

# The Application of Statistical and Machine Learning Techniques for DDoS Detection in Network Information Systems

Pheeha Machaka



A thesis submitted in fulfilment of the requirements of  
the University of Western Cape,

for the award of

UNIVERSITY of the  
Doctor of Philosophy  
WESTERN CAPE

Computer Science

Supervised by

Prof. Antoine Bagula

April 2022

<https://etd.uwc.ac.za/>

## Declaration

---

I hereby declare that the work presented in this thesis has not been submitted for any other degree or professional qualification, and that it is the result of my own independent work.

Pheeha Machaka

---

Full Name Goes Here (Candidate)

April 2022

---

Date



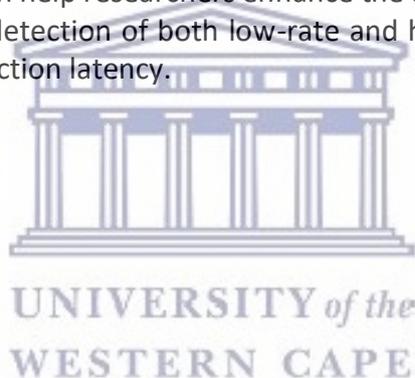
## Abstract

---

A surge of large-scale Distributed Denial of Service (DDoS) attacks has swept the internet in recent years, possibly presenting a serious threat to industry internet service offerings. These attacks take advantage of susceptible devices linked to the internet via the Transmission Control Protocol (TCP) and the Internet Protocol (IP). As a result, current Internet-of-Things (IoT) devices are no longer off-limits. DDoS attacks have become stealthier and more sophisticated as they aim to circumvent conventional detection systems as the number of connected devices has grown. They do this by deploying both low-rate and high-rate DDoS attack techniques.

The higher prevalence of DDoS attacks has motivated academics to investigate detection approaches for large-scale DDoS attacks. As a result, the goal of this research is to conduct an experimental investigation and assessment of detection algorithms capable of detecting both low-rate and high-rate DDoS attacks with high accuracy and low detection delay. We do this by proposing a framework that makes use of statistical characteristics of the incoming IP address, as well as building a testbed that simulates low-rate and high-rate DDoS attacks, on which detection strategies may be evaluated. We investigated the performance of classic statistical change-point detection approaches, machine learning detection techniques, and contemporary deep learning detection techniques in this work.

The study's findings can help researchers enhance the overall effectiveness of detection approaches for early detection of both low-rate and high-rate DDoS attacks with high accuracy and low detection latency.



## Publications Associated With This Research

---

1. Machaka P, Ajayi O, Maluleke H, Kahenga F, Bagula A, Kyamakya K. Modelling DDoS Attacks in IoT Networks using Machine Learning. arXiv e-prints. 2021 Dec:arXiv-2112 (submitted for review).
2. Machaka P., Bagula A. (2021) Statistical Properties and Modelling of DDoS Attacks. In: Vinh P.C., Rakib A. (eds) Context-Aware Systems and Applications, and Nature Of Computation and Communication. ICCASA 2020, ICTCC 2020. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 343. Springer, Cham. [https://doi.org/10.1007/978-3-030-67101-3\\_4](https://doi.org/10.1007/978-3-030-67101-3_4)
3. Machaka, P., & Nelwamondo, F. (2020). Data Mining Techniques for Distributed Denial of Service Attacks Detection in the Internet of Things: A Research Survey. In I. Management Association (Ed.), Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications (pp. 561-608). IGI Global. <http://doi:10.4018/978-1-5225-9866-4.ch030>
4. P. Machaka, A. Bagula and F. Nelwamondo, "Using exponentially weighted moving average algorithm to defend against DDoS attacks," 2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2016, pp. 1-6, doi: 10.1109/RoboMech.2016.7813157.
5. Machaka P., McDonald A., Nelwamondo F., Bagula A. (2016) Using the Cumulative Sum Algorithm Against Distributed Denial of Service Attacks in the Internet of Things. In: Vinh P., Alagar V. (eds) Context-Aware Systems and Applications. ICCASA 2015. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 165. Springer, Cham. [https://doi.org/10.1007/978-3-319-29236-6\\_7](https://doi.org/10.1007/978-3-319-29236-6_7)

## **Acknowledgements**

---

To my Supervisor, Dr Antoine Bagula, I thank you and truly appreciate the support and guidance throughout this research. Thank you for nurturing me and imparting your knowledge and experience.

To my Family for the continued love and support throughout this journey.

To my friends for encouraging me: you all are amazing!



# Table of contents

---

<b>Declaration</b> .....	<b>i</b>
<b>Abstract</b> .....	<b>ii</b>
<b>Publications Associated With This Research</b> .....	<b>iii</b>
<b>Acknowledgements</b> .....	<b>iv</b>
<b>Table of contents</b> .....	<b>v</b>
<b>List of figures</b> .....	<b>viii</b>
<b>List of tables</b> .....	<b>ix</b>
<b>Chapter 1: Introduction</b> .....	<b>1</b>
1.1 Background.....	1
1.2 Problem Statement and Research Questions .....	1
1.3 Research Contributions .....	2
1.4 Thesis structure .....	4
<b>Chapter 2: Background of DDoS Attacks</b> .....	<b>6</b>
2.1 Introduction.....	6
2.2 The Internet of Things .....	7
2.2.1 Evolution of the internet.....	7
2.2.2 What is the Internet of Things (IoT)? .....	7
2.2.3 Characteristics of IoT.....	8
2.2.4 IoT Applications Domain .....	8
2.2.5 IoT Security Concerns.....	10
2.2.6 Summary .....	11
2.3 Introduction to DDoS Attacks.....	12
2.4 DDoS Attacks Background .....	13
2.4.1 Denial of Service (DoS) Attacks .....	13
2.4.2 Distributed Denial of Service (DDoS) Attacks .....	13
2.4.3 The Prevalence of DDoS Attacks .....	14
2.4.4 Attackers' Motives .....	15
2.4.5 DDoS Attack Classifications and Targets.....	16
2.4.6 Common Types of DDoS Attacks.....	20
2.4.7 Commonly Used DDoS Tools.....	22
2.5 DDoS Defence Mechanisms .....	24
2.5.1 Classification based on Defence Points (or Defence Location).....	24
2.5.2 Defence Approaches and Strategies (Point in time defence).....	25
2.6 Summary.....	30

<b>Chapter 3: Literature Review .....</b>	<b>31</b>
3.1 Introduction.....	31
3.2 Change Point Statistical Techniques .....	32
3.2.1 Cumulative Sum (CUSUM) Technique.....	34
3.2.2 Exponentially Weighted Moving Average (EWMA) Technique .....	36
3.3 Machine Learning .....	38
3.4 Deep Learning Techniques .....	42
3.5 Research Challenges.....	45
3.6 Summary.....	46
<b>Chapter 4: Methodology .....</b>	<b>47</b>
4.1 Introduction.....	47
4.2 Research Design .....	47
4.3 DDoS Attack Detection using IP Address.....	47
4.4 DDoS Attack Modelling.....	49
4.4.1 TCP SYN DDoS Attack Modelling.....	49
4.4.2 Synthetic DDoS Attack Traffic Modelling .....	50
4.5 Change Point Detection Algorithms: CUSUM and EWMA .....	52
4.6 Machine Learning Techniques.....	53
4.6.1 Data Processing and Labelling.....	53
4.6.2 Supervised Learning .....	54
4.6.3 Unsupervised Learning.....	56
4.6.4 Semi-Supervised Learning.....	56
4.6.5 Prediction .....	56
4.7 1D Convolutional Network for Time Series and Sequential Data Classifications	57
4.8 Summary.....	59
<b>Chapter 5: Research Findings and Discussions .....</b>	<b>61</b>
5.1 Introduction.....	61
5.2 Algorithm Performance Metrics.....	61
5.3 Algorithm Design (Environment Setup for Algorithm Development).....	62
5.4 Statistical Change Point Detection .....	62
5.4.1 Cumulative Sum (CUSUM) Technique.....	62
5.4.2 Exponentially Weighted Moving Average (EWMA) Technique .....	67
5.4.3 Summary of Statistical Change Point Detection .....	72
5.5 Machine Learning .....	72
5.5.1 Supervised Learning .....	73
5.5.2 Unsupervised Learning.....	74

5.5.3	Semi-Supervised Learning.....	75
5.5.4	Prediction .....	75
5.5.5	Summary .....	79
5.6	1D Convolutional Neural Network Technique .....	79
5.6.1	False Positive Rate and Detection Rate .....	80
5.6.2	Detection Rate and Detection Delay.....	81
5.6.3	Summary .....	82
5.7	Discussions.....	82
5.8	Summary.....	85
<b>Chapter 6: Conclusion .....</b>		<b>86</b>
6.1	Summary of Contributions .....	86
6.2	Future Work.....	88
<b>References .....</b>		<b>90</b>



## List of figures

---

Figure 1 A typical DDoS architecture.....	14
Figure 2 Amplification-based Attack Mechanism .....	17
Figure 3 Reflection-based DDoS Attack Mechanism .....	18
Figure 4 Elements of a Botnet .....	18
Figure 5 TCP Three-way Handshake .....	21
Figure 6 A schematic drawing of the biological neuron .....	38
Figure 7 The Artificial Neuron .....	39
Figure 8 Structure of a Deep Learning Neural Network .....	42
Figure 9 Structure of a Recurrent Neural Network.....	43
Figure 10 Architecture of a CNN.....	43
Figure 11 Low intensity attacks (red line) seen on interval 20-40. ....	51
Figure 12 High Intensity attacks (red line) seen on intervals 20-40. ....	52
Figure 13 Change Point Detection Framework .....	53
Figure 14 Machine Learning System Architecture.....	53
Figure 15 Data Labels for Normal and Attack Traffic.....	54
Figure 16 Data Frame Samples .....	54
Figure 17 ANN Supervised Learning Process.....	55
Figure 18 Architecture of a TCN [162] .....	59
Figure 19 Detection rate and False Alarm rate for varied amplitude factor value ( $\alpha$ ) for both Low rate attacks and High rate attacks. ....	64
Figure 20 Detection rate and False Alarm rate for varied Weighting factor ( $\beta$ ), for both Low rate attacks and High rate attacks .....	64
Figure 21 ROC Curves for low rate attacks .....	65
Figure 22 ROC Curves for high rate attacks .....	66
Figure 23 Graph displaying the trade-off between Detection Rate and average Detection Delay for low rate attacks .....	66
Figure 24 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks .....	67
Figure 25 Detection rate for varied alarm threshold value ( $\alpha$ ), for both Low rate attacks and High rate attacks.....	68
Figure 26 Detection rate for varied EWMA factor ( $\beta$ ), for both Low rate attacks and High rate attacks...69	69
Figure 27 ROC Curves for low rate attacks .....	70
Figure 28 ROC Curves for high rate attacks .....	70
Figure 29 Graph displaying the trade-off between Detection Rate and average Detection Delay for low rate attacks .....	71
Figure 30 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks. ....	71
Figure 31 Average Detection Delay .....	74
Figure 32 K-Means Elbow Method .....	74
Figure 33 KRR predictions for upcoming 15 minutes.....	76
Figure 34 LGR predictions for upcoming 15 minutes.....	77
Figure 35 SVR predictions for upcoming 15 minutes.....	77
Figure 36 KRR predictions for upcoming 3 to 4 hours .....	77
Figure 37 LGR predictions for upcoming 3 to 4 hours .....	78
Figure 38 SVR predictions for upcoming 3 to 4 hours .....	78
Figure 39 ROC Curves for Low Rate Attacks.....	80
Figure 40 ROC Curves for High Rate Attacks.....	80
Figure 41 Graph displaying the trade-off between Detection Rate and average Detection Delay for low rate attacks. ....	81
Figure 42 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks. ....	82

## List of tables

---

<i>Table 1 Advantages and disadvantages of prevention approaches</i> .....	26
<i>Table 2 Advantages and disadvantages of responsive techniques</i> .....	28
<i>Table 3 Advantages and disadvantages of survival approaches</i> .....	29
<i>Table 4 CUSUM and EWMA Overall Performance</i> .....	72
<i>Table 5 Summary of Results for Supervised Learning Models</i> .....	73
<i>Table 6 Summary of Results for Semi-Supervised Learning Models</i> .....	75
<i>Table 7 Summary of Results for the Prediction Models</i> .....	76
<i>Table 8 1-D CNN Overall Performance</i> .....	82
<i>Table 9 Summary of All Algorithm Performance</i> .....	84



# Chapter 1: Introduction

---

## 1.1 Background

The first distributed denial of service (DDoS) assault on the public Internet happened in August 1999 [1]. A year after the initial DoS event, in February 2000, a handful of commercial websites, including Yahoo, CNN, and eBay, saw their first DDoS attacks. A high number of requests overloaded the websites, forcing the company's services to go offline, resulting in considerable financial losses. The July 4 2009 cyber-attack is a well-known example of a DDoS attack [2]. In South Korea and the United States, prominent government, news media, and financial websites were targeted in a series of cyber-attacks. These attacks caused service interruptions and, in some cases, loss of millions of dollars each hour while companies fought to restore their internet services.

Due to the Internet's phenomenal development over the last decade, attackers now have access to a growing number of vulnerable devices. Attackers may now use a large number of these susceptible devices to initiate an attack on the victim server. These attacks have various modes of intensity, more especially those attacks that are designed with low intensity of attack in order to evade detection from current detection techniques. Attackers have diverse motivations for instigating a DDoS attack and this thesis will explore more of these attacker motives in section 2.4.4. However, attackers' tools and attack methods have likewise become more sophisticated. Some of these tool and attack methods are also explored in detail in section 2.4.5 and section 2.4.7.

Parallel to the attackers' mode of operation becoming more sophisticated, researchers have been investigating and developing DDoS attack defence mechanisms. There are various types of defence mechanisms that have been developed so far, the details of which will be further explored in section 2.5. They all have different functions and locations in the network. However, the challenge for the researcher is to develop a detection technique that will detect the start of a DDoS attack accurately and efficiently.

## 1.2 Problem Statement and Research Questions

The ultimate objective of detection techniques is to identify DDoS attacks in real time in order to minimize potential network damage. Therefore, an ideal detection technique must be able to detect an attack with high accuracy (a high detection rate and a low false positive rate) and minimal detection delay.

Designing and implementing an effective DDoS detection technique can be difficult. Even though modern computing capabilities can improve these detection techniques, there is a number of open challenges that still exist as described in [3, 4]. This thesis will focus on the following challenges that are faced by researcher when designing and implementing an effective detection technique:

DDoS attacks can compromise a large number of devices very fast and can damage the network instantly. Almost every day new attack modes (often a variation of those described in section 2.4.5.4) come into existence and the nature of DDoS attacks keeps changing over time as attackers adapt their network attack strategies in order to evade detection. More especially those attack techniques that are designed to evade detection methods by launching low intensity and high intensity attacks. Therefore, the design and

implementation of detection techniques must be adaptable to high-rate and low-rate attacks. This gives rise to the following research question:

- ***RQ 1 : How can we detect low-rate and high-rate DDoS attacks with accuracy and with minimal detection delay?***
- ***RQ 2 : How does the performance of statistics-based techniques, machine learning techniques compare to modern deep learning convolutional neural network techniques when detecting low-rate and high-rate DDoS attacks?***

Researchers have contended with the task of designing methods that identify optimal sets of features while not compromising on efficiency when designing detection techniques.

Researchers also discovered that the source IP address of incoming packets can be used as a useful parameter for detecting the start of an attack. However, the high dimensionality of IP address features, as well as the complicated association between them, results in significant computational overheads, making identification difficult. This is as a result of the fact that many network traffic features typically have low variation or correlations that cause dependencies among the network traffic features. Moreover, the issue with scalability becomes more important when we consider the use of IPv6 address space. Therefore, in this thesis we ask the following question:

- ***RQ 3 : How can the onset of a DDoS attack be identified on the basis of simple features of the source IP address?***

Current DDoS attacks datasets have constraints: these are privacy and legal concerns involved with the sharing of recorded datasets. Thus, there is a lack of actual intrusion data that could be used to simulate attacks and to test and validate new detection techniques. From this challenge, we therefore ask:

- ***RQ 4 : What are the key statistical features of a DDoS attack?***
- ***RQ 5 : How do we model the characteristics of DDoS attacks so that we can simulate and generate practical attack traffic datasets?***

### **1.3 Research Contributions**

The research conducted for this thesis has resulted in a number of research contributions that covers aspects of designing and implementing DDoS attack detection techniques, more specifically focused on the use of statistical and machine learning techniques. The research further investigated and developed a DDoS simulation testbed in MATLAB which uses synthetically generated attack datasets that simulate high intensity and low intensity attacks.

#### **Contribution 1:**

**Chapter 2:** of the thesis presented a detailed background of DDoS attacks and an analysis of the security concerns that are presented by the modern digital transformation, more specifically on the Internet of Things (IoT). The chapter presented an investigation into the ubiquity of DDoS attacks in modern computing. It further

explored the background of DDoS in terms of the motives for initiating an attack and the targets of these attacks. It further investigated the methods, tools and mechanisms used to initiate these attacks. The background of strategies that are currently used to mitigate DDoS attacks were also studied, listing their advantages and disadvantages. **Chapter 3:** presented a theory base for the investigation, a literature survey that covers published research of common detection techniques for DDoS attacks. The chapter presented a comprehensive and critical review of detection techniques that are based on statistics, machine learning and the modern neural networks and deep learning techniques. The chapter further presented research challenges pertaining to DDoS detection and providing the rationale for attempting our research.

The work presented in **Chapter 2:** and **Chapter 3:** highlights key insights into our attempts to partially answer the first research question (**RQ 1**). It provides an understanding of the work that other researchers in this field have already developed and published. Therefore, we provide a platform on which we carry out further investigation in this field.

A significant part of the work presented in **Chapter 2:** and **Chapter 3:** of this thesis was published in the following article and book chapter:

- Machaka, P., & Nelwamondo, F. (2016). Data Mining Techniques for Distributed Denial of Service Attacks Detection in the Internet of Things: A Research Survey. In O. Isafiade, & A. Bagula (Ed.), *Data Mining Trends and Applications in Criminal Science and Investigations* (pp. 275-334). IGI Global. <http://doi:10.4018/978-1-5225-0463-4.ch010>

#### **Contribution 2:**

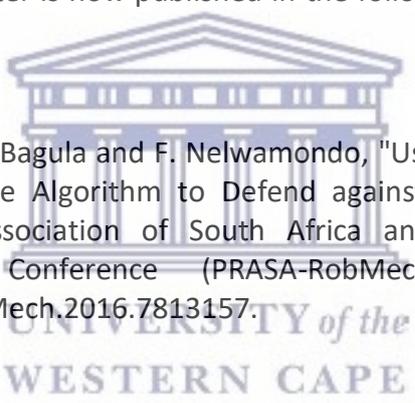
**Chapter 4:** of the thesis describes in detail, the simulation modelling research design methodology that was used in this research. The chapter provided a critical analysis of the published literature on the use of the source IP address for detecting DDoS attacks, and it further justified the use of a source IP address as a key feature for modelling DDoS attacks. The work presented in this section of the chapter assisted with answering the third research question (**RQ 3**). The chapter further investigated reliable benchmark datasets that are used in DDoS attack detection. The insights gathered from the literature survey and investigations provided the basis for analysing and understanding the statistical properties of DDoS attacks. This section answered the fourth research question (**RQ 4**). The chapter also described, in detail, the modelling techniques and the methods used for generating synthetic attack traffic, together with the framework for the testbed. The chapter then described the environmental setup for the development of the testbed and testing environment for the simulations. The work presented in this chapter assists with answering the fifth research question (**RQ 5**).

A significant part of the work presented in **Chapter 4:** was further published in the following research article and book chapter:

- Machaka P., Bagula A. (2021) Statistical Properties and Modelling of DDoS Attacks. In: Vinh P.C., Rakib A. (eds) *Context-Aware Systems and Applications, and Nature of Computation and Communication*. ICCASA 2020, ICTCC 2020.

### Contribution 3:

**Chapter 5:** presents the core findings of the simulations that were proposed for the detection algorithms. It provides a performance analysis of the experiment that simulates a set of high intensity and low intensity DDoS flooding attacks on a network information system. The performance is then analysed for their accuracy and efficiency, and a contextual analysis of these findings is also presented. The research investigated the efficacy of statistical-based techniques, machine learning (supervised, unsupervised and semi-supervised) techniques, and deep learning techniques against low intensity and high intensity DDoS attacks simulation. In order to transfer IP network security from reactive to proactive, we investigated the effectiveness of regression models for predicting possible DDoS attacks. A semi-supervised learning model capable of auto-labelling traffic and accurately classifying malicious traffic was built. This is accomplished through the integration of supervised and unsupervised machine learning models. The work presented in this chapter of the thesis assisted with answering the first research question (**RQ 1**) and the second research question (**RQ 2**). A significant part of the work presented in this chapter is now published in the following research articles and book chapter:

- 
- P. Machaka, A. Bagula and F. Nelwamondo, "Using the Exponentially Weighted Moving Average Algorithm to Defend against DDoS attacks," 2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2016, pp. 1-6, doi: 10.1109/RoboMech.2016.7813157.
  - Machaka P., McDonald A., Nelwamondo F., Bagula A. (2016) Using the Cumulative Sum Algorithm Against Distributed Denial of Service Attacks in Internet of Things. In: Vinh P., Alagar V. (eds) Context-Aware Systems and Applications. ICCASA 2015. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 165. Springer, Cham. [https://doi.org/10.1007/978-3-319-29236-6\\_7](https://doi.org/10.1007/978-3-319-29236-6_7)
  - Machaka, P., Ajayi, O., Maluleke, H., Kahenga, F., Bagula, A. and Kyamakya, K., 2021. Modelling DDoS Attacks in IoT Networks using Machine Learning. arXiv e-prints, pp.arXiv-2112 (submitted for review).

## 1.4 Thesis structure

The remainder of the thesis is organised as follows:

**Chapter 2:** presents an investigation into the security concerns of the Internet of Things (IoT), together with how the technology can be used as a platform to perpetrate DDoS attacks. The research further investigates the landscape of DDoS attacks.

**Chapter 3:** presents some of the most important techniques that are used for DDoS detection. The chapter further presents an investigation into literature as to how researchers have used the classical statistics-based techniques, machine learning techniques and neural networks DDoS detection, their advantages and drawbacks. It further explores the use of modern deep learning techniques for DDoS techniques.

**Chapter 4:** presents the simulation modelling and design that will be implemented in this research. It further describes the modelling techniques and the methods used for generating synthetic attack traffic, together with the framework for the testbed.

**Chapter 5:** presents the core findings of the simulations and the algorithms performance is analysed for their accuracy and efficiency. A contextual analysis of these findings is also presented.

**Chapter 6:** concludes and summarises the key research contributions and proposes possible directions for future research work.



## Chapter 2: Background of DDoS Attacks

---

### 2.1 Introduction

We live in a fast evolving digital information age in which information is at our fingertips. The adoption of Information and Communications Technology (ICT) has enabled on-demand access to information which is very simple and inexpensive. Computing has progressed from the age of the typical desktop computer to the Internet of Things (IoT) paradigm. Many of the items that surround us will be connected to the internet network in some form or another in the Internet of Things. The usage of technology in the IoT has resulted in massive volumes of data that must be saved, processed, and displayed in a flawless, efficient, and readily interpretable manner. Having so many gadgets linked to the internet, however, presents fascinating internet security concerns.

The ubiquitous usage of the internet and network technologies has resulted in a substantial reliance on ICT systems by society. As a result, any failure or disruption in the services supplied by these systems has a direct impact on significant parts of society. Even if it is only for a brief duration, this interruption may be felt strongly. An interruption in a corporate organization's or government's ICT infrastructure, for example, may have a significant impact on their day-to-day operations. This might result in considerable financial losses (business and legal actions) as well as higher operating costs as a result of fraudulent operations.

The resultant interruptions might be the consequence of a hacker attempting to disrupt systems through Denial of Service (DoS) attacks. A DoS attack is a malicious attempt by an attacker to disrupt a service provider's online services, rendering them inaccessible to legitimate users. The Distributed Denial of Service is a large-scale version of DoS (DDoS). This type of attack on a company might have disastrous consequences and result in an unsatisfactory service to consumers and significant financial losses. It may also lead to intellectual property losses for an organization, affecting the long-term competitiveness of corporations and governments in industrial and military espionage events [5]. As a result, it is critical that organizations and governments implement procedures and strategies that will assist them in correctly and reliably detecting the beginning and occurrence of DDoS attacks.

This chapter investigates the Internet of Things phenomenon in relation to DDoS attacks. It gives a historical overview of the internet and how it developed into the Internet of Things. In addition, the chapter delves into the features of an IoT system. In addition, the present application of IoT applications in households and businesses is explored. The study delves deeper into the Internet of Things' security concerns, as well as how the technology may be utilized as a platform to perpetrate and even inject threats and attacks. The study expands on the environment of distributed denial-of-service attacks by seeking to address the following questions:

- What are DDoS attacks? How pervasive are these attacks?
- Why are DDoS attacks executed? How are DDoS executed?
- What is the attack targeting? Which DDoS attacks are common?
- What strategies and mechanisms are used for a successful DDoS attack? Which tools are used to conduct a DDoS attack?
- What defence mechanisms are currently used to combat DDoS attacks? What are their advantages and disadvantages?

