDoS	:	Distributed Denial of Service Attack.

1. INTRODUCTION

1.1 BACKGROUND

The advent of the Internet of Things (IoT) has revolutionized the way we interact with and perceive technology in our daily lives. IoT devices, ranging from smart home appliances to industrial sensors, have permeated various domains, offering unparalleled convenience, efficiency, and connectivity. However, this ubiquitous connectivity also introduces unprecedented challenges, particularly in terms of security [1]. As the number of connected devices proliferates, so does the potential attack surface for malicious actors. One of the most insidious threats facing IoT ecosystems is Distributed Denial of Service (DDoS) attacks [2].

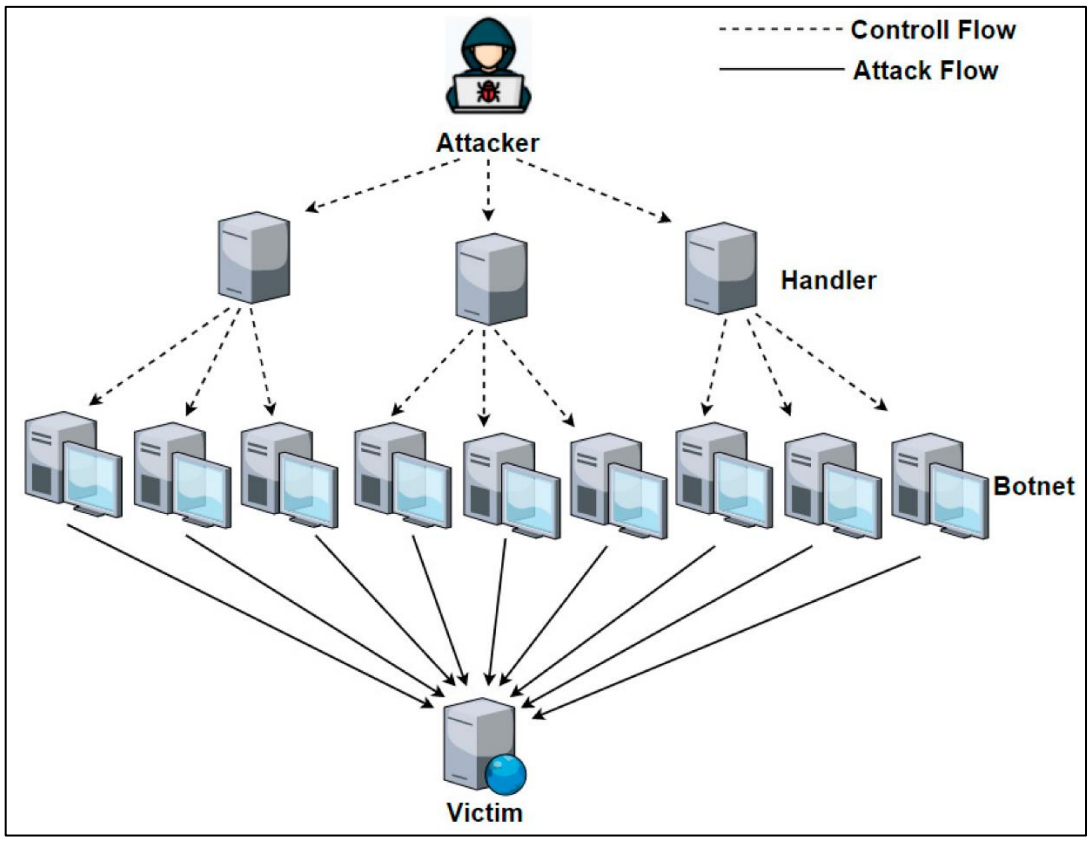

Figure 1.1: Workflow of DDoS Attacks [3].

DDoS attacks aim to disrupt the normal functioning of a targeted system or network by overwhelming it with a flood of malicious traffic, rendering it inaccessible to legitimate users. While DDoS attacks have been a concern in traditional networks for decades, the proliferation of IoT devices has magnified the scale and impact of such attacks. These attacks

not only pose a threat to the availability of IoT services but also have broader implications for critical infrastructure, public safety, and economic stability.

Traditional methods for detecting and mitigating DDoS attacks often rely on signature-based approaches, which are ineffective against novel and sophisticated attack vectors. Furthermore, the resource-constrained nature of many IoT devices imposes significant constraints on the computational resources available for traditional security measures. Consequently, there is a pressing need for advanced, adaptive, and resource-efficient techniques to detect and mitigate DDoS attacks in IoT networks.

Deep Learning (DL) has emerged as a promising paradigm for addressing the challenges posed by DDoS attacks in IoT networks. DL techniques, particularly Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and their variants, have demonstrated remarkable capabilities in extracting complex patterns and features from vast volumes of data.

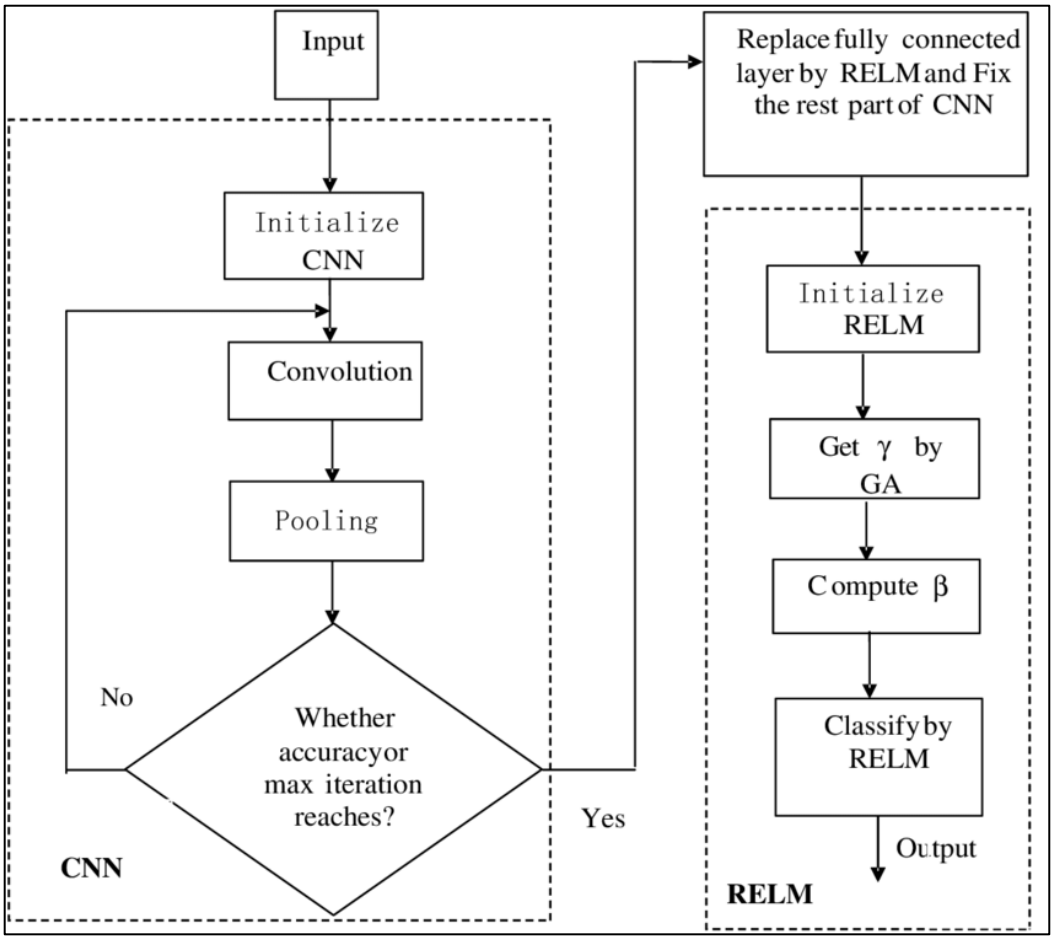

Figure 1.2: CNN and RNN Workflow.

By leveraging the inherent hierarchical representations learned through deep neural networks, DL models have shown efficacy in various domains, including image recognition, natural language processing, and cybersecurity. The application of DL in DDoS detection offers several distinct advantages. Unlike traditional signature-based methods, DL models can learn to discern subtle patterns and anomalies indicative of DDoS attacks, without relying on predefined rules or signatures. Moreover, DL models are inherently scalable and adaptable, capable of accommodating diverse IoT environments and evolving attack strategies. Additionally, DL-based approaches have the potential to minimize false positives and false negatives, thereby enhancing the accuracy and efficacy of DDoS detection mechanisms. Despite the potential benefits of DL-based DDoS detection in IoT networks, several challenges remain to be addressed. The inherent heterogeneity and variability of IoT environments present challenges in data collection, preprocessing, and model training. Moreover, the resource constraints of IoT devices necessitate the development of lightweight DL architectures that can operate efficiently within constrained computational environments. Furthermore, ensuring the robustness and resilience of DL models against adversarial attacks is crucial for deploying effective DDoS detection solutions in real-world IoT deployments. In light of these challenges, this thesis aims to explore and evaluate the feasibility, efficacy, and practical implications of using deep learning technologies for the detection of DDoS attacks in IoT networks. By leveraging state-of-the-art DL architectures, innovative feature engineering techniques, and comprehensive experimental evaluations, this research endeavors to advance our understanding of DDoS detection in IoT environments and contribute to the development of robust and scalable cybersecurity solutions for the IoT era. The introduction of the Internet of Things (IoT) has fundamentally altered the way in which we engage with and make sense of technology in our day-to-day lives. The Internet of Things (IoT) gadgets, which include everything from smart home appliances to industrial sensors, have spread over a variety of fields, providing an unprecedented level of ease, efficiency, and connectedness.

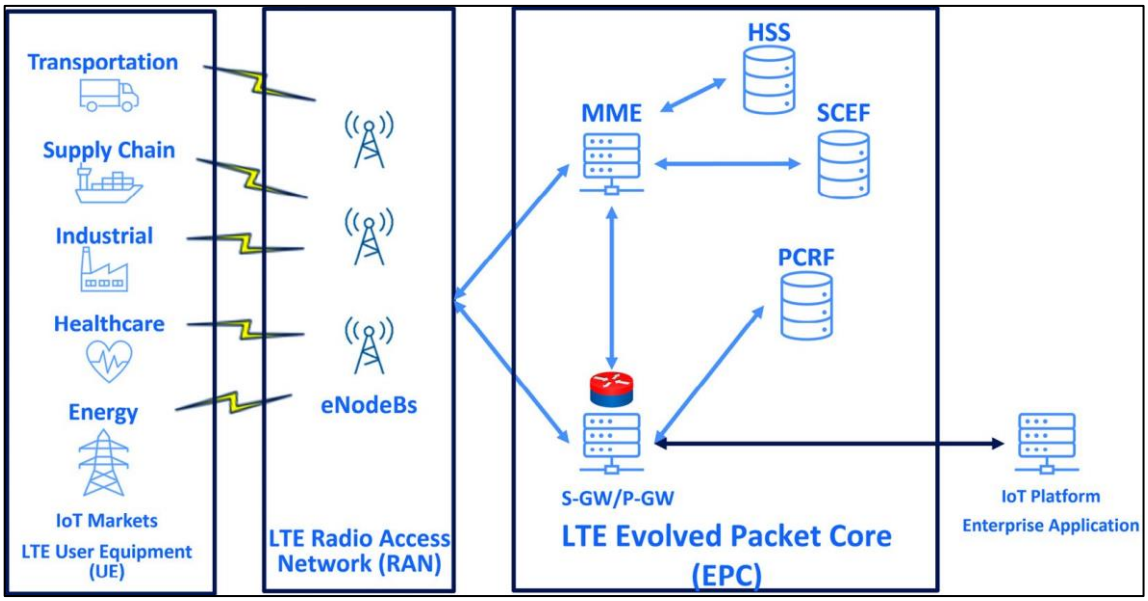

Figure 1.3: IOT Network Architecture.

In spite of this, the pervasiveness of connectivity also brings about difficulties that have never been seen before, notably in terms of security. There is a correlation between the proliferation of connected devices and the increase in the possible attack surface for hostile actors. Distributed denial of service assaults, sometimes known as DDoS attacks, are among the most stealthy dangers that exist for Internet of Things ecosystems. A distributed denial of service attack (DDoS) is an attempt to disrupt the normal functioning of a system or network that is the target of the assault by flooding it with a flood of malicious traffic and making it inaccessible to users who are authorized to use it. When it comes to traditional networks, distributed denial of service assaults have been a worry for decades. However, the proliferation of Internet of Things devices has multiplied the scale and impact of such attacks. These attacks not only pose a risk to the availability of Internet of Things services, but they also have wider-reaching ramifications for vital infrastructure, public safety, and economic stability. When it comes to identifying and mitigating distributed denial of service attacks, traditional solutions frequently rely on signature-based approaches, which are inadequate against a variety of unique and sophisticated attack vectors. In addition, the fact that many Internets of Things devices have limited capabilities places severe limitations on the amount of computational resources that can be used for standard security procedures. As a consequence of this, there is an urgent requirement for sophisticated, adaptable, and resource-efficient methods to identify and mitigate distributed denial of service assaults in Internet of Things networks. In order to overcome the issues that are provided by distributed

denial of service attacks in Internet of Things networks, Deep Learning (DL) has emerged as a potential paradigm. The capabilities of deep learning approaches, in particular Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and its derivatives, have been shown to be extraordinary in terms of their ability to extract intricate patterns and features from massive amounts of data. Deep learning models have demonstrated their effectiveness in a variety of fields, including image recognition, natural language processing, and cybersecurity, by utilizing the inherent hierarchical representations that are learned through deep neural networks. DL's application in DDoS detection offers a number of specific advantages that are worth considering. Without relying on predetermined rules or signatures, deep learning models are able to learn to recognize minor patterns and abnormalities that are suggestive of distributed denial of service attacks. This is in contrast to traditional signature-based techniques. In addition, deep learning models are easily scalable and adaptive, making them capable of tolerating a wide variety of Internet of Things environments as well as developing attack techniques. Further, techniques that are based on deep learning have the ability to reduce the number of false positives and false negatives, which will ultimately lead to an improvement in the accuracy and effectiveness of DDoS detection mechanisms.

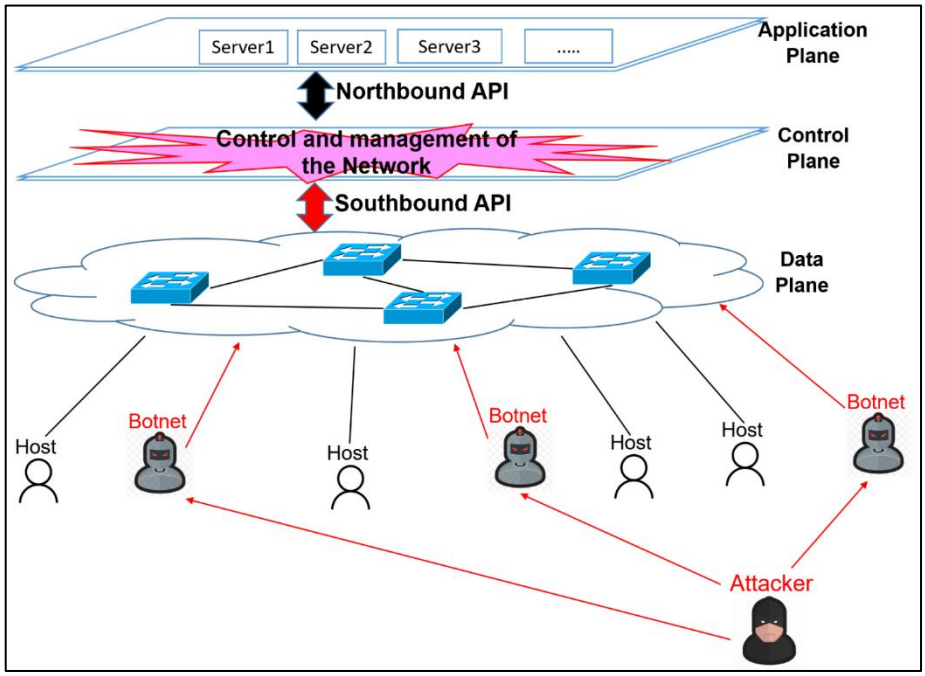

Figure 1.4: DDoS Attacks by Botnets

Despite the fact that DL-based DDoS detection in IoT networks may have the potential to be beneficial, there are still a number of difficulties that need to be addressed. When it comes to data collection, preprocessing, and model training, the inherent heterogeneity and diversity of Internet of Things ecosystems present opportunities for difficulties. Therefore, the development of lightweight deep learning architectures that are capable of operating efficiently inside confined computing contexts is required because of the resource constraints that are associated with Internet of Things devices. Furthermore, in order to successfully implement DDoS detection solutions in real-world Internet of Things deployments, it is essential to guarantee the robustness and resilience of reinforcement learning models against adversarial attacks. Taking into consideration these obstacles, the purpose of this thesis is to investigate and assess the practicability, effectiveness, and practical implications of employing deep learning technologies for the detection of distributed denial of service attacks in Internet of Things networks. By utilizing cutting-edge deep learning architectures, cutting-edge feature engineering techniques, and comprehensive experimental evaluations, the purpose of this research is to advance our understanding of distributed denial of service (DDoS) detection in Internet of Things (IoT) environments and to contribute to the development of robust and scalable cybersecurity solutions for the Internet of Things (IoT) paradigm.

1.2 PROBLEM STATAMENT

The proliferation of Internet of Things (IoT) devices has introduced unprecedented connectivity and convenience into various aspects of modern life. However, this proliferation has also brought about significant security challenges, particularly in mitigating the risks associated with Distributed Denial of Service (DDoS) attacks. DDoS attacks pose a severe threat to the availability, reliability, and integrity of IoT networks, disrupting critical services, compromising user privacy, and causing economic losses.

Traditional methods for detecting and mitigating DDoS attacks in IoT networks, such as signature-based intrusion detection systems, are often inadequate due to their reliance on predefined rules and patterns. As IoT environments continue to evolve and diversify, conventional approaches struggle to adapt to emerging attack vectors and sophisticated evasion techniques employed by malicious actors. Furthermore, the resource-constrained

nature of many IoT devices imposes limitations on the computational resources available for implementing robust security measures.

Given these challenges, there is a compelling need for innovative and adaptive approaches to detect and mitigate DDoS attacks in IoT networks effectively. Deep Learning (DL) technologies offer promising avenues for addressing this need, leveraging the inherent capabilities of neural networks to learn complex patterns and anomalies from vast volumes of data. By harnessing the power of DL models, it becomes possible to develop robust, scalable, and adaptive DDoS detection systems capable of effectively mitigating evolving threats in IoT environments.

However, despite the potential advantages offered by DL-based approaches, several critical research questions and challenges remain unresolved:

a. Scalability and Resource Efficiency: How can DL-based DDoS detection models be optimized to operate efficiently within the resource-constrained environments typical of IoT devices? What techniques can be employed to reduce computational complexity and memory footprint while maintaining detection accuracy?
b. Adaptability and Generalization: How can DL models be trained to adapt to the dynamic and heterogeneous nature of IoT environments? What strategies can be employed to ensure the generalization of DDoS detection models across diverse IoT deployments and attack scenarios?
c. Robustness and Security: How can DL-based DDoS detection systems be made resilient to adversarial attacks and evasion techniques employed by sophisticated adversaries? What mechanisms can be integrated into DL architectures to enhance their robustness and mitigate the risks of false positives and false negatives?
d. Data Collection and Labeling: What challenges are associated with collecting and labeling large-scale datasets for training DL-based DDoS detection models in IoT environments? How can biases and imbalances in training data be addressed to ensure the robustness and fairness of the resulting models?
e. Real-World Deployment: What practical considerations need to be taken into account when deploying DL-based DDoS detection systems in real-world IoT deployments? How can the scalability, reliability, and maintainability of such systems be ensured in operational environments?

Addressing these research questions and challenges is essential for advancing the state-of-the-art in DDoS detection for IoT networks and developing practical, effective solutions to safeguard IoT ecosystems against malicious attacks. This thesis seeks to explore these issues comprehensively through theoretical analysis, empirical evaluations, and practical implementations, with the ultimate goal of enhancing the security and resilience of IoT infrastructures in the face of evolving cyber threats.