The sections that follow will give a background of how the internet evolved into the IoT. It will explore the characteristics and current applications of the IoT paradigm. The security concerns that are present with the IoT are also discussed. The chapter further introduces the concept of DDoS; the various attack types and its background. The rationale and motives, together with methods and tools used to conduct a large scale DDoS are also presented in this chapter. The various defence mechanisms that are currently implemented in industry are explored together with their advantages and disadvantages. The chapter further presents the state-of-the-art literature review of detection algorithms that are integrated into the defence mechanisms and network intrusion detection systems. The challenges of defending an information system against a DDoS in an IoT environment is also highlighted, together with the research questions that will be explored by this thesis.

## **2.2 The Internet of Things**

To comprehend the (IoT) concept, one must first understand the evolution of the internet.

### **2.2.1 Evolution of the internet**

The creation of the telegraph, telephone, radio, and electronic computer marked the beginning of communication technology. In the 1960s, the US Department of Defence granted contracts for ARPANET development (Advanced Research Projects Agency Network). The ARPANET was the first packet switching network to use the TCP/IP protocol suite. This technique formed the Internet's technological underpinning. The technology was commercialized in the early 1990s and evolved into what is today known as the World Wide Web (WWW).

The WWW grew in popularity, with massive use and rapid expansion. Mobile devices became capable of connecting to the internet, resulting in the formation of the mobile internet. This technology enabled gadgets and people to be linked from remote areas, and social networking platforms arose as a result. This enabled an increasing number of devices to be linked and communicate with one another over the internet. This was the beginning of the internet of things (IoT).

### **2.2.2 What is the Internet of Things (IoT)?**

The "Internet of Things" was first mentioned by Kevin Ashton in 1998 [6]; and the term was formally introduced by the ITU (International Telecommunications Union) in their internet report of 2005 [7]. Since then there have been several debates on the precise definition of the IoT. This is due to the influence of several contributing fields, from information technology engineering, academia and marketing. For the purposes of this work we will define IoT as [8]:

"The Internet of Things allows people and things (or objects) to be connected Anytime, Anyplace, with anything and anyone; ideally using any path/network and any service."

The number of gadgets that humans utilize to generate and receive information has increased dramatically during the last decade. According to reports, there were 500 million devices in 2003, which rose to 12.5 billion internet-enabled devices in 2010. [9]. It is projected that by 2020, between 50 billion and 100 billion such gadgets would be connected to the internet [10].

### **2.2.3 Characteristics of IoT**

The key properties of the IoT are investigated in this section of the work. Although this is not an exhaustive list of IoT characteristics, those that are relevant to internet and network security are highlighted as follows: size and population; complexity and intelligence; heterogeneity and interoperability.

#### *2.2.3.1 Size and Population*

The number of devices or entities participating in IoT is rapidly increasing. It is projected that by 2020, there will be 50-100 billion internet-connected gadgets [10]. These are information-embedded devices with the capability of transferring data through a specific method or channel of transmission, which may be abused by an attacker. The growing number and population of internet-connected devices creates a sea of accessible options for an attacker to employ throughout the internet, in addition to that specific domain application, to launch an attack.

#### *2.2.3.2 Complexity and Intelligence*

IoT devices communicate with one another autonomously. The complexity and intelligence capabilities of the gadget determine how this interaction varies from one domain application to the next. A device's capabilities, such as storage and computing power, may be constrained. As a result, the device has low-complexity and has a narrow attack space. A device, on the other hand, may have a lot of storage and processing power, making it a highly sophisticated device with a lot of attack space. The population of these intelligent devices is growing, and they are becoming more affordable to purchase, posing a danger to the internet and the domain in which they are deployed.

#### *2.2.3.3 Heterogeneity and Interoperability*

Because of their pervasiveness, the demand for and population of sophisticated intelligent devices is projected to rise fast. They may function in heterogeneous environments and offer chances for interoperability between entities inside an environment as well as across the IoT platform. There are advocates for the need for universal interoperability and open standards in the IoT, but progress has been hindered by industries' unwillingness to collaborate. This is due to the fact that universal interoperability is accompanied with the difficulty of accountability and security management. As a result, industries and stakeholders are fully accountable and liable for identifying, preventing, and resolving security concerns in a decentralized form.

### **2.2.4 IoT Applications Domain**

The vision and idea of IoT provide a wide range of capabilities that will allow ICT innovation in a variety of application fields. By spanning a wide variety of business areas, IoT has the potential to increase competitiveness across diverse application domains

and open up new venture opportunities. As a result, IoT technologies have an influence on a variety of application areas, which may be characterized based on the kind of network accessible, coverage, scalability, heterogeneity, repetition, user participation, and impact [11]. The work of authors in [12-14] attempted to identify IoT application areas, and all of their classifications have common components; hence, the classification below will be used for the sake of our study:

- Personal and Home Applications
- Utilities and Environment applications
- Smart Business and Smart Cities Applications

The subsections that follow will describe and offer examples of the various application domains.

#### *2.2.4.1 Home and Personal Applications*

The IoT platform also enables users to create smart homes by utilizing IoT technology. IoT may be utilized in household applications such as those that allow users to operate home appliances such as refrigerators, air conditioners, washing machines, and others. This enables the user to better control his or her house and energy consumption. Authors in [15] have created a prototype design and architecture for an internet-based application that automates functioning home gadgets.

IoT technology may be used in home applications such as Ubiquitous Health and e-Health to aid in the development of assisted-living solutions. These programs enable clinicians to monitor the elderly and sick in their own homes, lowering the expense of in-hospital monitoring and allowing for earlier treatment and intervention. For example, the elderly and sick will be equipped with medical sensors that will monitor health indicators such as blood pressure, body temperature, respiratory activities, and mobility. Smartphones and other wearable devices include accelerometer sensors, position and proximity sensors, and gyroscope sensors, which may be used to collect data about their patients' activities in their homes. Local medical staff utilize the observed data to make decisions about the patient's health and to respond quickly when necessary [16].

Wearable technology and personal body area networks are two more common examples of IoT personal applications. These sensor technologies are linked to cell phones using Wi-Fi or Bluetooth technology to measure everyday activities such as exercise routine performance, distance ran, steps walked, and calories burnt. This data may be used to track a patient's or user's progress toward certain goals in order to improve their lifestyle or prevent health issues [17].

#### *2.2.4.2 Utilities and Environment Applications*

IoT provides a viable platform for utility and environmental monitoring applications. Municipalities and utility corporations use some of these applications in the form of smart metering for water and energy supply. These applications are made up of large networks of sensors, usually on a regional and national scale. These applications are used to monitor consumer consumption in order to better manage resources and optimise services [18].

The same technology can be extended and applied to monitor drinking water and irrigation systems. Sensors are strategically positioned to monitor vital water and soil

characteristics. This can assist to avoid water and soil pollution, as well as make better decisions about drinking water and agriculture [19].

IoT technologies enable the widespread installation of sensors in a variety of critical locations in a seamless and self-managing way. Sensor technologies enable them to communicate with one another and report on real-time occurrences. These sensors can be put in high-risk regions where human presence could be hazardous (examples: volcanic areas and oceanic abysses). This can help monitor and detect anomalies in the environment (such as wildfires [20]) that can jeopardize human and animal life.

#### *2.2.4.3 Smart Business and Smart Cities Applications*

IoT technology has the potential to enable a wide range of innovative and diversified applications in smart city scenarios. Sensor technologies may be installed and used to offer sophisticated traffic control systems; this monitors vehicle traffic in large cities and on highways so that alternate traffic routing recommendations can be supplied in instances of congestion. This application may be expanded to include sensors that monitor the wear and tear on highway and road network infrastructure. Sensors may be installed in a metropolitan environment to further monitor and identify pollution levels in the air. They can accomplish this by measuring carbon dioxide, nitrogen oxide, ammonia, and other contaminants. [13].

Businesses may also profit from the usage of IoT technology; for example, in the supply and delivery sectors, RFID and sensor technologies have been utilized for inventory management. These enable firms to trace the flow of commodities and products throughout the various stages of their life-cycle. RFID has been utilized in retail to minimize theft and the counterfeiting of items. Bio sensor technologies have been utilized to monitor the manufacturing process of products in the food industry. The sensors can assist in ensuring that the manufacturing process of food items fulfils the established quality requirements and, as a result, can determine the degradation or shelf-life of the products [12].

Security surveillance is another significant application of IoT technology that helps both the government and companies. Security surveillance is a popular camera network technique in commercial buildings, retail malls, airports, road intersections, parking lots, and other public areas. When IoT technologies such as ambient sensors, infrared, and advanced video analytics are integrated, companies will be able to follow targets, detect suspicious activity, and locate left luggage, as well as prevent unauthorized access [12, 13].

#### **2.2.5 IoT Security Concerns**

The security of IoT technology and applications is a key problem for their mainstream adoption. Industries and stakeholders are reluctant to embrace IoT technology and applications if system and data security, privacy, and trust are not guaranteed. Several security issues have arisen as a result of the lobbying for an open IoT platform. We focus on security problems in terms of cyber-attacks and their implications for the IoT platform in our research.

### *2.2.5.1 Data Confidentiality*

Data is a valuable asset that must be safeguarded in various IoT application fields. In a business application domain, for example, data is an asset that is utilized to boost an organization's market competitiveness. In this instance, data confidentiality is crucial since it ensures that only those stakeholders and entities with permission may access and alter the data created by the program. Because of the nature of IoT, entities may be mobile and interoperable across application domains. This makes authentication and access control to the application system and information more difficult. This increases the vulnerability to the IoT application by allowing attackers with access to the system to compromise its entities and launch an attack against the environment.

### *2.2.5.2 Privacy*

Privacy is another critical security problem for the IoT platform. Data privacy is concerned with setting data access rules; which types of data can be accessed by which person or organization in the application area. This is especially crucial in an application domain like Ubiquitous-Healthcare, because these apps handle personal and sensitive data. As a result, it is critical to put in place proper data privacy safeguards. The Internet of Things platform enables the usage of ubiquitous wireless communication channels. However, if these communication channels are used inappropriately, they may pose dangers and may violate data privacy regulations. This might be because remote access capabilities are susceptible to eavesdropping, masquerade assaults, and denial of service attacks.

### *2.2.5.3 Trust*

Trust in information systems relates to security policies that govern access to resources and the credentials necessary to comply with the regulations [21]. To exchange credentials, trust negotiation techniques are employed. This is to allow persons or devices requesting services or resources to communicate with the device or entity delivering those services or resources. This is a typical decentralized and static approach to trust negotiation and management. It may not work well in an IoT environment where various individuals, entities and devices are heterogeneous, mobile and broadly distributed in physical space. Owing to the IoT's dynamic and decentralized nature, trustworthiness is particularly difficult to achieve because an attacker may construct a masquerade attack in order to acquire access and control of the physical and logical system, and therefore get access and control of the data created by the system.

## **2.2.6 Summary**

The Internet of Things is a new paradigm of internet computing that is rapidly expanding. This constant progress in IoT technology will be interwoven into our lives, changing the way we live and the way businesses operate. However, as the number of networked and computational devices grows, so does the potential danger and attack space. As a very large number of these computer devices are becoming more mobile and dispersed in their design, present security methods and technologies may need to be significantly improved.

Because of the features and structure of IoT, attackers have more possibilities and a wider range of target systems to exploit. As a result, more complex tools and attacks will become accessible for attackers to use in order to carry out an attack. This will make

ensuring data security, privacy, and trust a difficult endeavour. As a result, new security techniques for information systems must be continually investigated and developed in order to enhance and assure the safety and security of the IoT environment.

The attacks and threats that are relevant to this research are those that interrupt, degrade, deny, or destroy the IoT application's services. Denial-of-Service (DoS) attacks are the most widely utilized suite of attacks for the goal of disrupting and degrading systems. Because of the distributed nature of the IoT platform, the attacker can utilize it to carry out and even insert a distributed denial-of-service (DDoS) attack in the IoT application. The next section will provide a thorough study of the nature of DDoS attacks and their defence methods.

## **2.3 Introduction to DDoS Attacks**

The first DoS attack on the public Internet occurred in August 1999 [1]. A year after the initial DoS event, in February 2000, a handful of commercial websites, including Yahoo, CNN, and eBay, saw their first DDoS attacks. A significant number of requests overloaded the websites, forcing the company's services to go offline, resulting in considerable financial losses. The July 4th, 2009 cyber-attack is a well-known example of a DDoS attack. [2]. In South Korea and the United States, prominent government, news media, and financial websites were targeted in a series of cyber-attacks. These assaults caused service interruptions and, in some cases, loss of millions of dollars each hour while companies scrambled to restore their internet services.

In 2009, Forrester Consulting conducted a survey [22] to investigate DDoS threats and prevention in the United States and Europe. Four hundred IT decision-makers and security experts participated in the survey. These responses came from a wide range of industries and organizations that conduct business online or have invested heavily in their online brand and reputation. Online banking (e-Banking), e-commerce, online multimedia (news, video, and audio), and entertainment were all covered (online gaming). DDoS attacks have been demonstrated to be so common that they have risen to the top of the list of IT security concerns. In the preceding 12 months, over 75% of respondents indicated they had suffered one or more "targeted" DDoS attacks on their companies.

According to the Forrester survey [22], many companies are concerned about the potential of DDoS attacks. Only a few businesses have dedicated DDoS defence systems in place, according to the survey. Those that tackled the DDoS issue discovered that the existing systems lacked the capacity and agility to neutralize assaults quickly before they reached the whole network. The inability of standard DDoS defence systems to properly and quickly protect networks may be ascribed to the DDoS terrain's exceptional and rapid changes.

The nature of DDoS evolves over time as hackers update their attack tactics to circumvent current intrusion detection measures. DDoS assaults have grown larger and are stealthier, more targeted and smarter than ever before. Almost every day, new and sophisticated DDoS attacks with unknown motivations are launched (for social, political or criminal motives). The difficult task is thus to design adaptive DDoS defence measures capable of detecting large-scale attacks adequately and quickly. As a result, it is critical

to create security systems capable of detecting DDoS attacks accurately (with a low false alarm rate) and reliably (with low detection delay).

## **2.4 DDoS Attacks Background**

A DoS attack is one that is intended to render a computer or network incapable of providing normal services to its legitimate users. It is the outcome of malicious activity carried out by an attacker with the goal of preventing or degrading a computer's or network's functions. The next section examines the features of DoS attacks as well as the tools used to coordinate these operations.

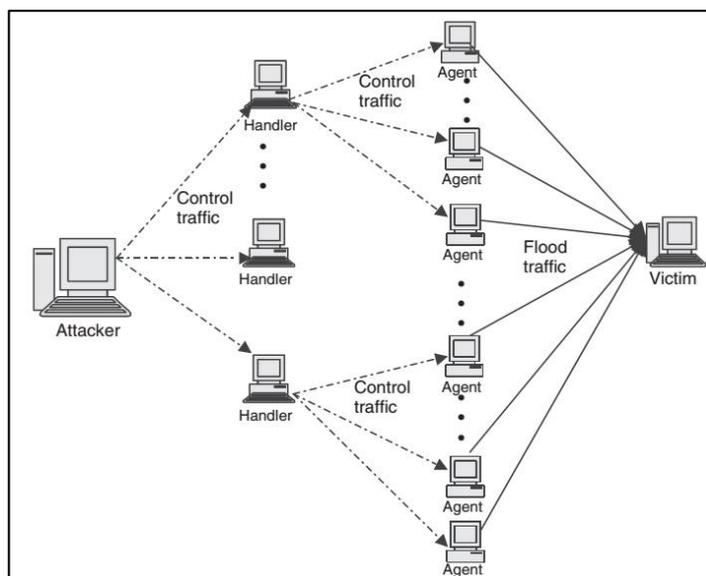
### **2.4.1 Denial of Service (DoS) Attacks**

A DoS attack is a malicious attempt by an attacker to disrupt a service provider's online services, rendering them inaccessible to legitimate users [23, 24]. These attacks constitute a serious threat to the provision of internet services. The attack renders certain network services (such as the internet and email) inaccessible by depleting the resources such that the targeted service is no longer accessible to its authorized users. The DoS attack might be directed against computing resources such as the CPU or a network resource such as bandwidth.

The first documented DoS attack occurred during the week of 1 February 2000[1]. This was a premeditated attack targeting e-commerce sites such as Amazon.com and eBay.com. These attacks employed computers in various locations to overwhelm the merchants' systems and prevent legitimate commercial traffic from reaching their websites. A DoS assault can cause anything from a slight increase in service response time to full inaccessibility, as well as financial consequences for organizations that rely significantly on the availability of their internet services.

### **2.4.2 Distributed Denial of Service (DDoS) Attacks**

The Distributed Denial of Service (DDoS) attack is a large scale variant of DoS attack. To generate a large-scale DoS, the attacker employs a network of compromised machines (bots, zombies, slaves, or agents) to attack a single target-victim. The compromised machines will be infected (without the owner's knowledge) with a Trojan or backdoor application that will allow the attacker (bot-master) to take control of them. The master-bot will direct all bots to flood the target system with a high number of false requests at the same time and repeatedly. This will eventually result in a DDoS attack on genuine users of the targeted victim's system. This is seen in Figure 1, which depicts a typical DDoS assault architecture.



**Figure 1 A typical DDoS architecture**

A DDoS attack depletes the victim's essential resources. This is accomplished by initiating a high-rate flooding (HRF) attack on the target with a huge number of packets, therefore depleting the victim's available network or computer resources. The most popular HRF attacks include buffer overrun, ping of death (PoD), TCP SYN floods, UDP floods, smurf, Neptune, teardrop, and land. For example, in February 2000, Yahoo suffered a large advertising income loss as a result of a two-hour-long DDoS attack. In December 2000, financial institutions such as PayPal, Visa.com, and MasterCard.com were subjected to a DDoS flooding attack. In another case, 9 major US banks were constantly subjected to a series of DDoS flooding attacks as of September 2012 [2].

DDoS attacks have been used by hackers for a long time, and their attack patterns have evolved significantly in order to circumvent existing defence systems. DDoS assaults have become more sophisticated, diverse, and unconnected, making them a significant concern.

### 2.4.3 The Prevalence of DDoS Attacks

According to Prolexic Technologies' Q4 2012 study [25] on Global DDoS assaults [25], there has been a general increase in DDoS tools and attacks. The majority of assaults were directed against critical infrastructure services with a high volume of online service rendering. Clients include those in the financial industry, e-commerce, software-as-a-service (SaaS), and energy sectors, as well as government agencies. When compared to the fourth quarter of 2011, the overall number of new DDoS attacks grew by 19% in 2012. When compared to the third quarter of the previous year, the average duration of a DDoS attack has increased from 19.2 hours to 32.2 hours.

The capability and ubiquity of DDoS attacks stems from the fact that DDoS attacks primarily exploit the Internet infrastructure. The Internet was created with usability in mind, not security. The Internet is a straightforward network designed for packet forwarding. The Internet's design adheres to the end-to-end paradigm. The duty for establishing security features is left to the sender and receiver (end-users) of the two-

way connection in this architecture. As a result, many security flaws have arisen, which have been exploited by attackers for DDoS assaults. According to the authors' work in [26] and [23], the following architectural flaws render the internet vulnerable to DDoS attacks:

- *Internet security is highly interdependent* – Regardless of how secure the victim's system is, it is still vulnerable to a DDoS attack since it is dependent on the overall security of the worldwide Internet.
- *Internet resources are limited* - Each internet system has a finite quantity of resources that can be utilized by a sufficient number of users.
- *Accountability is not enforced* – IP spoofing allows attackers to carry out attacks while avoiding accountability for their activities.
- *Intelligence and resources are not collocated* – The Internet's end-to-end communication architecture has minimized the amount of processing on the intermediate network, allowing packets to be sent effectively and at a low cost. The end host was in charge of ensuring intelligence for service. Meanwhile, research and the need for high throughput have resulted in the development of high bandwidth paths in the intermediate network. As a result, attackers are given the chance to take advantage of the plentiful resources on an innocent network in order to overwhelm a victim with malicious messages.

#### 2.4.4 Attackers' Motives

The motives for DDoS attacks vary and few researchers [27] have attempted to analyse and infer attacker motives in order to improve decision-making, risk-assessment and proactive cyber defence. The attacker's motive for launching DDoS attacks may then be divided into the five categories listed below [23]:

- *Financial/economic gain*: These sorts of attacks are a big source of concern for businesses which conduct their operations online or have made a considerable investment in their online branding. This is mostly due to the fact that the attacker who conducts this sort of assault is the most technically advanced and experienced. These assaults are hazardous, difficult to detect, and difficult to stop.
- *Revenge*: Typically, the root of an attack is frustration. Attackers in this group are likely to be persons with few technical abilities who carry out the attack in response to a rumoured suppression.
- *Ideological belief*: Some attackers are driven by ideological beliefs. For example, the political motives behind DDoS attacks on the Wikileaks website.
- *Intellectual Challenge*: This group of attackers consists of young hackers who are experimenting and learning how to carry out various sorts of attacks.
- *Cyber Warfare*: Cyber warfare attackers are thought to be extremely hazardous, well-trained, and well-equipped. They are generally political in nature and are members of a military or terrorist organization. They target telecommunications companies, energy infrastructure, and financial institutions. These sorts of attacks devote the majority of their time and resources to disrupting services. This may cause a country's vital services to be disrupted, and as a result, the organization may incur significant economic losses.

## 2.4.5 DDoS Attack Classifications and Targets

DDoS attacks have various outcomes and target different locations in a network; nevertheless, they always aim to disrupt and terminate a service of their targets. DDoS attacks might vary depending on their target. A DDoS attack against a network is distinct from an attack against an application. DDoS attacks have been described in a number of ways in literature [4, 23, 24, 26, 28-32]; DDoS assaults are classified in the surveys based on their target, degree of automation, exploited weakness, attack rate dynamics, and impact on the victim. The various categories are discussed in the subsections that follow [26].

### 2.4.5.1 Classification based on DDoS Attack Targets

DDoS attacks can target many places in the communication channel's OSI model. The attack may have different consequences at each location, but the ultimate goal is to degrade and interrupt services for their intended consumers. DDoS targets are classified as either network-transport layer targets or application level targets.

#### 2.4.5.1.1 Network-Transport Level Targeting DDoS Attacks

The volume of traffic supplied to the victim, rather than the substance of the packets, is the primary objective of Network/Transport layer DDoS attacks. They are aimed towards the OSI model's Network Layer (Layer 3) and Transport Layer (Layer 4). The goal of these attacks is to exhaust the target victim's incoming network capacity by delivering enormous numbers of packets. An attacker may also use IP address spoofing techniques to avoid detection. TCP SYN, UDP, and ICMP flooding attacks are examples of this sort of attack (discussed in a section that follows).

#### 2.4.5.1.2 Application Level Targeting DDoS Attacks

DDoS attacks at the application level take advantage of a flaw in the design and implementation of the program running on the target victim system. They are not 'volume-based' attacks and they often use less bandwidth. They employ low-rate attack methods to deny legitimate users access to the services supplied by the target victim computer. They employ 'apparently genuine' attack packets and are difficult to detect. The HTTP request attack and the DNS Amplification assault are two examples of these attacks.

This research will look at the most frequent DDoS flooding attacks, in which the attacker tries to interrupt a legitimate user's connectivity by depleting bandwidth, router processing capacity, or network resources (network and transport layer flooding attacks). This study does not focus on attacks that exhaust server resources such as sockets, memory, and database bandwidth (application layer flooding attacks).

### 2.4.5.2 Classification Based on Degree of Automation

Attacks can also be categorised based on how automated they are. There are several ways for launching an HRF DDoS attack; they can be conducted manually through human coordination or automatically through the deployment of botnets [26].

#### 2.4.5.2.1 Attack Coordination

*Manual Attacks:* The first DDoS attacks were launched manually. This sort of attack necessitates extensive human cooperation in order to be effective against the target. The attacker will actively search for susceptible and compromised remote workstations

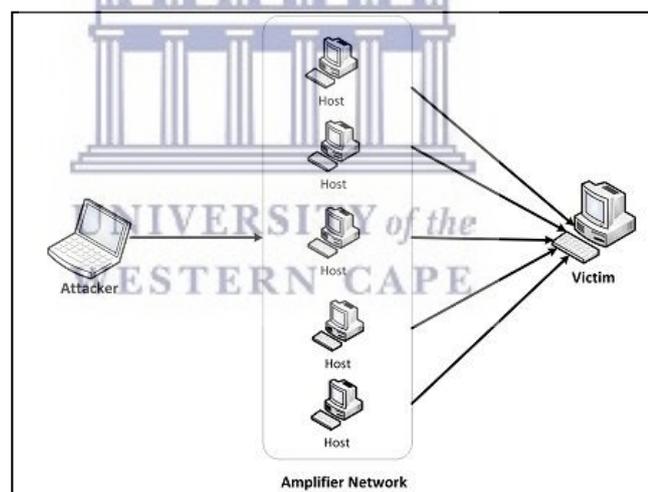
and break into them in order to install an attack code/program that will control the attack's initiation. The overall volume of traffic delivered to the target may crush its online services, depending on the number of computers engaged in the attack, their computing power, and the sophistication of the attack tool utilized. This attack technique was used in early DDoS attacks. This coordinating method, however, has become outdated due to the availability of powerful computers and rapid internet connections.

*Semi-automated and Automated Attacks:* A master machine and agents (slave, daemon, and zombie) machines are used in these attack launch techniques. The installed attack code pre-programmes the attack's initiation, length, attack type, and victim. Because the attacker is only involved at the beginning of the recruiting process, he is exposed to the least amount of risk.

Amplification and reflection-based attacks are the most often employed automated attack methods [2].

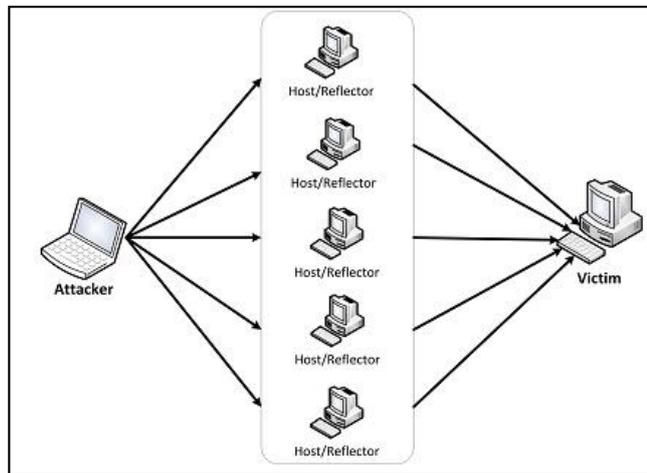
#### 2.4.5.2.2 Attack Strategy

*Amplification-based Attacks:* Attackers take advantage of the network services to allow them to generate multiple messages, for every message sent (Internet Control Message Protocol-ICMP Echo request), in order to amplify traffic towards the victim. In this attack mechanism, the amplification network is a network of host machines that enable IP broadcast messages. The broadcast message is sent using a spoofed IP address, and it triggers a response from every host machine in the amplification network directed towards the victim. This is depicted by Figure 2.



**Figure 2 Amplification-based Attack Mechanism**

*Reflection-based attacks:* A group of hosts (reflector network) are employed to launch this attack. A reflector is one that may send a reply message (SYN+ACK) to the packet's spoofed source IP address in response to an incoming message (e.g. SYN) (of the intended victim). Reflectors are often web servers and DNS servers. The reflection attack will cause a huge amount of network traffic to be directed towards the chosen victim. Figure 3 illustrates this.

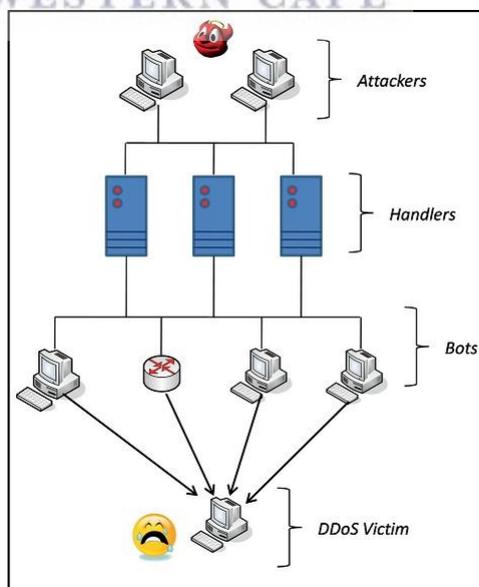


**Figure 3 Reflection-based DDoS Attack Mechanism**

#### 2.4.5.2.3 Attack Components

To coordinate a large-scale DDoS assault, reflection and amplification attack methods may be employed together. Botnets have long been utilized to engage in both reflection and amplification attacks. Botnets are made up of three components: masters, handlers, and bots [23, 26, 33]. Figure 4 illustrates the elements of a Botnet-based attack [2].

- *Masters (Attackers)*: These are attackers who transmit command and control instructions to both handlers and bots in order to launch and carry out an attack.
- *Handlers*: Malicious applications known as handlers are installed on infected host PCs. Attackers (the master) employ handlers to interface with and control the bots indirectly. Attackers interface with handlers and bots through a variety of techniques, most recently Instant Messaging (IM) and Internet Relay Chat (IRC).
- *Bots*: These are infected host machines with the malicious handler software installed. They will be the machines employed to carry out the large-scale coordinated attack.



**Figure 4 Elements of a Botnet**

The coordination and launch of a DDoS attack include numerous phases [23, 34].

- *Recruitment and Selection of Agents.* In this phase, the attacker chooses the agents who will carry out the attack. This choice is made depending on the nature of the machine's current vulnerabilities. These devices have been hacked in order to be deployed as agents. Normally, these machines are leveraged for their considerable resources in order to produce a strong attack stream.
- *Compromise.* The attacker installs the attack software by exploiting security weak points in the agent machines. The attack software must be installed in such a way that it can elude detection and removal (in the form of a Trojan, malware or virus). Except when utilizing a sophisticated defence mechanism (for example anti-virus and anti-malware software), it is typically impossible for the users and owners of the agent machines to understand that they are engaging in a DDoS attack system. In terms of processing power, memory, and bandwidth, the attack software utilized to infiltrate the agent computers are highly cost efficient. As a result, they have little effect on the system's performance.
- *Communication.* The attacker interfaces with a broad range of handlers using different protocols such as ICMP, TCP, and UDP. In this interaction, the attacker wants to know which agents are operational, when to launch attacks, and when to upgrade the handlers. In recent years, the attacker and the agents have interacted using an online, multi-user messaging system known as the Internet Relay Chat (IRC) channels. The use of IRC channels for DDoS attacks has grown in popularity owing to three primary advantages: it gives a high level of anonymity, it is difficult to detect, and it provides a robust, assured delivery mechanism.
- *Attack.* To begin the attack, the attacker sends a command. The victim, duration of the attack, and unique aspects of the attack, such as type, length, TTL, and port numbers, can all be customized. If there are significant differences in the characteristics of attack packets, it is advantageous to the attacker since it resists detection.

UNIVERSITY of the  
WESTERN CAPE

#### 2.4.5.3 Classification based on Exploited Vulnerability

DDoS attacks leverage a variety of system vulnerabilities; nevertheless, categorization based on exploited vulnerability might be semantic or brute-force attacks. Semantic attacks use a specific feature or implementation fault of a protocol or program installed on the target to deplete an excessive amount of the victim's available resources. Brute-force attacks can be used instead of semantic attacks. They are carried out by sending a large number of ostensibly valid packets to the target. Because the intermediary network can supply higher amounts of traffic than the victim can manage, this will gradually or instantly exhaust the victim's available resources.

There is a significant distinction between semantic and brute-force assaults. They both deplete the victim's limited resources. The distinction is that a victim can significantly reduce the impact of a semantic attack by changing the exploited protocol or using proxies. In a brute-force attack, the victim may be powerless to stop the attack. An attacker would typically decide to employ a brute-force technique if a victim mitigates a semantic attack by changing an exploited protocol or implementing proxies.

#### *2.4.5.4 Classification based on DDoS Attack Rate Dynamics*

DDoS attacks may be categorized depending on attack rate dynamics: constant rate attacks and variable rate attacks. This classification is explained further below.

##### *2.4.5.4.1 Constant Rate Attacks*

This format is used by a large number of attacks. Attack packets from the attack agents are sent at a steady and consistent pace in this type of attack and from the start of the attack. The agent machines create attack packets leveraging all available resources at the same time, with no breaks or variations in attack rate. The effect of such an attack on a person is immediate, constant, and unexpected.

##### *2.4.5.4.2 Variable Rate Attacks*

Variable rate attacks do not employ an instantaneous offensive tactic. Their attack tactic has a variable rate of occurrence. They are able to evade detection in this manner. Variable rate attacks can modify a defence mechanism training model in such a way that it avoids detection. Variable rate attacks can be further classified as increasing rate attacks or fluctuating rate attacks.

Increasing rate attacks begin with a modest rate of attack and steadily grow until all resources are depleted. This approach postpones discovery of the attack by gradually diminishing the victim's services over time. Fluctuating rate attacks change their attack rate dependent on the victim's behaviour and response to the attack. In this type of assault, the victim will experience intermittent service disruptions since the attacker may sometimes alleviate the attack impact from the victims in order to avoid detection.

#### *2.4.5.5 Classification based on DDoS Attack Impact on Victim*

A DDoS assault can have two effects on a victim. The effect might be disruptive or deteriorating. A disruptive attack is one that causes the victim's resources to be completely depleted, preventing service to the victim's clients. This category contains the vast majority of attacks. A degrading attack is one that depletes the victim's resources. The attack will slow down the victim's services, preventing genuine users from accessing them. Because the attack does not completely interrupt service, it can go unnoticed for a long time.

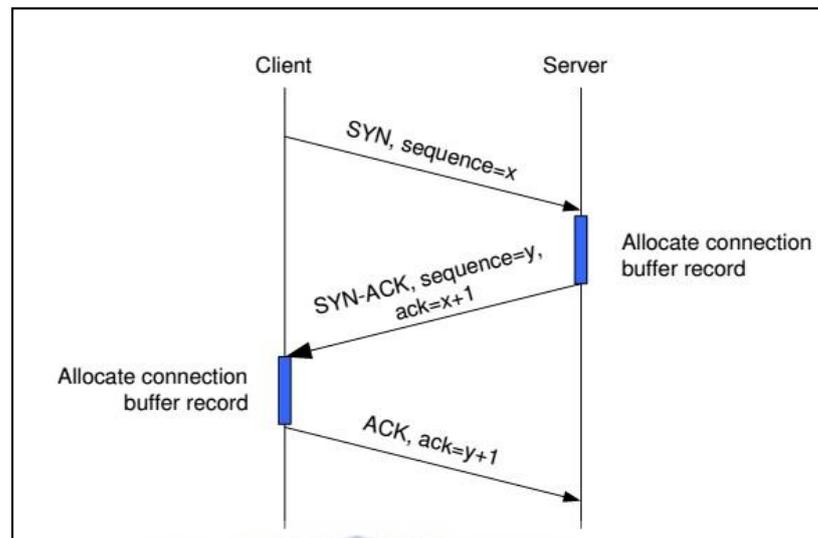
### **2.4.6 Common Types of DDoS Attacks**

According to a Kaspersky Lab [35] 2020 second quarter report, the most prevalent attacks were TCP SYN flood, HTTP flood, UDP flood, and ICMP flood. These most prevalent forms of flooding attacks are detailed in the subsections that follow.

#### *2.4.6.1 TCP SYN Flooding Attacks*

TCP SYN flooding is a type of network layer flooding assault that is one of the most popular and potent flooding tactics. It makes use of the flaws in the TCP three-way handshake; this is illustrated by Figure 5 below. In a typical TCP connection, the client requests a connection by sending a SYN packet to the server. The server will start a connection session and respond with a SYN ACK packet upon receiving the connection request; by doing so, the server saves information of the desired TCP connection in the memory stack and assigns resources to this open session [23].

The connection is still half-open, i.e. in the SYN RECVD state. The client must confirm the connection and respond with an ACK packet to complete the three-way handshake with the server. The server will then examine the memory stack for an existing connection request before moving the TCP connection from the SYN RECVD to the ESTABLISHED state. If no ACK packet is provided within a certain amount of time, the connection will timeout, releasing the assigned resources [34].



**Figure 5 TCP Three-way Handshake**

The attacker sends a large number of SYN packets to the target server in a TCP SYN flooding attack. These packets typically include spoofed IP addresses, which are IP addresses that are either non-existent or are not in use. TCP SYN floods can also be launched from compromised computers with valid IP addresses; however, the machines must be configured so that they do not react to or acknowledge a SYN ACK packet from the target server. As a result, the server will not receive any ACK packets from clients in response to the 'half-open' connection request.

During a high-rate flooding attack, the server will keep a significant volume of unfinished three-way handshakes and assign resources to fake connection requests for a length of time. The server will continue to receive fake requests until its resources are depleted. This will prohibit the server from processing new requests as well as genuine client requests [36].

#### 2.4.6.2 HTTP Flooding Attacks

HTTP flooding is an example of a high rate flooding attack that targets the application layer. The attacker employs a massive army of compromised computers and botnets to send a high volume of ostensibly valid HTTP requests (GET and POST) to the target server for a web page or a huge file. To fulfil this request, the server may dedicate substantial resources in the CPU, memory, and bandwidth. An attacker's HTTP requests may masquerade as legitimate requests. As a result, the server will treat them as usual, making them exceedingly difficult to filter and eventually exhausting the server's resources [2].

#### 2.4.6.3 UDP Flooding Attacks

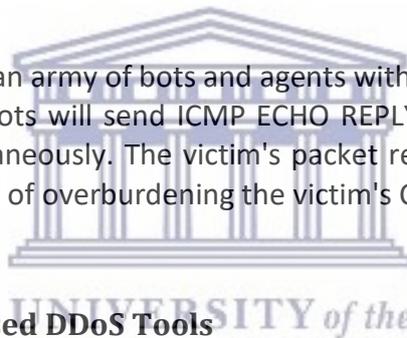
A UDP Flooding attack sends a large number of UDP packets to the victim system. The attacker employs a large number of bots and agents to saturate the victim system's network and use up all of its available bandwidth for legitimate service requests. To send UDP packets to the victim, the attacker typically employs bots and agents with spoofed IP addresses.

When the packet is received by the victim system, it determines which program is supposed to be waiting on the destination port. When it detects that no application is listening on the port, it will send a "destination unreachable" packet to the faked source address. The system will be saturated with an army of bots and a huge number of UDP packets transmitted to the victim's ports [36].

#### 2.4.6.4 ICMP Flooding Attack

The Internet Control Message Protocol is used in this type of flooding attack (ICMP). A user computer can utilize the protocol to transmit an ICMP ECHO REPLY packet (also known as a "ping" message) to a target host or remote server. This is done to test the host's reachability and to assess the roundtrip time of the message delivered back to the user computer. If the destination host can be reached, a response will be returned to the user computer. Response packets will provide statistical information such as the packet roundtrip's minimum, maximum, mean, and standard deviation. The target host uses its CPU (Central Processing Unit) and network resources to create a response packet [36].

The attacker employs an army of bots and agents with spoofed IP addresses in an ICMP flooding attack. The bots will send ICMP ECHO REPLY packets to the intended victim frequently and simultaneously. The victim's packet request response (to a spoofed IP address) has the effect of overburdening the victim's CPU and network resources [23].



### 2.4.7 Commonly Used DDoS Tools

DDoS attackers have access to a number of well-known DDoS tools. The literature has a comprehensive list of DDoS tools [23, 34, 37-41]. In architecture and design, these tools are similar to some extent. Some tools are hacker adaptations and enhancements to existing attack tools. This section examines the most prevalent DDoS tools employed by hackers.

#### 2.4.7.1 Trinoo

Trinoo [42, 43] is a popular and extensively used hacking tool among hackers. It is a bandwidth depletion tool that is used to execute coordinated UDP flood assaults against a single or many IP addresses. The program employs fixed-size UDP packets that are delivered at random to the target machine's ports. It secures the communication channel through encryption and password protection. Trinoo does not allow source IP address spoofing by default, however it may be changed to support source IP spoofing.

#### 2.4.7.2 Tribe Flood Network (TFN)

Another widely distributed and utilized DDoS attack tool is Tribe Flood Network (TFN) [44]. It is used to conduct coordinated UDP flooding attacks, TCP SYN flooding attacks, ICMP echo request flooding attacks, and directed ICMP flooding attacks. TFN may launch attacks against one or more target IP addresses. It is unable to encrypt the

communication route between attack components, making detection possible. It has the ability to generate spoofed source IP addresses.

#### *2.4.7.3 TFN2K*

The TFN2K [45] is a more sophisticated TFN version. It is constructed using the TFN architecture. The TFN2K is an improvement on the TFN, with characteristics developed particularly to make its traffic harder to detect and filter. It allows remote command execution and hides the real source of the attack by spoofing IP addresses. TFN2K traffic is routed through a variety of transport protocols, including UDP, TCP, and ICMP. Attacks entail flooding the victim's system and crashing or destabilizing it by delivering erroneous or invalid packets, such as those used in the Teardrop and Land attacks. TFN2K interfaces using the master-agent architecture, and all communications between the master and the agent are encrypted.

#### *2.4.7.4 Stacheldraht*

Stacheldraht [45] (a German term for 'barbed wire') is an attack tool that combines characteristics from the TFN and Trinoo attack tools. It eliminates some of the TFN's flaws and integrates the Trinoo's handler/agent feature. It is capable of conducting agent updates automatically. Stacheldraht additionally supports a secure telnet connection between the attacker and handler computers using symmetric key encryption. This aids in evading detection. The attack tools may carry out attacks such as UDP floods, TCP SYN floods, and ICMP echo request floods.

#### *2.4.7.5 Mstream*

Mstream [46] is a simple point-to-point attack program that sends TCP ACK Floods to its target. To attack the target, it sends spoofed TCP packets with the ACK flag set. It communicates between the master and the agent using UDP/TCP packets over an unencrypted telnet communication channel.

#### *2.4.7.6 Shaft*

Shaft [47] is based on the Trinoo attack tool and has a similar design (agent/handler). An unencrypted telnet connection is used by the handler to interact with the agent. It employs faked source IP addresses and has the capacity to change IP addresses in real time during an attack. It also has the capability of supplying the attacker with statistical information about the initiated flood attack. These statistics let the attacker determine when the victim's system is saturated. It is capable of carrying out UDP, ICMP, and TCP flooding attacks.

#### *2.4.7.7 Trinity v3*

Trinity v3 [48] is an advanced attack tool capable of conducting different sorts of flooding attacks against a victim system. UDP, TCP SYN, and other flooding attacks are among those that may be launched. It establishes a link between the handler and the agent through the IRC communication channel.

#### *2.4.7.8 Knight*

The Knight is a small yet strong IRC-based DDoS attack tool. The program can conduct TCP SYN attacks as well as UDP flooding attacks. It is intended to operate on Windows operating systems and is generally deployed through the use of a Trojan horse application known as Back Orifice.

#### *2.4.7.9 Low Orbit Ion Cannon (LOIC)*

The Low Orbit Ion Cannon (LOIC) [49] is an attack tool that uses IRC. It has the ability to launch UDP floods, TCP floods, and HTTP floods. It is accessible in binary and web-based formats.

## **2.5 DDoS Defence Mechanisms**

The nature of DDoS evolves over time as hackers modify their attack tactics to circumvent current intrusion detection measures. DDoS assaults have grown larger, are stealthier, more targeted, and smarter than ever before. Almost every day, new and sophisticated DDoS attacks with unknown intentions are launched (be it social, political or criminal motives). The difficult job is thus to create adaptive DDoS defence measures capable of detecting large-scale attacks adequately and quickly. To do this, one must first learn how a technique is created and implemented. We ask the following questions in this section:

- Where is the attack detected?
- What is the response mechanism?
- How is the attack detected?

### **2.5.1 Classification based on Defence Points (or Defence Location)**

It is critical to comprehend and recognize the point or location at which a defence mechanism is activated. A defence mechanism can be installed at the victim, source, or intermediary network. Each deployment site has benefits and drawbacks. DDoS attacks can be detected throughout the way between the target (attack victim) and the source of the attack. Because all traffic packets may be viewed at the target, detecting a DDoS assault might be comparably easy. In contrast, it is difficult for a single source network to identify an attack unless the majority or all attacks are launched from that source network. Although it would be preferable to detect an attack closer to its source, there is a trade-off between detection accuracy and the detection mechanism's closeness to the attack source. The next section examines the benefits and drawbacks of the three lines of defence.

#### *2.5.1.1 Victim-end Defence Mechanism*

Victim-end defence systems identify the attack at its target. Historically, most defence mechanisms were meant to function at the victim-end since the victim is the one who suffers the consequences of the attack. These methods are deployed at the target ISP's edge outside or access router. They may simulate the behaviour of the victim's network traffic and therefore distinguish attack activity from normal traffic. The authors suggested architectures and systems for victim-end defence mechanisms in [50-56].

#### *2.5.1.2 Source-end Defence Mechanism*

Source-end defence systems identify attacks at the source in order to prevent an ISP's clients from participating in a large-scale DDoS attack. These methods might be deployed on the routers of the source network or on the access and edge routers of the ISP. The optimum point of defence is the source-end defence point because it is close to the source of the attack; the mechanisms can ease a trace back in an examination of the

attack and reduce the damage caused by the attack. Detecting attacks at their source has advantages; nevertheless, distinguishing genuine traffic from attack traffic at the source might be difficult for a variety of reasons [2]:

- As sources are scattered across domains, it is difficult for a single source to identify an assault properly.
- Also because the volume of traffic at the source is insufficient, distinguishing between genuine and attack traffic may be difficult.
- While source-end defence measures prohibit the source network from taking part in a DDoS attack, the advantage is mostly for the potential victim and not fully for the source network. As a result of the high cost of deploying these systems, there is little incentive to do so.

In the literature, many source-end defence mechanisms have been proposed [36, 50, 57, 58].

### *2.5.1.3 Intermediate Network Defence Mechanism*

In most cases, intermediate network defence measures are deployed on an ISP's core routers. They are more effective than victim-end defence mechanisms because they can reduce the victim's impact on the attack. The core routers filter and rate-limit the traffic that passes through them separately. However, because core routers process highly aggregated and huge quantities of data, they are unable to detect and filter for every potential victim. Core routers will need a lot of storage and processing power to be able to distinguish between attacks in the intermediate network, because these routers handle a lot of data. As a result, intermediary network defence measures have the problem of properly distinguishing between attack and legitimate traffic. Zhang et al. developed an architecture for attack detection in the intermediate network in [59].

## **2.5.2 Defence Approaches and Strategies (Point in time defence)**

Defence techniques may be roughly categorized into three kinds based on reaction time, namely Prevention Approaches, Survival Approaches, and Responsive Approaches, which are addressed in the paragraph that follows.

### *2.5.2.1 Prevention Approaches (Proactive Strategies) – Before the attack*

The best method to defend against a DDoS attack is to prevent it from happening in the first place. As a result, the goal of these techniques is to identify attacks and thwart attackers' attempts before they reach the target machine. With this method, the person is unaware that they are being attacked. Routers play a critical role in these techniques by detecting and filtering attack traffic before it reaches the target. These are routers that are closer to the source of the traffic (attack), core routers, or end-routers that are closer to the target computer. Routers use methods to identify and filter harmful traffic from legal traffic. The following are some of the specific methods:

*Ingress/Egress Filtering* [50]. This is a filtering method used to block attack packets before they reach the target system. Ingress filtering is a technique that prevents inbound packets from illegal and spoofed IP address sources from accessing the network. This method can greatly minimize DDoS assaults by discarding packets from questionable IP addresses. It does, however, occasionally lose valid packets. Egress filtering is a similar process to Ingress, except it is concerned with outgoing-outbound

packets. It guarantees that traffic packets only originate from IP address space that has previously been assigned. The Egress filtering approach does not defend the network against attacks in particular, but it does prevent the domain network from participating in a DDoS attack on other domains.

*Route-based distributed packet filtering* [53]. This filtering method extends the Ingress filtering function to internet core routers. The method filters spoofed IP packets using routing information. This method can filter a significant number of spoofed IP packets at the border router before they reach the target network.

*Disabling Unused Services* [51]. Attackers may utilize unused services to launch a fully-fledged DDoS attack. UDP echo, ping, and IP broadcast services are a few examples. As a result, it is critical that any unnecessary services be removed from the network.

*Beaver* [54]. This is a defensive mechanism added to the application and session levels. To safeguard network services, it employs authentication and cryptography techniques. Authors in [60] implemented this method in client-server services, together with public and private keys, to reduce network traffic. The client must be registered with the Admission Server in order to initiate a communication session with a device on the server. By preventing non-registered devices from connecting the server, this approach is thought to prevent anomalous and attack traffic from reaching the network.

*History-based Filtering*. This technique to DDoS attack avoidance was presented by Peng et al [27]. The edge router maintains a database of IP addresses based on past connection history in this method. Incoming packets with a source address that matches an IP address in the database will be allowed.

*Secure Overlay Services (SOS)* [61]. This method is a distributed system that needs target systems to alter their design in order to provide DDoS prevention. In this design, a client must be authenticated with one of the SOAP (Service Overlay Access Points) nodes in order to obtain access. This gives the client access to the overlay network, which filters out all other unauthenticated traffic packets. The overlay network filters genuine traffic using servlets, and the target system will only accept traffic packets from the small number of servlet nodes.

*Puzzling Mechanisms*. OverDose [62] is a technique that employs a unique computational perplexing architecture to block DDoS attacks before they reach the target network. Between the clients and the server in this network is a series of overlay nodes. To connect, the client selects one of the overlay nodes and requests a connection. The overlay node will react with a computational perplexing challenge that the requesting client must solve. Once the solution has been confirmed, the connection request is sent to the server. Cookies and flow specifications will be returned by the server. The overlay nodes enforce the flow requirements as a set of rules that govern the established connection between the client and the server.