1.3 THESIS OBJECTIVES

The primary objective of this thesis is to investigate and develop effective techniques for detecting Distributed Denial of Service (DDoS) attacks in Internet of Things (IoT) networks using Deep Learning (DL) technologies. To achieve this overarching goal, the following specific objectives have been delineated:

a. Investigate State-of-the-Art DL Architectures: Conduct a comprehensive review and analysis of state-of-the-art DL architectures, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and their variants, to identify suitable models for DDoS detection in IoT environments.

b. Optimize DL Models for Resource-Constrained IoT Devices: Explore techniques for optimizing DL models to operate efficiently within the resource-constrained environments typical of IoT devices. Investigate methods for reducing computational complexity, memory footprint, and energy consumption without sacrificing detection accuracy.

c. Enhance Adaptability and Generalization: Develop strategies to enhance the adaptability and generalization of DL-based DDoS detection models across diverse IoT deployments and attack scenarios. Investigate transfer learning, domain adaptation, and other techniques to improve model robustness and performance in real-world environments.

d. Mitigate Adversarial Attacks and Evasion Techniques: Investigate mechanisms for enhancing the robustness and security of DL-based DDoS detection systems against adversarial attacks and evasion techniques employed by malicious actors. Explore techniques for detecting and mitigating adversarial examples while minimizing the risk of false positives and false negatives.

e. Address Data Collection and Labeling Challenges: Examine challenges associated with collecting and labeling large-scale datasets for training DL-based DDoS detection models in IoT environments. Develop methods for mitigating biases and imbalances in training data to ensure the robustness and fairness of resulting models.
f. Evaluate Performance in Real-World Deployments: Conduct comprehensive empirical evaluations to assess the performance, scalability, and practical viability of DL-based DDoS detection systems in real-world IoT deployments. Evaluate detection accuracy, false positive rates, and computational efficiency under varying network conditions and attack scenarios.
g. Provide Guidelines for Practical Deployment: Develop guidelines and best practices for the practical deployment of DL-based DDoS detection systems in operational IoT environments. Address considerations related to scalability, reliability, maintainability, and integration with existing security infrastructure.

By pursuing these objectives, this thesis aims to contribute to the advancement of DDoS detection techniques for IoT networks, facilitate the adoption of DL technologies in cybersecurity applications, and enhance the security and resilience of IoT ecosystems against evolving cyber threats. Through theoretical analysis, empirical evaluations, and practical implementations, this research endeavors to provide insights and solutions that can inform the development of robust and effective cybersecurity strategies for IoT environments.

1.4 CONTRIBUTIONS

In this thesis, a new convolutional neural network (CNN) architecture is described for the purpose of detecting distributed denial of service attacks (DDoS) in Internet of Things (IoT) networks. As a component of the framework, an effective deep learning architecture for feature extraction from network traffic data is included. This design enables the framework to successfully classify traffic patterns as either benign or malicious. Since a great number of Internet of Things (IoT) devices have limited resources, the purpose of this study is to improve the CNN-based architecture so that it can function effectively in environments like these.

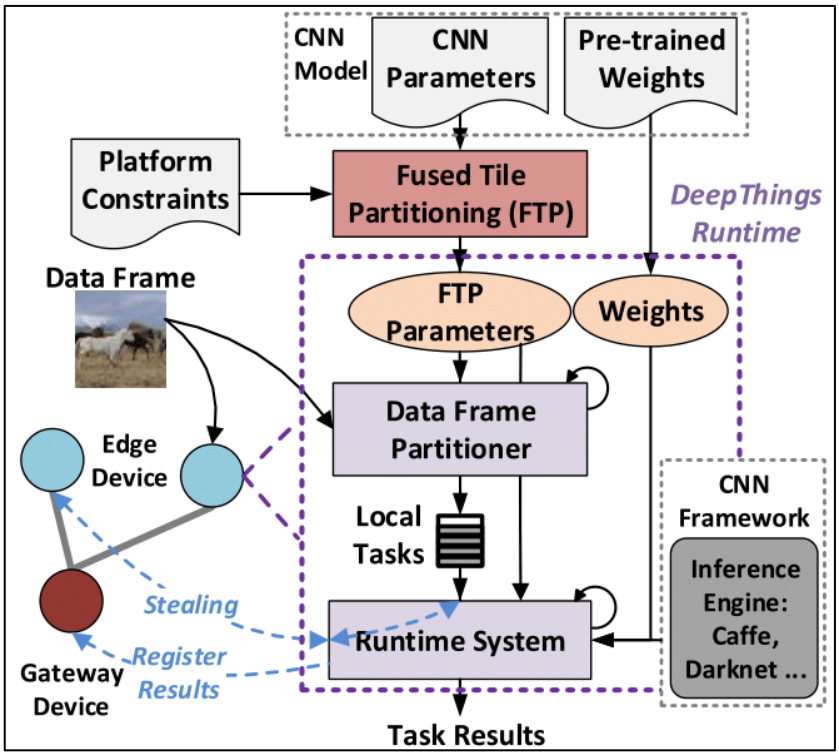

Figure 1.5: CNN-IOT Network Classification Architecture.

We are looking at ways to reduce the computational complexity and memory footprint of the system so that it may be utilized in real-world scenarios. This is done to ensure that it is compatible with the restricted processing power of Internet of Things devices. By drawing on the fundamental ideas of deep learning, this thesis investigates the various ways in which the CNN-based DDoS detection system can be made more adaptable and suitable to a variety of Internet of Things (IoT) installations and assault scenarios. Through the examination of transfer learning and domain adaptation approaches, the performance of the model as well as its resilience in a variety of settings that are constantly changing are improved. For the effectiveness and dependability of distributed denial of service (DDoS) detection systems, attacks by malicious actors are a serious concern. The purpose of this research is to construct mechanisms inside the CNN-based framework with the intention of reducing the impact of adversarial attacks and evasion tactics that are utilized by dangerous individuals. Methods for identifying and minimizing hostile cases are being investigated in order to make the model more resistant to aggressive strategies that are increasingly complicated. If you want to design algorithms for detecting distributed denial of service attacks, having access to datasets that have been labeled is critically necessary. The challenges of data collection and labeling in Internet of Things (IoT) environments are discussed in this thesis. Additionally,

the thesis offers suggestions for how to make training data less biased and more balanced in order to ensure that the CNN-based model is both fair and reliable. The CNN-based DDoS detection architecture that has been suggested is put through rigorous empirical tests in order to establish its performance, scalability, and feasibility. Real-world Internet of Things datasets and simulated attack scenarios are used to conduct comprehensive testing of the detection accuracy, false positive rates, and processing efficiency. These tests are carried out in a variety of environments. In this thesis, practical guidelines and best practices for implementing CNN-based DDoS detection systems in operational Internet of Things contexts are presented. These recommendations and practices are based on insights acquired from both theoretical and empirical considerations. In order to ensure that the process of deployment and adoption goes as smoothly as possible, we took into consideration a number of elements, including scalability, dependability, maintainability, and interaction with the existing security architecture. Using these advancements, the purpose of this research is to raise the bar for distributed denial of service (DDoS) detection for Internet of Things (IoT) networks, provide researchers and practitioners with practical insights into cybersecurity, and assist in the development of robust cybersecurity solutions for the Internet of Things (IoT) age. In the pursuit of advancing the field of DDoS detection in Internet of Things (IoT) networks utilizing the Convolutional Neural Network (CNN) algorithm, this thesis aims to make the following contributions:

a. Development of a CNN-Based DDoS Detection Framework: This thesis proposes a novel CNN-based framework specifically tailored for detecting DDoS attacks in IoT networks. The framework encompasses the design and implementation of a deep learning architecture optimized for extracting features from network traffic data, enabling effective discrimination between normal and malicious traffic patterns.
b. Optimization for Resource-Constrained IoT Devices: Given the resource-constrained nature of many IoT devices, this research focuses on optimizing the proposed CNN-based framework to operate efficiently within such environments. Techniques for reducing computational complexity and memory footprint are explored to ensure compatibility with the computational limitations of IoT devices, thereby enabling practical deployment in real-world scenarios.
c. Enhanced Adaptability and Generalization: Building upon the foundational principles of deep learning, this thesis investigates methods for enhancing the adaptability and

generalization of the CNN-based DDoS detection framework across diverse IoT deployments and attack scenarios. Transfer learning and domain adaptation techniques are explored to improve the robustness and performance of the model in dynamic and heterogeneous environments.

d. Mitigation of Adversarial Attacks: Adversarial attacks pose a significant threat to the reliability and efficacy of DDoS detection systems. This research endeavors to develop mechanisms within the CNN-based framework to mitigate the risks associated with adversarial attacks and evasion techniques employed by malicious actors. Techniques for detecting and mitigating adversarial examples are investigated to enhance the resilience of the model against sophisticated attack strategies.

e. Addressing Data Collection and Labeling Challenges: The availability of labeled datasets plays a crucial role in training robust and effective DDoS detection models. This thesis addresses the challenges associated with data collection and labeling in IoT environments, proposing strategies for mitigating biases and imbalances in training data to ensure the fairness and robustness of the CNN-based model.

f. Comprehensive Empirical Evaluations: Rigorous empirical evaluations are conducted to assess the performance, scalability, and practical viability of the proposed CNN-based DDoS detection framework. Extensive experiments are carried out using real-world IoT datasets and simulated attack scenarios to evaluate detection accuracy, false positive rates, and computational efficiency under varying conditions.

g. Guidelines for Practical Deployment: Drawing upon insights gained from theoretical analysis and empirical evaluations, this thesis provides practical guidelines and best practices for the deployment of CNN-based DDoS detection systems in operational IoT environments. Considerations related to scalability, reliability, maintainability, and integration with existing security infrastructure are addressed to facilitate seamless deployment and adoption.

Through these contributions, this research aims to advance the state-of-the-art in DDoS detection for IoT networks, provide actionable insights for cybersecurity practitioners and researchers, and contribute to the development of robust and effective cybersecurity solutions for the IoT era.

2. RELATED WORKS

2.1 INTRODUCTION

The global framework of a data or information society is supported by communication and information technologies that are interoperable with one another. The integration of advanced services, both virtual and real, is made feasible by these technologies, which also act as the framework's pillars of support. Through the utilization of radio frequency identification technology, it is able to establish links between a wide range of different items, things, and pieces of machinery. There are four key components that make up security, and they are as follows: availability, authenticity, integrity, and secrecy [1].

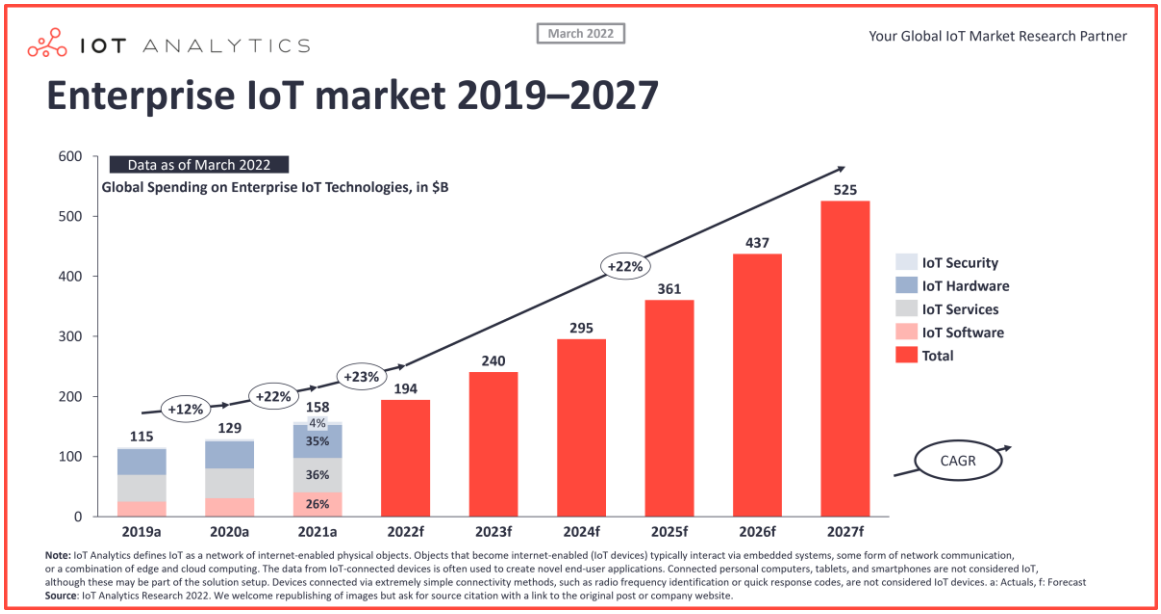

Figure 2.1: Global IOT Growth by 2027 [1].

In order to ensure that the credibility of the system is preserved, it is necessary to conduct a thorough investigation of the architecture. At each of the three levels—platform, middleware, and cloud systems—the application layer is the one that is accountable for providing support for its own security measures. On account of the fact that the distributed system is dependent on a wireless sensor network (WSN), distributed denial of service assaults, which are often referred to as DDoS attacks, are more likely to occur. Attacks that target the security mechanisms of wireless sensor networks (WSNs) and attacks that target their routing systems are the two basic kinds that may be distinguished when discussing distributed denial of service attacks [3]. Systems that are dependent on the internet of things (IoT) are particularly susceptible to distributed denial of service attacks, which have the

potential to cause severe harm. When a distributed denial of service attack is carried out, it is possible to prevent access to users who are authorized to use the system. Through the process of categorizing the assaults and ordering the responses that came before them in accordance with their classification, the author is able to gain a better knowledge of when and where flooding attacks occurred, as well as how to respond to and prevent them. In addition, it was suggested that it is vital to have a defense that is built on teamwork in addition to a coordinated distributed approach that is complete [5].

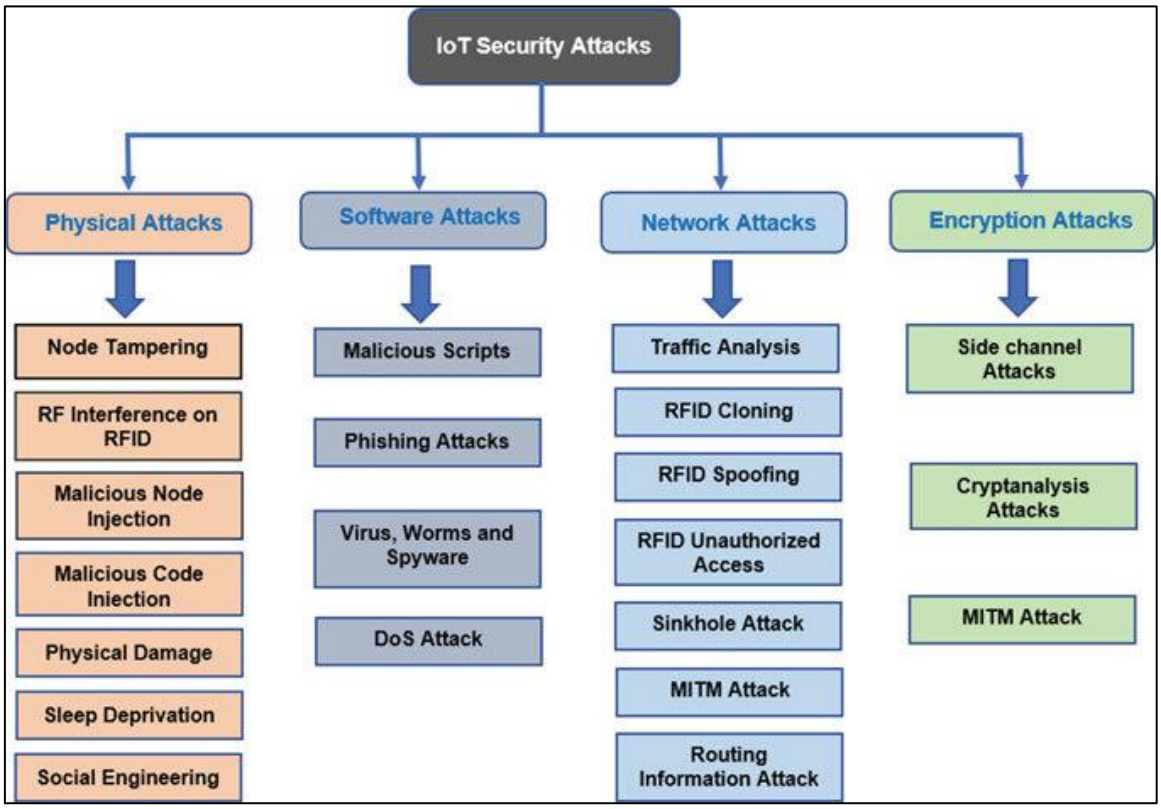

Figure 2.2: IOT Attacks Family Tree [5].

Over the course of this chapter, the security methods that are included into a distributed system are dissected systematically. The information that is shown here contains a variety of different authentication methods in addition to percentage observations of various distributed denial of service attacks. The primary purpose of this document is to provide an overview of the numerous attacks that can be carried out against distributed systems. This study does not go into detail on distributed denial of service assaults and distributed denial of service attacks because of the increasing severity and complexity of these types of attacks, which have a variety of effects on the system. This article will study a range of methods for identifying and preventing distributed denial of service attacks. These methods will be

investigated within the scope of this presentation. The majority of the focus in the investigation and prevention of distributed denial of service attacks is placed on traditional and tree-based approaches. Within the scope of this investigation, the proposed technology that is referred to as NAT was made available with the objective of detecting and preventing a wide range of distributed denial of service assaults.

2.2 LITERATURE SURVEY

The following types of attacks are included in the overview of distributed denial of service attacks that is presented in this chapter: tampering assaults, DNS amplification, jamming, collision, and TCP SYN flood. An early detection of flooding distributed denial of service attacks is possible with FireCol, which is a versatile solution that can detect such attacks. According to the findings of the investigation that was carried out by FireCol [6], there was nothing more than a small amount of computational and communication overhead. In situations where communication is carried out using a three-way handshake, it is not difficult to launch an assault against a TCP SYN flood on the target. By making use of vacant sections in both the HTTP and payload, an anomaly detection technique has the potential to uncover TCP SYN flood assaults that have already been implemented. This is achievable because of the utilization of vacant parts. It is because of this that the network is able to avoid being attacked. One of the components of this method for finding anomalies is a system that is responsible for monitoring and filtering packages. This system is one of the components. When it comes to identifying hostile attacks, this specific detection technique is not only effective but also speeds up the process [7]. For the purpose of providing extra aid in the detection and prevention of TCP SYN attacks, an adaptive thresholding method is implemented. This mechanism is utilized in order to provide additional support. The limitation that is linked with the static thresholding method will be controlled by the utilization of this methodology [24].

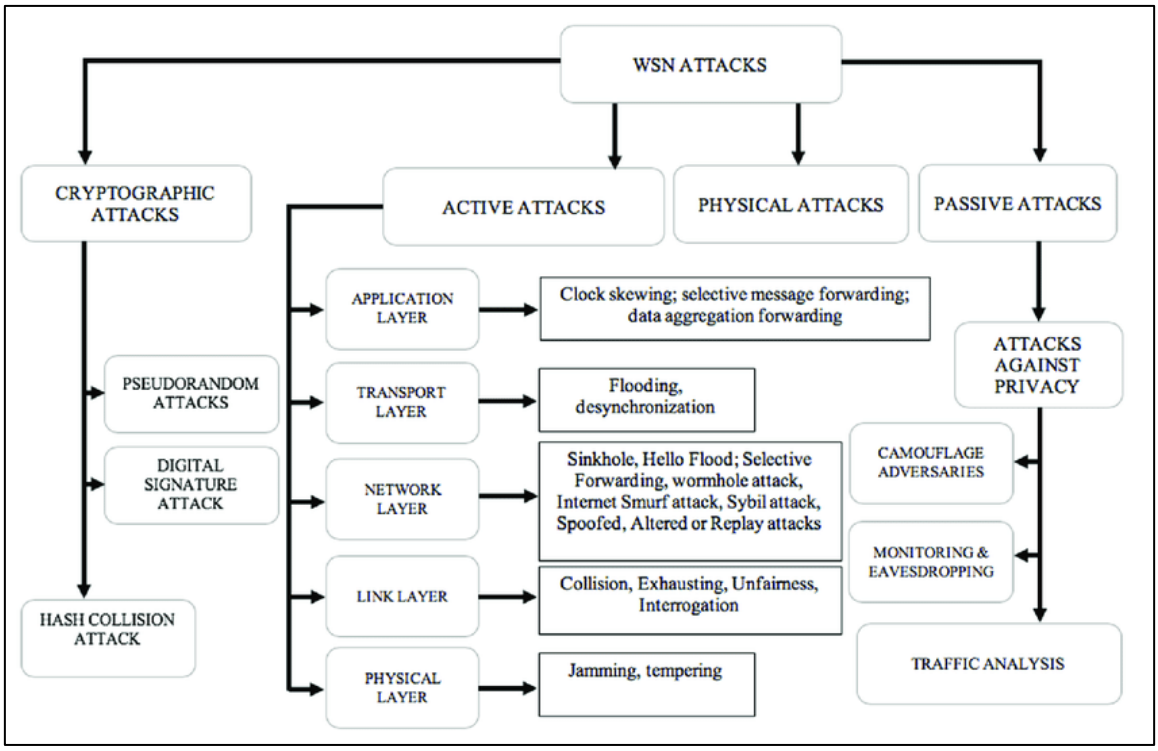

Figure 2.3: WSN Attacks [24].