**Table 1 Advantages and disadvantages of prevention approaches**

Advantages	Disadvantages
------------	---------------

<ul style="list-style-type: none"> <li>• Effective in such a way that the victims are unaware that they are being attacked.</li> <li>• They are simple to deploy since the ISP is aware of the address space given to each client.</li> <li>• Filtering may be carried out depending on a variety of parameters, such as IP address, protocol type, port number, and so on.</li> <li>• It drastically reduces spoofed IP addresses before they reach the intended victim.</li> <li>• Filtering methods can prevent attack traffic from entering the source network.</li> <li>• More accurate attack signatures may be created with ISP participation. As a result, the false positive rate will be reduced.</li> </ul>	<ul style="list-style-type: none"> <li>• The majority of the approaches may be used effectively with widespread deployment and coordination with other ISPs.</li> <li>• Attackers generate attacks using technologies that allow them to use non-spoofed IP addresses.</li> <li>• Some implementations may benefit the ISP but degrade network performance.</li> <li>• Some techniques provide no advantage to the deployed ISP but may prohibit them from engaging in a DDoS attack.</li> <li>• The majority of these techniques have a high false positive rate, which causes genuine traffic packets to be deleted from the network.</li> <li>• An attacker may be able to understand the preventive strategy and create techniques to defeat it.</li> </ul>
--	---

### 2.5.2.2 Responsive Techniques (Reactive Strategies) – During the Attack

When an ISP experiences a DDoS attack on its services, it will implement (invoke) a detection and mitigation mechanism. The invoked process is intended to aid in the control of attack traffic flow by tracking and finding the source of the attack and filtering traffic based on the discovered source. The majority of DDoS defence methods fall into this category [63], and the most common are discussed below.

*Pushback Technique.* A local Aggregate Congestion Control (ACC) detects congestion and creates an attack signature that may be utilized for router filtering at the router level. The signature defines a traffic aggregate for a collection of traffic flows having comparable characteristics. Given the established signature, a suitable rate limit for the ACC, which is shared by nearby routers, is also specified. If a router becomes congested, a rate limit request is issued to the neighbouring router in order to rate restricted traffic that fits the ACC's signature; this request is also broadcast to all upstream neighbouring routers [64, 65].

*Hop Count Packet Filtering (HCF).* This method, proposed in [52, 66], is based on determining the time-to-live (TTL) value of packets. When a packet is transmitted to the target, it is launched with a TTL value from the sender (typical values: 30, 32, 60, 64, 128 and 255). The victim guesses the packet's initial TTL value when it was launched. The hop count is calculated by the difference between the original and current TTL values. When the victim is not under attack, a database of the most frequent genuine users and their hop count is produced. During an attack, however, spoofed packets are ones that are not saved in the database and have a hop count that does not match their

appropriate launch address. It is difficult for an attacker to predict the TTL value of a forged packet in order to overcome the filtering mechanism using this technique.

*K-Max-Min.* This technique, proposed in [67, 68] in which a DDoS attack is treated as a resource management challenge. This is a more advanced variant of the Pushback method. The recommended approach is to spread the victim's congested bandwidth resources among the level-k routers. Level-k routers are those that are directly linked to the host but are either k-hops or fewer than k-hops from the destination. To divide the congested bandwidth across the level-k routers, several approaches are deployed.

*IP Traceback.* These approaches are used to determine the real source of the assault [69, 70]. Because an assault consists of a large number of zombie computers with faked IP addresses, utilizing IP addresses to track the real location of the attacker machine may be ineffective. As a result, several trace back approaches, such as link testing [57, 71]; Probabilistic Packet Marking (PPM) [72, 73]; and Algebraic based trace back techniques [74], have been suggested.

**Table 2 Advantages and disadvantages of responsive techniques**

Advantages	Disadvantages
<ul style="list-style-type: none"> <li>• The method works with routers near to the victim server. As a result, they can precisely filter attack packets.</li> <li>• They can successfully identify the edge network holding the DDoS sources.</li> <li>• When attacker machines are collected in close proximity, they may successfully mitigate DDoS attacks.</li> <li>• They are easy, lightweight, and inexpensive to deploy for DDoS prevention on a network.</li> </ul>	<ul style="list-style-type: none"> <li>• Rate-limiting methods used during an attack may have a significant impact on genuine traffic.</li> <li>• For these approaches to be extremely effective, they must be widely deployed on the internet and supported by ISP collaboration. This results in significant implementation costs.</li> <li>• A large number of zombie computers with faked IP addresses can be used by an attacker, as can a reflector attack using genuine IP addresses. As a result, the trace back approach is rendered useless since the true source of the attack cannot be identified.</li> </ul>

### 2.5.2.3 Survival Techniques (Survival Strategies) After the Attack

An ISP that employs survival tactics will acquire and deploy equipment and systems to increase their resources in the event of a DDoS attack [26]. More resources, such as CPU power, memory, and bandwidth, are obtained statically or dynamically with this method, allowing services to be duplicated.

During an attack, the expanded resource pool provides the ISP with enough resources to service both legal and malicious packet requests. This guarantees that genuine users may continue to utilize the service even if the ISP is under attack. The replication

technique allows the ISP to successfully defend static services but not dynamic services from an attack. However, because dynamic services are targets for attackers, the replication method is no longer useful for DDoS defence.

The success of this approach is determined by whether the increased resources outnumber the assault traffic volume. This may be difficult to do since an attacker can recruit thousands of zombie machines to assist them in continuing the attack and potentially depleting the increased victim resources.

**Table 3 Advantages and disadvantages of survival approaches**

Advantages	Disadvantages
<ul style="list-style-type: none"> <li>• To deal with a DDoS attack, more resources are made accessible.</li> <li>• Even when the service is under assault, genuine users can still access it.</li> <li>• Strengthens resilience against DDoS attacks.</li> <li>• Provides effective DDoS protection and load balancing for static internet content.</li> </ul>	<ul style="list-style-type: none"> <li>• Not all services can be replicated.</li> <li>• Replication for some services may need a significant amount of time and money, and the approach is costly to deploy.</li> <li>• They defend victims against small-scale attacks, but they cannot deflect large-scale attacks.</li> <li>• May be ineffective if an attacker generates more traffic than the enhanced capacity of the resources.</li> </ul>

Given that acquiring more devices to extend network resources may be insufficient in the face of a large-scale DDoS attack, the victim may need to implement augmenting defence methods. These defence systems must detect and identify the source of the attack, as well as implement reaction methods. Because DDoS attacks cannot be entirely avoided or halted, most DDoS response tactics nowadays are aimed at reducing the impact of the attack on the target. Following the detection of an attack, there are two more phases in the defence process. These phases are attack source detection and attack defence reaction, which are described further below.

Techniques for identifying the source of an attack have been presented in the literature [69, 70, 72, 74-78]. They attempt to determine the origin of the attack. Some approaches seek to address the issue of spoofing IP addresses. These trace back techniques detect the attack source by traversing all routers in reverse order from the victim to the source, distinguishing between valid routes and packets and illegitimate ones.

Attack Defence Reaction is a stage in the defence against DDoS attacks that is intended to launch an appropriate response to the attack upon identifying its source. Throttling (rate limitation) and packet filtering are the most frequent response mechanisms. Various throttling or rate-limiting strategies [54, 55, 79-82] and packet filtering approaches [64, 68, 79, 83, 84] have been suggested in the literature.

## 2.6 Summary

This chapter began with outlining society's widespread use of the Internet of Things (IoT) and how it helps society. It went on to investigate the key properties of IoT technology, as well as how attackers are exploiting these qualities to leverage it as a platform for distributing stealthier attacks. The chapter also included background information on DDoS attacks as well as the motives for attackers to conduct a DDoS attack. We also investigated DDoS attack categories based on their targets, the most prevalent types of DDoS attacks, and the most popular tools used by attackers to begin an attack. We carried out further investigations into the methods and techniques used by organizations to protect against DDoS attacks, as well as their classifications, benefits, and drawbacks.

The next part searches the literature for relevant detection techniques and algorithms that have been effectively implemented and incorporated into Network Intrusion Detection Systems (NIDS). These are entropy-based detection approaches and algorithms, as well as change-point detection, artificial neural networks, and deep learning techniques.



## Chapter 3: Literature Review

---

### 3.1 Introduction

A Network Intrusion Detection System (NIDS) is a hardware or software program that monitors aberrant network traffic patterns or patterns that violate network protocols. Several Network Intrusion Detection Systems now include intrusion detection methodologies (NIDS) [85-89]. Signature-based NIDS and anomaly-based NIDS are the two forms of NIDS classifications. Signature-based detection systems try to create a collection of templates (signatures or rules) that may be used to determine if a particular network traffic pattern represents an intruder. If the attack falls into one of the attack classes specified in the database, it can be effectively detected or recognized. As a result, signature-based systems are capable of detecting intrusions with high accuracy and a low number of false positives. Signature-based detection, on the other hand, fails to detect novel attacks or variations of existing attacks [90].

The limitations of signature-based intrusion detection motivated the development of anomaly-based NIDS. Anomaly-based intrusion detection systems (ABIDS) are concerned with detecting occurrences that appear to be out of the ordinary in terms of system behaviour. When a divergence from regular traffic behaviour is noticed, an attack is reported.

Because signature-based NIDS look for known intrusions, whereas anomaly-based NIDS look for unexpected or suspicious patterns in network performance and behaviour, they are complimentary in their application. Since anomaly-based NIDS may identify both known and unknown attacks, the intrusion detection research community has focused on enhancing their performance (i.e. detection rate and detection latency) in recent years [34]. As a result, ABIDS are relevant in our planned investigation.

Anomaly-based network intrusion detection systems seek to discriminate between regular network activity and abnormal network activity. This is accomplished by developing the system's normal profile based on previous data and monitoring for substantial departures from normal profile activities. Sudden variations in network traffic might be detected during a DDoS attack. Similarly, there is a sudden shift in the statistical characteristics of detection parameters. As a result, the challenge of anomaly detection may be modelled as a change point detection problem [91, 92].

Several strategies for defending against DDoS attacks have been developed over the last decade, but early and effective detection of DDoS attacks has remained a difficult problem. As a result, the development of systems for detecting DDoS attacks in a timely and effective manner remains an important field of study. It is critical to create ways for properly detecting an attack sooner, so that corrective steps may be performed before severe damage is incurred.

For accurate detection, well-known and proven defence solutions are built on algorithms based on entropy, change point, neural network, and deep learning approaches. The next section will search the literature for noteworthy studies conducted by other researchers in these disciplines.

## 3.2 Change Point Statistical Techniques

In the event of a DDoS attack, abrupt changes in network traffic data may be detected. Similarly, an abrupt change in statistical properties of detection parameters will be observed. Entropy in network traffic captures the abnormal distributional changes in traffic features in a single value, where adequate measurements of the changes in value will clearly identify the network anomalies. Entropy is used to accumulate useful features for distinguishing between attack and non-attack traffic. Entropy-based techniques are typically developed by applying the entropy metric to raw traffic data. For example, raw traffic data may include source and destination IP addresses as well as destination port numbers. Entropy can summarize the amount of traffic data within a given time bin as compared to a number of time bins. A high entropy measure of measured traffic data indicates a high variance, while a low entropy measure indicates a low variation in traffic data trends [93].

Entropy-based techniques for DDoS detection are commonly based on the Shannon [94], Tsallis [95] and Renyi [96] entropies. The Shannon entropy was developed to measure information gain and reduce ambiguity in communication. It has commonly been used to categorize network traffic anomalies and is defined by the equation below:

$$H(X) = - \sum_{i=1}^N P(x_i) \log_2 P(x_i)$$

Where  $X = x_1, \dots, x_n$  is a finite set and each element has a probability of  $P(x_i)$ .

The Shannon entropy equation was generalized in [96] to produce the Renyi entropy, and the Tsallis entropy in [95] is likewise a one-parameter generalisation of the Shannon entropy. More variants of the entropy measure have been developed, and we investigate significant uses of these approaches in the realm of DDoS attack detection.

Li et al. [97] proposed a technique for identifying DDoS assaults based on the generalised Renyi entropy. The technique examines variations in probability distribution changes over a particular time period for both the source IP address-based method and the IP packet size-based method. Then, rather than utilizing an a priori threshold value, it determines the change in information entropy value trends over a certain time frame. A trend is characterized as anomalous if there are substantial fluctuations in the information entropy values over a lengthy period of time. Experiments show that the Renyi entropy metric, as opposed to the traditional Shannon entropy meter, reduces both false positive and false negative rates.

Qi et al. [98] presented a new hierarchical entropy-based DoS detection methodology. The model is premised on the principle of "alive" communication and combines netflow conversation association features with the complicated entropy model. The authors generate "alive" network states by computing the request-to-response ratio in netflow. Their investigations indicate that dynamic entropy remains constant under normal traffic circumstances. However, it varies significantly during a DoS attack. The dynamic and static entropy shift rates were studied in anomaly detection, and it was revealed that the dynamic entropy technique is more robust and can forecast unexpected DoS attacks. The research does not indicate if the system was evaluated on a distributed or

large-scale DoS model, whether it was tested on low-rate DDoS attacks, or whether detection delay was recorded.

For anomaly detection, the work of Gu et al [99] employs maximum entropy calculation and relative entropy. Two-dimensional packet groups were created for network packets. First, packets are organized according to packet protocol, and then packets are organized by destination port number. The authors generate a baseline distribution and the relative entropy of normal network traffic performance using feature selection and parameter estimation. The procedure demonstrated a high accuracy rate with low false positive and false negative rates; however, it demands a large amount of memory and processing time and therefore cannot be done in real time.

Xinlei Ma et al. [100] illustrated a DDoS detection system that makes use of the Tsallis entropy and a variant of the Lyapunov exponent. The Tsallis entropy measures variations in the distribution of source and destination IP addresses. The Lyapunov Exponent variance quantifies the exponent distinction of the source and destination IP address entropies. The source and destination IP addresses will have equal entropy values over the same time period. The experimental findings reported a high true positive rate and a low false positive rate in detecting DDoS attacks; no comparisons with other datasets were made.

The work of Zhang et al. [101] detects low-rate DoS attacks and flash crowd events using an advanced entropy-based tool. To differentiate typical traffic from DoS attacks and Flash crowd incidents, the method employs the Shannon entropy at three levels of adaptive threshold. Each adaptive threshold value is modified and tailored on a regular basis to the current network conditions. While this approach produced reliable results, it requires significantly more computational resources than current DDoS attack detection systems.

Xiang et al. [102] introduced a generalized entropy and a generalized information distance metric to differentiate between low-rate DDoS attacks and normal traffic. The approach calculates the information distance between valid traffic and low-rate attack traffic using packet header features such as source IP, destination IP, and protocol type. As compared to the current Shannon entropy and KL divergence metrics, the algorithm detects low-rate DDoS with a lower false positive rate. The researchers, however, did not consider the detection of high and low rate DDoS attacks, and they did not report on the algorithm's success in terms of detection delay.

Bhuyan et al. [103] created an expanded entropy metric based on packet header features. This is a development on the work by authors in [102]. In order to detect high-rate DDoS attacks, the method computes the entropy difference between the source IP and the incoming packet rate. To detect high-rate DDoS attacks, the algorithm employs a split window size sampling approach and three kinds of expanded entropy functionality, including the source IP address. The method demonstrated high accuracy in identifying four types of DDoS attacks: persistent, pulsing, increasing, and dynamic. However, this method has not been tested against DDoS attacks with low-intensity attack traffic.

Mousavi et al. [104] devised an early warning system for DDoS attacks. To detect attack traffic, the method evaluates the destination IP address entropy of incoming packets within a given time frame. The authors choose a time window size of 50 packets to

reduce the amount of computing required for each time window. The detection methods begin by defining a standard traffic flow entropy value as a baseline. A DDoS attack is considered present in the network if the destination IP address entropy falls below the threshold five times in a row. The authors discovered that this approach would detect DDoS attacks as soon as the first 250 attack packets arrive. The system yielded a detection rate of 100%. It can, however, not be suitable for detecting low-rate DDoS attacks. Furthermore, the system computes the baseline threshold value at the start, but network traffic flows change often, necessitating a more constructive or continuous method of computing the baseline entropy threshold.

We found that the entropy approaches are more effective with acquiring the detailed distributions and patterns that typical volume-based techniques cannot acquire. Entropy calculations are flexible and can be implemented across a number of network traffic features, like IP address, ports, protocol, network flows, and number of packets. The main advantage of using entropy-based techniques is for their low computational overheads. However, they have a few limitations. Entropy-based detection techniques compute and detect unexpected changes in distribution probability of network traffic data [105]. This change is represented using a single value and researchers are concerned that some information about the distribution change may be lost [106]. In some cases the anomaly may be concealed and go undetected [3].

Change-point challenges are concerned with the identification of changes in the statistical behaviour of operations. This problem has a wide range of important applications, including biomedical signal and image processing, financial markets, anomaly and intrusion detection in communication and information systems, detection of the beginning of an infectious disease outbreak, surveillance systems, econometrics, and seismology, to name a few [107].

The aim of change detection techniques is to help detect a change in statistical properties of observed parameters with minimal detection delay and false positive rate [108]. The approach first starts by applying a filter to the traffic data according to desired parameters and arranging the data into time series data. For change detection, if there were a DDoS attack at time  $\lambda$ , the time series would show a significant statistical change around or at a time greater than  $\lambda$  [109].

Detecting changes in statistical properties of observed parameters has been studied extensively and applied in various fields like image processing, network traffic and financial analysis. There are a number of techniques that are used for change detection and amongst them the most common techniques used for the detection of DDoS attacks are Cumulative Sum (CUSUM) and Exponentially Weighted Moving Average (EWMA) [110].

### **3.2.1 Cumulative Sum (CUSUM) Technique**

The CUSUM algorithm was first introduced by Page in [111]. The CUSUM algorithm is based on hypothesis testing and was developed for independent and identically distributed random variables  $\{y_i\}$ . In the approach, an abrupt change occurring at any time can be modelled using two hypothesis,  $\theta_1$  and  $\theta_2$ . The first hypothesis  $\theta_1$  represents the statistical distribution before the abrupt change occurring; and the second

hypothesis  $\theta_2$  represents the statistical distribution after the abrupt change has occurred. The test for signalling a change is based on the log-likelihood ratio  $S_n$ .

$$S_n = \sum s_i$$

Where,

$$s_i = \ln \frac{P_{\theta_1}(y_i)}{P_{\theta_2}(y_i)}$$

According to Siris et. al. [112], the typical behaviour of the log-likelihood ratio  $S_n$  includes a negative divergence before an abrupt change and a positive divergence after the change. Therefore, the relevant information for detecting a change lies in the difference between the value of the log-likelihood ratio and its current minimum value [108]. The alarm condition for the CUSUM algorithm takes the form:

If  $g_n \geq h$  ( $h$  is a threshold parameter) then signal alarm at time  $n$ ;

$$\text{where } g_n = S_n - m_n \tag{1}$$

$$\text{and } m_n = \min_{1 \leq j \leq n} S_j. \tag{2}$$

In the above equations, it is assumed that  $\{y_i\}$  are independent Gaussian random variables with known variance  $\sigma^2$ , and  $\mu_0$  and  $\mu_1$  represents the mean before and after the abrupt change, respectively. Accordingly,  $\vartheta_0 = N(\mu_0, \sigma^2)$  and  $\vartheta_1 = N(\mu_1, \sigma^2)$ . Following an application of various calculations, Basseville et al [108] implemented the following CUSUM algorithm:

$$g_n = \left[ g_{n-1} + \frac{\mu_0 - \mu_1}{\sigma^2} \left( y_n - \frac{\mu_1 + \mu_0}{2} \right) \right]^+ \tag{3}$$

The above algorithm was adapted and applied to the problem of detecting SYN flooding attacks. This algorithm was applied as follows:

$$\tilde{x}_n = x_n - \bar{\mu}_{n-1} \tag{4}$$

where  $x_n$  represents the number of SYN packets in the  $n$ -th time interval, and  $\bar{\mu}_n$  represents the estimated mean rate at time  $n$ . The estimates mean rate is computed using an exponentially weighted moving average as follows:

$$\bar{\mu}_n = \beta \bar{\mu}_{n-1} + (1 - \beta)x_n \tag{5}$$

where  $\beta$  is the exponentially weighted moving average factor.

The mean value of  $\tilde{x}_n$  prior to a change is zero, therefore the mean in (3) is  $\mu_0 = 0$ . The mean traffic rate after a change cannot be known in advance. It can therefore be estimated with  $\alpha \bar{\mu}_n$ , where  $\alpha$  is an amplitude percentage parameter. The parameter equates to the most likely percentage increase of the mean rate after an attack has

occurred. For purposes of detecting SYN flood attacks, the algorithm in (3) has been adapted to:

$$g_n = \left[ g_{n-1} + \frac{\alpha \bar{\mu}_{n-1}}{\sigma^2} \left( x_n - \bar{\mu}_{n-1} - \frac{\alpha \bar{\mu}_{n-1}}{2} \right) \right]^+ \quad (6)$$

In the CUSUM algorithm, the tuning parameters are the amplitude factor,  $\alpha$ , the weighting factor,  $\beta$ , and the CUSUM algorithm threshold,  $h$ .

The CUSUM technique has been applied to various problems including DDoS detection. It calculates the cumulative sum of difference between actual and expected values of a sequence, the CUSUM value. This value is compared to a threshold value (an upper bound). A CUSUM value greater than the upper bound indicates a change in statistical properties of the parameter time series values.

There are a number of variations of the CUSUM technique, and Tartakovsky et al. [91] proposed fully-sequential and batch-sequential algorithms. They are both non-parametric variations of the CUSUM techniques adapted to detecting changes in multiple bins. The algorithms were found to be self-learning, which enables them to adapt to various network loads and usage patterns. They also allow for the detection of attacks with a small average delay for a given false-alarm rate and they are computationally feasible and thus can be implemented online.

Bo et al. [113] also used an algorithm which was a variation of the CUSUM technique to help with quick detection of worm attack incidents. In their experiments they observed the computer's degree of connection to estimate the CUSUM score. It was concluded that the algorithm could detect new attacks rapidly and effectively.

Siris et al. [112] proposed and investigated a change point detection algorithm, which is also based on the CUSUM technique. The algorithm revealed robust performance over various attack types; it was computationally feasible and not costly to implement. Wang et al. [66] also proposed an algorithm which is a variation of the CUSUM technique on an application for detecting DDoS attacks. Protocol behaviours of TCP SYN – FIN (RST) pairs were used to make detections. The experiment results revealed that the algorithm had low detection delays and high detection accuracy.

### 3.2.2 Exponentially Weighted Moving Average (EWMA) Technique

EWMA was first introduced by Roberts [114], it analyses whether the value of the parameters being observed, in a given time interval, exceeds a particular threshold value. The algorithm adaptively calculates the threshold value (the parameter mean value of recent observations in each sampling interval) instead of using a predefined threshold value.

$$\text{If } x_n \geq (\alpha + 1) \bar{\mu}_{n-1}, \quad (7)$$

then an alarm is signalled at time  $n$ , where the tuning parameter  $0 < \alpha \leq 1$  indicates the amplitude factor, a percentage above the mean value that is considered to be an indication of anomalous behaviour. This amplitude factor parameter is used for

computing the alarm threshold. The mean  $\bar{\mu}_n$  can be computed over some past time window or using an exponential weighted moving average of previous measurements.

$$\bar{\mu}_n = \beta \bar{\mu}_{n-1} + (1 - \beta) x_n, \quad (8)$$

Where the tuning parameter  $0 < \beta \leq 1$ , is the weighting factor parameter. The parameter  $\beta$  determines the rate at which “older” data enter into the calculation of the EWMA statistic. A value of  $\beta = 1$  implies that only the most recent measurement influences the EWMA. Thus, a large value of  $\beta = 1$  gives more weight to recent data and less weight to older data; a small value of  $\beta$  gives more weight to older data.

However, if the algorithm is applied in its original format, it will yield a higher rate of false alarms. To improve the performance a modification to the algorithm was to raise an alarm after a minimum number of successive violations of the threshold. Therefore,

$$\text{If } \sum_{i=n-k+1}^n 1_{x_i \geq (\alpha+1)\bar{\mu}_{i-1}} \geq k \quad (9)$$

Then an alarm is raised at time  $n$ , where  $k > 1$  is a tuning parameter that indicates the number of successive intervals the threshold must be violated before an alarm is signalled.

The tuning parameters for the EWMA algorithm are the threshold value (amplitude factor)  $\alpha$ , the EWMA factor  $\beta$ , and  $k$  which signifies the number of successive threshold violations before raising an alarm.

There have been a number of variations of the EWMA algorithm that was used for intrusion detection and flooding attacks. Siris et al [112] proposed an adaptive threshold algorithm, which is a variation of the EWMA technique. They used real traffic traces to analyse and compare the performance based on detection delay, false alarm rate and detection accuracy. The algorithm adaptively learns the normal behaviour of the network traffic. The algorithm revealed satisfactory results for high intensity attacks, however the performance declined for low intensity attacks. However, it is of paramount importance to detect the onset of an attack whose intensity increases slowly.

Ye et al. [56, 115] investigated and applied EWMA techniques to help detect anomalous changes in the events intensity for intrusion detections. The techniques were applied on the large DARPA datasets. Their findings revealed that the EWMA techniques can work well for detecting abrupt changes in event intensity, and also small mean shifts through the gradually increased or decreased event intensity.