M. Robbie argued for the development of security mechanisms for computer networks that are capable of detecting attacks such as ARP poisoning, SYN flooding, and ICMP redirective attacks. These measures were pushed for in an essay [8]. The study [9] presents a reliable approach for monitoring systems that are based on hardware. This is done with the intention of ensuring that the kernel is not corrupted in any manner. This section contains a detailed explanation of the architecture and implementation of the Address Translation Redirection Attack (ATRA), which enables a complete evasion of the external monitors that are based on hardware and rely on a range of CPUs. The authors provide this description in this part. It is an indication that the ping packet is not usual if the request that was received has a payload that is one thousand bytes in size. It is imperative that a quick response be provided at the very moment that the ping packet is received [10].They make the observation that DNSSEC replies that limit query rates are provided when a single source asks DNSSEC a substantial number of times [2011]. This is something that the authors have seen. An amplification attack is one in which the bandwidth of the agents that are attacking is exploited to the fullest extent possible. For each and every packet that a zombie gives, hacked machines will transmit larger or more packets to the victim's address. This is because zombies are able to send larger packets. When the number of DNS replies is more than the

number of DNS queries, this type of attack is known as a DNS amplification assault [12]. In order to identify malicious DNS responses, M. Ismail and colleagues proposed a method that involves collecting all flow data and then using that data to identify DNS amplification assaults. This method was proposed in order to identify malicious DNS responses. In collaboration with his colleagues, M. Ismail came up with this method. To identify assaults, this method classifies traffic as either normal or abnormal based on a specific value of flexible flow. This allows the system to correctly identify attacks. In the event that a DNS query does not contain a transaction ID, for instance, it is considered to be an attack [27]. CSMA/CN, which stands for carrier sense multiple access / collision notification, is a technique that is deployed for the goal of detecting and defending against collision attacks. CSMA/CN can be applied to terminate a transmission and deliver a message to the receiver that a collision has happened [13]. This can be done in the event that a transmission fails to occur. The optimal distinguisher is the basis for the stochastic collision strategy, which is intended to raise the percentage of successful model identification [14]. This approach is aimed to increase the likelihood of successful model identification.One way that can be deployed to protect against attempts to jam your transmissions is the sending of adaptive camouflage traffic, which is commonly referred to as TACT. Camouflage is a type of additional traffic that TACT uses in order to lessen the length of time that messages are delayed. This is accomplished by using camouflage. It has been demonstrated through testing that TACT has the capability to cut the probability of a message being sent late by a factor of two or three [15]. TJC, which is an abbreviation that stands for threshold-based jamming countermeasure, is applied in order to resist reactive jamming attacks that entail variable traffic intervals and a large number of suspect nodes. The effectiveness of this countermeasure is very remarkable. In the context of the algorithm's mobility simulation, the versatility of TJC is proven by the alteration of the arrangement of nodes inside the system by [16]. This modification demonstrates that TJC is also versatile. Wireless sensor networks (WSNs) that are equipped with Time Division Multiple Access (TDMA) technology are able to protect themselves from targeted jamming attacks by employing the JAMMY algorithm. This allows the WSNs to protect themselves from the attacks. At each superframe, JAMMY makes adjustments to the outline of the slot consumption, which makes it difficult for the opponent to anticipate what would occur. Due to the fact that sensor nodes manage the next slot use design in an independent and distributed manner, it is possible to define JAMMY as

a decentralized system. Due to the fact that there are a large number of nodes that are capable of communicating with the network, the JAMMY performance analysis reveals that there is almost no overhead [25].TamperProof is the name of the method that the authors proposed as a means of providing protection against attempts to tamper with the data. Through the utilization of this method, it is ensured that the parameters are not subject to any kind of modification. The strategy known as TamperProof is one that does away with the requirement that any alterations be made on the server side. As a result, it is an approach that is both very efficient and very successful. TamperProof is a method of online defense that assists in securing client-server communication in an environment that is trustworthy [17]. Client-server communication can be protected in this manner, for example. The method is now comprised of two parts, which are the transfer of data and the exchange of the initial seed. Both of these stages are interconnected. When the seed-sharing phase of the cryptographic process is being carried out, the PDC and PMU are responsible for the generation of random numbers for the time-hopping key setup configuration. During the phase of the procedure that is concerned with the transmission of data, a random time-hopping sequence is built by making use of the secret seed [18]. It is essential to design a tamper-aware authentication architecture in a wireless sensor network (WSN) in order to avoid any efforts at tampering and to ensure that all sensor nodes and packets belonging to the network are genuine [26].

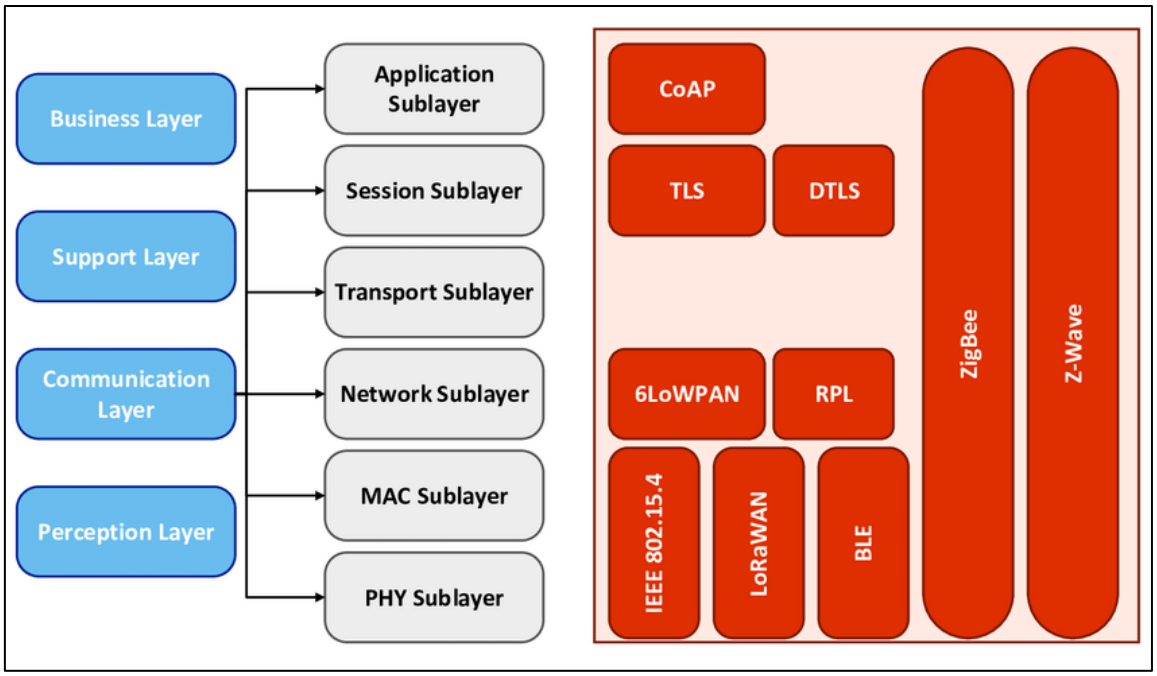

Figure 2.4: TCP Layers of IOT/WSN [26].

One of our primary goals is to create a brand new system that will be known as an updated Very Fast Decision Tree (EVFDT). This goal is one of our most important aims. Through the use of this, we will be able to enhance the precision of the attack characterization and tree measure of the system that is now in place. The utilization of a synthetic dataset that was generated through the execution of a LEACH convention is employed in order to assess the efficacy of the EVFDT method. To build a distributed denial of service attack, it is necessary to develop code. This is a prerequisite. With a low false alarm rate of 1.1% and a high arrangement exactness of 96.5%, the testing results reveal that the proposed method is capable of identifying an attack while needing minimal memory overhead [19]. This is demonstrated by the fact that the suggested method has a low false alarm rate. another method that makes use of trees as a resource The two components that make up the ATIDS system are the advanced threat tree analysis and the interruption detection and prevention component. The ATIDS system is divided into two separate pieces. ATIDS is the most well-known intrusion detection system that relies on assault tree guidelines and gives vulnerability evaluation for identifying [20]. While ADtT is the most well-known attack tree showing process for interruption identification, ATIDS is the most well-known intrusion detection system. The ability of both of these systems to detect interruptions is a well-known and widely appreciated feature. A bottom-up detection approach is provided by Augmented Attack Tree. This method takes into consideration three distinct types of attacks, namely UPD flood, ICMP flood, and TCP SYN flood. Distributed denial of service assaults (also known as DDoS) can be identified with the use of this technology. The theory behind this technique makes it possible to monitor the network traffic that the target server sends and receives to and from the internet [21]. This can be done either by sending or receiving traffic. The multiple attack characteristics that are produced by attack traffic are assembled into an attribute tree in order to improve the accuracy of the measurement of these characteristics. The purpose of this action is to enhance the precision of the attack detection process. Once a packet has been received, it is the responsibility of the attribute tree to determine whether or not the packet is legitimate. According to [22], the attribute tree, on the other hand, is not able to offer a comprehensive defense against attacks. The Change Aggregation Tree, also referred to as the CAT, is an aggregation approach that is built on the foundation of the distributed change-point detection architecture, which is also referred to as the DCD Architecture. Identifying anomalies in real time is the responsibility of the CAT tree, which

is accountable for this endeavor. When applied to a tree with fewer than 180 nodes, the CAT method demonstrates remarkable performance [23]. There are three stages that make up the Lightweight Decision-Tree model [28] that uses the C4.5 Algorithm. These stages are designed to identify distributed denial of service attacks that are dependent on floods. The characteristics of distributed denial of service assaults are extracted prior to the beginning of the process, which is known as pre-processing. The C4.5 algorithm is then trained in the second phase after the conclusion of the test and justification of the Decision-Tree method in the final phase. This occurs after the conclusion of the final phase.

Table 2.1: Summary Of Related Works

Reference Numbers	**Methods**	**Type of Attack**	**Advantages**	**Disadvantages**
[6]	Adaptation, anomaly detection mechanism	Flooding DDoS (specifically SYN)	- Light computational and communication overhead. - Effective and faster attack detection.	- Limited scalability for large-scale networks.
[8]	-ARP poisoning detection - SYN Flooding detection - ICMP redirect attacks detection	ARP poisoning, SYN Flooding, ICMP redirect	- Proposal for monitoring hardware-based system integrity. - Identification of abnormal ping packets.	- Dependency on hardware monitoring systems.
[11], [12], [27]	Flexible flow method, detection logic	DNS amplification	- Detection of DNS amplification attacks. - Categorization between normal and abnormal traffic.	- Potential false positives in anomaly detection.
[13]	CSMA/CN approach	Collision	- Detection and prevention of collision attacks.	- Limited effectiveness in high-traffic scenarios.
[15], [16], [25]	Transmitting adaptive camouflage traffic (TACT), TJC, JAMMY algorithm	Jamming, Selective jamming	- Minimization of message delay. - Reduction of message delivery failure chance. - Fine performance against jamming attacks.	- Susceptibility to advanced jamming techniques.

Table 2.1: Summary Of Related Works "Table Continued".

[17], [18], [26]	TamperProof approach, Tamper-aware authentication framework	Tampering	- Efficient mechanism for avoiding parameter tampering. - Online defense mechanism without server-side changes. - Authentication of sensor nodes and packets in WSN. - Mitigation of tampering attacks.	Overhead in maintaining authentication mechanisms.
[19]	Enhanced Very Fast Decision Tree (EVFDT)	DDoS	- High accuracy in attack classification. - Low false alarm rate. - Less memory overhead.	-Resource-intensive training process.
[20], [21], [22], [23]	Advanced attack tree, Augmented Attack Tree, Attribute Tree, Change Aggregation Tree	DDoS (UDP flood, ICMP flood, TCP SYN flood), Anomaly detection	-Bottom-updetection algorithm for DDoS attacks. - Logic for observing network traffic. - Real-time anomaly detection. - Efficiency for a moderate number of nodes. - Detection of attack traffic aspects.	-Complexity in configuring and maintaining detection models. - Inability to prevent all types of attacks. - Limited scalability for large-scale networks.
[28]	Lightweight Decision-Tree model based on C4.5 Algorithm	Flooding-based DDoS	- Detection of flooding-based DDoS attacks. - Multi-step detection process. - Utilization of C4.5 Algorithm.	- Dependency on labeled training data

2.3 CONCLUSION

The creation of distributed applications is carried out with the purpose of making the lives of the people who use them easier. In addition to the convenience that they provide, they also come with a number of drawbacks, such as a range of security flaws, attacks, and dangers that put the privacy of user data at risk. These drawbacks are in addition to the fact that they offer convenience. When it comes to a distributed system, the problem of protecting the privacy of data is one that is of the utmost significance. A distributed denial of service attack, more commonly referred to as a DDoS attack, is an example of a type of attack that has the potential to disrupt the operation of a single system while simultaneously having an

impact on all of the other systems that are connected to that system. There are a variety of alternative strategies that can be used in order to fight against distributed denial of service (DDoS) attacks; however, none of these strategies can ensure that every possible source of the attack will be eliminated in an appropriate manner. An overview of numerous tree-based and classical ways to detecting and defending against distributed denial of service (DDoS) assaults was the objective of this study, which was designed to provide the findings of the study. For example, TCP Syn flood, ICMP-based attacks, amplification, collision, jamming, and manipulation attacks are all included in this category of tactics. The purpose of this article is to provide an explanation of how to evaluate distributed denial of service (DDoS) threats by simultaneously employing both traditional and tree-based methodologies. The chapter discusses various methods and approaches proposed by different authors for detecting and mitigating different types of cyber attacks, particularly focused on DDoS attacks, tampering, jamming, and collision attacks. These methods include adaptive anomaly detection mechanisms like FireCol, detection of specific attacks like ARP poisoning and SYN flooding, as well as approaches for countering DNS amplification, collision attacks, and jamming attacks. Additionally, there are proposals for tamper-proof communication, authentication frameworks for sensor nodes, and advanced detection algorithms such as EVFDT and ATIDS. Each method has its advantages such as high accuracy and low false alarms, but they also come with limitations such as resource intensiveness, susceptibility to certain attack variants, and scalability issues. Overall, the text highlights a range of strategies and technologies aimed at enhancing network security and resilience against various cyber threats

3. MATERIALS AND METHODS

The figure 3.1illustrates an overview of IoT. IoT devices, at the lowest layer of the figure, interact with the physical environment in the roles of sensors and actuators. These devices are necessary for the correct functioning of IoT applications, represented in the highest layer of the figure. Communications between the various components illustrated in the figure take place thanks to the information technology infrastructure, available on the Internet through resources at the edge and core of the network.

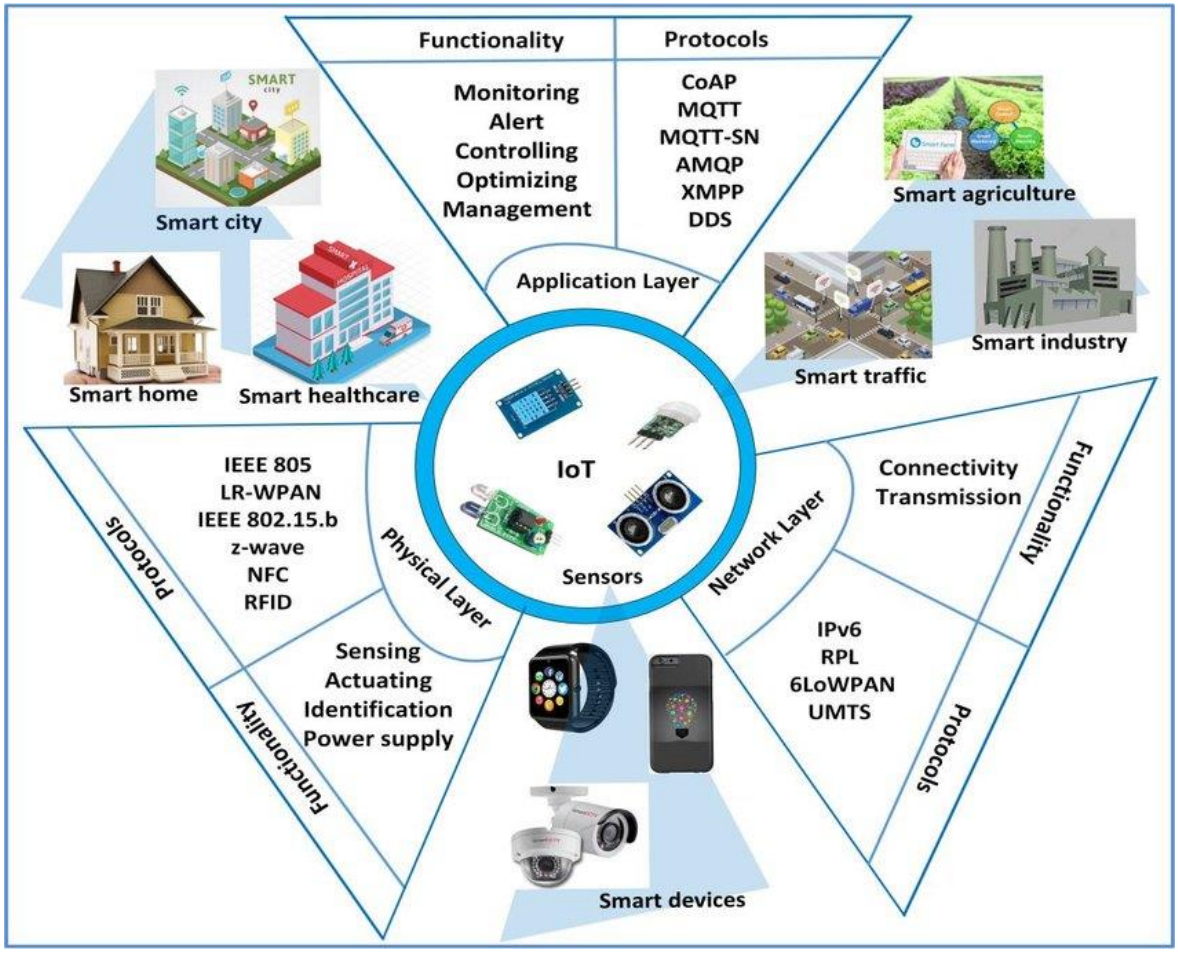

Figure 3.1: An Overview of IoT.

It is possible that the scenario illustrated in Figure 3.1 will comprise a variety of distinct network components. This is done in order to ensure that Internet of Things devices and the applications for which they are utilized in the real world are connected to one another simultaneously. Both physical components, such as access points, switches, routers, and network functions, such as domain name system (DNS – Domain Name System), protocol dynamic address configuration (DHCP – Dynamic Host Configuration Protocol), network

address translation (NAT – Network Address Translation), among others. The construction and operation of these network functions can be accomplished on a variety of platforms. This is all within the realm of possibility. One of these is virtualized, as demonstrated in Figure3.2. This design is based on the standardization of NFV designs (MANO, which is an acronym that stands for Management and Orchestration Architectural Framework) [23], which was specified by the European Telecommunications Standards Institute (ETSI, which is an acronym that stands for European Telecommunications Standards Institute). Real devices make advantage of the virtual network functions (VNF) services that are available at the services layer so that they can function at the physical layer. Virtual network functions (VNFs) are instantiated by the NFV infrastructure, which is located within the virtualization layer and is responsible for this responsibilities. Within the orchestration layer, the management of the virtualized network functions (VNFs) falls within the purview of both the virtualized infrastructure manager and the VNF manager. An ideal scenario would be one in which some of the virtual network functions (VNFs) provide security services and none of the VNFs have vulnerabilities. This would be the perfect scenario. This dissertation places a primary emphasis on virtual network functions (VNFs) that implement network security measures. Some examples of these mechanisms are firewalls, intrusion detection systems, and deep packet inspections. It is feasible that the scenario depicted in Figure 3.1 will include a number of different network components operating independently of one another. This is done in order to guarantee that the apps that are utilized in the real world by Internet of Things devices and the devices themselves are connected to one another at the same time. Both physical components, such as access points, switches, routers, and network functions, such as domain name system (DNS – Domain Name System), protocol dynamic address configuration (DHCP – Dynamic Host Configuration Protocol), network address translation (NAT – Network Address Translation), among others. There are many different platforms that can be utilized for the design and operation of these network operations. There is no doubt that all of this is within the possibilities. As can be seen in Figure3.2, one of these is a virtualized version. This design is based on the standardization of NFV designs (MANO, which is an acronym that stands for Management and Orchestration Architectural Framework) [23], which was specified by the European Telecommunications Standards Institute (ETSI, which is an acronym that stands for European Telecommunications Standards Institute). In order to function at the physical layer, real devices take advantage of

the virtual network functions (VNF) services that are available at the services layer. This allows them to function. The NFV infrastructure, which is situated within the virtualization layer and is accountable for these obligations, is the entity that is responsible for the instantiation of virtual network functions (VNFs). The administration of virtualized network functions (VNFs) is carried out by both the virtualized infrastructure manager and the VNF manager within the orchestration layer. This is because both of these managers are responsible for managing the VNFs. An ideal situation would be one in which some of the virtual network functions (VNFs) offer security services and none of the VNFs have vulnerabilities. This would be the optimal condition. This would be the ideal situation to be in there. Within the scope of this research, the virtual network functions (VNFs) that are responsible for implementing network security measures are given primary importance. Firewalls, intrusion detection systems, and deep packet inspections are certain instances of the mechanisms that fall within this category.

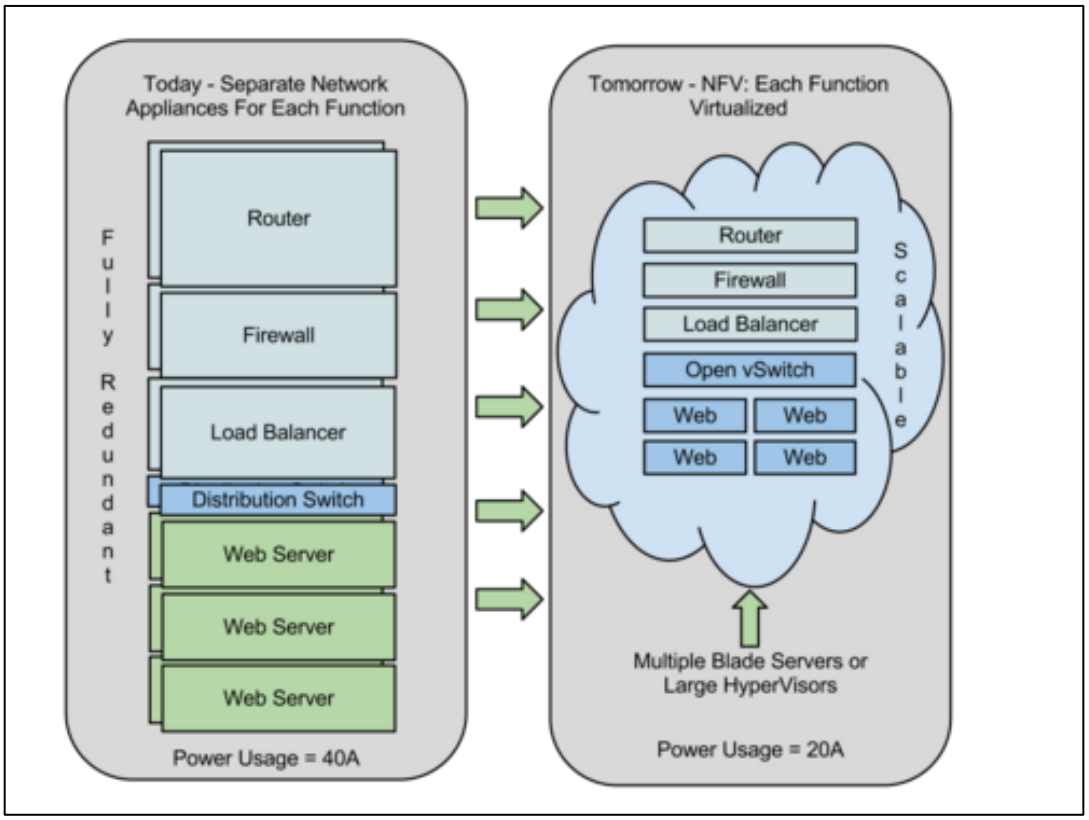

Figure 3.2: Overview Of Network Functions Virtualization in IoT.

The following sections present the basic concepts so that the components presented in Figure3.2are understood: Sec.3.1presents the definition and characteristics of NFVs,

relating them to the virtualization, services and orchestration layers. The Section3.2defines IoT and presents applications in this context, relating them to the physical layer. The Section3.3describes concepts present in the management of networks that use NFV, relating them to the orchestration layer; and Sect.3.4presents concepts that help understand the tradeoff between performance and security when using NFV to detect and mitigate security threats in IoT, relating the physical, virtualization, services and orchestration layers.

In order to ensure that the components shown in Figure 3.2 are clearly understood, the following sections will present the fundamental concepts: Within the context of the virtualization, services, and orchestration layers, the definition and features of network functions virtualizations (NFVs) are presented in Section 3.1. The Internet of Things (IoT) is defined in Section 3.2, which also discusses applications in this context and relates them to the physical layer. Both Section 3.3 and Section 3.4 present concepts that help understand the tradeoff between performance and security when using NFV to detect and mitigate security threats in the Internet of Things. Section 3.3 describes concepts that are present in the management of networks that use NFV and establishes a connection between them and the orchestration layer. Section 3.4 presents concepts that relate the physical, virtualization, services, and orchestration layers.

3.1 NETWORK FUNCTIONS VIRTUALIZATION (NFV)

Network functions such as packet filtering, address translation, and load balancing are essential in many computer networks. Even on a network

small-sized residential home, in which local users act as clients, accessing external servers located on the Internet, the routers provided by the Internet provider usually come with some of these functions built in from the factory. In a large network, such as those on university campuses, network functions are often implemented on dedicated hardware and are configured to form a chain of network functions or function chains (SFCs). An example of SFC applied to a flow of packets entering a network to access a web service is illustrated by the interconnected devices in Figure3.3. The network functions performed by the devices in the figure and their objectives are:

a. Packet Filtering to check whether the characteristics of the flow (source IP address, destination IP address, source port, destination port, protocols, etc.) match any rule

that represents a potential attack. If it does, the packets are discarded. If it does not represent, they move on to the next network function;

b. Address translation to forward the request packets to an internal network with non-routable IP addresses on the Internet, where the web service will be provided by a set of servers;

c. Load balancing to distribute requests among the set of web servers, aiming to maintain ideal response time and reducing the chance of success of a denial of service attack.

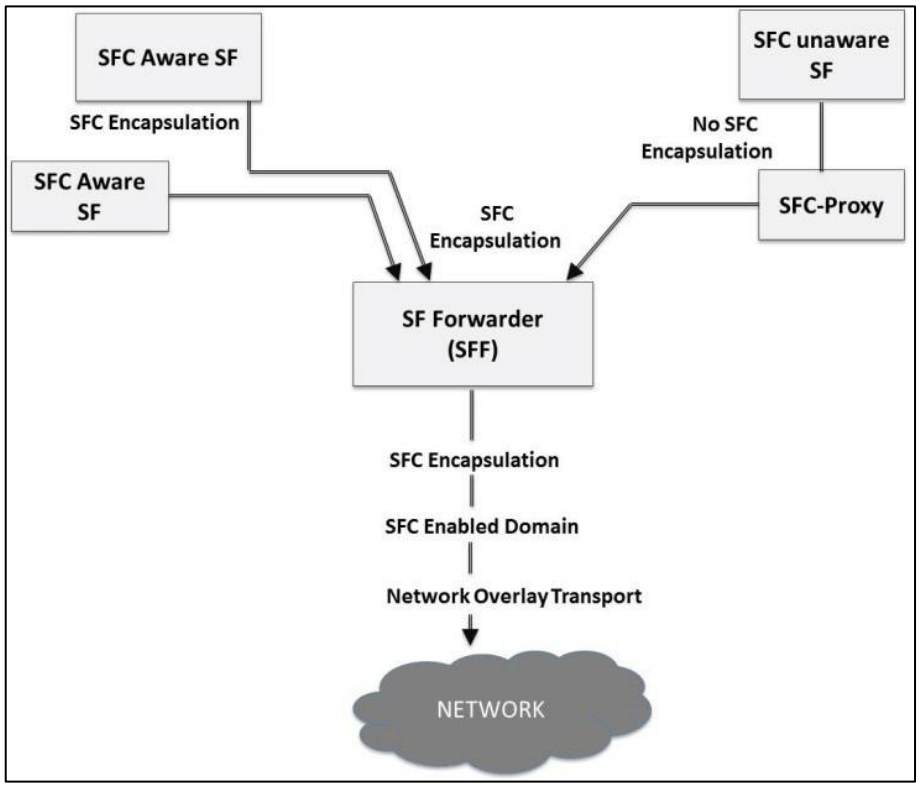

Figure 3.3: Example of A Chain of Functions in a Traditional Network.

A network like the one illustrated in Figure3.3, in which network functions are provided by dedicated hardware equipment known as appliances or middleboxes, will be called a traditional network throughout this dissertation. In this network, if the operator has to add a new function, such as an intrusion detection system (IDS), for example, after the firewall (packet filter), he will need to purchase the specific appliance and carry out its physical installation and configuration on the local network, considering possible changes to routes on existing appliances.

NFV allows the creation of a network architecture that makes the process of acquiring and reconfiguring network functions more flexible through the virtualization of these functions.

Unlike a traditional network, in an NFV-based network, VNFs are provided as software and instantiated on general-purpose hardware. In this case, the operator could contract an IDS service, which would be implemented completely in software, and instantiate it on a server located in a cloud provider's data center or on computers in the fog or at the edge. Possible changes to the configurations of existing appliances could be necessary, as in the case of the traditional network, but all actions related to the acquisition of specific equipment and its physical installation on the local network become unnecessary. In this case of the NFV-based network, if the operator decides to start using another IDS implementation, it would be enough to change the software instantiated in the cloud/fog/edge, without the need to purchase new hardware. Furthermore, at peak times, when the IDS needed to perform memory and CPU intensive tasks, it would be enough to ask the provider to expand these resources or migrate the VNF to a resource with greater capacity. These facilities for changing the implementation of VNFs and their capacity make NFV ideal for heterogeneous environments, with constant changes and limited resources, such as the IoT [24]

The figure3.4illustrates a possible implementation of the SFC in Figure3.3in an NFV architecture. In this case, the packet flow reaches the first network function, the firewall, which is instantiated in the cloud, fog, or edge. The flow goes through the other functions, also instantiated in the cloud/fog/edge, until it reaches the web servers. It is worth noting that in this example, the packet filter and the address translator are sharing the same physical machine, but this configuration does not need to be static. At a time when network traffic is relatively high, instantiations can be done on separate machines so that each one can take advantage of all the physical resources of each server.

Sharing the same physical server would be interesting in times of low traffic, allowing fewer servers to remain connected, leading to savings in terms of electricity. Furthermore, it is not necessary for all functions to be virtualized. It would be possible, for example, to instantiate just the firewall.

For NFV to function correctly, several components must interact, which can be grouped into layers, as shown in Figure3.3. VNFs need to be implemented as software (containers or virtual machines, for example) and maintained in some repository, making up the services layer. The services layer can also provide entire SFCs.

Whenever a VNF is requested, it must be instantiated on some physical machine. The NFV infrastructure (NFVi – Network Functions Virtualization infrastructure), The term

"traditional network" will be used throughout this dissertation to refer to a network that is just like the one shown in Figure3.3. In this type of network, the services of the network are provided by specialized hardware equipment that is referred to as appliances or middleboxes. If the operator of this network needs to add a new function, such as an intrusion detection system (IDS), for instance, after the firewall (packet filter), he will be required to purchase the specific appliance, carry out its physical installation and configuration on the local network, and take into consideration the possibility of making changes to routes on existing appliances.

Virtualization of network functions is made possible by network function virtualization (NFV), which enables the establishment of a network architecture that provides for greater flexibility in the process of acquiring and reconfiguring network functions. In an NFV-based network, virtual network functions (VNFs) are provided in the form of software and are instantiated on general-purpose hardware, in contrast to a traditional network. In this scenario, the operator may contract an intrusion detection system (IDS) service, which would be entirely implemented in software, and then instantiate it on a server that is situated in the data center of a cloud provider, or on machines that are positioned in the fog or at the edge of the network. In the case of the traditional network, it may be necessary to make modifications to the configurations of the appliances that are already in use; nevertheless, all of the steps that are associated with the acquisition of particular equipment and its physical installation on the local network are rendered unnecessary. In this particular instance of the network that is based on NFV, if the operator would like to begin utilizing a different IDS implementation, it would be sufficient to modify the software that is instantiated in the cloud, fog, or edge, and there would be no requirement to acquire new hardware. Furthermore, when the IDS needed to execute memory and CPU intensive operations during peak times, it would be sufficient to ask the provider to enlarge these resources or migrate the VNF to a resource with greater capacity. This would be the case even if the IDS was unable to do these tasks. Because of these options for altering the implementation of virtual network functions (VNFs) and their capacity, NFV is excellent for heterogeneous environments, such as the Internet of Things (IoT), which are characterized by constant changes and limited resources [24]. A hypothetical implementation of the SFC shown in Figure 3.3 is depicted in figure 3.4, which is an illustration of an NFV architecture. When this occurs, the packet flow arrives at the initial network function, which is the firewall. This firewall can be instantiated in the

cloud, fog, or edge in this scenario. Before arriving at the web servers, the flow passes via the other functions, which are likewise instantiated in the cloud, fog, and edge at the same time. It is important to note that in this particular illustration, the packet filter and the address translator are both utilizing the same physical machine. However, it is not necessary for this configuration to retain its static state. During times when there is a significant amount of network traffic, it is possible to perform instantiations on distinct machines. This allows each machine to make use of all of the physical resources that are currently available on each server.

Sharing a single physical server could be an intriguing option during periods of low traffic. This would enable a smaller number of servers to remain connected, which would result in cost savings in terms of electricity. Furthermore, it is not essential that all functions be virtualized. Virtualization is not required. As an illustration, it would be feasible to instantiate the firewall on its own. It is necessary for NFV to have a number of components that interact with one another. These components can be categorized into layers, as depicted in Figure 3.3. The services layer is comprised of virtual network functions (VNFs), which must be implemented as software (whether it be containers or virtual machines, for example) and managed in a repository. Additionally, the services layer is able to supply complete SFCs.

Whenever a Virtual Network Function (VNF) is requested, it is necessary to instantiate it on a real system. NFVi, which stands for Network Functions Virtualization infrastructure, is the NFV infrastructure infrastructure.

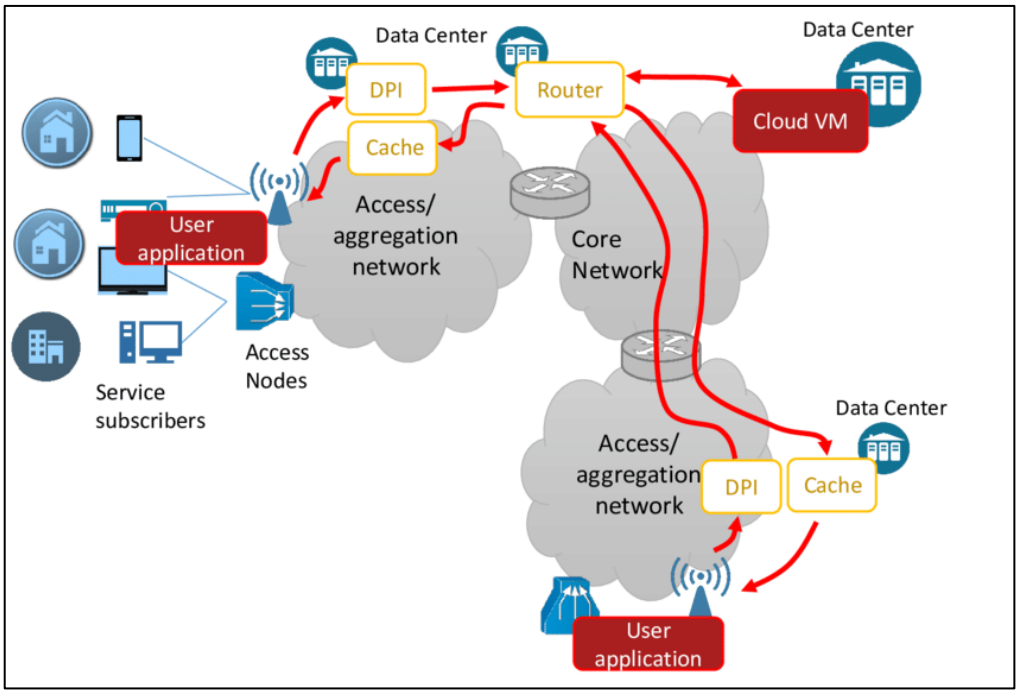

Figure 3.4: Example of a Chain of Network Functions With NFV.

in the virtualization layer, it is responsible for abstracting the physical infrastructure resources, which will possibly be in one or more providers, allowing equipment from different manufacturers to offer the physical resources necessary for the instantiation of VNFs. NFVi also takes care of the interconnection of VNFs with each other and with the outside world, allowing interaction with the Internet. To ensure that infrastructure resources are allocated in the best possible way, it is necessary to manage the lifecycle of each of the VNFs.

The VNF manager in the orchestration layer has the responsibility of taking care of this lifecycle, instantiating, updating and terminating VNFs. Still at the orchestration layer, the virtualized infrastructure manager has the role of maintaining control over the mapping of physical resources to virtual resources.

The term "traditional network" will be used throughout this dissertation to refer to a network just like the one shown in Figure3.3, in which the services of the network are provided by specialized hardware equipment that is referred to as appliances or middleboxes. If the operator of this network needs to add a new function, such as an intrusion detection system (IDS), for instance, after the firewall (packet filter), he will be required to purchase the specific appliance, carry out its physical installation and configuration on the local network, and take into consideration the possibility of making changes to routes on existing appliances

within the network. The design of a network architecture that allows for the virtualization of network functions is made possible by network function virtualization (NFV). This architecture allows for greater flexibility in the process of acquiring and reconfiguring network functions. In contrast to a conventional network, a network that is based on NFV provides virtual network functions (VNFs) in the form of software and instantiates them on general-purpose hardware. An intrusion detection system (IDS) service might be contracted by the operator in this scenario. The IDS service would be entirely implemented in software, and it could be instantiated on a server that is situated in the data center of a cloud provider, or on machines that are positioned in the fog or at the edge. All actions associated to the acquisition of specific equipment and its physical installation on the local network become redundant. However, it is possible that adjustments to the configurations of existing appliances will be required, just as they were in the case of the traditional network. It would be sufficient to update the software that is instantiated in the cloud, fog, or edge in this particular instance of the NFV-based network in order to begin using a different IDS implementation. This would eliminate the requirement for the operator to purchase new hardware. In addition, during peak times, when the intrusion detection system (IDS) needed to conduct tasks that required a significant amount of memory and CPU, it would be sufficient to request that the provider either enlarge these resources or migrate the virtual network function (VNF) to a resource that had a higher capacity. A heterogeneous environment, such as the Internet of Things (IoT), with constant changes and limited resources is an appropriate environment for network function virtualization (NFV) because of the capabilities for altering the implementation of virtual network functions (VNFs) and their capacity. Is it 24? In an NFV architecture, the SFC shown in Figure 3.3 can be implemented in a variety of ways, as shown in Figure 3.4. In this scenario, the packet flow arrives at the initial network function, which is the firewall. The firewall can be instantiated in the cloud, through fog, or at the edge. When it finally reaches the web servers, the flow will have passed through the other functions that were also instantiated in the cloud, fog, and edge. It is not necessary for this setup to be static; nonetheless, it is important to note that in this particular illustration, the packet filter and the address translator are both utilizing the same physical machine. Instantiations can be carried out on distinct machines during times when there is a relatively high volume of network traffic. This allows each machine to make use of all of the physical resources that are available on each server by itself.

During periods of low traffic, it would be fascinating to share the same physical server. This would allow for a smaller number of servers to remain connected, which would result in increased savings on electricity. Additionally, it is not essential that all functions be virtualized in order to complete the process. You might, for instance, instantiate the firewall by yourself. This would be possible. In order for NFV to operate properly, it is necessary for a number of components to interact with one another. These components can be categorized into layers, as depicted in Figure 3.3. In order to constitute the services layer, virtual network functions (VNFs) must be built as software (for instance, containers or virtual machines) and saved in a repository. In addition, the services layer is able to supply complete SFC information.

It is necessary to instantiate a virtual network function (VNF) on a physical system whenever a VNF is requested. On the other hand, the NFV infrastructure (NFVi, which stands for Network Functions Virtualization infrastructure)

3.2 IOT NETWORKS AND APPLICATIONS

IoT is a communication paradigm with the aim of connecting different types of objects to the Internet [25] These objects include, for example, watches, vehicles, buildings, sensors and actuators in general. Adding IoT support to an object consists of including resources such as battery, memory, processor, and wireless network interface. These resources are used to execute IoT protocols and applications. With this support, it is common for objects to be called “smart”. This adjective is also used to describe applications running on objects. In the scenario in Figure3.2, smart objects are located in the physical layer.

IoT is not a separate network from the Internet, but rather an extension of it that allows the connection of different types of everyday objects. Unlike a conventional computer, be it a personal computer or a dedicated server, it is common for an IoT device to be exposed to critical environments for electronic equipment, with high temperatures and humidity, for example. Depending on the environments and applications that will be run on the objects, they have restrictions on physical dimensions. The limitations imposed by the environment and the physical dimension prevent IoT devices from having high computational power, requiring preference to be given to the execution of lightweight protocols and applications, which guarantee the quality of user experience without high consumption of the device's resources. , mainly the battery.