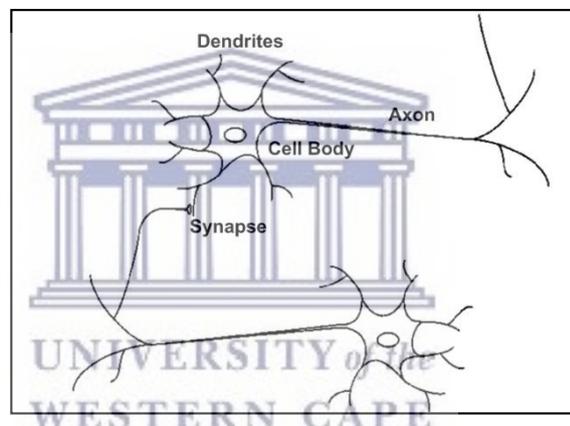
Münz et al. [116] investigated and evaluated the network traffic anomaly detection capabilities of the CUSUM and EWMA techniques. To deal with seasonal and serial fluctuation, a time series of forecast errors was utilized instead of a straight time series of traffic data. The traffic information was obtained from an Internet Service Provider (ISP). When applied to time-series of prediction errors, their studies revealed that CUSUM did not outperform Shewart and EWMA.

The subsection that follows will go into the literature for research on DDoS detection algorithms based on machine learning methods.

### 3.3 Machine Learning

Detection methods based on various models and theories are developed in the DDoS attack detection research community. The three key technologies that form the basis of the majority of today's detection techniques are machine learning, information theory, and statistical models [117]. Artificial neural networks (ANN), support vector machine (SVM), and other machine learning techniques in cybersecurity are helpful for decision making analysis [118]. The passage that follows highlights some of the related work in the DDoS research field.

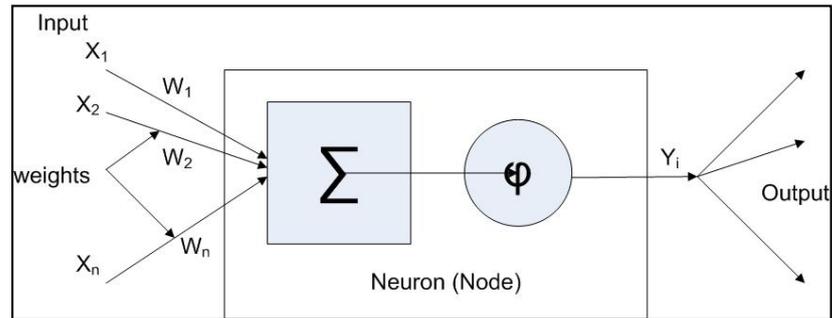
The human brain is a densely packed network of around 100 billion biological neurons. Figure 6 depicts a network of biological neurons that aids in critical human body activities such as reading, breathing, mobility, and thinking [119]. Some neural structures are present in the human body from birth, whereas others are formed via experience [120]. All biological neural characteristics, including memory, are stored in neurons and their connections. Learning is thus seen as the formation of a new neuronal connection or the change of existing neurons. This changes over one's life. Humans refer to this as intelligence.



**Figure 6 A schematic drawing of the biological neuron**

Researchers in computer science have been working to enable computer systems to do analytical tasks, learn, and classify as precisely as the human brain. To begin with, the ability to see and interpret dynamic patterns of linked information from all dimensions, comparable to the human brain. Second, in order for a computer to achieve human-like intelligence, it needs have access to a massive amount of data. The perceptron was created by Frank Rosenblatt [121], and the perceptron represented in Figure 7 went on to serve as the framework for the construction of neural networks, which served as the foundation for what is now known as Deep Learning [122].

The perceptron is an artificial neuron that serves as the foundation for the biological model utilized in traditional Artificial Neural Networks (ANN). The perceptron has been utilized in several ANNs, including the well-known Multi-Layer Perceptron (MLP), a model that performs linear transformations with scalar weights and a weighted summation. The evolution of the MLP has led in well-known improvements in learning and generalization performance for a wide range of applications. This includes, among other things, aerospace, accountancy, medical, and voice recognition [123].



**Figure 7 The Artificial Neuron**

The MLP has a perceptron (or node) structure that is interconnected in layers, with the output from one layer becoming the input to the next. The model is composed of three layers: the input layer, the hidden layer, and the output layer.

The Input Layer - this layer's nodes are input nodes; they encode and prepare raw network data for further processing.

The Hidden Layer — this layer is hidden from the input and output layers. This layer's nodes or neurons compute an internal representation of the data. When MLP algorithms, such as back propagation, are used, the values of the connections (or weights) are adjusted appropriately, and algorithms may include non-linearity features, as opposed to linear best-fit algorithms, which try to minimize the distance between the misclassified or incorrectly predicted.

The Output Layer - the neurons in this layer encode data representations based on the outcomes of the calculations performed in the hidden layer [123].

The essential structure of the MLP is that of a perceptron interconnected in layers with output from one layer as input into another layer. The general structure is made up of three layers i.e. the input layer, the hidden layer and the output layer. The first layer contains nodes of the input layer, which use an encoding scheme of zeros and ones and prepares the raw data for further processing. In other words, it is a more abstract representation of the data patterns. The second layer is the hidden layer and it holds nodes that use the various MLP algorithms to compute a more abstract and internal representation of the data. The output layer contains nodes that uses results from the hidden layer to encode the final data representations.

In Li et al (2012 DDoS-neural) a Learning Vector Quantization (LVQ) neural network was used to detect DDoS attacks. LQV is a supervised version of the vector quantisation; it uses known target output classifications for each input pattern. It has been applied to pattern recognition, multi-class classification and data compression tasks. In their simulations the dataset was pre-processed into numerical format so that it can be fed to the neural network for training and testing. The LVQ neural network achieved a 99.7% detection rate while Back Propagation (BP) neural network achieved 89.9% detection rate. However, the data collected was simulated using a small number of simulation PCs; and this may not be representative of a real DDoS attack. The pre-processing detail of the data was not clarified.

The work of authors in [124] proposed an attack traffic classification method that is based on Probabilistic Neural Network; a feedforward network that is used for non-linear pattern classification. In this method, the network uses the Bayesian decision rule for Bayes inference; this was coupled with the Radial Basis Function Neural Network (RBFNN) because it is suitable for attack traffic classification problems. The proposed method was used to effectively classify attack traffic patterns from legitimate traffic using single and joint distributions of various packet attributes. It successfully classified TCP SYN, UDP and ICMP flooding attacks. In their experiments the average detection rate was below 80% with an average false alarm rate of 1%. However there is room for improvement in order to achieve a higher detection rate.

The work of Siatelris et al. in [125] presented DDoS detection as a classification problem and they proposed a Multilayer Perceptron Neural Network as a data fusion algorithm. Current DDoS research attempts to identify a single detection metric that can reliably detect DDoS attacks, however in their research a multiple set of passive network measurements are used as input and fused with the MLP algorithm to successfully detect a DDoS attack. The proposed method was found to have a detection rate of greater than 74% and a false alarm rate of less than 3%. However, the authors used a self-generated dataset of traffic using DDoS tools, and it was not clear how the data was generated and what were the configuration details of the data generation tools.

Ali et al. [126] developed an ANN-based machine learning strategy for detecting DDoS attacks. The backpropagation strategies employed by the ANN were Bayesian Regularization (BR) and Scaled Conjugate Gradient (SCG). The approach effectively detects DDoS attacks with an accuracy of 99.6% using BR and 97.7% using SCG backpropagation algorithms, according to their experiment results. Soe et. al [127] created a system for detecting numerous large-scale IoT attacks in sequential order. They proposed using different specified classifiers for each attack type instead of a single classifier. For experimentation, they presented a single-layered artificial neural network (ANN) on a publicly available dataset. They used a series of ANN models to detect specific assault types. They achieved a 99% accuracy by using the sigmoid function.

In order to detect DDoS attacks, Wang et al. [128] proposed combining feature selection with an ANN MLP (multilayer perceptron) model, and the MLP was combined with the sequential feature selection technique. These strategies were used to choose the best features during the training phase, and they created a feedback system to reconstruct the detector when significant detection faults were detected dynamically. With a 98% accuracy rate, the proposed methodology proved effective. Ioannou et al [129] proposed a supervised learning anomaly detection model that combines a Radial Basis Function (RBF) kernel with a C-support optimizer (c-SVM) to differentiate between benign and malicious traffic data. With Blackhole and Sinkhole attacks, the model was 100% accurate, whereas with other attack types, it was 81% accurate. The researchers did not compare the outcomes of their experiment with those of other machine learning models.

Chaudhary et al [130] further suggested a machine learning technique for detecting DDoS assaults that involves filtering crucial network packet parameters such as packet size, interval size, and so on. Support Vector Machine (SVM), Random Forest, Decision Tree, and Logistic Regression were all used. Random forest surpassed other machine learning approaches in their research study, detecting DDoS attacks with 99.17%

accuracy. Kokila et al. [131] created a method for detecting DDoS attacks using the SVM classifier. The SVM classifier has a 0.8% false alarm rate and a classification accuracy of 95.11%. Wehbi et al. [132] used flow features of network traffic such as packet size, packet interval, protocol, bandwidth, and destination IP to construct a model to detect DDoS attacks. They used SVM, K-Nearest Neighbour (KNN), Random Forest, Decision Tree, and ANN in their models. The results of the experiment show that Random Forest and ANN have 99% accuracy in detecting malicious traffic.

For detecting DDoS attacks in Software Defined Networks, Polat et al. [133] employed SVM, KNN, ANN, and Naive Bayes (SDN). Initially, the authors specified twelve features, and the algorithms chose a subset of these features based on threshold values provided to the algorithms. The algorithms analysed flow traffic data and detected DDoS with 98.3% accuracy. In a study by Aytac et al [134], the Artificial Neural Networks (ANN), Support Vector Machine (SVM), Logistic Regression, K-nearest neighbour (KNN), Gaussian Naive Bayes, Bernoulli Naive Bayes, Multinomial Naive Bayes, Decision Tree (entropy-gini), and Random Forest algorithms were all investigated for DDoS attack detection. They looked at data from twelve different aspects and discovered that only a small fraction of them, such as cumulative count and descriptive statistics, was enough to detect a DDoS attack. In their tests, they discovered that the SVM algorithm had the highest accuracy rate of 99.7%.

Tonkal et al [135] used 23 traffic flow features to look into DDoS attack detection in SDN. They employed the Neighbourhood Component Analysis (NCA) to determine the most important flow data characteristics for the pre-processing and feature selection stage. Following that, they classified DDoS attacks using the KNN, Decision Tree (DT), ANN, and SVM algorithms. They discovered 14 features to be important in their findings, and the DT algorithm was able to attain 100% detection accuracy. Churcher et al. [136] used KNN, SVM, decision tree (DT), naïve Bayes (NB), Random Forest (RF), ANN, and logistic regression (LR) algorithms to explore the detection of DDoS attacks in IoT networks. Their research looked into the effectiveness of algorithms for binary and multi-class classification. They also tested the algorithms' performance against a weighted and non-weighted Bot-IoT dataset. For non-weighted datasets, their testing revealed that the RF algorithm has a 99% accuracy. The ANN performed better for binary classification accuracy on weighted datasets. KNN, on the other hand, surpassed other ML algorithms in multi-class classification, with an accuracy of 99%, which is 4% higher than RF.

The work by Chung-Lung et al. [137] which presented an early warning system for DDoS attack detection was developed for integration into traditional IDS. The system developed was based on a Time Delay Neural Network (TDNN). TDNN is a kind of neural network that has the time delay factor integrated and implicitly represented inside the signal. In their experiments the TDNN was implemented in a two-layer structure; this enables the system to take proactive action against a DDoS attack like initiating an integrated IPS. In their implementation, nodes were dispatched at the demilitarized zone or between the first and second layer of the firewall. Each node's activities are monitored by neighbouring nodes and attack data is conveyed to the expert module for further analysis. The approach was found to have an 82.7% detection rate; however the information about the false alarm rate was not available.

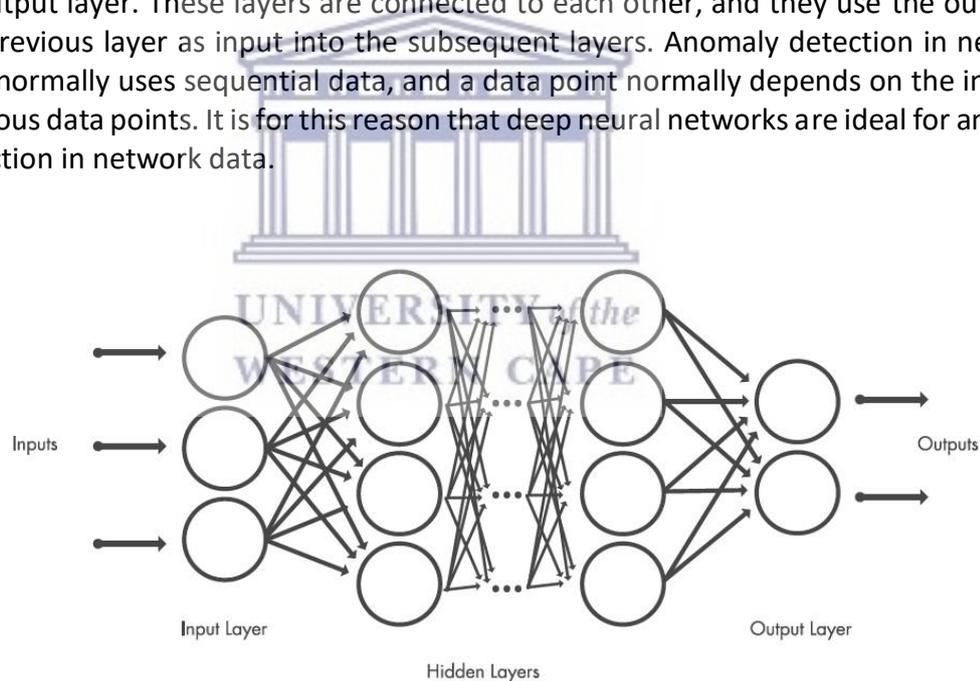
In machine learning, artificial neural network algorithms are considered the best in classifying DDoS attack models [138]. Several researchers have proposed various

algorithms that are based on neural networks. However, these algorithms have resulted in high false positive rate and lengthy detection times. Traditionally, artificial neural networks have two or three layers in the network, and the general assumption is that the inputs and outputs are independent. However, this assumption may not be always true for some tasks, especially in anomaly detection.

Researchers in [139] further found that traditional neural networks (with up to three layers) limit the use of data in its raw form, and it requires a feature extractor. Normally, developing a feature extractor requires decades of research work and expertise in order to transform raw data into a suitable abstract representation that will be processed for classification or prediction. Furthermore, they discovered that the ANN's architecture can be too simplistic for tasks with complex functional dependencies that cannot be represented analytically in a straightforward manner. Thus rose the need for a network architecture with more than three layers, called a deep neural network. This is detailed in the next section.

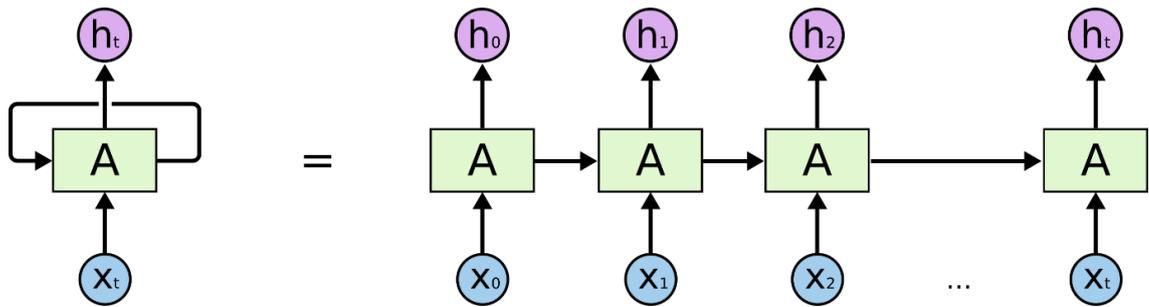
### 3.4 Deep Learning Techniques

Deep neural networks are a modern adaptation of artificial neural networks that integrate several nonlinear processing layers in the network to extract features from raw data. As depicted in Figure 8, it consists of an input layer, several hidden layers and an output layer. These layers are connected to each other, and they use the output of the previous layer as input into the subsequent layers. Anomaly detection in network data normally uses sequential data, and a data point normally depends on the input of previous data points. It is for this reason that deep neural networks are ideal for anomaly detection in network data.



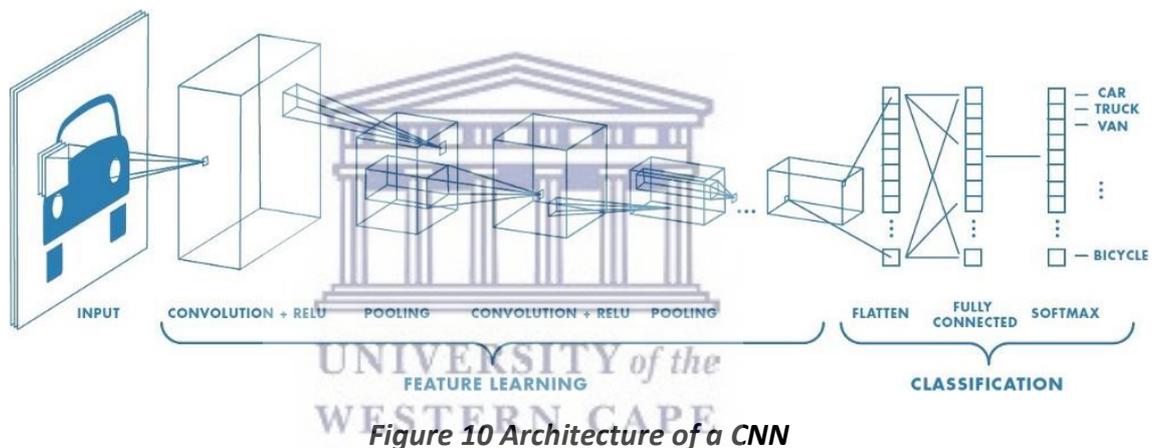
**Figure 8 Structure of a Deep Learning Neural Network**

Deep neural network models are used to learn and perform classification tasks on raw data such as images, text, and sound. Face recognition, text translation, voice recognition, driver assistance systems for lane classification, and traffic sign recognition are all examples of effective classification applications. The Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN) are two popular deep learning models that are extensively used in these applications.



**Figure 9 Structure of a Recurrent Neural Network**

Figure 9 illustrates the basic structure of a RNN. The node A, which receives an input  $X_t$  and outputs a value  $h_t$ . The loop allows information to be passed and memorised from one step of the network to the next. RNNs are originally designed to capture a sequence of inputs without a fixed limit on size. One input item from the sequence is connected to other inputs and probably has an effect on them. The loops in the neural network allows for memory. Therefore, the network has the ability to memorise past decisions and influence the outcome.



**Figure 10 Architecture of a CNN**

The CNN is analogous to the interconnection model of the neurons in the human brain and it was adapted from the visual cortex. It is a popular model and it is implemented successfully in various fields and one such field is computer vision, deep learning for images and video. Figure 10 shows the architecture of a CNN for video and image classification. The architecture uses various algorithms for feature learning. This stage of feature detection is repeated over many layers with each layer learning and detecting different features. Once features are detected, the classification layers compile probabilities for each classification class and produce the final classification output.

The work of researchers in [140, 141] had various detection modules infused to detect DDoS attacks. The initial detection module examines the relevant statistical features of the DDoS incoming traffic. This data is then streamed through multiple Bayesian classifiers that evaluate and predict the probability of an attack. These classifiers use the selected traffic features to estimate the probability density functions and the likelihood ratios for each traffic feature. For final decision making, the researchers further infused a RNN with the statistical data. When trained with sufficient data, the RNN has strong

classification and estimation capabilities. Therefore, this technique works well distinguishing normal traffic from attack traffic. The work of these researchers has achieved success, however, the researchers did not consider adaptive low-rate attacks.

The work of researchers in [142] proposed an artificial intelligence based detection system that defends a system against SYN flooding DDoS attacks. The proposed method uses a RNN ensemble that detects attacks on the client side and its intermediary nodes. The detection method monitored the nodes for the volume of rejected connection requests and the deviations in resource usage (particularly for CPU usage, Physical Memory usage and NIC usage). The RNN ensemble processes the information in order to classify the state of the network into one of six states that range from 'normal' state to 'attack' state. The RNN ensemble promises to produce good results, however this is based on assumptions that an increase in volume of rejected packets and an increase in resources usage may signify an attack. However, this can also take place under normal state of the network, in the case of a flash crowd. Therefore there is a need to have a self-adaptive and methods that do not depend on predefined threshold states.

The work of [143] proposed a solution that implements multiple agents which are distributed at different monitoring points of the network. These individual agents perform DDoS attack monitoring and detection from selected parameters observed in network traffic flows. The observations from the agents are occasionally synchronized and eventually aggregated in order to assist with an all-inclusive decision making of observed network traffic patterns. The solution is fault-tolerant and robust in that a failure from a single agent does not notably interrupt the functioning of the whole system. The researchers uses predefined network thresholds to model the network traffic flow patterns. These thresholds are based on the incoming traffic flows' maximum number of packets that can be processed in a specified timeframe. The solution could perform well in DDoS attack detection, however it may not perform well against those DDoS attacks that adapt their rate of attack, in the case of low-rate attacks.

The researchers in [144] proposed and implemented a DDoS detector that is made up of three components, the detection, defence and sharing components. The detection component was designed using ANN algorithm for detecting known and unknown attacks; the algorithm could detect, with 98% accuracy, the specific DDoS attacks and distinguish attack traffic from normal traffic in real time. The defence component would thwart forged packets from getting to the attack target. Since these DDoS detectors will be deployed across different parts of the network, the knowledge component will then share important DDoS information with other detectors in order to diminish the strength of the attack. The authors report that, in terms of accuracy, the detection technique performed 5% better than related techniques. In the experiments, they did simulate for low rate DDoS attacks, however, the authors do not report on how the techniques performed on detection delay.

The work of Chuanlong et.al in [145] modelled an intrusion detection system based on RNN. They studied the performance of the detection system in binary classifications (normal and anomaly) and multiclass classifications (for Normal, DoS, R2L, U2R and Probe attacks). The experiment results reveal that the detection systems obtain an accuracy of 68.55% to 83.28% for binary classification. Meanwhile, for multiclass classification the detection system obtained an accuracy of 64.6% to 81.29%. Under the same experiment conditions, the authors found that the RNN had higher accuracy and

lower false positive rate. This performance was compared against previously used machine learning methods proposed by other researchers, namely ANN, naive Bayesian, random forest, multi-layer perceptron, support vector machine and other machine learning methods. The detection model looked at detecting the DoS attack, however, it did not incorporate the distributed variant of the DoS or its low-rate attack variant.

The work of Yuan et al in [146] proposed a deep learning approach called *DeepDefence*. The proposed system makes use of Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unit Neural Networks (GRU). The author extracted the 20 features from packet data, transformed and concatenated them into a sequence classification problem that is based on window detection. The extraction of features is transformed into matrices of size  $m*n$ , where  $m$  represents the number of packets and  $n$  represents the new features. The CNN was then used to train the model for classification. The detection method realised a classification accuracy of 97.61% with an average error rate of 2.4%; the model outperformed the Random Forest detection method.

Similarly, in [147], the authors proposed a lightweight CNN-based DDoS detection system with low processing overhead and a higher attack detection rate. The method gathers network traces into sliding time windows, which are then converted into feature arrays and fed to the CNN. The algorithm was effective in detecting DDoS attacks under the given conditions. It had a detection accuracy of 99.67% and a false alarm rate of 0.59%.

Roopak et. al. [148] further suggested several hybrid methods for detecting anomalous traffic packets and initiating an attack in an IoT network. In their testbed, the hybrid methods employed MLP, CNN, and LSTM. The combined LSTM and CNN algorithms performed better in the research experiments, with an accuracy of 97.16%. The subsection that follows will highlight research challenges for efficiently and accurately identifying high-rate and low-rate DDoS.

### 3.5 Research Challenges

The previous passage provided details of DDoS detection methods that have been used so far, however only a few of them have been effective. Designing and implementing an effective DDoS detection technique can be difficult, and even though modern computing capabilities can improve these detection techniques, there are a number of open challenges that still exist [3]. However, the work of this thesis will focus on the following research challenges:

Internet technologies are continuously being developed to improve the lives of people, for example the introduction of the Internet of Things where almost any object could be connected to the internet. These improvements have also presented an opportunity for attackers to develop more sophisticated attack schemes day by day, more especially those attack techniques that are designed to evade detection methods by launching low-rate and high-rate attacks. These kind of attacks can vary between zeros to maximum and have the ability to cripple a service provider within a matter of seconds [34]. This gives rise to the following research question:

- How can we detect low-rate and high-rate DDoS attack with accuracy and with minimal detection delay?
- How does the performance of statistical techniques and machine learning compare to modern deep learning convolutional neural network techniques when detecting low-rate and high-rate DDoS attacks?

Researchers have contended with the task of designing methods that identify optimal sets of features while not compromising on efficiency when designing detection techniques. This is as a result of the fact that many network traffic features typically have low variation or correlations that causes dependencies among the network traffic features. Researchers also discovered that the source IP address of incoming packets can be used as a useful parameter for detecting the start of an attack. However, the high dimensionality of IP address features, as well as the complicated association between them, results in significant computational overheads, making identification difficult. Moreover, the issue with scalability becomes more important when we consider the use of IPv6 address space [4, 149]. Therefore, in this thesis we ask the following question:

- How can the onset of a DDoS attack be identified on the basis of a simple feature of the source IP address?