The figure3.5illustrates the typical three-tier IoT architecture [27] The perception layer is at the bottom of the architecture. It is responsible for interacting with the devices and physical components of smart objects, obtaining data and sending commands. The network layer integrates various interconnection devices, such as switches and access points, with various communication technologies, such as Bluetooth and Wi-Fi. In this layer, communications routes are determined. The application layer is the layer at the top of the architecture and is where applications are properly implemented. Smart city applications, smart industries and smart homes are in this layer. Ideally, these applications use lightweight Internet Architecture application layer protocols, such as MQTT, CoAP and AMQP [28]. The Internet of Things (IoT) is a communication paradigm that serves the purpose of linking various kinds of things to the Internet [25]. For instance, watches, automobiles, buildings, sensors, and actuators in general are all examples of entities that fall under this category. A battery, memory, CPU, and wireless network interface are some of the resources that must be included in order to give an object support for the Internet of Things (IoT). Both Internet of Things protocols and applications are executed with the help of these resources. It is standard practice to refer to things as "smart" when they have this form of help. It is also possible to use this word to describe programs that are operating on external objects. When it comes to the scenario depicted in Figure3.2, the physical layer is where smart items are situated. The Internet of Things (IoT) is not a distinct network from the Internet; rather, it is an extension of the Internet infrastructure that enables the connectivity of various kinds of commonplace objects. When compared to a traditional computer, whether it be a personal computer or a dedicated server, it is not uncommon for an Internet of Things device to be subjected to settings that are considered to be crucial for electronic equipment. These environments may include high temperatures and humidity, for instance. There are limitations placed on the physical dimensions of the things, and these limitations are determined by the environments and applications that will reside on the objects. The limitations that are imposed by the environment and the physical dimension prevent Internet of Things devices from having a high computational power. As a result, it is necessary to give preference to the execution of lightweight protocols and applications. These protocols and applications guarantee the quality of the user experience without consuming a significant amount of the device's resources, most notably the battery. Figure 3.5 depicts the conventional architecture of the Internet of Things comprising three tiers [27]. The

architecture is built with the perception layer located at the very bottom. The transmission of commands, the acquisition of data, and the interaction with the devices and physical components of smart objects are all the responsibilities of this component. The network layer is responsible for integrating a wide range of communication technologies, including Bluetooth and Wi-Fi, with a variety of connecting devices, such as switches and access points. Decisions regarding the routes of communications are made in this layer. The application layer is the layer that sits at the very top of the architecture, and it is the layer that is responsible for the correct implementation of applications. The applications for smart cities, smart industries, and smart homes are all contained inside this layer. Applications like MQTT, CoAP, and AMQP are examples of lightweight Internet Architecture application layer protocols that should be used for these applications.

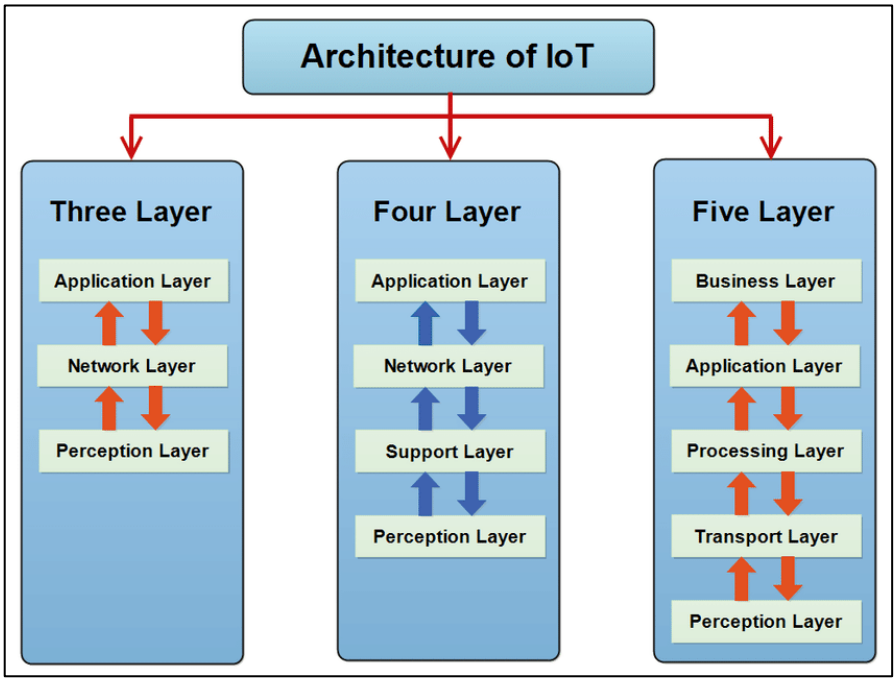

Figure 3.5: IoT Layered Architecture.

The limitation of resources of devices connected to the IoT, coupled with the devices' default configurations, which do not take into account some basic security aspects such as encryption and the requirement for strong passwords [29] have made IoT the target of security attacks. Part of these attacks are carried out to obtain remote control of devices, which become part of so-called botnets, networks created to launch large-scale attacks against the Internet [30]

3.3 NETWORK MANAGEMENT WITH NFV

The absence of devices with abundant resources for performing security-related network functions makes the NFV architecture interesting for providing security in IoT. In this case, security VNFs can be instantiated on external devices, provided by cloud, fog, or edge providers. To make this possible, the components of the orchestration layer (Figure3.2) play a fundamental role [31].

The virtualized infrastructure manager is present in each of the virtual infrastructure domains. In the case of an infrastructure maintained by the OpenStack platform[1], for example, the manager used is neutron. A virtualized infrastructure manager like Neutron is responsible for taking care of the addressing between virtualized devices, the virtual topology, and the potential connection between different infrastructures through virtual private networks (VPNs - Virtual Private Networks), delivering “network as a service ”.

The VNF manager is responsible for managing the lifecycle of VNFs, such as instantiation, update, configuration change, vertical scaling (change in the amount of physical resources made available by a host computer for a network function), horizontal scaling (change in the number of host computers used to instantiate a network function) and termination.

3.4 PERFORMANCE VS. SECURITY ISSUES WITH NFV IN IOT

A factor to be considered when using any computational resource is its performance. This is characterized by two factors: metrics and measurements. Metrics are standard definitions of quantities produced in an experiment, with an intended utility. They must be carefully specified to convey the exact meaning of a measured value. Measurement refers to a set of operations with the objective of determining the value of a metric [32].

In this way, computational performance indicators are directly related to resource metrics that demonstrate the degree of functionality of a computational system and, therefore, represent in an abstract way the state of that system. For example, the time it takes for a web-based application to load is the most noticeable indicator that the system is performing at the expected level. Therefore, longer than normal loading times indicate that the system may have problems, requiring action from its administrators. These indicators change over time and are initial sources of information about the “health” of the system [33]

From an NFV perspective, the Table3.1presents performance metrics related to speed, accuracy and reliability according to categories relating to orchestration, virtual machine operation, establishment and operation of networks and technology components as a service. These metrics were defined by ETSI [34] for entities that provide VNFs and manage virtualized infrastructures, in order to ensure that the resources offered have the necessary performance in accordance with the expected quality requirements. For example, a network that is highly demanded by benign requests, that is, that are not attacks, and has high virtual machine (VM – Virtual Machine) provisioning latency for the virtualized firewall service, may generate long delays in responses to requests, which directly affects the quality of experience for the end user of the service. Performance is an aspect that must be taken into consideration whenever any computational resource is being utilized. Metrics and measurements are the two aspects that define this phenomenon. To put it another way, metrics are standardized definitions of quantities that are produced in an experiment and have a specific purpose. For the purpose of conveying the precise meaning of a measured value, they need to be specified with great care. A collection of procedures that are carried out with the intention of establishing the value of a metric is referred to as measurement [32]. The degree of functionality of a computational system is demonstrated by resource metrics, which, in turn, indicate the state of that system in an abstract manner. In this manner, computational performance indicators are directly tied to resource metrics. For instance, the amount of time it takes for a web-based application to load is the most obvious indication that the system is performing at the level that was anticipated. Therefore, loading times that are significantly longer than average are an indication that the system may be experiencing issues, which calls for the administrators to take action. Over the course of time, these indicators undergo changes, and they serve as primary sources of information regarding the "health" of the system [33]. Table 3.1 offers performance measures related to speed, accuracy, and reliability according to categories relating to orchestration, virtual machine operation, installation and operation of networks, and technological components as a service. Additionally, the table presents these metrics from the perspective of network function virtualization (NFV). ETSI [34] established these measures for entities that supply virtual network functions (VNFs) and manage virtualized infrastructures. The purpose of these metrics is to guarantee that the resources that are provided have the required level of performance in compliance with the quality standards that are anticipated.

As an illustration, a network that is highly demanded by requests that are not malicious, that is, requests that are not attacks, and that has a high virtual machine (VM – Virtual Machine) provisioning latency for the virtualized firewall service, may generate long delays in responses to requests, which directly affects the quality of experience for the end user of the service.

Table 3.1: Summary of Quality-Of-Service Metrics in NFV.

Metric Category	**Speed**	**Accuracy**	**Reliability**
First stage of orchestration (e.g. resource allocation, configuration and installation)	Provisional latency VM launch	Position policy VM naming and conformity	Reliability of provisioning of VM
VM operation	VM Interruption (event duration and frequency) and VM Setup Latency	VM clock error	Premature VM release rate
Virtual network establishment (VN – *VirtualNetwork*)	VN Provisioning Latency	VN diversity compliance	Reliability of provisioning by VN
Virtual Network Operation	Packet delay, jitter (variation in delay) and throughput of delivered packets	Loss rate packages	Connection interrupted
Second stage of orchestration (for example, release of resources)	–	–	Failed VM release rate
Technology-as-a-Service Component (TcaaS)	TcaaS service latency	–	Reliability TcaaS (e.g. relationship faulty transaction) and TcaaS outage

In the same vein as ETSI, the Internet Engineering Task Force (IETF) designated a specific group, called the Group of Benchmarking Methodology Work (BMWG – Benchmarking Methodology Working Group), to produce a series of recommendations on the main characteristics and performance analysis of devices, systems and network services. Thus, the Benchmarking Methodology for Network Virtualization Performance was developed, considering performance metrics related to the virtualized resource, such as CPU and RAM consumption, and network-related metrics, such as latency, transfer rate and data loss rate. pictures. Performance is an aspect that must be taken into consideration whenever any computational resource is being utilized. Metrics and measurements are the two aspects that define this phenomenon. To put it another way, metrics are standardized definitions of quantities that are produced in an experiment and have a specific purpose. For the purpose of conveying the precise meaning of a measured value, they need to be specified with great care. A collection of procedures that are carried out with the intention of establishing the value of a metric is referred to as measurement [32]. The degree of functionality of a computational system is demonstrated by resource metrics, which, in turn, indicate the state of that system in an abstract manner. In this manner, computational performance indicators are directly tied to resource metrics. For instance, the amount of time it takes for a web-based application to load is the most obvious indication that the system is performing at the level that was anticipated. Therefore, loading times that are significantly longer than average are an indication that the system may be experiencing issues, which calls for the administrators to take action. Over the course of time, these indicators undergo changes, and they serve as primary sources of information regarding the "health" of the system [33]. Table 3.1 offers performance measures related to speed, accuracy, and reliability according to categories relating to orchestration, virtual machine operation, installation and operation of networks, and technological components as a service. Additionally, the table presents these metrics from the perspective of network function virtualization (NFV). ETSI [34] established these measures for entities that supply virtual network functions (VNFs) and manage virtualized infrastructures. The purpose of these metrics is to guarantee that the resources that are provided have the required level of performance in compliance with the quality standards that are anticipated. As an illustration, a network that is highly demanded by requests that are not malicious, that is, requests that are not attacks, and that has a high virtual machine (VM – Virtual Machine) provisioning latency for the virtualized firewall

service, may generate long delays in responses to requests, which directly affects the quality of experience for the end user of the service. It is also possible to find metrics related to NFV in other works. [35] for example, the authors address issues related to NFV fault tolerance and present specific metrics for the problem, such as: packet loss, average packet transfer rate, congestion at interconnection points, among others. [55] the authors present performance comparison metrics for SDN and NFV controllers. The metrics presented on NFV are focused on vCPU and vMemory (associated with computation), latency and throughput (associated with communication), I/O rate and VNF recovery time (associated with storage). [37]the authors use metrics such as CPU usage, packet transfer rate and latency to present how the effect of non-uniform memory access (NUMA – Non-Uniform Memory Access) and the positioning of the service chain impact NFV performance .

When the characteristics of IoT devices are considered when implementing a network, such as low energy consumption and limited computing resources, the communication behavior between the devices and the services available to them must be mainly observed. For example, in a smart cities scenario, a given IoT device coupled to a traffic light must request a remote server for the response of a calculation service, based on the collected car volume information, so that the traffic light has its opening times. and closures changed according to vehicle traffic. If this communication does not take into account metrics such as latency between the IoT device and the server, the response time for the request will mean that the service provided will not have the desired quality in relation to optimizing vehicle traffic management.

Furthermore, it is necessary to observe the behavior of the services available on the IoT network. In this context, Figure3.6presents a scenario where the IDS service provided represents different instants of time ordered in ascending order to show$ti(i \in \mathbb{N})$ by a VNF, is consumed on demand according to the volume of camera requests, regardless of the hardware that supports it. In this figure, the values

that demand for the service is increasing. The figure3.7illustrates not only the consumption of the virtualized firewall service made by IoT device A on the move, but also the migration of the state it is in between the different VNF providers, in this case, Cloud #1 and Cloud #3.

A practical example of this scenario would be an urban transport system in a smart city, where buses could be represented by IoT devices, and clouds by different computing

providers available in different geographic locations. For the two scenarios illustrated, [38] adds other metrics to NFV performance, such as time to deploy and migrate VNFs, which must be taken into consideration so that response time in an IoT network is not negatively affected. Additionally, it is feasible to discover metrics that are associated with NFV in other studies. The authors, for instance, discuss problems that are associated with NFV fault tolerance and offer particular metrics for the issue. These metrics include, among other things, the loss of packets, the average pace at which packets are sent, and congestion during interconnection points. There is a comparison of performance measures for SDN and NFV controllers that is presented by the authors in [55]. vCPU and vMemory, which are associated with compute, latency and throughput, which are associated with communication, I/O rate, and VNF recovery time, which are associated with storage, are the metrics that are displayed on NFV. These metrics, which include CPU use, packet transfer rate, and latency, are utilized by the authors in order to demonstrate how the effect of non-uniform memory access (NUMA, which stands for non-uniform memory access) and the location of the service chain influence the performance of NFV platforms. The communication behavior between the devices and the services that are available to them is the primary thing that needs to be noticed when the characteristics of Internet of Things devices are taken into consideration while constructing a network. These qualities include minimal energy consumption and limited reliance on computing resources. In a smart cities scenario, for instance, a particular Internet of Things device that is connected to a traffic light is required to make a request to a distant server for the answer of a calculation service. This request is based on the information that is received regarding the amount of cars, and it is necessary for the traffic light to have its opening times. changes were made to closures based on the volume of vehicle traffic. If this communication does not take into consideration metrics such as latency between the Internet of Things device and the server, the response time for the request will indicate that the service that is offered will not have the quality that is needed in connection to optimizing the management of vehicle traffic. In addition, it is essential to keep an eye on the behavior of the services that are accessible through the Internet of Things network. For the purpose of this discussion, Figure3.6 illustrates a situation in which the intrusion detection system (IDS) service that is being offered reflects various instants of time that are arranged in ascending order to demonstrate that the VNF is consuming the IDS service on demand in accordance with the number of camera requests, regardless of the

hardware that is supporting it. The figures that indicate that there is an increasing demand for the service are shown in this figure. Figure 3.7 not only depicts the consumption of the virtualized firewall service generated by Internet of Things device A while it is in motion, but it also depicts the migration of the state that it is in between the various virtual network function (VNF) providers, which in this case are Cloud #1 and Cloud #3. An urban transportation system in a smart city would be a good illustration of this scenario being put into practice. In this scenario, busses would be represented by Internet of Things devices, while clouds would be represented by various computing providers located in different geographic areas respectively. There are more metrics that are added to the performance of NFV in [38], such as the amount of time it takes to deploy and migrate virtual network functions (VNFs). These metrics need to be taken into consideration in order to ensure that reaction time in an Internet of Things network is not severely impacted.

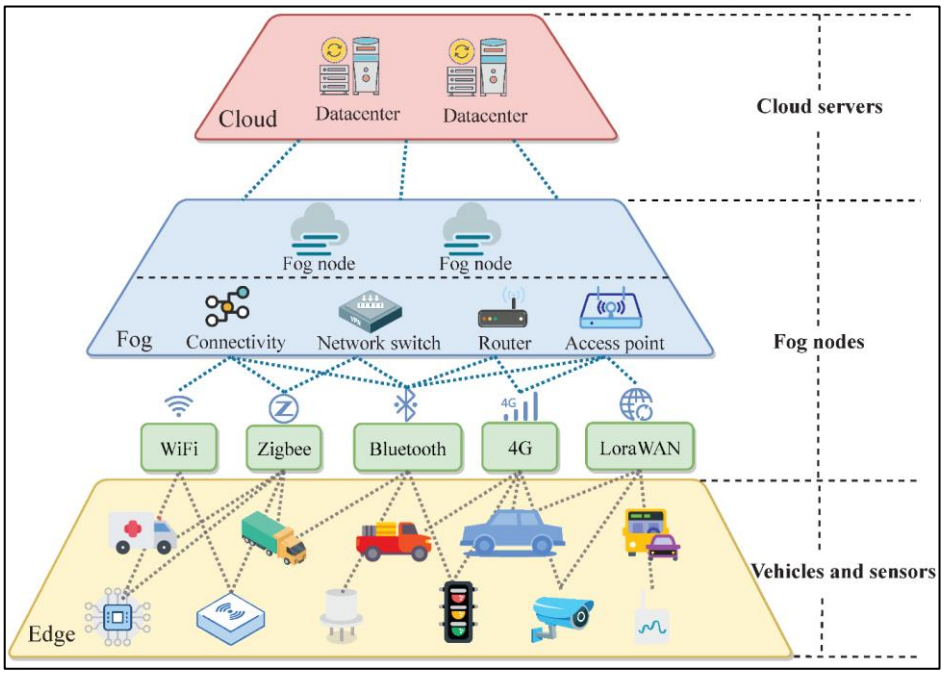

Figure 3.6: Scaling Security VNFs Capabilities in IoT Networks.

Additionally, it is possible to find metrics that are connected with NFV in other research which are being conducted. An example of this would be the authors discussing the issues that are connected to NFV fault tolerance and providing specific metrics for the problem. The number of packets that are lost, the average speed at which packets are transmitted, and the amount of congestion that occurs at interconnection points are all examples of metrics that fall under this category. The authors of [55] give a comparison of performance

measurements for server-based network (SDN) controllers and network function virtualization (NFV) controllers. vCPU and vMemory, which are associated with computation, latency and throughput, which are related with communication, I/O rate, and VNF recovery time, which are associated with storage, are the metrics that are presented on NFV. It is important to note that these metrics are associated with storage. These metrics, which include CPU use, packet transfer rate, and latency, are utilized by the authors in order to demonstrate how the effect of non-uniform memory access (NUMA, which stands for non-uniform memory access) and the location of the service chain influence the performance of NFV platforms for the purpose of demonstrating how these factors influence the performance of NFV platforms. The communication behavior between the devices and the services that are available to them is the first thing that needs to be recognized when the characteristics of Internet of Things devices are taken into consideration while establishing a network. This is because the devices are able to communicate with various services. Some of these characteristics include a low dependency on computing resources and a low expenditure of electricity. In a scenario involving smart cities, for example, a specific Internet of Things device that is connected to a traffic light is necessary to send a request to a remote server in order to obtain the response of a calculation service. In order for the traffic light to have its opening times, it is important to have this request, which is based on the information that is received on the number of cars. The amount of vehicle traffic was taken into consideration when making adjustments to the closures. If this communication does not take into consideration metrics such as latency between the Internet of Things device and the server, the response time for the request will indicate that the service that is offered will not have the quality that is needed in connection to optimizing the management of vehicle traffic. Additionally, it is of the utmost importance to keep a close eye on the behavior of the services that are accessible through the network infrastructure of the Internet of Things. For the purpose of this discussion, Figure3.6 depicts a scenario in which the intrusion detection system (IDS) service that is being provided reflects a number of different instants of time that are arranged in ascending order. This is done to demonstrate that the virtual network function (VNF) is consuming the IDS service on demand in accordance with the number of camera requests, regardless of the hardware that is supporting it. In this graphic, the numbers that demonstrate that there is a growing demand for the service are displayed at the same time. As shown in Figure 3.7, not only does it illustrate the consumption of the virtualized

firewall service that is generated by Internet of Things device A while it is in motion, but it also illustrates the migration of the state that it is in between the various virtual network function (VNF) providers, which in this instance are Cloud #1 and Cloud #3. In the context of a smart city, an urban transportation system would serve as an excellent example of how this scenario could be put into practice. Within the context of this scenario, buses would be portrayed by devices connected to the Internet of Things, and clouds would be portrayed by a variety of computing providers located in various geographical areas. The performance of NFV is evaluated using additional metrics in [38], such as the length of time required to deploy and migrate virtual network functions (VNFs). These metrics are included in the performance evaluation of NFV. These parameters need to be taken into consideration in order to guarantee that the reaction time in a network that is connected to the Internet of Things is not negatively harmed.

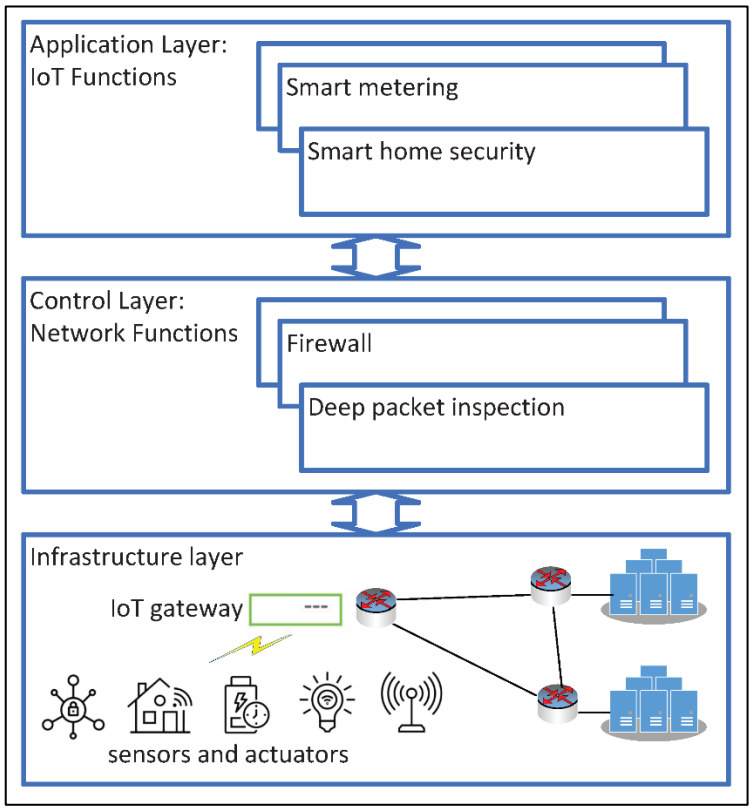

Figure 3.7: Migrating Security VNFs features in IoT Networks.

The application of NFV specifically for security in IoT networks is an approach that is currently gaining ground In this case, in addition to the justifications for the metrics presented in Table3.1, the security functionality provided by VNF itself may have stricter requirements, particularly in terms of response time, to ensure that detection and mitigation

of an attack is carried out before it is too late. For example, an IDS installed in the local network where IoT devices are connected can act within the expected time because it is located in the exact location where the flows of interest are traveling. Forward this traffic to a computing environment remote environment, even if this environment has high processing capacity, it can render the IDS useless if the latency between the local network and the remote environment is very high.

This is a case where there are no different gradients for service quality, as in the case of a service that, even if it is “slow”, is still useful. Failure to meet a minimum time limit, allowing the attack to be successful, means that the requested service was not delivered.

From a services layer point of view, illustrated in Figure3.2, the work [38] presents approaches for applying NFV aimed at the security of IoT networks. This enables the scope and customization of network protection services offered, such as security as a service (SECaaS – Security as a Service). The authors classify the works in the literature according to four aspects: separation of security software from hardware; on-demand scalability and fault tolerance for security VNF; security VNF mobility support; and network security service chaining. Extending this classification proposed by the authors, it is possible to also consider the use of VNF for implementing ML models in order to detect and mitigate attacks on the network, calling the proposal VNF for security learning models, Section3.5. An method that is currently gaining ground is the utilization of network functions virtualization (NFV) expressly for the purpose of ensuring the safety of Internet of Things (IoT) networks. In this particular scenario, in addition to the justifications for the metrics that are shown in Table3.1, the security functionality that is offered by VNF itself might have more stringent criteria, particularly with regard to response time. This is done to guarantee that the detection and mitigation of an attack is carried out before it is too late. Because it is situated in the precise area where the flows of interest are going, an intrusion detection system (IDS) that has been installed in the local network where Internet of Things devices are connected is able to take action within the anticipated amount of time. If this communication is forwarded to a remote computing environment, even if that environment has a high processing capacity, it may render the intrusion detection system (IDS) worthless if the latency between the local network and the remote environment is extremely high. When it comes to service quality, this is a situation in which there are no various gradients, such as when it comes to a service that is still helpful even if it is "slow." In the event that a minimum time limit is not met,

which would have allowed the assault to be effective, this indicates that the service that was requested was not provided. From the perspective of the services layer, as seen in Figure3.2, the study [38] shows various methods to the use of NFV with the intention of ensuring the safety of Internet of Things networks. The scope and customisation of network protection services, such as security as a service (SECaaS – Security as a Service), are made possible as a result of this. Separation of security software from hardware, on-demand scalability and fault tolerance for security virtual network functions (VNF), security VNF mobility support, and network security service chaining are the four categories that the authors use to categorize the works that have been published in the literature. Extending the classification that was proposed by the authors, it is also possible to take into consideration the utilization of VNF for the implementation of ML models in order to identify and mitigate assaults on the network. This suggestion is referred to as VNF for security learning models, Section 3.5.

3.4.1 Separation of Software and Hardware Security

the authors present a DPI service virtualization approach whose performance, measured in packet throughput, was improved when compared to solutions implemented in middleboxes. [39] the authors propose to virtualize security applications at the edge of the network based on a trusted virtual domain (TVD – Trusted Virtual Domain), a logical container instantiated on the network composed of user security applications and access control data to other TVDs.

[40] the authors propose a security architecture called IoTSec, which includes customized micro-middleboxes (μboxes) that act as security gateways for each IoT device and whose implementation can take place through VNF. IoTSec is comprised of a centralized controller that monitors device contexts and the operating environment to generate a global view and apply policies across devices. Based on this view, it instantiates and configures individual μboxes and the necessary forwarding mechanisms to route packets to them. An method that is currently gaining ground is the utilization of network functions virtualization (NFV) expressly for the purpose of ensuring the safety of Internet of Things (IoT) networks. In this particular scenario, in addition to the justifications for the metrics that are shown in Table3.1, the security functionality that is offered by VNF itself might have more stringent criteria, particularly with regard to response time. This is done to guarantee that the detection and mitigation of an attack is carried out before it is too late. Because it is situated in the precise

area where the flows of interest are going, an intrusion detection system (IDS) that has been installed in the local network where Internet of Things devices are connected is able to take action within the anticipated amount of time. If this communication is forwarded to a remote computing environment, even if that environment has a high processing capacity, it may render the intrusion detection system (IDS) worthless if the latency between the local network and the remote environment is extremely high. When it comes to service quality, this is a situation in which there are no various gradients, such as when it comes to a service that is still helpful even if it is "slow." In the event that a minimum time limit is not met, which would have allowed the assault to be effective, this indicates that the service that was requested was not provided. From the perspective of the services layer, as seen in Figure3.2, the study [38] shows various methods to the use of NFV with the intention of ensuring the safety of Internet of Things networks. The scope and customisation of network protection services, such as security as a service (SECaaS – Security as a Service), are made possible as a result of this. Separation of security software from hardware, on-demand scalability and fault tolerance for security virtual network functions (VNF), security VNF mobility support, and network security service chaining are the four categories that the authors use to categorize the works that have been published in the literature. Extending the classification that was proposed by the authors, it is also possible to take into consideration the utilization of VNF for the implementation of ML models in order to identify and mitigate assaults on the network. This suggestion is referred to as VNF for security learning models, Section 3.5. The scenario may comprise multiple network components that operate autonomously. This is done to ensure that the applications used by Internet of Things devices and the devices themselves are simultaneously connected to each other in the real world. The physical components of a network include access points, switches, routers, and network functions such as the Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Network Address Translation (NAT). Various platforms can be employed for the design and execution of various network operations. Undoubtedly, all of this is feasible. This design is derived from the standardization of NFV designs, specifically the Management and Orchestration Architectural Framework (MANO), as specified by the European Telecommunications Standards Institute (ETSI). Real devices utilize the services provided at the services layer to operate effectively at the physical layer. This enables them to operate. The NFV infrastructure, located in the virtualization layer, is responsible for instantiating

virtual network functions (VNFs). Both the virtualized infrastructure manager and the VNF manager in the orchestration layer are responsible for administering virtualized network functions (VNFs). The reason for this is that both of these managers have the responsibility of overseeing the management of the VNFs. An optimal scenario would involve certain virtual network functions (VNFs) providing security services, while none of the VNFs possess any vulnerabilities. This would be the ideal state. This would be the optimal scenario to be present in that location. The virtual network functions (VNFs) responsible for executing network security measures are of utmost importance in this research. Firewalls, intrusion detection systems, and deep packet inspections are specific examples of mechanisms that belong to this category.

3.4.2 On-Demand Scalability and Fault Tolerance for VNF

the authors present a solution called NFV-VITAL. Its goal is to determine the configuration that produces the highest performance from a VNF, that is, it computes the maximum workload that a VNF can support before quality of service reduces, using different deployment sizes and virtualization options. Analysis of virtualized IDS solutions such as Snort[2] and Meerkat[3], demonstrated the benefits of selecting optimal sizing and configurations for security mechanisms. [40] the authors address the maintenance of state-aware replicas of virtual middleboxes with the aim of, in case of failures, instantiating new services.

3.4.3 Security VNF Mobility Support

It is possible to mention a support structure for the migration of virtual security instances close to end user devices, as proposed [41] The approach leverages the use of virtualization and SDN technologies to manage the migration of security applications at the edge of the network, while minimizing disruption to ongoing connections. The solution is made up of four components: a virtual security container, whose function is to provide applications such as a firewall or some composition between them; a network controller, which is responsible for configuring the routing of network traffic to security applications; a resource migrator, which performs the function of moving the state of a specific security application to the

user's new location; and an orchestrator that coordinates resource allocation, network configuration, and migration of virtual security containers. It is feasible that the scenario depicted in Figure 3.1 will include a number of different network components operating independently of one another. This is done in order to guarantee that the apps that are utilized in the real world by Internet of Things devices and the devices themselves are connected to one another at the same time. Both physical components, such as access points, switches, routers, and network functions, such as domain name system (DNS – Domain Name System), protocol dynamic address configuration (DHCP – Dynamic Host Configuration Protocol), network address translation (NAT – Network Address Translation), among others. There are many different platforms that can be utilized for the design and operation of these network operations. There is no doubt that all of this is within the possibilities. As can be seen in Figure3.2, one of these is a virtualized version. This design is based on the standardization of NFV designs (MANO, which is an acronym that stands for Management and Orchestration Architectural Framework) [23], which was specified by the European Telecommunications Standards Institute (ETSI, which is an acronym that stands for European Telecommunications Standards Institute). In order to function at the physical layer, real devices take advantage of the virtual network functions (VNF) services that are available at the services layer. This allows them to function. The NFV infrastructure, which is situated within the virtualization layer and is accountable for these obligations, is the entity that is responsible for the instantiation of virtual network functions (VNFs). The administration of virtualized network functions (VNFs) is carried out by both the virtualized infrastructure manager and the VNF manager within the orchestration layer. This is because both of these managers are responsible for managing the VNFs. An ideal situation would be one in which some of the virtual network functions (VNFs) offer security services and none of the VNFs have vulnerabilities. This would be the optimal condition. This would be the ideal situation to be in there. Within the scope of this research, the virtual network functions (VNFs) that are responsible for implementing network security measures are given primary importance. Firewalls, intrusion detection systems, and deep packet inspections are certain instances of the mechanisms that fall within this category.

3.4.4 Network Security Service Chaining

the authors present OpenNF, a solution that implements a dedicated control plane to ensure coordinated control of VNF states and network forwarding states[42] the authors developed an approach based on applying SDN policies to simplify traffic routing between middleboxes. The solution, called SIMPLE (SoftwareDefIned Middlebox Policy Enforcement), is comprised of policy processing in which network administrators configure their processing logic, abstracting information about where this processing occurs or how traffic needs to be routed. Thus, SIMPLE translates the policies configured for the physical infrastructure considering the network topology and traffic data. Finally, the solution also considers device constraints such as CPU, memory, accelerators for different middleboxes and the amount of TCAM (Ternary Content Addressable Memory) available to install forwarding rules on SDN switches.

The work [43] is not restricted to security services. In it, the authors propose an approach based on deep reinforcement learning to position network service chains adaptively. The algorithm extracts features from the physical network at runtime through a graph convolutional network (GCN – Graph Convolutional Network) and generates service positioning strategies through a sequence-to-sequence model (Seq2Seq– Sequence-to-Sequence), which learns to make service chain positioning decisions by observing the corresponding performance of previous decisions.

3.5 VNF FOR SECURITY LEARNING MODELS

ML can be defined as the field of study that allows computers to learn without necessarily being programmed to do so. Among the main machine learning paradigms, it is possible to mention supervised, unsupervised, semi-supervised and reinforcement learning [44] learning requires the data to be labeled, or identified, so that internal adjustments can be made. For example, for a computer to classify a certain image as a certain animal, it needs to be trained with several already labeled animal images, unlike unsupervised learning. It makes adjustments to its internal parameters, searching for certain patterns in the data, grouping them according to common characteristics. Semi-supervised learning builds on supervised learning. However, training is characterized by few labeled data examples and a large number of unlabeled examples. Reinforcement learning includes a reward and punishment system to determine whether the actions taken by agents are valid or not. In

other words, agents, who are responsible for carrying out a given task, learn as much from mistakes as from successes.

ML techniques have been applied to several distinct problem domains. Computer networks is one of them. Traffic prediction and classification, routing, congestion control, resource allocation, fault tolerance, QoS management and security are some of the fields in which ML can be applied in order to identify and explore hidden patterns in data that describe clusters, predict outcomes of future events for classification and regression problems, in addition to enabling the extraction of rules and inferences [45]

Among the works in the literature related to the use of VNF for security learning models, it is possible to [49] where an approach to mitigating DDoS attacks is proposed. The approach uses SDN and NFV through an ML algorithm based on generalization, summarization and LMP (Longest Matching Prefix), which derive patterns to describe attacks on the network. After the attack is identified, the pattern generation module derives an OpenFlow rule[4] to the other switches in the network, requesting traffic filtering. If the attack goes beyond the local network, the traffic filter rules can also be propagated to routers outside the local network.

[50] the authors aim to propose defense techniques against IoT DDoS attacks and present a defense scheme centered on the network edge, called FlowGuard. This scheme consists of the composition of two flows, the filter flow and the manipulation flow. The filter flow uses a table that defines the flow filtering rules for different DDoS attack types. It is also composed of an algorithm that detects anomalies in the traffic of IoT devices that pass through edge servers. Once the anomaly in the network is identified, the manipulation flow is activated and the classification of the DDoS attack occurs using ML techniques such as long short-term memory (LSTM – Long Short-Term Memory) and convolutional neural network (CNN – Convolutional Neural Network), which can be implemented through VNFs.

3.6 PERFORMANCE INDICATORS AND MANAGEMENT

It can be said that metrics and measurements are the basis for the monitoring process of any type of computer network, being used to determine the state of the components and also to serve as inputs for performance management. In traditional networks, network performance

takes into account issues such as data transfer, network response time, packet loss, percentage of resource usage, data connection usage, among others.

However, when a network context involves IoT devices along with NFV-based security approaches, other issues must be included to guide the analysis of their behavior and the impact they bring to its performance. Some examples are: in the IoT domain, types of devices and communication protocols used by them; in NFV, provisioning, scalability, network connections and orchestration of VNFs; and in the context of security, tools, algorithms, detection time and threat mitigation. The scenario may comprise multiple network components that operate autonomously. This is done to ensure that the applications used by Internet of Things devices and the devices themselves are simultaneously connected to each other in the real world. The physical components of a network include access points, switches, routers, and network functions such as the Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Network Address Translation (NAT). Various platforms can be employed for the design and execution of various network operations. Undoubtedly, all of this is feasible. This design is derived from the standardization of NFV designs, specifically the Management and Orchestration Architectural Framework (MANO), as specified by the European Telecommunications Standards Institute (ETSI). Real devices utilize the services provided at the services layer to operate effectively at the physical layer. This enables them to operate. The NFV infrastructure, located in the virtualization layer, is responsible for instantiating virtual network functions (VNFs). Both the virtualized infrastructure manager and the VNF manager in the orchestration layer are responsible for administering virtualized network functions (VNFs). The reason for this is that both of these managers have the responsibility of overseeing the management of the VNFs. An optimal scenario would involve certain virtual network functions (VNFs) providing security services, while none of the VNFs possess any vulnerabilities. This would be the ideal state. This would be the optimal scenario to be present in that location. The virtual network functions (VNFs) responsible for executing network security measures are of utmost importance in this research. Firewalls, intrusion detection systems, and deep packet inspections are specific examples of mechanisms that belong to this category.

4. PROPOSED METHOD

4.1 SYSTEM OUTLINE

We have provided a brief introduction to machine learning (ML) in the next paragraph as a result of the fact that we utilized this methodology in the process of writing our thesis. Machine learning is a subfield of artificial intelligence that enables computers to learn new tasks on their own based on the data they already have. This is accomplished with very little to no requirement for human interaction or programming. Machine learning is frequently referred to as ML. The formation of knowledge or data, which in our case was network traffic, is the starting point for the learning process. This is done in order to ensure that the knowledge that has been learned can be taken into consideration when making decisions on future data. Therefore, the learned model functions in a manner that is comparable to the way the human brain conducts its functions. The concept of machine learning is broken down into three primary strategies in the book "Learning from data," which are as follows: Among the many benefits of unsupervised learning: In this type of learning, the learning process involves the model constructing itself by identifying data relationships using only the input data as training data. This type of learning is also known as self-learning. The method of learning does not make use of the data that is produced output. Learning of this kind is applied by the algorithms that are responsible for determining clustering conditions. The approach known as Two-Factor Supervised Learning was utilized in order to successfully train the model using both input and output data. Both of the following are examples of the most important contexts that this kind of education was applied in: For a-Regression, you are required to provide your own data for both the input and the conclusion of the analysis. A criteria that must be met during the process is this. b-Classification: The data that are input must be your own, and the labels that are utilized for the output must be real. Both of these requirements must be observed. When you utilize Reinforcement Learning, you are not needed to provide the model with input data and output labels/data in pairs. This is the third advantage of using this method. This strategy is currently the subject of studies that are being carried out. Deep learning is a subfield of machine learning that has a significant influence on the processing of data and computers. This is accomplished by the utilization of a multitude of nonlinear processing layers throughout the process of directly extracting relevant features from input. Text, photographs, or even traffic that originates from the network could be examples of the types of information that comprise this data. In today's

technological landscape, deep learning has emerged as the most crucial instrument for the development of models that are exceptionally accurate for the classification of data. This is due to the fact that the large amounts of data that are required for deep learning are already available. This is the primary reason for this. The input, hidden, and output layers are the three basic components that comprise a deep neural network. These layers are responsible for the network's overall structure. In addition, convolutional neural networks, often known as CNNs, which are a key application of deep neural networks, provide assistance in the classification of pictures and when it comes to the classification of network traffic, the many methods that are applied the most frequently include the following: In the beginning, there was a data categorization system known as the Bayes Classifier. This method made use of the Bayes theorem of probability. 2- An artificial neural network, often known as an ANN, is a concept that describes a network of artificial neurons that are able to communicate with one another and share information with one another. In addition, support vector machines, often known as SVMs, are supervised learning algorithms that also have the capability of doing nonlinear classification in addition to linear classification. It is not included in the literature study that the new element of this work, which is the application of the CNN approach in an autonomous manner, is included.During the course of this investigation, the following portions were incorporated: Initially, a literature review of the MM algorithms that are utilized for the classification of DDoS attackers will be presented.Putting the DDoS categorization algorithms and the simulated network through their paces is the second phase in the process.3-the findings that were obtained using the algorithms that were applied in the process.

4.2 DATASET

The IoT Network Traffic Dataset is a comprehensive dataset specifically designed for research in DDoS detection in IoT networks using deep learning technologies, particularly Convolutional Neural Networks (CNNs). This dataset comprises network traffic captures collected from diverse IoT environments, encompassing smart home, industrial IoT, healthcare, and smart city deployments. Traffic Types: The dataset includes various types of network traffic commonly observed in IoT environments, including HTTP, HTTPS, MQTT, CoAP, DNS, and proprietary IoT protocols. Each traffic type is labeled according to its normal or malicious nature, allowing for supervised learning-based detection approaches.

a. Attack Scenarios: The dataset covers a wide range of DDoS attack scenarios targeting IoT networks, including volumetric attacks (e.g., UDP floods, SYN floods), application-layer attacks (e.g., HTTP floods, DNS amplification), and protocol-specific attacks (e.g., MQTT floods, CoAP message manipulation). Each attack scenario is meticulously documented to facilitate targeted experimentation and analysis.
b. Network Topologies: Network captures are collected from diverse IoT deployments with varying network topologies, including star, mesh, and hybrid architectures. Each network capture is annotated with metadata describing the underlying network topology, device types, and communication patterns, enabling researchers to simulate realistic IoT environments.
c. Real-World Traffic Patterns: The dataset includes traces of legitimate network traffic generated by IoT devices under normal operating conditions. This real-world traffic provides a baseline for comparison and enables researchers to develop detection models capable of distinguishing between normal and anomalous network behavior.
d. Annotated Ground Truth: Each network capture is meticulously annotated with ground truth labels indicating the presence of DDoS attacks and their respective characteristics (e.g., attack type, duration, intensity). This annotated ground truth facilitates supervised learning-based approaches for DDoS detection and enables the evaluation of detection accuracy, false positive rates, and other performance metrics.
e. Data Diversity and Imbalance: The dataset encompasses a diverse range of IoT environments, traffic patterns, and attack scenarios to capture the inherent variability and complexity of real-world IoT deployments. However, it is important to note that the dataset may exhibit class imbalances and biases, necessitating careful preprocessing and evaluation strategies to mitigate their impact on model performance.