Current DDoS attacks datasets have constraints: these are privacy and legal concerns involved with the sharing of recorded datasets. Thus, there is a lack of actual intrusion data that could be used to simulate attacks and to test and validate new detection techniques [4, 150]. From this challenge, we therefore ask:

- What are the key statistical features of a DDoS attack?
- How do we model the characteristics of DDoS attacks so that we can simulate and generate practical attack traffic datasets?

### 3.6 Summary

This chapter of the dissertation explored some of the most important techniques that are used for DDoS detection. We investigated literature to observe how researchers have used the classical statistics based change-point detection techniques and machine learning for DDoS detection; these are the EWMA, CUSUM, ANN, SVM, Logistic Regression algorithms. We explored their advantages and drawbacks as described in literature. We further explored the use of modern deep learning techniques for DDoS techniques; we looked at the structures of the recurrent neural network and the convolutional network. We further highlighted the research gaps and challenges that the dissertation will address, more specifically towards the development of detection techniques that will detect low-rate and high-rate DDoS attacks accurately and within a reasonable detection time delay. The next chapter describes and highlights the methodologies that are used to carry out the research.

## Chapter 4: Methodology

---

### 4.1 Introduction

The aim of this study is to design a testbed that will use comprehensible source IP address features to rigorously assess the efficacy of learning algorithms in detecting low-rate and high-rate DDoS attacks. The testbed will compare efficacies of entropy-based techniques and deep learning techniques based on their ability to detect DDoS attacks accurately with minimal detection delay and low false positive rate.

This chapter of the study describes the methods used in this research.

### 4.2 Research Design

In order to answer the research questions posed in this study the research design will follow a simulation modelling approach. The research will use statistical models that capture and describe the process of DDoS attack detection and run various simulation scenarios for both low-rate and high-rate DDoS attacks. This approach is best suited and widely used in the field of anomaly detection. The subsections that follow will explain the settings that were used to conduct various DDoS experiments for the purposes of learning and investigations. The experiments will further contribute towards understanding and determining the key distinguishing characteristics that differentiate authentic network traffic data from anomalous network traffic data.

### 4.3 DDoS Attack Detection using IP Address

The Internet Protocol (IP) is a standard internet communication protocol that devices utilize to connect across great distances. The IP address is a unique identifier for a device on the internet or a local network; researchers in [54, 149, 151-154] exploited its characteristics to identify abnormal behaviour. These academics' work highlights the significance of picking a variety of feature sets that may be utilized to distinguish normal network behaviour from abnormal network behaviour.

Researchers have proposed different combinations, including utilizing basic data such as source IP address traffic volume, a variety of source and destination address pairs, and the fraction of new source IP addresses per client, to name a few. However, when more feature combinations are employed, the solution space gets more sophisticated and scaling challenges arise. This study will examine the usage of source IP addresses and a combination of methods to construct a TCP SYN DDoS attack detection testbed, building on the work of the researchers in [149, 152, 155].

Feature selection is critical for information exploration in machine learning. In the feature selection process, researchers choose a subset of important data attributes to create robust and strong machine learning models for intrusion detection. By defining key data features and how they connect, we may gain a better understanding of the data. This will improve the learning model's efficiency in a variety of ways while also reducing the impact of the high dimensionality issue. With the rise of large data and the resulting requirements for effective machine learning techniques, new DDoS attacks have surfaced, and inventive detection measures are required [156].

The significance of feature selection cannot be overstated. It is essential to any detection technique and algorithm since it aids in distinguishing between network attack traffic and normal network traffic. The primary problem that researchers confront in developing an effective detection approach is the issue of high dimensionality in network traffic data, as well as the significant computing costs that this entails. The high dimensionality of network traffic data, along with the intricate correlation between the characteristics, results in significant computing overheads, making the design of an anomaly detection system extremely challenging.

The main difficulty with adopting IP addresses as a detection approach is scalability. For an IPv4, the researcher must compute and save statistical data for 232 elements of the IP address space. This necessitates significant computational and storage overheads, as well as monitoring fewer IP addresses both during regular traffic and during an attack. The design of detection techniques becomes considerably more difficult when dealing with an IPv6 address space, where the number of components in the address space grows to 2128 [149].

Researchers have utilized several factors linked to IP addresses to build detection algorithms for DDoS attacks over time. Attempts have been attempted to leverage IP address features such as traffic volume to change the number of distinct network flows, i.e. a grouping of destination and source IP, destination and source port, and protocol type [153]. Researchers in [138] concentrate their detection efforts on inbound traffic quantities and IP address dispersion. Entropy was also utilized by the researchers to assess the distribution and homogeneity of IP addresses. They categorize traffic flows based on their destination IP addresses and compare the traffic volumes of each category to the anticipated chi-square statistic. A deviation from the expected traffic profile indicates an attack. Researchers in [93, 157] examined entropy measurements across IP header characteristics as well. Entropy is a measure of feature distribution that is used to detect whether there is a divergence in network traffic performance.

Some academics have created a list of legitimate IP addresses using historical database techniques. These are IP addresses that have completed a three-way TCP handshake. This approach employs a sliding window update to keep an up-to-date IP address database. Only packets from IP addresses specified in the database are allowed during an attack or when the network is overloaded [54]. A cunning attacker might outsmart this approach by initiating a TCP handshake with the intent of subsequently launching an assault with other IP addresses.

Using the IP address, several researchers attempted to distinguish flash occurrences from DDoS attacks. Flash events (FE) occur when a network server experiences a surge in traffic requests from real and authentic clients; nonetheless, this surge can be compared to a high-rate DDoS attack. Researchers in [158] utilized an IP address aggregation method to distinguish FEs and identify DDoS attacks. The method assumes that during a FE, most customers' IP addresses would be geographically close together, but during a DDoS attack, IP addresses will be widely spread. The section that follows will look at how to simulate a DDoS attack using IP addresses and probability distribution fitting.

## 4.4 DDoS Attack Modelling

The field of network anomaly detection research is evolving, and with excellent datasets, it is critical to assess established detection algorithms. Several network intrusion datasets have been generated by well-known research organizations. The attempts were made to aid in the evaluation and validation of created approaches and algorithms. A superior dataset aids researchers in determining the efficacy of established approaches for detecting attacks when used in real-world operational situations.

Several public datasets, private datasets, and network simulation datasets were employed by the researchers. Most researchers, however, have cited publicly accessible datasets such as the DARPA Dataset, the KDD Cup dataset, the NSL-KDD Cup dataset, the DEFCON dataset, and the CAIDA dataset. These are yardstick datasets generated in experimental settings [156]. The DARPA dataset was chosen for this study owing to its prominence in the field of DDoS attack detection. This dataset, however, presents its own set of difficulties.

The lack of freely available real-world network traffic statistics is a natural challenge for academics when assessing the proposed approaches for detecting DDoS attacks. The statistics that generally earn reference are frequently out of date in terms of accurately displaying the most recent traffic directions. They have been excluded from sensitive data due to legal and privacy concerns. As a result, the majority of research on this issue is assessed using open-source traffic generators, simulation-based testbeds, and publicly available statistics. Each of these evaluation techniques has its own set of constraints. As a result of these constraints, researchers have developed low-cost, configurable, and scalable testbeds [159].

We selected this dataset because we previously deduced that IoT systems have an underlying IP network on which they function, making them vulnerable to typical IP attacks like DDoS. All network activity, including the whole content of each packet, were captured and submitted for evaluation using the tcpdump format. Sniffed network traffic, Solaris BSM audit data, and Windows NT audit data were employed in these studies. Finally, the test network was comprised of both real and simulated machines, with the real and simulated machines artificially generating background traffic while the attacks were carried out on the real computers.

### 4.4.1 TCP SYN DDoS Attack Modelling

In anomaly-based detection systems, researchers design detection systems by modelling the normal behaviour of an attack free network traffic. If an abnormality in standard behaviour is observed, then an attack will be detected. A DDoS attack brings about unexpected changes in the network traffic. Likewise, an unexpected variation in the statistical features of network traffic performance can be noticed. Should there be a DDoS occurrence at a particular time  $\lambda$ , the data will depict a substantial statistical variation about or from the time which is greater than  $\lambda$  [92].

The literature [160] has shown that statistical and mathematical behaviour of these attacks is usually regarded as entropy and the information theory of network traffic characteristics. These metrics capture the unusual distributional changes of the traffic data features in a single value. Therefore, sufficient observations of the changes in the

value can distinctly reveal the anomalies in the network therefore distinguishing attack traffic from normal traffic. Researchers have studied various probability distributions to capture the intricacy of the statistical properties of a DDoS attack. The Weibull, Gaussian and Logistic distributions are the most popular. The Weibull probability distribution function is generally used to represent and to model traffic features. To represent network traffic data, the Gaussian probability distribution model is often commonly used, while logistic regression distribution models are often used in the field of network traffic and attack modelling [161].

It is important for researchers to further understand which distribution model is best suited for DDoS attack detection. This will assist with building and designing more accurate detection techniques and algorithms. The work of researchers in [161] looked at implementing probability fitting and parameter estimation on many features of a DDoS attack. They provided probabilistic behaviour of traffic features of DDoS attacks. They found that the best fitted probability distribution for TCP SYN DDoS attacks is the Weibull, Gaussian and Logistic distribution. This is further confirmed by the work of researchers in [97]. Based on their analysis, they found that under normal legitimate network traffic, where humans were participating, and the probabilistic characteristics of the network data was that of a Gaussian distribution. In the instance where a DDoS attack is launched, where the network traffic data is auto-generated by bots or agents, the probabilistic character of the network data resembled that of a Poisson distribution. It is for this reason that for the design of the DDoS attack testbed, this study used the Poisson distribution for generating synthetic attack data.

This study further models the attack detection problem as a sequence classification problem [162]. In a sequence classification problem, let us assume that you have an input sequence  $x_0, \dots, x_T$ , and we want to classify an equivalent output  $y_0, \dots, y_T$  at each time step. The primary drawback is that in order to estimate the output  $y_t$  for any time  $t$ , we can only use inputs that have previously been recorded, i.e.  $x_0, \dots, x_t$ . In other words, the value of  $y_t$  relies on past values of  $x_0, \dots, x_t$  and not on the future inputs of  $x_{t+1}, \dots, x_T$ . Therefore, to model the sequence classification problem we use the function:

$$f: X^{T+1} \rightarrow Y^{T+1}$$

The function constructs the equivalent output:

$$\hat{y}_0, \dots, \hat{y}_T = f(x_0, \dots, x_T)$$

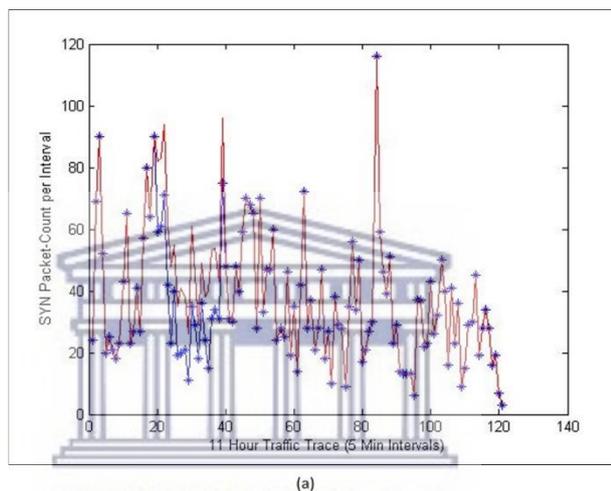
#### 4.4.2 Synthetic DDoS Attack Traffic Modelling

Researchers are faced with the challenge of obtaining realistic datasets, and the datasets that are currently available are too old and do not reflect the latest trends. To overcome this challenge many researchers have developed a customized traffic generation testbed simulation in order to evaluate designed techniques. It is for this reason that we used a similar approach in this research.

In this study, we evaluated attack-free data from multiple source IP addresses in the DARPA dataset to simulate attack traffic data. This is a dataset created by the MIT Lincoln Laboratory utilizing real-world traffic statistics. Trace data from network traffic captured during regular network operation is included in the data. In this experiment,

we looked at traffic data where there were notable traffic activities. As a result, 11 hours of traffic data were evaluated between 08h00 and 19h00. The dataset was filtered based on the source IP address and TCP protocol, and the TCP SYN was recorded. SYN packets were considered while examining the start of a TCP SYN flood attack.

The investigation focused on the computation of SYN packets at 10-second intervals. This was done so that these attacks may be synthetically produced in order to investigate an algorithm's performance across multiple attack characteristic circumstances. They were generated using a homogeneous Poisson process, which creates independent and exponentially dispersed delays between packet arrivals. The attack was planned to last 300 seconds and span 30 time intervals (each period is 10 seconds) (five minutes). Every five minutes of traffic was injected with attack data in order to consider all potential attacks employed in these experiments. In these studies, we analyse and simulate two types of assault characteristics: high and low intensity attacks. The specifics of these traffic characteristics will be discussed next.



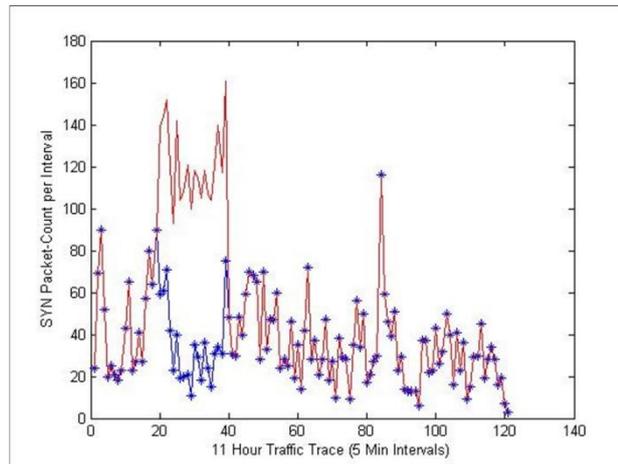
**Figure 11 Low intensity attacks (red line) seen on interval 20-40.**

Low intensity (or low rate) attacks are those whose intensity gradually develops until all resources are exhausted. This approach delays the detection of the attack by gradually degrading the victim's services over a long period of time. In the simulations, we discovered that a low-intensity assault had a mean amplitude that is 50% greater than the mean of regular traffic throughout a five-minute timeframe. This is seen in Figure 11. Attacks were synthetically introduced at 20-40 second intervals.

High rate attacks were discovered to have a sharp rise and the greatest amplitude in a single attack period. During the assault, packets are directed at a constant and continuous pace, with no interruptions or deviations in attack rate. The victim impact is rapid and abrupt [163]. In our experiments, high rate attacks were 250% higher than the typical packet rate in that particular timeframe. This may be seen between the same intervals 20 and 40 in Figure 12.

We applied the Poisson distribution flow to simulate DDoS attack traffic in our experiments, while we assumed a Gaussian distribution for normal network traffic. The Poisson distribution flow employed the following function:  $f_p(k; \lambda)$  where the non-

negative integer  $k \in [0; \infty]$  and the positive real number  $\lambda$  is the average packet rate per second for a given timeframe.



(b)

**Figure 12 High Intensity attacks (red line) seen on intervals 20-40.**

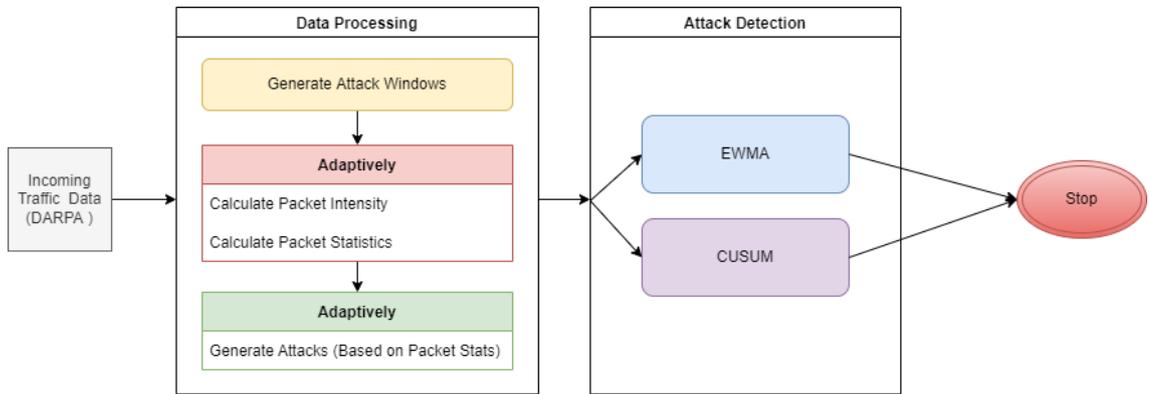
A sliding time window approach was used for this investigation, as shown in Figure 11 and Figure 12. Due to the assumption of stable time series used by the bulk of statistical and signal processing techniques, this approach is very beneficial for examining time series data. Time series, on the other hand, are never stationary in the actual world. A time series can be employed as long as it is at least locally stationary, that is, its statistical characteristics change slowly over time [164]. Previous research [138, 165-167] used the sliding time window technique to detect DDoS assaults and abnormal traffic data.

#### 4.5 Change Point Detection Algorithms: CUSUM and EWMA

The CUSUM algorithm and the EWMA algorithm, which was adopted for this study, was described in detail in section 3.2.1 and section 3.2.2. Both algorithms share similar tuning parameters and these are the amplitude factor,  $\alpha$ , the weighting factor,  $\beta$ , and the CUSUM algorithm threshold,  $h$ , and the EWMA algorithm factor,  $k$ . In order to address the research questions for this study, an investigation must be carried out to determine the optimum values for the tuning parameters for this specific study. The investigation assumed that other parameters are held constant in order to assess partial variation in the one parameter that is caused by variation in the other parameter while other variables do not change. The investigation tested the following parameter's effects on both the detection rate and false positive rate:

- The effect of the amplitude factor,  $\alpha$ . The parameter value was modified to vary between 0.05 and 1.0, while all other parameters were held constant.
- The effect of the weighting factor,  $\beta$ . The weighting factor was varied between 0.8 to 1.0, while other parameters values did not change.

The EWMA and CUSUM algorithms effects of parameter tuning and simulation for low-rate and high-rate DDoS attacks will be presented in greater detail in the research findings section of the algorithms.

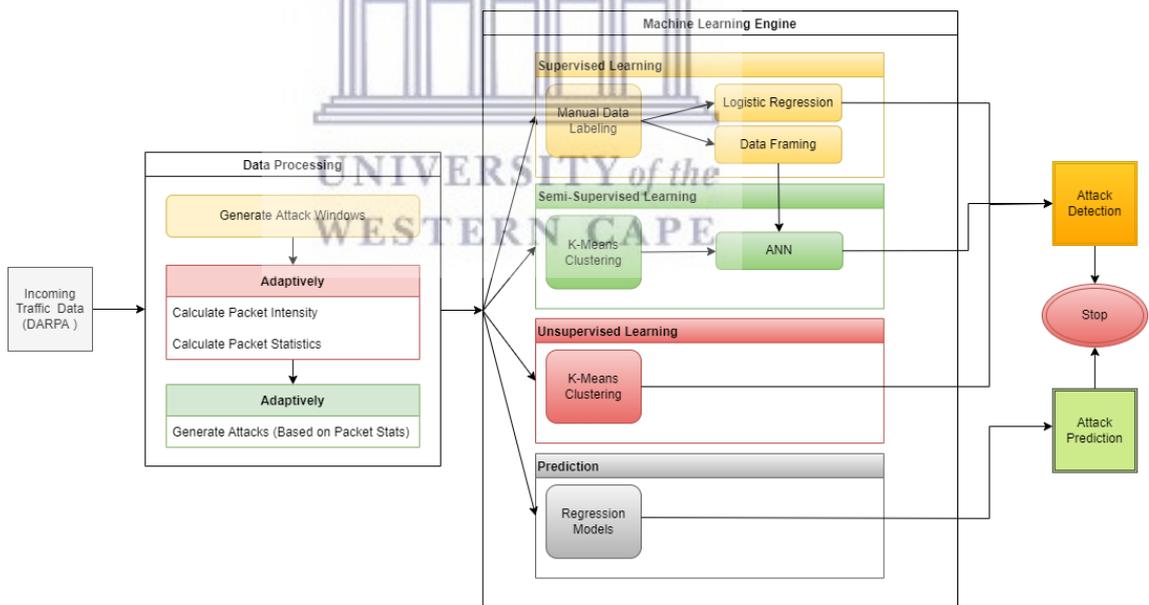


**Figure 13 Change Point Detection Framework**

The change point framework and testbed design are depicted in Figure 13. Incoming traffic data is obtained from a variety of IP addresses. Attack detection sliding windows are established, and then packet statistics and packet intensity statistics are generated for each sliding window as a baseline for developing adaptive attacks. The attack detection module accurately classifies and detects attacks for each sliding time window using any of the algorithms.

## 4.6 Machine Learning Techniques

Figure 14 depicts the proposed machine learning system's primary components, which include data pre-processing, supervised learning, semi-supervised learning, unsupervised learning, and prediction. Each of these components is outlined below:



**Figure 14 Machine Learning System Architecture**

### 4.6.1 Data Processing and Labelling

The raw dataset was processed by developing a Python script that recorded the number of network packets that arrived at a specific host each time interval of 10 seconds. We adopted this as the data baseline and labelled each interval (10-second block) as zero (0), indicating that there was no DDoS attack. The dataset was then injected with attacks

traffic by manually increasing the amount of packets arriving at randomly selected intervals. These intervals were labelled as one (1), signifying a DDoS attack. Figure 15 visually represents the process and shows the packet count for the first 2 minutes (120 seconds) of traffic flow. The first image shows genuine traffic, whereas the bottom image shows the presence of DDoS attack traffic (highlighted in yellow).

LEGITIMATE TRAFFIC	
Time	0 10 20 30 40 50 60 70 80 90 100 110   120 130 140 150 160 170 180 190 200 210 220 230...
Packet#	2 7 0 2 0 2 9 0 1 3 10 9   8 10 5 9 8 6 7 2 9 8 8 4...

ATTACK	
Time	0 10 20 30 40 50 60 70 80 90 100   110 120 130 140 150 160 170 180 190 200 210 220 230...
Packet#	2 7 0 2 0 20 90 100 110 75 90   9 8 10 5 9 8 6 7 2 110 88 91 4...

**Figure 15 Data Labels for Normal and Attack Traffic**

#### 4.6.2 Supervised Learning

The supervised learning component in Figure 14 is marked in yellow, and it involves manual labelling of data processing and machine learning modelling. Supervised learning is a type of machine learning (ML) in which an ML model is trained using labelled data that serves as "instances" for the ML model. Once trained, the model may be applied to test data for classification or prediction. We evaluated Logistic Regression (LGR) and Artificial Neural Network (ANN) models in our system.

<b>DF1</b>	<b>Label = 0</b>	<b>DF3</b>	<b>Label = 1</b>																								
<table border="1"> <tr><td>2</td><td>7</td><td>0</td></tr> <tr><td>2</td><td>0</td><td>2</td></tr> <tr><td>9</td><td>0</td><td>1</td></tr> <tr><td>3</td><td>10</td><td>9</td></tr> </table>	2	7	0	2	0	2	9	0	1	3	10	9	$\sigma = 3.86$	<table border="1"> <tr><td>2</td><td>7</td><td>0</td></tr> <tr><td>2</td><td>0</td><td>20</td></tr> <tr><td>90</td><td>100</td><td>110</td></tr> <tr><td>75</td><td>90</td><td>9</td></tr> </table>	2	7	0	2	0	20	90	100	110	75	90	9	$\sigma = 45.94$
2	7	0																									
2	0	2																									
9	0	1																									
3	10	9																									
2	7	0																									
2	0	20																									
90	100	110																									
75	90	9																									
<b>DF2</b>	<b>Label = 0</b>	<b>DF4</b>	<b>Label = 1</b>																								
<table border="1"> <tr><td>8</td><td>10</td><td>5</td></tr> <tr><td>9</td><td>8</td><td>6</td></tr> <tr><td>7</td><td>2</td><td>9</td></tr> <tr><td>8</td><td>8</td><td>4</td></tr> </table>	8	10	5	9	8	6	7	2	9	8	8	4	$\sigma = 2.34$	<table border="1"> <tr><td>8</td><td>10</td><td>5</td></tr> <tr><td>9</td><td>8</td><td>6</td></tr> <tr><td>7</td><td>2</td><td>110</td></tr> <tr><td>88</td><td>91</td><td>4</td></tr> </table>	8	10	5	9	8	6	7	2	110	88	91	4	$\sigma = 40.98$
8	10	5																									
9	8	6																									
7	2	9																									
8	8	4																									
8	10	5																									
9	8	6																									
7	2	110																									
88	91	4																									

**Figure 16 Data Frame Samples**

##### 4.6.2.1 Data Framing

Data framing was performed just for ANN, and three variations were explored. The first did not consider data framing, but the second divided the dataset into "data frames" of size 12, equivalent to 120 seconds of traffic flow (10 seconds interval). The standard deviation of the values in the data frame was computed and attached to the frame in the second, giving it a size of 13. The data frames were then supplied into the ANN model as input. The algorithm below summarizes the data framing procedure using pseudocode.

---

**Algorithm 1: Data Framing Algorithm**

---

Divide the entire dataset into data blocks of 120 seconds.

For each 120 second data block in the dataset:

    Create a 3 by 4 data frame as follows:

        Set  $t = 0$

        For row = 1 to 4

            a.  $col1 = Packet\_Count(t), t += 10$

            b.  $col2 = Packet\_Count(t), t += 10$

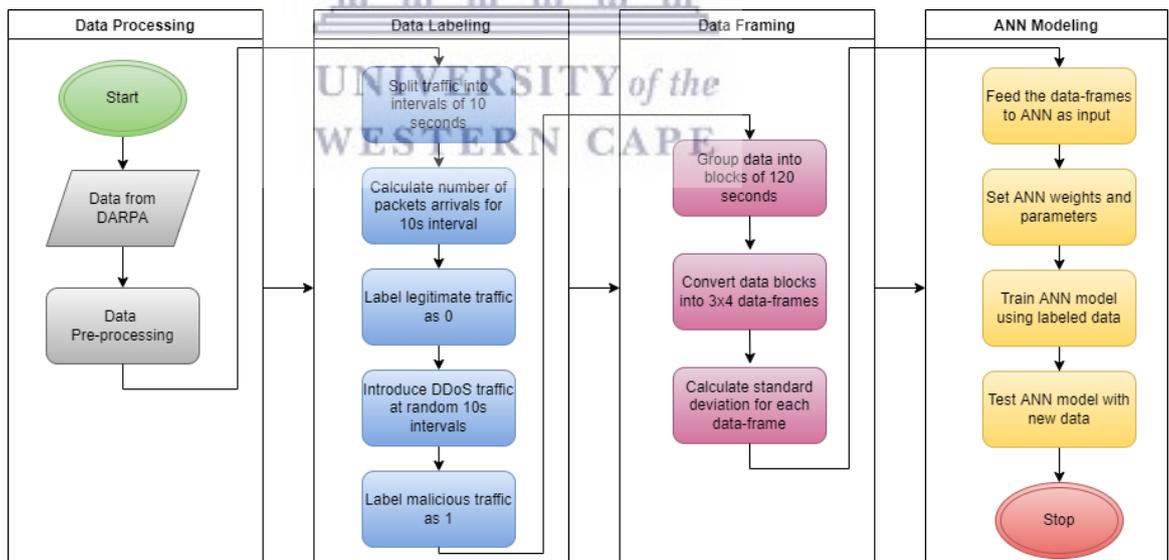
            c.  $col3 = Packet\_Count(t), t += 10$

        Calculate the standard deviation ( $\sigma$ ) for the data block.

    end

---

As an example, by executing Algorithm 1 on the sample data in Figure 15, we get four data frames, as shown in Figure 16. DF1 and DF2 in the figure are data frame representations of normal traffic (top image in Figure 15) and are labelled 0, indicating that there is no DDoS attack. Both DF3 and DF4 reflect traffic flows involving malicious attacks, and are thus labelled 1. The standard deviation ( $\sigma$ ) is computed for each data frame. This standard deviation is used to confirm the likelihood of a malicious attack. If the data points deviate from the mean, the deviation within the dataset increases, indicating an attack. The reverse is true for nearby data points, which have smaller variance and are less likely to constitute an attack. Finally, a threshold value is applied when all 12 elements in a data frame are high (during high rate DDoS attacks) and near to each other, resulting in a decreased standard deviation value. If the estimated value is more than this threshold value, the frame is considered to be under attack.



**Figure 17 ANN Supervised Learning Process**

#### 4.6.2.2 Machine Learning Models

In this study, the LGR and ANN were considered. The One-vs-rest (OvR) training scheme and limited memory BFGS (Broyden–Fletcher–Goldfarb–Shanno) algorithm solver were

adopted for LGR, with an 80:20 split for training and test data. ANN was modelled with 3 layers – the input layer (13 nodes), the hidden layer (12 nodes) and the output layer (1 node). The data frames acquired from algorithm 1 are fed in, with the 13th node being the corresponding standard deviation value. The input and hidden layers were activated with the (Rectified Linear Unit) ReLu function, while the output layer was activated with the Sigmoid function. The processes involved in our ANN supervised learning component are depicted in Figure 16 and Figure 17.

#### **4.6.3 Unsupervised Learning**

We investigated using the K-means clustering algorithm to automatically label the dataset as an alternative to manually labelling it. In our system design, depicted in Figure 14, this component is indicated in orange. K-Means is a centroid-based clustering technique that evaluates cluster membership based on data point closeness to a centre point (centroid) [168]. It has been used to solve a variety of classification challenges, including network classification [169], and intrusion detection [170]. Millions of packets pass the network every unit time in IP network security, and must be classified (labelled) as either genuine or malicious traffic. Manual labelling would be slow and arduous in such instances, thus using an automatic classifier, in our case K-Means, is preferable. In our work, traffic flow is classified as either genuine or malicious, hence the k value is set to 2.

#### **4.6.4 Semi-Supervised Learning**

Our semi-supervised learning component, shown in green in Figure 14, is identical to the supervised learning outlined previously. The main change is that instead of employing manually labelled data as input to the ML models, we supplied the models which are the result of unsupervised learning (k-Means clustering). K-Means is essentially used to automatically label (classify) the data, which is subsequently employed to train the supervised model. This results in a hybridization of a supervised and unsupervised model, or, in this case, a semi-supervised model. The output of this model is then compared to the output of the other two models (supervised and unsupervised).

#### **4.6.5 Prediction**

After successfully classifying and distinguishing between normal and malicious attacks, the next natural step may be to forecast the likelihood of such attacks occurring. This would assist the network administrator in implementing preventative steps to minimize them, basically shifting the defence strategy from reactive to proactive. In Fig. 2, this is indicated in grey. For prediction, three regression models were used in this study: Logistic Regression (LGR), Kernel Ridge Regression (KRR), and Support Vector Regression (SVR).

LGR calculates the likelihood of an occurrence, known as the dependent variable, based on one or more independent variables ( $s$ ). LGR was chosen because it is ideally suited for determining binary output probabilities, i.e. True or False (1 or 0), and it does not require a linear connection between the dependent and independent variables. When we applied it to our work, we employed it to predict whether an attack will occur at a

specific period in the future. The independent variables were traffic 'count' and 'status' (i.e. normal (0) or attack (1)), whereas the output variable was 'time'.

SVR is a support vector machine variant proposed in [171] that attempts to reduce the predictor coefficient to a value less than or equal to a predefined threshold. To discover ideal settings, we employed the Gaussian Radial Basis Function (RBF) as the kernel and grid search.

KRR, like SVR, employs the kernel technique (RBF), but instead of the epsilon employed by SVR, it employs a ridge as the loss function. KRR is quicker than SVR because it combines the kernel technique with least square regression [172, 173]. We selected KRR because it is comparable to SVR and because there are substantial variances (differences from the mean) between data points in the dataset. When the packet count in normal traffic flow (labelled 0) is compared to that of attacks, the differences become more noticeable (labelled 1).

#### **4.7 1D Convolutional Network for Time Series and Sequential Data Classifications**

Section 3.4 of this study looked at the CNN literature. Much of the research on CNNs has been concentrated on 2D signals such as image and video frame processing. Krizhevsky et al. launched the first Deep CNN model, AlexNet [174]. On the ImageNet benchmark database, the AlexNet reported a false positive rate of 16.4% (10% lower than conventional ML techniques), and it was designed with an 8-layer CNN (5 convolutional-pooling layers and 3 fully-connected layers). The authors developed the Rectified Linear Unit (ReLU), a novel and now common architectural feature that replaced activation functions such as the Sigmoid (*sigm*) and Tangent Hyperbolic (*tanh*). In a brief period of time, their proposed deep learning CNN architecture rapidly displaced standard ML techniques. Their influence gradually led to them becoming the industry standard for a variety of ML applications such as computer vision, natural language processing, and speech recognition [175].

The CNN and LSTM architectures were used to simulate high and low rate DDoS attacks in our work. We experimented with different convolution layer settings in order to optimize the models proposed in [176] for time series classification tasks. We also experimented with and used two data feature preparation modelling approaches, lag features and window features, for data and feature modelling. Lag features is a time series transformation technique that predicts or classifies the current time value based on the value of the previous time value; and rolling window statistics features adds extra statistical features to the lag features. Lag features is a time series transformation technique that predicts or classifies the current time value based on the value of the previous time value; and rolling window statistics features adds extra statistical parameters to the lag features.

The CNN was first intended for 2D signal processing, picture classification, and natural language processing applications. Meanwhile, LSTM includes feedback connections that can analyse numerous data points in pictures as well as sequence data such as voice and video. The modelling framework for our experimental testbed is most likely unsuitable for the CNN and LSTM architectures. Some researchers have adapted CNN and LSTM to

meet the problem description and specification, and we will offer a few examples below; further examples may be found in section 3.4.

Chen et al [177], evaluated the performance of the Random Forest and SVM models and a CNN model for DDoS attack detection. In their tests, they created a multiclass classification CNN that separates benign traffic from five different forms of DDoS attacks: MSSQL, NETBIOS, NTP, SYN, and UDP Lag attacks. In these studies, they chose 24 features that had a high correlation with the type of attack. The performance of CNN simulations was compared to that of SVM and Random Forest. The results show that the CNN models perform similarly to the Random Forest algorithm, with 94% precision and recall. In terms of accuracy, the CNN model achieved 98%, while the Random Forest model achieved 94%. Both models exceeded the SVM (92%) by a slight margin of accuracy. Since they selected 20 strongly correlated features in their investigations, the CNN model built for these studies may be resource intensive and hence suffer from high dimensionality of data features. The authors did not divulge the detection delay or the false positive rate performance of the CNN models in the investigations.

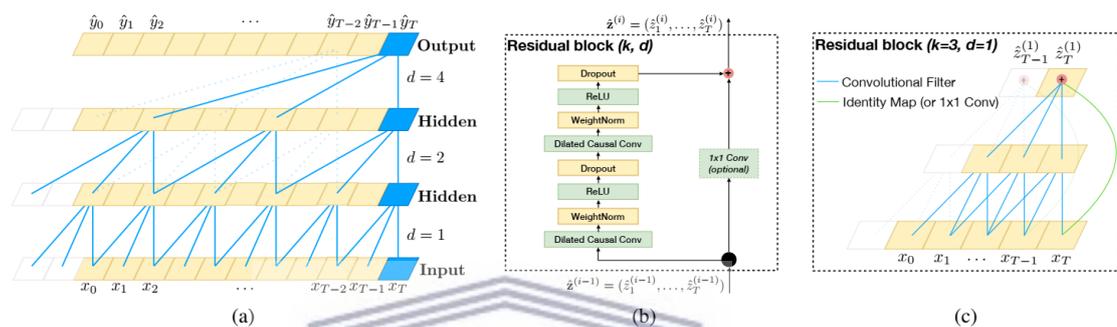
McCullough et al. [178] deployed an LSTM CNN to identify TCP DDoS attacks in an IoT network setting. The detector's architecture is based on the idea that CNN was initially created for picture classification; hence, the approach of displaying network traffic data as a graphical heat map was introduced. They employed a 255x255 matrix with three colours to indicate the volume of traffic in a particular time frame as incoming and outgoing packets. In their studies, they discovered that this technique had a 99.8% accuracy rate and a 0.1% false alarm rate, outperforming the SVM simulation. The simulations did not show how the technique would perform on DDoS attacks with low and high attack intensities, nor did they provide the average detection delay for an attack.

Deep CNN's popularity has motivated researchers to use it for 1D signal processing applications, which necessitated a 1D to 2D design conversion because deep CNN is modelled and configured for 2D signal processing. Applications such as electrocardiogram (ECG) beat classification and arrhythmia diagnosis are examples of these conversion methods. Kiranyaz et al. [179] suggested the first 1D CNNs capable of processing ECG signals. 1D CNNs achieved state-of-the-art efficiency in a limited period of time in a variety of signal processing applications, including structural damage detection, high-power engine fault detection, and real-time monitoring of high-power circuits [180].

The current study, as explained in section 4.4.1, resembles the sequence classification problem and it is for this reason that this study adopts the 1D CNN for DDoS attack detection. The significant advantage of adopting 1D CNNs over 2D CNNs is their computational complexities [180]. A 2D CNN application with  $N \times N$  dimensions will convolute with a  $K \times K$  kernel that will result in a computational complexity of  $\sim O(N^2 K^2)$ . On the other hand, a 1D application with the same dimensions  $N \times K$  will result in a significantly lower computational complexity of  $\sim O(NK)$ . There is very little published research on the topic of applying 1D CNN for DDoS attack detection, however, this study adopts the 1D fully convolutional (FCN) deep learning architecture that was designed by Bai et. al. [162].

The deep learning architecture adopted for this study is a variation of the CNN that is designed for sequence modelling problems and it is called Temporal Convolutional

Networks (TCN). With its temporality and broad receptive fields, the TCN architecture utilizes casual convolutions and dilated convolutions to be adaptive for sequential data [181]. Figure 18 depicts the building blocks of the architecture of a TCN, Figure 18 (a) is a dilated casual convolution with dilation factors  $d = 1, 2, 4$  and filter size  $k = 3$ . Usually, several convolutional layers are stacked on top of each other. In order to cover the receptive field of all values from the input sequence, the dilation factor of subsequent convolution layers is exponentially increased. The dilated causal convolution layer, acts over each sequence's time steps. Figure 18 (b) is a TCN residual block each of which comprises two sets of dilated causal convolution layers of the same dilation factor, followed by normalization, ReLU activation, and spatial dropout layers. The  $1 \times 1$  convolutional residual block is appended when the input and output sequence have dimensions that do not match. Then a final activation function is applied. Figure 18 (c) is an example of a residual connection; the filters in the residual function are represented by the blue line, while identity mapping is represented by green lines.



**Figure 18 Architecture of a TCN [162]**

For the purposes of this study, the TCN architecture was designed with four residual blocks of 1D convolutional layers and a fully connected convolutional layer. A residual block contained three input channels and each dilated causal convolution layer comprised 175 size 3 filters, and a dropout factor of 0.05. The training parameters of the study were setup such that each custom loop will train for 30 epochs with a one mini-batch size. The **initial** learning rate was set to 0.001 and the learning rate drop-factor was set to 0.01 for every 12 epochs, and the gradient threshold was set to one. The training and testing data-split was set to 70:30 ratio, respectively.

## 4.8 Summary

This chapter has described methods that will be used in this study. The chapter began by describing the simulation modelling research design that will be implemented in this investigation. It further described the dataset that was used for this investigation, and it further justified the use of a source IP address as a key feature for modelling TCP SYN DDoS attacks. The modelling techniques and the methods used for generating synthetic attack traffic was also described, together with the framework for the testbed. The chapter then described the environmental setup for the development of the testbed and testing environment for the simulations. The simulation parameters settings for the CUSUM, EWMA, machine learning (LGR, K-Means, ANN, KRR and SVR) and the 1D CNN

algorithms were also explained and the results obtained from these simulations are described in the chapter that follows.



## Chapter 5: Research Findings and Discussions

---

### 5.1 Introduction

In the previous chapter we outlined the methods and the simulation modelling approach that will be applied in this study. In this chapter, we present the core findings of the simulations that were proposed for the three machine learning algorithms. The algorithms' performance is analysed for their accuracy and efficiency, and a contextual analysis of these findings is also presented in this chapter.

### 5.2 Algorithm Performance Metrics

For statistical techniques we will measure the detection rate, false positive (FP) rate, false negative (FN) rate, and detection latency for the objectives of this study, and we will also employ Receiver Operating Characteristics (ROC) curves. (ROC) curves are a popular method for depicting the relationship between an intrusion detection system's True Positive (TP) and False Positive (FP) rates. It may also be used to compare the accuracy of two or more classifiers. It uses the orthogonal coordinate system to visually display a classifier's detection performance [4].

For machine learning techniques, we further use the F1-score; the F1-score is another measure of accuracy for a binary classification model. It is the harmonic mean between precision and recall. For prediction, we will use regression models; therefore we will specifically use the accuracy, coefficient of determination ( $R^2$ ), and the Root Mean Square Errors (RMSE) to evaluate the performance of the machine learning techniques.  $R^2$  measures the percentage of variation that is explained by the relationship between the dependent variable and the independent variable. The RMSE measure represents the magnitude of the error of a model in predicting quantitative data. It is the standard deviation of the errors of predictions.

The network anomaly detection algorithm should perform its duty and generate an alarm in a timely basis, allowing the system administrator to take necessary action before causing permanent harm to the system. The detection delay is the average time between the occurrence of an attack and the generation of an attack signal or reaction by the detection system. The formulae below define the evaluation metrics that will be used.

$$\text{Detection rate} = \frac{\text{Total number of correctly classified instances}}{\text{Total number of testing instances}}$$

$$\text{Accuracy} = \frac{(TP + TN) * 100}{TP + TN + FP + FN}$$

$$\text{False Positive (FP) Rate} = \frac{FP * 100}{(FP + TN)}$$

$$\text{False Negative (TN) Rate} = \frac{FN * 100}{FN + TP}$$

$$F1 - \text{score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

$$\text{Coefficient of Determination (R}^2\text{)} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

$$\text{Root Mean Square Errors} = \sqrt{\frac{\sum_{i=1}^N (y_i - \bar{y}_i)^2}{N}}$$

### 5.3 Algorithm Design (Environment Setup for Algorithm Development)

This study's experiments were carried out in two technical settings. The testbed and algorithms were created in a desktop version of Matlab R2021a, and the simulations were run on the High Performance Computer (HPC). The desktop Matlab R2021a was run on a 64-bit Windows 10 Enterprise Operating System (OS), 8GB RAM, Intel® Core™ i5-8350U CPU @ 1.70GHz. For simulations of the EWMA, CUSUM and 1D CNN, the HPC runs the Red Hat Enterprise Linux 7.4 and is made up of 128 standard computer nodes, 28 cores per node, and thus consists of 3584 cores. Each node uses 256GB RAM and the Intel® Xeon® E5-2690 v4 CPU @ 2.60 GHz. The Graphical Processing Unit (GPU) accelerator node consists of the NVidia® Tesla® P100 with 16GB on-board RAM. The simulations for the LGR, K-Means, ANN, KRR, and SVR algorithms were performed on Google Colab, which was operating on a Python 3 Google Compute backend with 12GB of RAM, a 2.3GHz 2 Core Intel Xeon CPU, and GPU hardware accelerators. For machine learning, Keras and Sci-Kit learn were used; Smote was employed for data balance; Pandas and NumPy were used for data manipulation; and Matplotlib was used for data visualization.

### 5.4 Statistical Change Point Detection

#### 5.4.1 Cumulative Sum (CUSUM) Technique

The CUSUM algorithm is a change detection technique that calculates the entropy and log-likelihood ratio for detecting abrupt changes in sequence data. It is based on the hypothesis that an abrupt change in statistical distribution properties will be observed at a point in time, before and after a DDoS attack has begun. The algorithm was developed for independent and identically distributed random variables; therefore, in the modelling and adaptation of the CUSUM algorithm we assumed that the TCP SYN sequence data are independent Gaussian random variables with known variance and

known mean. In the CUSUM algorithm, the tuning parameters are the amplitude factor,  $\alpha$ , the weighting factor,  $\beta$ , and the CUSUM alarm threshold,  $h$ .

The experiments setup in this study were designed to investigate the efficiency of the CUSUM algorithm for detecting both low and high intensity attacks. We further investigated the effect on the algorithm efficacy of the tuning parameters.

- The effect of the amplitude factor  $\alpha$ , on the detection rate.
- The effect of the weighting factor  $\beta$ , on detection rate.
- The trade-off analysis between detection rate and the false positive rate.
- The trade-off analysis between the detection rate and detection delay.

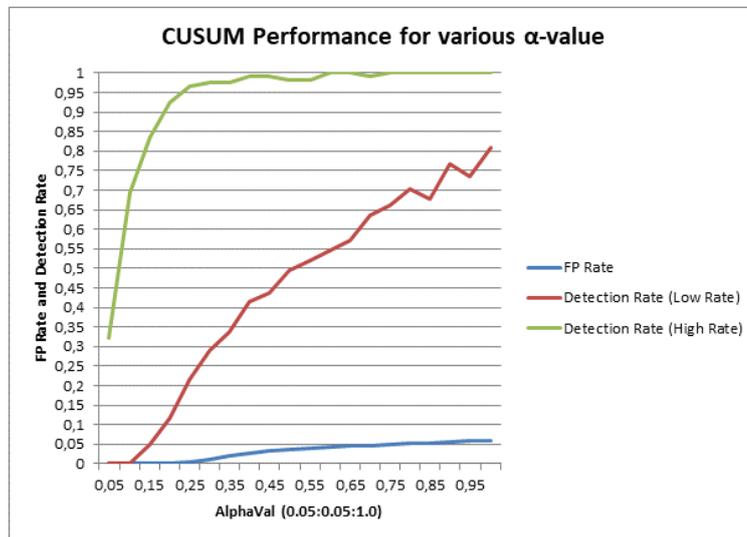
The findings and analysis from the experiments and simulations will be elaborated in the sub-sections that follow.

#### 5.4.1.1 *The effect of the Amplitude factor ( $\alpha$ )*

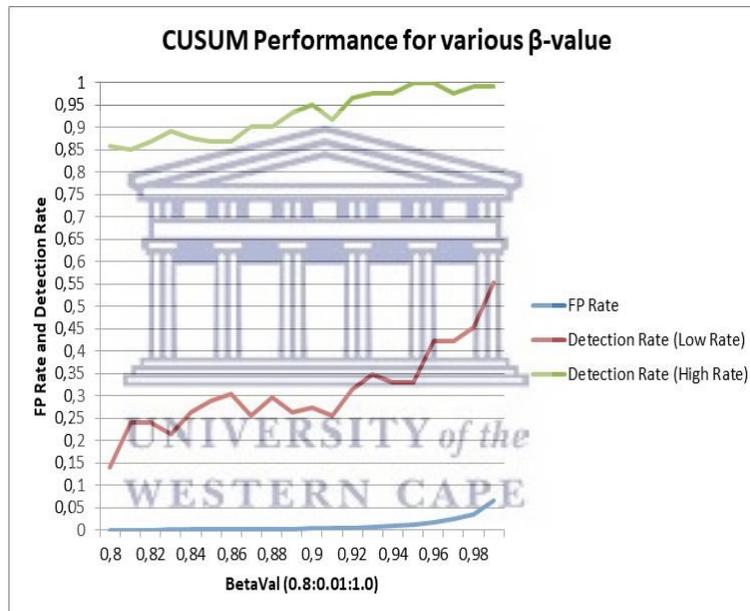
In this part of the simulation experiment, we seek to determine the effect of the amplitude factor ( $\alpha$ ) on the efficacy of the algorithm, i.e. the detection rate. The control variables for this experiment were the weighting factor, and it was held constant  $\beta = 0.8$ ; the CUSUM alarm threshold, which was also held constant,  $h = 3$ . The independent variable of the experiment was the amplitude factor, for which a linear increment and iteration was applied from 0.05 up to 1.0. The dependent variable, in this case the detection rate, was then observed throughout the experiment.

Figure 19 depicts a line chart of the results of this experiment. From the line chart it can be observed that for low-rate attacks, the detection algorithm yields a detection rate of between 0%-81% and a false positive of between 0%-7% for values of  $0 < \alpha \leq 0.5$ . Again, it can also be observed that for high-rate attacks and values of  $0 < \alpha \leq 0.95$ , the detection algorithm yields a 32%-100% detection rate while having a false positive rate of between 0%-6%.

The above experiment signifies that when the value of the amplitude factor increases, the values of the detection rate and the false positive rate also increases simultaneously. The false positive rate for both low rate and high rate attacks increases sharply starting from  $\alpha \geq 0.25$  and it plateaus at the point when  $\alpha \geq 0.5$ . For values of  $\alpha \geq 0.5$ , the algorithm experiences a higher accuracy rate and a plateaued false positive rate. Therefore, for further experiments in this investigation, the value of the amplitude factor will be set to 0.5.



**Figure 19** Detection rate and False Alarm rate for varied amplitude factor value ( $\alpha$ ) for both Low rate attacks and High rate attacks.



**Figure 20** Detection rate and False Alarm rate for varied Weighting factor ( $\beta$ ), for both Low rate attacks and High rate attacks

#### 5.4.1.2 The effect of the Weighting factor ( $\beta$ )

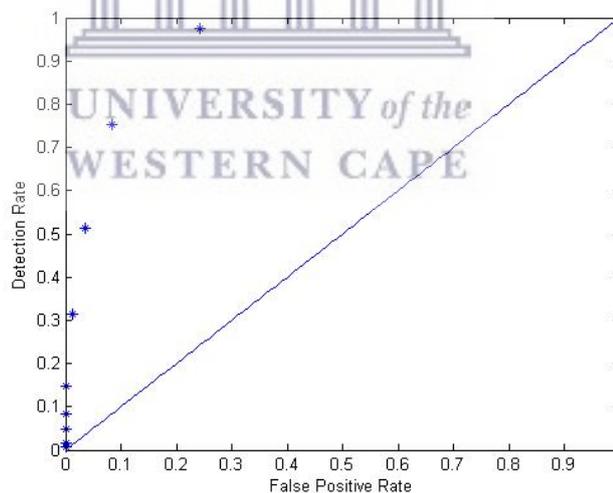
The aim of this simulation experiment is to determine the impact of the weighting factor ( $\beta$ ) on the algorithm's detection rate. The control variables were the amplitude factor, which was constant  $\alpha = 0.5$ ; the CUSUM alarm threshold, which was also constant,  $h = 3$ . The weighting factor was the independent variable, and a linear increment from 0.80 to 1.00, was used. Throughout the experiment, we monitored the detection rate as the dependent variable.