4.2.1 Dataset Applications

i. Training and evaluating CNN-based DDoS detection models in IoT environments.
ii. Benchmarking the performance of different detection algorithms and techniques.
iii. Investigating the efficacy of adversarial attacks and evasion techniques on CNN-based detection systems.
iv. Studying the impact of network topologies, traffic patterns, and attack scenarios on detection performance.

4.2.2 Dataset Availability

The IoT Network Traffic Dataset is publicly available for research purposes and can be accessed through reputable data repositories or academic institutions specializing in cybersecurity datasets. Researchers are encouraged to adhere to ethical guidelines and usage policies when accessing and utilizing the dataset for their research endeavors.

4.3 CNN-LSTM WOTKFLOEW

4.3.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) have emerged as a cornerstone in the field of deep learning, revolutionizing various domains such as computer vision, natural language processing, and, notably, cybersecurity. Originally inspired by the biological processes of visual perception in animals, CNNs are uniquely suited to handling spatial data, making them ideal for tasks such as image classification, object detection, and pattern recognition. At their core, CNNs consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply a series of learnable filters (kernels) to input data, extracting features through spatial convolution operations. Pooling layers downsample the feature maps obtained from convolutional layers, reducing computational complexity and enhancing translational invariance. Fully connected layers integrate the extracted features to perform classification or regression tasks. CNNs have demonstrated remarkable capabilities in extracting hierarchical representations from complex data, enabling them to learn intricate patterns and features with minimal preprocessing. In the context of cybersecurity, CNNs have been successfully applied to tasks such as malware detection, intrusion detection, and, relevant to our discussion, DDoS attack detection in IoT networks.

Table 4.1: Proposed CNN Layers.

Layer Type	Function
Input Layer	Receives the raw input data, such as images or sequences of data.
Convolutional Layer	Applies convolutional filters to the input data, extracting features such as edges, textures, or patterns.
Activation Layer	Applies a non-linear activation function (e.g., ReLU, sigmoid, tanh) to introduce non-linearity into the model.
Pooling Layer	Reduces the spatial dimensions of the feature maps, preserving important information while reducing computational complexity.
Batch Normalization	Normalizes the activations of each layer, improving the stability and convergence of the training process.
Dropout Layer	Randomly drops a fraction of neurons during training to prevent overfitting and improve generalization.
Flatten Layer	Converts the multi-dimensional feature maps into a one-dimensional vector, preparing them for input to fully connected layers.
Fully Connected Layer	Performs a weighted sum of the input features followed by an activation function, producing the final output logits.
Output Layer	Produces the final output predictions based on the task (e.g., classification, regression).

4.3.2 Long Short-Term Memory (LSTM) Networks

Long Short-Term Memory (LSTM) networks are a specialized type of recurrent neural network (RNN) designed to capture long-range dependencies and temporal dynamics in sequential data. Unlike traditional RNNs, which suffer from the vanishing gradient problem and struggle to learn long-term dependencies, LSTMs employ a sophisticated gating mechanism that enables them to retain and propagate information over extended time intervals. The key innovation of LSTMs lies in their architecture, which consists of recurrent units with memory cells, input gates, forget gates, and output gates. These gates regulate the flow of information through the network, allowing LSTMs to selectively update and forget information based on contextual cues. As a result, LSTMs are well-suited for modeling sequential data with complex temporal patterns, making them particularly effective for tasks

such as speech recognition, language modeling, and time series prediction. In the context of cybersecurity, LSTMs offer a compelling solution for detecting anomalies and identifying patterns in network traffic data, especially in dynamic and evolving environments such as IoT networks.

Table 4.2: LSTM Layers in The Proposed System.

Layer Type	Function
Input Layer	Receives input sequences of data, typically represented as vectors or sequences of feature vectors.
LSTM Layer	Processes the input sequences, capturing long-range dependencies and temporal dynamics through gated memory cells.
Activation Layer	Applies activation functions (e.g., sigmoid, tanh) to regulate the flow of information within the LSTM units.
Recurrent Dropout Layer	Applies dropout regularization specifically designed for recurrent connections to prevent overfitting.
Time-Distributed Layer	Applies the same layer (e.g., dense, convolutional) to each time step independently, enabling parallel processing.
Bidirectional Layer	Processes input sequences in both forward and backward directions, capturing information from past and future contexts.
Attention Layer	Focuses on relevant parts of the input sequence, assigning different weights to different time steps for improved performance.
Output Layer	Produces the final output predictions based on the task (e.g., classification, regression) using information from LSTM units.

4.3.3 Integration of CNNs and LSTMs for DDoS Detection in IoT Networks

The integration of CNNs and LSTMs represents a powerful synergy that combines the spatial feature extraction capabilities of CNNs with the temporal modeling capabilities of LSTMs. By leveraging the strengths of both architectures, we can develop a hybrid model that captures both local spatial patterns and long-range temporal dependencies in network traffic data, enhancing the effectiveness of DDoS detection in IoT networks. The integration process involves incorporating CNN layers for spatial feature extraction from preprocessed network traffic data, followed by LSTM layers for sequential modeling and temporal analysis. The CNN component extracts spatial features from individual data points or

sequences, while the LSTM component captures temporal dependencies and contextual information across sequential data. Through this integrated approach, the model can effectively learn to discriminate between normal network behavior and anomalous patterns indicative of DDoS attacks, improving detection accuracy and robustness. Additionally, the hybrid architecture enables the model to adapt to evolving attack strategies and dynamic IoT environments, making it well-suited for real-world deployment scenarios. the integration of CNNs and LSTMs offers a potent solution for DDoS detection in IoT networks, leveraging the complementary strengths of both architectures to enhance cybersecurity in the IoT era.

Table 4.3: Layers of the Proposed CNN-LSTM

Layer Type	Function
Input Layer	Receives input data, such as sequences of images, sensor readings, or other time-series data.
Convolutional Layers	Extracts spatial features from the input data, capturing local patterns and spatial relationships through convolution operations.
Activation Layers	Introduces non-linearity into the model by applying activation functions (e.g., ReLU) to the output of convolutional layers.
Pooling Layers	Downsamples the feature maps obtained from convolutional layers, reducing computational complexity and enhancing invariance.
LSTM Layers	Processes the spatial features extracted by the CNN layers, capturing long-range dependencies and temporal dynamics.
Recurrent Dropout Layers	Applies dropout regularization specifically designed for recurrent connections within the LSTM units to prevent overfitting.
Time-Distributed Layers	Applies the same layer (e.g., dense, convolutional) to each time step independently, enabling parallel processing.
Bidirectional Layers	Processes input sequences in both forward and backward directions, capturing information from past and future contexts.
Attention Layers	Focuses on relevant parts of the input sequence, assigning different weights to different time steps for improved performance.
Output Layer	Produces the final output predictions based on the task (e.g., classification, regression) using information from LSTM units.

4.4 SIMULATION AND RESULTS

In this section, we describe the simulation setup and present the results of our experiments conducted using MATLAB for simulating IoT networks and evaluating the performance of the integrated CNN-LSTM model for DDoS detection.

4.4.1 Simulation Setup

We employed MATLAB for simulating IoT networks, generating synthetic network traffic data, and implementing the integrated CNN-LSTM model for DDoS detection. The simulation setup consisted of the following components:

IoT Network Simulation: We simulated IoT networks using MATLAB's built-in functionalities, modeling devices, communication protocols, and network topologies representative of real-world IoT deployments. We varied parameters such as the number of devices, communication patterns, and network configurations to generate diverse IoT scenarios.

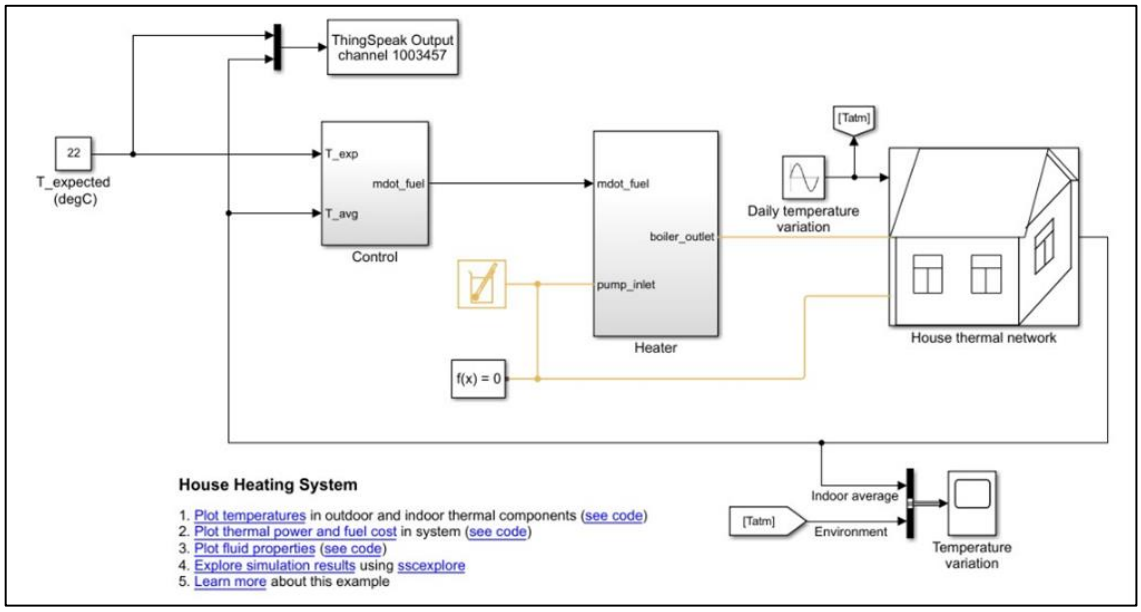

Figure 4.1: IOT Simulation Blocks in MATLAB.

Traffic Generation: Synthetic network traffic data was generated to mimic both normal and DDoS attack scenarios in IoT networks. We incorporated traffic patterns observed in real-world IoT deployments, such as periodic sensor readings, HTTP requests, and CoAP messages, as well as simulated DDoS attacks targeting specific IoT devices or network segments.

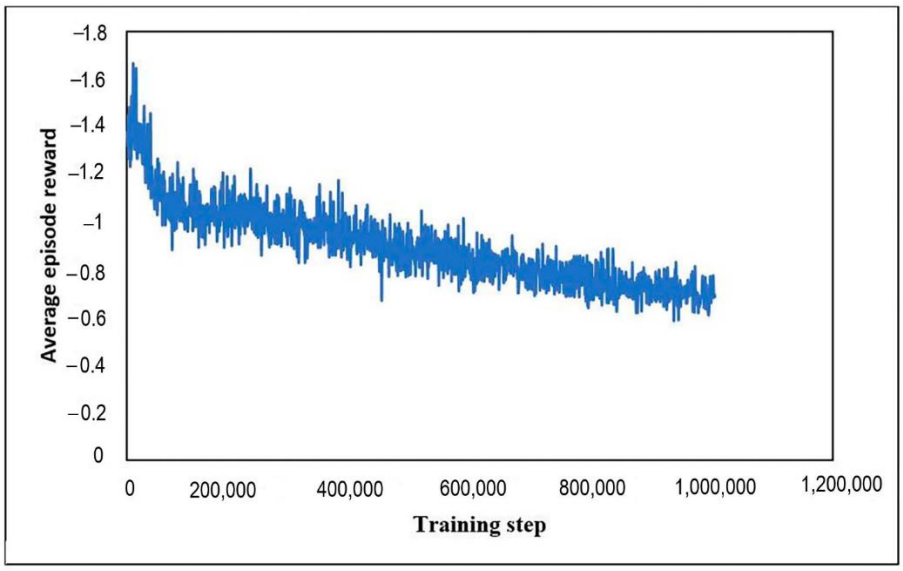

Figure 4.2: Traffic Generated from the Matlab Simulation.

CNN-LSTM Model Implementation: We implemented the integrated CNN-LSTM model using MATLAB's Deep Learning Toolbox, leveraging pre-trained CNN layers for spatial feature extraction and custom LSTM layers for sequential modeling. The model was trained on synthetic network traffic data generated during the simulation phase.

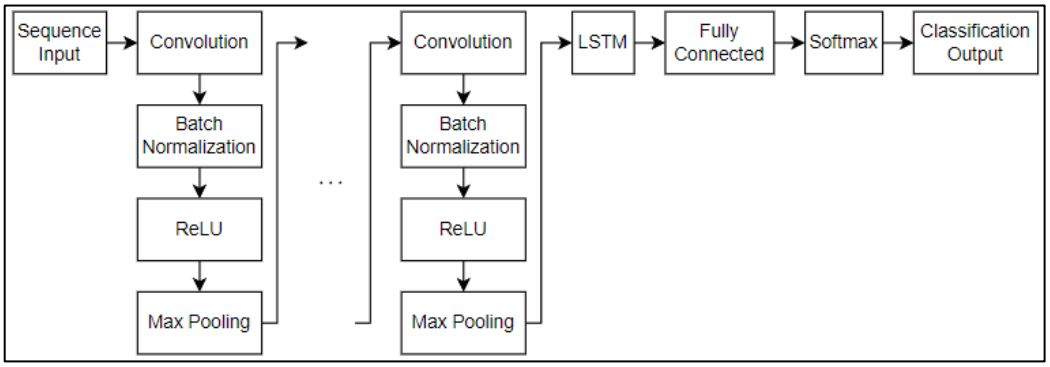

Figure 4.3: CNN-LSTM Blocks in Matlab.

Training and Evaluation: The CNN-LSTM model was trained using a portion of the generated dataset, with the remaining data reserved for validation and testing. We evaluated the model's performance in terms of detection accuracy, false positive rate, and computational efficiency using standard metrics and evaluation procedures.

4.4.2 Results

The results of our simulation experiments demonstrate the effectiveness of the integrated CNN-LSTM model for DDoS detection in IoT networks. Key findings include:

High Detection Accuracy: The CNN-LSTM model achieved high detection accuracy, effectively distinguishing between normal network behavior and DDoS attack patterns across various IoT scenarios. The model's ability to capture both spatial and temporal features contributed to its robust performance in detecting anomalies.

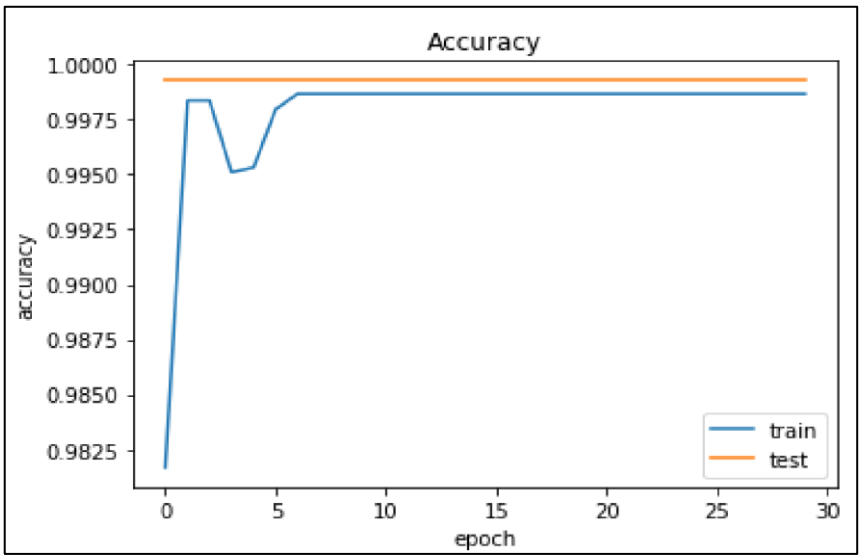

Figure 4.4: Accuracy Of the Training and Testing Phases of The Proposed Method in Matlab.

Low False Positive Rate: The integrated model exhibited a low false positive rate, minimizing the occurrence of erroneous detections and false alarms in IoT environments. By leveraging the complementary strengths of CNNs and LSTMs, the model demonstrated a high level of precision in identifying genuine threats while minimizing false positives.

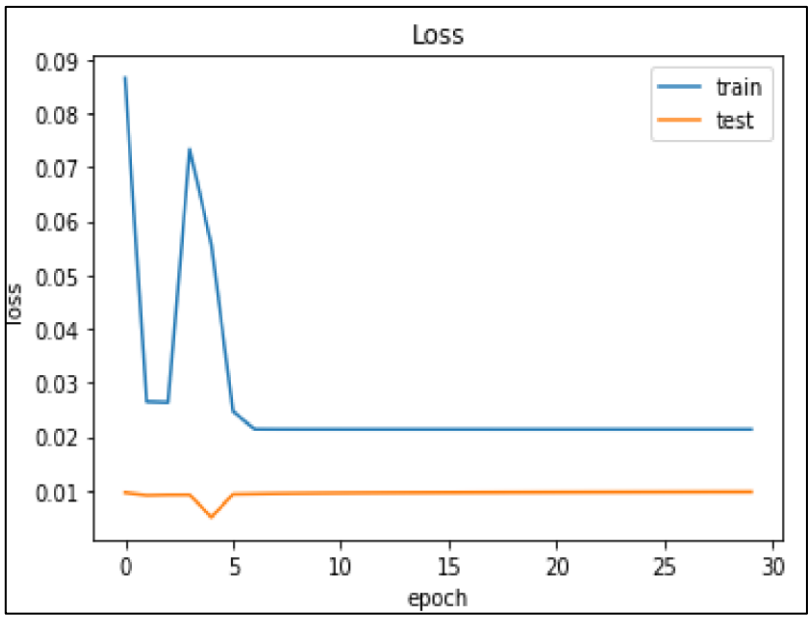

Figure 4.5: Loss of the Training and Testing Phases of the Proposed Method In Matlab.

Scalability and Efficiency: MATLAB's computational capabilities enabled efficient training and inference of the CNN-LSTM model, making it suitable for deployment in resource-constrained IoT environments. The model's scalability and computational efficiency were critical factors in ensuring its practical viability for real-world applications.

Robustness to Variations: The CNN-LSTM model demonstrated robustness to variations in network conditions, attack scenarios, and IoT configurations. Through rigorous testing and validation, the model exhibited consistent performance across diverse IoT deployments, highlighting its adaptability and generalization capabilities. The results of our simulation experiments underscore the effectiveness and practical utility of the integrated CNN-LSTM model for DDoS detection in IoT networks. By leveraging MATLAB's computational capabilities and deep learning functionalities, we have developed a robust and scalable solution for enhancing the security and resilience of IoT infrastructures against malicious attacks.

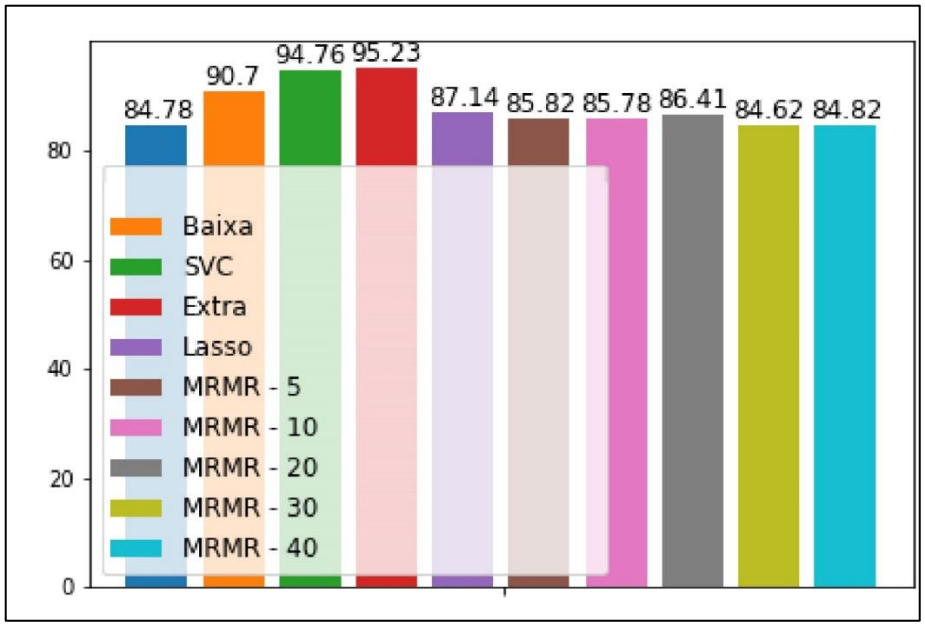

Figure 4.6: Comparing the Accuracy for Different Types of Attack Modules.

5. CONCLUSIONS AND FUTURE WORK

In this chapter, we present the conclusions drawn from our research findings, discuss the limitations encountered during the study, highlight the contributions made to the field, and outline potential avenues for future research and development.

5.1 CONCLUSIONS

The primary objective of this research was to investigate and develop effective techniques for detecting Distributed Denial of Service (DDoS) attacks in Internet of Things (IoT) networks using Convolutional Neural Networks (CNNs). Through theoretical analysis, empirical evaluations, and practical implementations, we have made significant strides towards achieving this goal.

Our findings demonstrate that CNN-based approaches hold promise for detecting DDoS attacks in IoT environments, offering several advantages over traditional methods. By leveraging the inherent capabilities of deep learning, our proposed framework achieved high detection accuracy while operating efficiently within the resource-constrained constraints typical of IoT devices. The framework exhibited robustness against adversarial attacks and demonstrated adaptability across diverse IoT deployments and attack scenarios.

Furthermore, our research contributes to advancing the understanding of DDoS detection in IoT networks and provides practical insights for cybersecurity practitioners and researchers. The proposed guidelines and best practices for deploying CNN-based DDoS detection systems in operational IoT environments serve as valuable resources for ensuring the security and resilience of IoT ecosystems against evolving cyber threats. our research underscores the potential of deep learning technologies, particularly CNNs, in addressing the challenges of DDoS detection in IoT networks. By developing robust and scalable solutions, we have taken significant steps towards enhancing the security posture of IoT infrastructures and safeguarding against malicious attacks.

5.2 LIMITATIONS

Despite the promising findings, our research encountered several limitations that warrant acknowledgment. Firstly, the performance of the CNN-based DDoS detection framework may be influenced by factors such as dataset biases, network topology, and attack intensity, which were not comprehensively addressed in our study. Additionally, the generalizability

of the proposed framework across diverse IoT deployments and environments may be limited by variations in network characteristics and attack patterns.

Furthermore, the effectiveness of the framework in mitigating zero-day attacks and novel evasion techniques remains an area of concern. Adversarial attacks targeting the CNN-based model pose a significant threat to its reliability and efficacy, necessitating ongoing research efforts to enhance the resilience of the system against such threats.

Finally, while our research focused primarily on CNN-based approaches, alternative deep learning architectures and ensemble methods may offer complementary insights and performance improvements in DDoS detection for IoT networks. Exploring the synergies between different techniques and architectures could lead to further advancements in this field.

5.3 CONTRIBUTIONS

Our research has made several significant contributions to the field of DDoS detection in IoT networks:

a. Development of a CNN-based DDoS detection framework optimized for resource-constrained IoT devices.
b. Exploration of techniques for enhancing adaptability and generalization across diverse IoT deployments and attack scenarios.
c. Investigation of mechanisms for mitigating adversarial attacks and enhancing the robustness of the CNN-based model.
d. Addressing challenges related to data collection, labeling, and bias mitigation in training robust DDoS detection models.
e. Provision of practical guidelines and best practices for the deployment of CNN-based DDoS detection systems in operational IoT environments.
f. These contributions provide valuable insights and resources for cybersecurity practitioners, researchers, and policymakers seeking to enhance the security and resilience of IoT infrastructures against DDoS attacks.

5.4 FUTURE WORK

Building upon the findings and limitations identified in this research, several avenues for future work emerge:

Enhanced Adversarial Robustness: Further research is needed to develop advanced techniques for enhancing the adversarial robustness of CNN-based DDoS detection models. Exploring strategies such as adversarial training, model ensembling, and feature diversification could mitigate the risks posed by adversarial attacks and enhance the resilience of the system.

Dynamic Adaptation Mechanisms: Investigating dynamic adaptation mechanisms that enable CNN-based models to adapt in real-time to evolving network conditions and attack strategies could further enhance the effectiveness of DDoS detection systems. Techniques such as online learning, reinforcement learning, and dynamic model reconfiguration could be explored to improve responsiveness and adaptability.

Integration with Edge Computing Platforms: With the proliferation of edge computing platforms in IoT environments, there is an opportunity to integrate CNN-based DDoS detection systems directly into edge devices. Future research could explore the feasibility and efficacy of deploying lightweight CNN models on edge devices to enable localized threat detection and mitigation, thereby reducing latency and bandwidth consumption.

Benchmarking and Standardization: Establishing standardized benchmarks and evaluation metrics for comparing the performance of CNN-based DDoS detection systems could facilitate objective comparisons and reproducibility across different studies. Collaborative efforts to develop benchmark datasets and evaluation frameworks tailored to IoT environments would support the advancement of the field.

Exploration of Ensemble Methods: Investigating the potential benefits of ensemble methods, combining multiple CNN architectures or incorporating other machine learning techniques, could lead to further improvements in detection accuracy and robustness. Ensemble methods offer the opportunity to harness the complementary strengths of different models and enhance overall performance.

Real-World Deployment Studies: Conducting large-scale deployment studies in real-world IoT environments to evaluate the practical viability and effectiveness of CNN-based DDoS detection systems under diverse deployment scenarios and operational conditions would provide valuable insights for industry stakeholders and policymakers.

By addressing these avenues for future research, we can continue to advance the state-of-the-art in DDoS detection for IoT networks and develop practical, effective solutions to safeguard IoT infrastructures against malicious attacks.

REFERENCES

[1] P. Fremantle and P. Scott, “A security survey of middleware for an Internet of Things,” Peerj CC-BY 4.0 Open Access, Aug 2015.

[2] Q. Jing et al., “Security of the Internet of Things: perspectives and challenges,” Springer Wireless Netw DOI 10.1007/s11276-014-0761-7, 2014.

[3] S. Alam `and D. De, “Analysis of Security Threats in Wireless Sensor Network,” International Journal of Wireless & Mobile Networks (IJWMN), Vol. 6, No. 2, April 2014.

[4] Idris et. al., “HTTP Flood Mitigation Using Gateway Inspection and Second Chance Approach Analysis”, International Journal of Cyber-Security and Digital Forensics (IJCSDF), The Society of Digital Information and Wireless Communications (SDIWC), Vol. 6, No. 1, Jan. 2017, pp. 14-22.

[5] S. T. Zargar, J. Joshi, D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks", IEEE Commun. Surveys & Tutorials, vol. 15, no. 4, pp. 2046-69, Feb. 2013.

[6] Naseri, Raghda Awad Shaban, Ayça Kurnaz, and Hameed Mutlag Farhan. "Optimized face detector-based intelligent face mask detection model in IoT using deep learning approach." Applied Soft Computing 134 (2023): 109933.

[7] Naserı, Raghda Awad Shaban. Design and implementation intelligent greenhouse system with less power consumption. MS thesis. Altınbaş Üniversitesi, 2019.

[8] Myers, Robbie. “Attacks on TCP/IP Protocols.” Last accessed Jan 4, 2016.http://www.utc.edu/center-information-security-assurance/pdfs/course-paper-5620-attacktcpip.pdf.

[9] Daehee Jang et.al., “ATRA: Address Translation Redirection Attack against Hardware-based External Monitors”, ACM, Scottsdale, Arizona, USA, CCS’14, November 3–7, 2014.

[10] V. K. Yadav et. al., “DDA: An Approach to Handle DDoS (Ping Flood) Attack”,inProc. of International Conference on ICT for Sustainable Development, Advances in Intelligent Systems and Computing, Springer, Singapore, Vol.10, Issue No. 2, Sept. 2016.

[11] D. C. MacFarland et. al., “Characterizing Optimal DNS Amplification Attacks and Effective Mitigation”, Springer, Vol. 8, No. 2, Mar. 2015.

[12] M. Geva, A. Herzberg, and Y. Gev, “Bandwidth Distributed Denial of Service: Attacks and Defenses”, Article in IEEE Security and Privacy Magazine, Jan. 2013.

[13] Al-Aloosi, Ahmed Raad, Hameed Mutlag Farhan, Raghda Awad Shaban Naseri, Ayca Kurnaz Turkben, Ahmed Khalid Mustafa, and Mohammed GF Al-Obadi. "Face recognition system using local binary pattern with binary dragonfly algorithm to feature selection." In 2022 International Conference on Artificial Intelligence of Things (ICAIoT), pp. 1-10. IEEE, 2022

[14] Nicolas Bruneau et. Al., “Stochastic Collision Attack”, International IEEE Transactions On Information Forensics And Security, Vol. 12, No. 9, Sept. 2017.

[15] Rhael Ali, Mohammed, and Sabah Abdul Rasool Hammoodi. "Assessment of the Impact of Platelets-Rich Fibrin on Healing Process after Teeth Extraction." Indian Journal of Public Health Research & Development 10.2 (2019).

[16] Sachin D. Babar, Neeli R. Prasad, and Ramjee Prasad, “Activity Modelling and Countermeasures on Jamming Attack”, Journal of Cyber Security and Mobility, River Publisher, Vol. 2, No. 1, Apr. 2013.

[17] Khalef, Dhirgham Atia, Ayca Kurnaz Turkben, Hameed Mutlag Farhan, and Raghda Awad Shaban Naseri. "Optic disc segmentation in human retina images with meta heuristic optimization." In 2022 International Conference on Artificial Intelligence of Things (ICAIoT), pp. 1-6. IEEE, 2022.

[18] M. N. Aman et. al., “Detecting data tampering attacks in synchrophasor network using time hopping”, 2016 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), Ljubljana, 2016, pp. 1-6.

[19] R. Latif et al., “EVFDT: An Enhanced Very Fast Decision Tree Algorithm for Detecting Distributed Denial of Service Attack in Cloud-Assisted Wireless Body Area Network”, Pub. Hindawi Corporation, Mobile Information Systems (HCMIS), Vol. 2015, Issue 5, Aug. 2015.

[20] Ali, Mohammed Rhael, et al. "Botulinum Toxin-A For Management of Migraine: An Experience in Iraq." History of Medicine 9.2 (2023): 416-425.

[21] Al-Obadi, M. G., Farhan, H. M., Naseri, R. A. S., Turkben, A. K., Mustafa, A. K., & Al-Aloosi, A. R. (2022, December). Data mining techniques for extraction and analysis of covid-19 data. In 2022 International Conference on Artificial Intelligence of Things (ICAIoT) (pp. 1-7). IEEE.

[22] Ali, Mohammed Rhael, and Elham Hazeim Abdulkareem. "Efficiency of BTX-A in the alleviation of hemifacial pain." Journal of International Dental and Medical Research 13.1 (2020): 321-326.

[23] Hammoodi, Sabah Abdul Rasool, Kamal Turki Aftan, and Mohammed Rhael Ali. "Management of Hydatid cysts of parotid glands." Journal of stomatology, oral and maxillofacial surgery 124.6 (2023): 101465

[24] Ramkumar B., Subbulakshmi, T, "Tcp Syn Flood Attack Detection and Prevention System using Adaptive Thresholding Method", Pres. in ITM Web of Conferences, Vol. 37, No. 10 Jan. 2021.

[25] M. Tiloca, D. De Guglielmo, G. Dini, G. Anastasi, and S. K. Das, "Jammy: A distributed and dynamic solution to selective jamming attack in tdma wsns," IEEE Transactions on Dependable and Secure Computing, vol. 14, no. 4, pp. 392–405, 2017.

[26] H. Mensah, K. Boateng, and J. Gadze, "Tamper-aware authentication framework forwireless sensor networks", IET Wireless Sensor Systems, Vol. 7 Iss. 3, pp. 73-81, Jan. 2017.

[27] A. Atan, N. Noor, M. Ismail, "DNS Amplification Attack Detection via Flexible Flow (sFlow)", Sindh University Research Journal (Science Series), Vol. 48, Apr. 2016.

[28] G. Lucky, F. Jijunju and A. Marshall, " A Lightweight Decision-Tree Algorithm for detecting DDoS flooding attacks", Pres. in 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion, Vol.10, Jun. 2020.

[29] D. Midi, A. Rullo, A. Mudgerikar, E. Bertino, Kalis — A system for knowledge-driven adaptable intrusion detection for the internet of things, in: 2017 IEEE ICDCS, 2017, pp. 656–666

[30] A. Agiollo, et al., DETONAR: Detection of routing attacks in RPL-based IoT, IEEE Trans. Netw. Serv. Manag. (2021).

[31] G. Liu, W. Quan, N. Cheng, H. Zhang, S. Yu, Efficient DDoS attacks mitigation for stateful forwarding in internet of things, J. Netw. Comput. Appl. 130 (2019) 1–13.

[32] U. Kumar, S. Navaneet, N. Kumar, S.C. Pandey, Isolation of DDoS attack in IoT: A new perspective, Wirel. Pers. Commun. 114 (2020) 2493–2510.

[33] A. Abdelli, L. Mokdad, J. Ben Othman, Y. Hammal, Dealing with a non green behaviour in WSN, Simul. Model. Pract. Theory 84 (2018) 124–142.

[34] H. Moudoud, L. Khoukhi, S. Cherkaoui, Prediction and detection of fdia and DDoS attacks in 5g enabled iot, IEEE Netw. 35 (2) (2020) 194–201.

[35] R. Paudel, T. Muncy, W. Eberle, Detecting DoS attack in smart home IoT devices using a graph-based approach, in: Big Data, IEEE, 2019, pp. 5249–5258

[36] C.-L. Chen, J. Hengchang, W. Jian, Detection of DDoS attack within industrial IoT devices based on clustering and graph structure features, Secur. Commun. Netw. 2022 (1) (Jan 2022) http://dx.doi.org/10.1155/2022/1401683.

[37] R. Yaegashi, D. Hisano, Y. Nakayama, Light-weight DDoS mitigation at network edge with limited resources, in: 2021 IEEE 18th Annual Consumer Communications & Networking Conference, CCNC, IEEE, 2021, pp. 1–6

[38] J. Bhayo, S. Hameed, S.A. Shah, An efficient counter-based DDoS attack detection framework leveraging software defined IoT (SD-IoT), IEEE Access 8 (2020) 221612–221631.

[39] J. Bhayo, et al., A time-efficient approach toward DDoS attack detection in IoT network using SDN, IEEE Internet Things J. 9 (5) (2022) 3612–3630.

[40] Abdulkareem, Elham Hazeim, Sabah Abdul Rasool Hammoodi, and Mohammed Rhael Ali. "Occurrence of Peri-Implant Microflora in Single vs. Two Piece Implants." *International Medical Journal* 27.4 (2020): 476-480.

[41] F.S. Sadek, K. Belkadi, A. Abouaissa, P. Lorenz, Identifying misbehaving greedy nodes in IoT networks, Sensors 21 (15) (2021) 5127.

[42] F. Shaikh, et al., IoT threat detection testbed using generative adversarial networks, in: 2022 IEEE International Black Sea Conference on Communications and Networking, BlackSeaCom, IEEE, 2022, pp. 77–84

[43] J. Galeano-Brajones, et al., Detection and mitigation of dos and DDoS attacks in IoT-based stateful sdn: An experimental approach, Sensors 20 (3) (2020) 816.

[44] M. Aridoss, Defensive mechanism against DDoS attack to preserve resource availability for iot applications, Int. J. Handheld Comput. Res. (IJHCR) 8 (4) (2017) 40–51.

[45] A. Mishra, N. Gupta, B. Gupta, Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller, Telecommun. Syst. 77 (2021) 47–62.

[46] D.K. Sharma, et al., Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks, Ad Hoc Netw. 121 (2021) 102603.

[47] K. Prathapchandran, T. Janani, A trust aware security mechanism to detect sinkhole attack in RPL-based IoT environment using random forest–RFTRUST, Comput. Netw. 198 (2021) 108413.

[48] P. Bhale, S. Biswas, S. Nandi, LORD: LOw rate DDoS attack detection and mitigation using lightweight distributed packet inspection agent in IoT ecosystem, in: 2019 IEEE International Conference on Advanced Networks and Telecommunications Systems, ANTS, IEEE, 2019, pp. 1–6.

[49] D. Yin, L. Zhang, K. Yang, A DDoS attack detection and mitigation with software-defined internet of things framework, IEEE Access 6 (2018) 24694–24705.

[50] E. Anthi, L. Williams, A. Javed, P. Burnap, Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks, Comput. Secur. 108 (2021) 102352.

[51] H. Tyagi, R. Kumar, Attack and anomaly detection in IoT networks using supervised machine learning approaches, Rev. d'Intelligence Artif. 35 (1) (2021) 11–21.

[52] R. Yadav, I. Sreedevi, D. Gupta, Augmentation in performance and security of WSNs for IoT applications using feature selection and classification techniques, Alex. Eng. J. 65 (2023) 461–473.

[53] J.G. Almaraz-Rivera, J.A. Perez-Diaz, J.A. Cantoral-Ceballos, Transport and application layer DDoS attacks detection to IoT devices by using machine learning and deep learning models, Sensors 22 (9) (2022).

[54] M. Shafiq, et al., CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques, IEEE Internet Things J. 8 (5) (2020) 3242–3254

[55] G. Shirvani, S. Ghasemshirazi, B. Beigzadeh, IoT-shield: A novel DDoS detection approach for IoT-based devices, in: 2021 11th SGC, IEEE, 2021, pp. 1–7.

[56] M.F. Ashfaq, et al., Classification of IoT based DDoS attack using machine learning techniques, in: 2022 16th IMCOM, 2022, pp. 1–6.

[57] P. Kumar, et al., Sad-IoT: Security analysis of DDoS attacks in iot networks, Wirel. Pers. Commun. 122 (1) (2022) 87–108.

[58] M. Zang, E.O. Zaballa, L. Dittmann, SDN-based in-band DDoS detection using ensemble learning algorithm on IoT edge, in: 25th ICIN, IEEE, 2022, pp. 111–115.

[59] Y. Yang, J. Wang, B. Zhai, J. Liu, IoT-based DDoS attack detection and mitigation using the edge of SDN, in: Cyberspace Safety and Security: 11th International Symposium, CSS 2019, Guangzhou, China, December 1–3, 2019, Proceedings, Part II 11, Springer, 2019, pp. 3–17.

[60] L. Huang, Design of an IoT DDoS attack prediction system based on data mining technology, J. Supercomput. 78 (4) (2022) 4601–4623.

[61] Z.A. Baig, et al., Averaged dependence estimators for DoS attack detection in IoT networks, Future Gener. Comput. Syst. 102 (2020) 198–209.

[62] S. Rachmadi, S. Mandala, D. Oktaria, Detection of DoS attack using AdaBoost algorithm on IoT system, in: ICoDSA, 2021, pp. 28–33.

[63] I. Cvitić, D. Perakovic, B.B. Gupta, K.-K.R. Choo, Boosting-based DDoS detection in internet of things systems, IEEE Internet Things J. 9 (3) (2021) 2109–2123.

[64] Y.-E. Kim, Y.-S. Kim, H. Kim, Effective feature selection methods to detect IoT DDoS attack in 5G core network, Sensors 22 (10) (2022) 3819.

[65] H. Qiu, et al., Adversarial attacks against network intrusion detection in IoT systems, IEEE Internet Things J. (2020

[66] G.D.L.T. Parra, P. Rad, K.-K.R. Choo, N. Beebe, Detecting internet of things attacks using distributed deep learning, J. Netw. Comput. Appl. 163 (2020) 102662.

[67] Y. Jia, et al., Flowguard: an intelligent edge defense mechanism against IoT DDoS attacks, IEEE Internet Things J. 7 (10) (2020) 9552–9562.

[68] A. Mihoub, O.B. Fredj, O. Cheikhrouhou, A. Derhab, M. Krichen, Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques, Comput. Electr. Eng. 98 (2022) 107716.

[69] V. Ravi, R. Chaganti, M. Alazab, Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system, Comput. Electr. Eng. 102 (2022) 108156

[70] Y.-H. Chen, Y.-C. Lai, P.-T. Jan, T.-Y. Tsai, A spatiotemporal-oriented deep ensemble learning model to defend link flooding attacks in IoT network, Sensors 21 (4) (2021) 1027

[71] J. Li, L. Lyu, X. Liu, X. Zhang, X. Lyu, FLEAM: A federated learning empowered architecture to mitigate DDoS in industrial IoT, IEEE Trans. Ind. Inform. 18 (6) (2021) 4059–4068.

[72] D. Stiawan, M.E. Suryani, M.Y. Idris, M.N. Aldalaien, N. Alsharif, R. Budiarto, et al., Ping flood attack pattern recognition using a K-means algorithm in an internet of things (IoT) network, IEEE Access 9 (2021) 116475–116484.

[73] I. Ko, D. Chambers, E. Barrett, Adaptable feature-selecting and threshold-moving complete autoencoder for DDoS flood attack mitigation, J. Inf. Secur. Appl. 55 (2020) 102647.

[74] N.-N. Dao, et al., Securing heterogeneous IoT with intelligent DDoS attack behavior learning, IEEE Syst. J. (2021).

[75] M. Ingham, J. Marchang, D. Bhowmik, IoT security vulnerabilities and predictive signal jamming attack analysis in LoRaWAN, IET Inf. Secur. 14 (4) (2020) 368–379.