From Figure 20 it was observed that the performance of the detection algorithm was generally poor and unsatisfactory. For low-rate attacks, the algorithm had a detection rate of between 14%-56%. For high-rate attacks, shown by Figure 20, the detection rate was between 85% and 100%; while the false positive rate was between 0% to 7%. The CUSUM algorithm reached a 100% detection rate for values of  $\beta \geq 0.95$ . Even though the false positive rate for the experiment was admissible, the higher detection rate was accompanied by a sharp increase in the false positive rate. As a result, there is a trade-off between a better detection rate and a higher false positive rate, and the weighting factor value will be adjusted to 0.98 for further experiments.

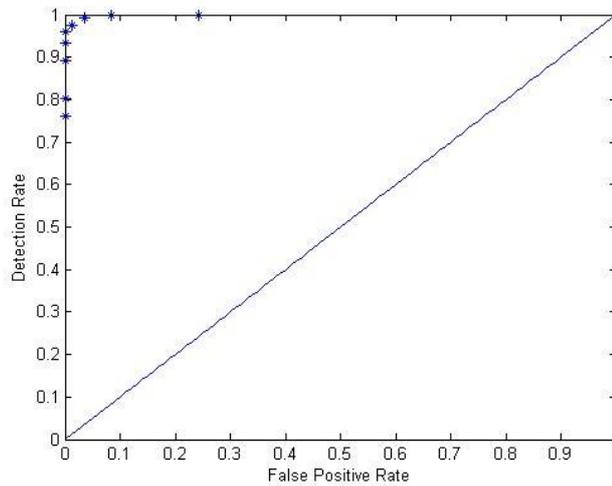
#### 5.4.1.3 False Positive Rate and Detection Rate

In this set of experiments we investigated the trade-off between the false positive rate and the detection rate. As determined in past experiments, the values for the tuning parameters were as follows: the amplitude factor  $\alpha = 0.5$ ; the Weighting factor  $\beta = 0.98$ . The CUSUM alarm threshold  $h$  was varied from 1 to 10.

Figure 21 and Figure 22 represent the receiver operating curves for both the low rate and high-rate attack experiments, with each point representing a distinct value of  $h$ . The operational points on the graph that are closest to the upper-left corner of the graph are the best. Figure 21 shows results for the experiments simulating low-rate attacks. When it came to low-rate attacks, an improvement in the algorithm's detection accuracy was followed by a significant rise in the false alarm rate. As a result, greater detection accuracy will result in a higher false alarm rate. This is not an ideal performance in a detection method. As a result, the CUSUM algorithm performed poorly in scenarios of low-rate attacks.



**Figure 21 ROC Curves for low rate attacks**

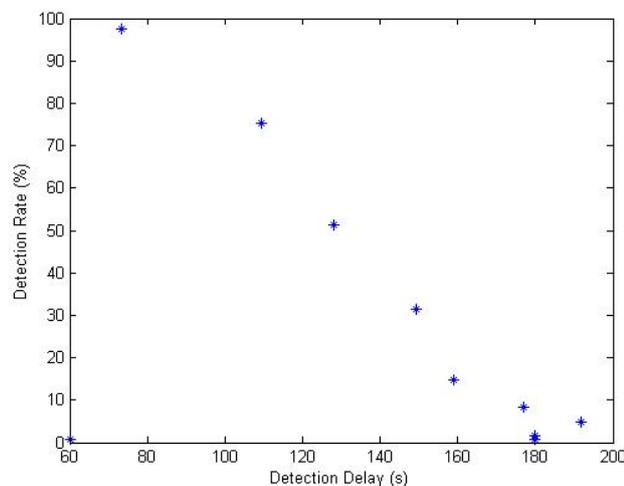


**Figure 22 ROC Curves for high rate attacks**

Figure 22 demonstrates the CUSUM algorithm's performance in the simulation of high-rate attacks. The majority of the operational points are located in the graph's upper-left corner. This also suggests that a greater detection rate is associated with a modest rise in the false alarm rate. When compared to the low rate attack simulations, this represents an enhanced performance.

#### 5.4.1.4 Detection Rate and Detection Delay

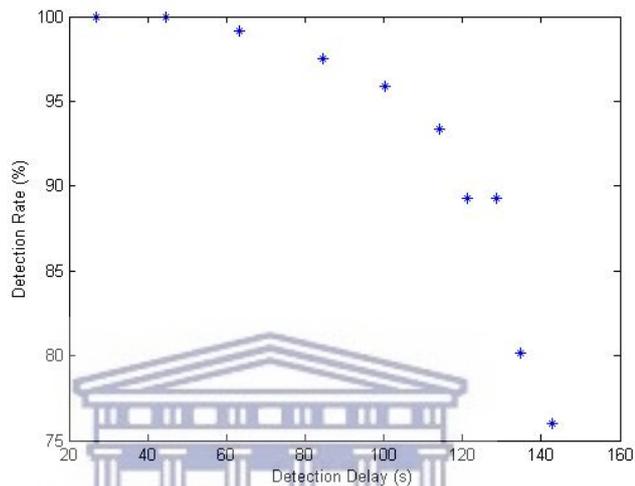
In the following set of experiments, we looked more closely at the trade-off between detection rate and detection latency. The outcomes are depicted in Figure 23 and Figure 24. In this scenario, detection delay is the average time it takes the algorithm to effectively identify an attack from the start of that attack. Each point represents a detection rate and an average detection delay. The tuning parameter values were as follows: the amplitude factor  $\alpha = 0.5$ ; the weighting factor  $\beta = 0.98$ . The value of the alarm threshold  $h$  was varied from 1-10.



**Figure 23 Graph displaying the trade-off between Detection Rate and average Detection Delay for low rate attacks**

Figure 23 displays the CUSUM algorithm's performance trade-off between detection rate and average detection delay for low-rate attack simulations. According to the graph, as the detection rate increases, so does the detection delay. The CUSUM failed to achieve a 100% detection rate in the simulation with low-rate attacks, although the best average detection delay was little under 80 seconds (73.3s).

Figure 24 depicts the outcomes of simulations with high-rate attacks. For high-rate attacks, the algorithm showed improvement. The average detection times for a 100% detection rate were 26.94s and 44.71s for varied alarm threshold levels. It can also be shown that when the detection rate performance falls, so does the average detection delay.



**Figure 24 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks**

#### 5.4.1.5 Summary

The CUSUM detection algorithm's performance against low-rate attacks was not satisfactory. The ROC curves revealed that the algorithm was not able to attain a 100% detection rate for the varied set of low-rate DDoS attack simulations, and the highest detection rate for the simulations was 97.52%. However, for all low-rate DDoS attack simulations, the algorithms managed to maintain an average false positive rate of 3.75%, which is relatively low. The algorithm further attained a best detection delay that was a little under 80 seconds for low rate attacks.

For high-rate DDoS attack simulations, the CUSUM algorithm attained a 100% detection rate on some of the simulations and at the same time maintained a relatively low average false positive rate of 6.01%. For simulations with 100% detection rate, the algorithm had the lowest detection delay under 27 seconds from the onset of an attack.

#### 5.4.2 Exponentially Weighted Moving Average (EWMA) Technique

The EWMA algorithm is also a change point detection technique that assumes that the TCP SYN sequence data are independent Gaussian random variables with known

variance and known mean. In the CUSUM algorithm, the tuning parameters are the amplitude factor,  $\alpha$ , the weighting factor,  $\beta$ , and the EWMA alarm threshold,  $k$ .

The experiments setup in this study were designed to investigate the efficiency of the EWMA algorithm for detecting both low and high intensity attacks. We further investigated the effect on the algorithm efficacy of the tuning parameters.

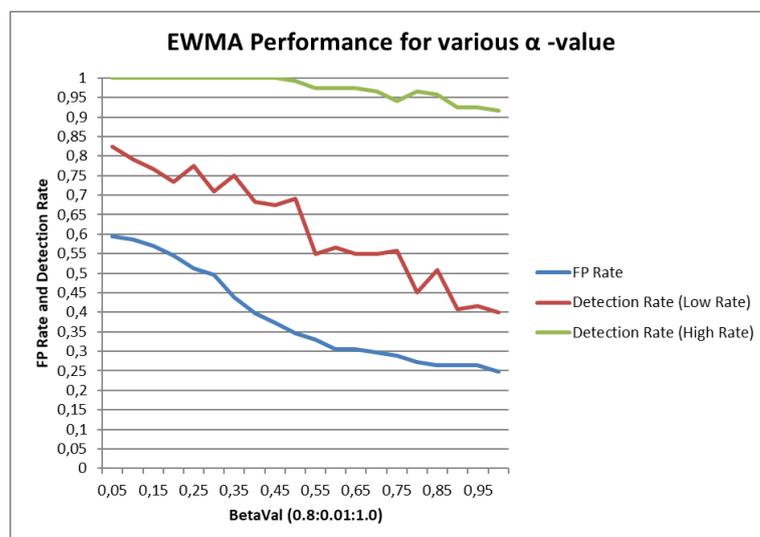
- The effect of the amplitude factor  $\alpha$ , on the detection rate.
- The effect of the weighting factor  $\beta$ , on the detection rate.
- The trade-off analysis between detection rate and the false positive rate.
- The trade-off analysis between the detection rate and detection delay.

The findings and analysis from the experiments and simulations will be elaborated in the sub-sections that follow.

#### 5.4.2.1 The effect of the Amplitude factor ( $\alpha$ )

In this section of the simulation experiment, we would like to examine the impact of the amplitude factor ( $\alpha$ ) on the algorithm's efficacy, which is its false positive rate and detection rate. The independent variable of the experiment was the amplitude factor, for which a linear increment and iteration was applied from 0.05 up to 1.0. The control variables for this experiment were the weighting factor, and it was held constant  $\beta = 0.8$ ; the EWMA alarm threshold, which was also held constant,  $k = 4$ . The dependent variable, in this case the detection rate, was then observed throughout the experiment.

Figure 25 depicts the results of the experiments. From Figure 25 it was observed that for low-rate attacks, the detection algorithm yields a detection rate of between 60%-85% and a false positive of between 30%-60% for values of  $0 < \alpha \leq 0.5$ . At the value of  $0.5 < \alpha \leq 1.0$  the detection rate deteriorates further while the false positive rate improves. Similar performance was also observed for high-rate attacks. For high-rate attacks and values of  $0 < \alpha \leq 0.5$ , the detection algorithm yields a 100% detection rate while having a false positive rate of between 30%-60%. However, for values of  $0.5 < \alpha \leq 1.0$ , the algorithm's detection rate deteriorates while the false positive rate improves. These results are indicative of the trade-off between detection rate and false positive rate with respect to the effect of the amplitude factor on the efficacy of the algorithm.

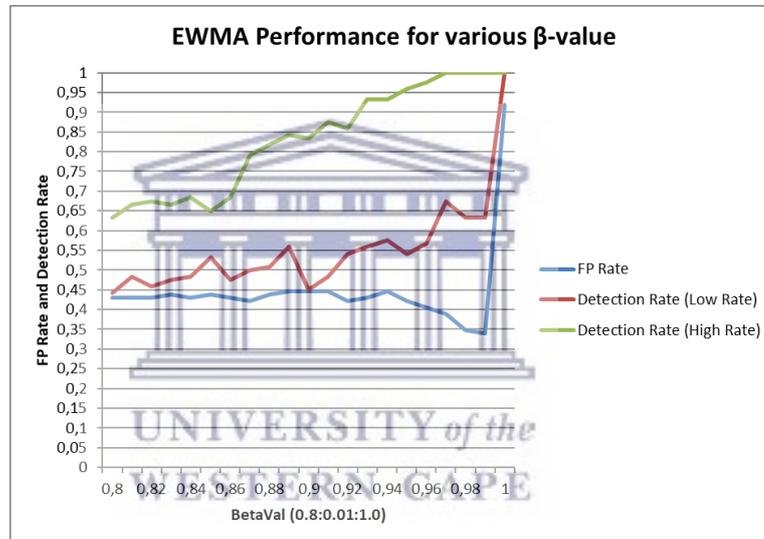


**Figure 25 Detection rate for varied alarm threshold value ( $\alpha$ ), for both Low-rate attacks and High-rate attacks**

### 5.4.2.2 The effect of the Weighting factor ( $\beta$ )

The goal of this simulation experiment is to investigate the effect of the weighting factor ( $\beta$ ) on the algorithm's detection rate. The weighting factor was the independent variable, and a linear increment from 0.80 up to 1.00, was applied. The control variables was the amplitude factor, and it was constant  $\alpha = 0.5$ ; the EWMA alarm threshold, was also constant,  $k = 4$ . The detection rate and the false positive rate performance was monitored throughout the experiment.

From Figure 26 it can be observed that the performance of the detection algorithm performs better at high values of the EWMA factor ( $\beta$ ). For low-rate attacks, the algorithm reached a 100% detection rate for  $\beta \geq 0.98$ , while the false positive rate was below 40% for values of  $\beta < 0.98$ . For high rate attacks the detection rate is higher than that of low-rate attacks. The detection rate reached 100% for values of  $\beta \geq 0.95$  while the false positive rate remained below 40%. In both cases of low and high-rate attacks, for the values of  $\beta \geq 0.99$ , the false positive rate deteriorated substantially. The improved detection rate was also accompanied by a sharp decline in false positive rate performance.



**Figure 26** Detection rate for varied EWMA factor ( $\beta$ ), for both Low-rate attacks and High-rate attacks

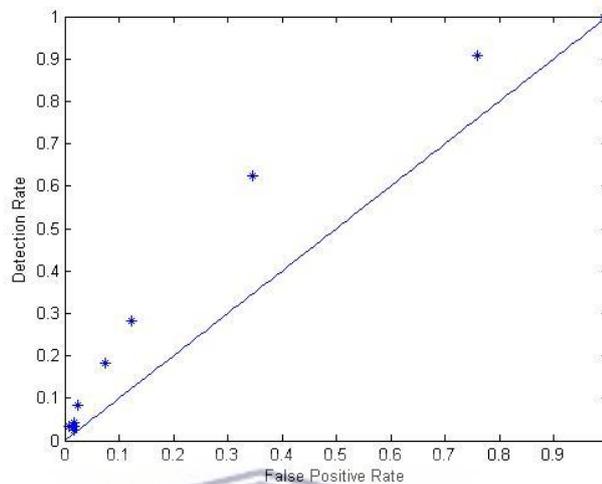
### 5.4.2.3 False Positive Rate and Detection Rate

We were investigating the trade-off between the false positive rate and the detection rate in this set of experiments. The results of the previous experiments determined that the values for the tuning parameters for this set of experiment are as follows: the alarm threshold  $\alpha = 0.5$ ; the weighting factor  $\beta = 0.98$ . The value of  $k$  was varied from 1-10.

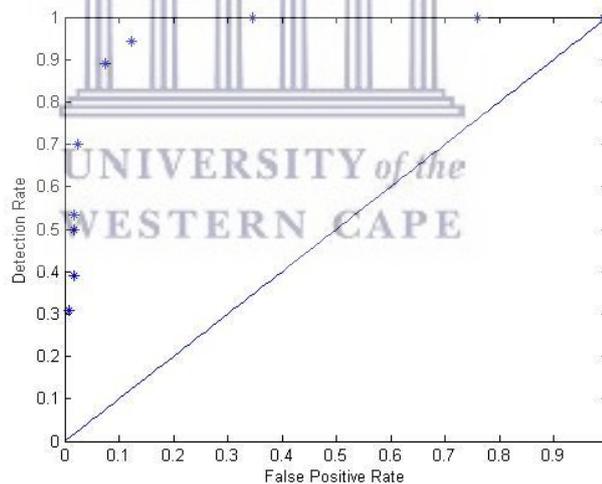
Figure 27 depicts results for the experiments simulating low-rate attacks. Each point corresponds to a varying value of  $k$ . As previously stated, good functioning points on the graph are those that are closer to the graph's upper-left corner. In the case of low-rate attacks, an increase in the algorithm's detection accuracy was accompanied by a sharp increase in the false alarm rate. Therefore higher detection accuracy will also result in a

higher false alarm rate. This is not a desirable performance in a detection method. As a result, the EWMA algorithm performed poorly in circumstances of low-rate attacks.

Figure 28 depicts the performance of the EWMA algorithm in the case of high-rate attack simulation. Most of the operating points are closer to the upper-left corner of the graph. This is also indicative that for a higher detection rate there is a slight increase in the false alarm rate. This is an improved algorithm performance when compared with the low-rate attack simulations. The average false positive rate using the set tuning parameters was 34%.



**Figure 27 ROC Curves for low-rate attacks**



**Figure 28 ROC Curves for high-rate attacks**

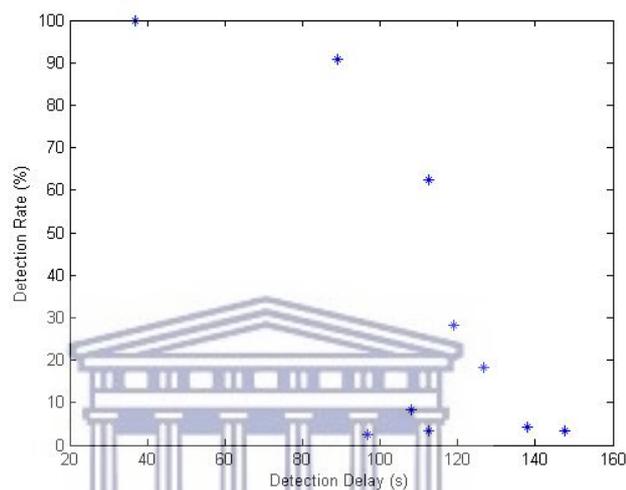
#### 5.4.2.4 Detection Rate and Detection Delay

In the next set of experiments we further analysed the trade-off between detection rate and detection delay. The results are shown in Figure 29 and Figure 30 below. Detection delay in this case is the average time taken by the algorithm to successfully detect an attack, from the onset of that attack. Each point corresponds to a pair of detection rates

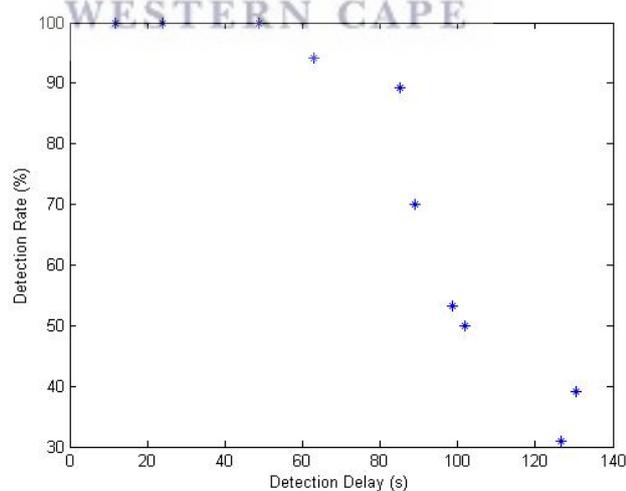
and average detection delay. The values for the tuning parameters were as follows: the alarm threshold  $\alpha = 0.5$ ; the EWMA factor  $\beta = 0.98$ . The value of  $k$  was varied from 1-10.

Figure 29 depicts the EWMA algorithm's performance trade-off between detection rate and average detection delay for low-rate attack simulations. According to the graph, as the detection rate decreases, so does the detection delay. For the low-rate attack simulation, the detection delay was just under 40 seconds (36.9s) in the experiment with 100% detection rate.

Figure 30 shows the outcomes of simulations with high-rate attacks. The method performed better in high-rate attacks. For a 100% detection rate, the average detection delays were 11.75s, 23.83s, and 48.67s for varied  $k$ -values. It can also be seen that with lower detection rate performance, the average detection latency increases for some of the alarm threshold values.



**Figure 29 Graph displaying the trade-off between Detection Rate and average Detection Delay for low-rate attacks**



**Figure 30 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks.**

#### 5.4.2.5 Summary

In the simulations for low-rate DDoS attacks, the EWMA detection algorithm's performance was also not satisfactory. DDoS attack simulations with higher accuracy rates were also accompanied by an increased false positive rate. This means that the chances that the detector algorithm will classify a random network occurrence as an attack becomes equally probable as classifying that occurrence as a normal network occurrence. This is depicted by the performance point at the top right corner of Figure 27: for a 100% detection rate, the classifier also experienced a 91.7% false positive rate. The detection delay performance for a 100% detection rate was slightly faster at 36.9 seconds from the start of the attack.

For experiments simulating high-rate DDoS attacks, the performance of the EWMA algorithm was an improvement from the experiments simulating low-rate DDoS attacks. For high-rate attack simulations, the algorithm also experienced a high false positive rate, ranging from 34.71% to 99.17%. For the simulation setup in these experiments, the algorithm was able to attain a 100% detection rate, with a 34.71% false positive rate and a detection delay of 48.67 seconds for high rate DDoS attacks.

#### 5.4.3 Summary of Statistical Change Point Detection

For an overall performance on all experiments, the CUSUM algorithm attained an 86.39% accuracy with a relatively high false positive rate, while the EWMA algorithm attained a 78.23% overall detection accuracy. For the applications in this research both algorithms may not be suitable as they produce a relatively high false positive rate and a lower accuracy rate. The overall performance for both EWMA and CUSUM algorithms for all experiments is depicted in Table 4.

Table 4 CUSUM and EWMA Overall Performance

	Accuracy (%)	False Positive Rate (%)	False Negative Rate (%)	F1-Score
CUSUM	86.392	80.149	3.126	0.92
EWMA	78.225	29.583	19.601	0.85

### 5.5 Machine Learning

The findings of our experiments are presented in this section. The models' performance was measured using seven metrics: false positive, false negative, average detection latency, F1-score, accuracy,  $R^2$ , and Root Mean Square Error (RMSE). The first five are unique to classification models, whereas the latter three (accuracy included) are unique to regression models. The number of malicious traffic misclassified as normal traffic is referred to as false positives. This is significant since it indicates how well the model

detects attacks. False negative measures the amount of normal traffic that was mistakenly labelled as an attack. The average execution time is the amount of time it takes the model to correctly classify traffic data as anomalous. Precision and recall are well-known measures in literature, however they are not evaluated in this study because they both assess how accurate a model is. Though this is significant, the inaccuracy of a model is more critical to us since undetected attacks might have negative consequences. Accuracy is a measure of the model's classification (or regression) performance, which is the number of times the model successfully detected traffic.  $R^2$  is used to compare a model's performance to a baseline, whereas RSME is the square root of the mean squared difference between predicted and actual values.

### 5.5.1 Supervised Learning

The results of the supervised learning models are summarized in Table 5. Logistic Regression (LGR) has the highest accuracy of 99.192% and the highest number of false positives in the table. In essence, despite the great accuracy, deploying LGR would allow more DDoS attacks to go undetected when compared to the other model. The LGR model, on the other hand, performed well in terms of false negative since no real traffic was misclassified.

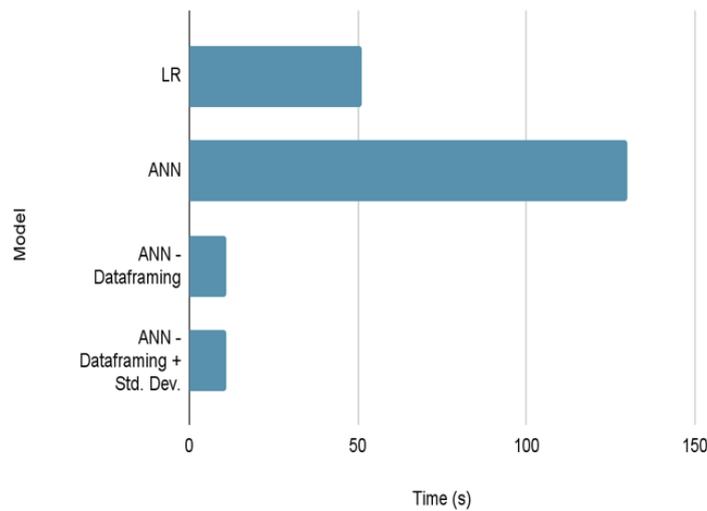
**Table 5 Summary of Results for Supervised Learning Models**

	Accuracy (%)	False Positives Rate (%)	False Negatives Rate (%)	F1-Score
LGR	99.192	1.622	0	0.999
ANN	99.414	0.670	0	0.999
ANN - Data framing	98.842	0.130	2.181	0.999
ANN - Data framing + Standard Deviation	99.405	0.297	0.957	0.999

The ANN model variation without data framing yielded the highest accuracy (99.414%) but the highest false positive rate of the three ANN models investigated. This means that, similar to LGR, some harmful attacks will go undetected; but, not as many as LGR. When compared to the other two variations, the one with standard deviation had the highest accuracy (99.405%) and the fewest false positives. However, it misclassified nearly 1% of all normal traffic as an attack. The variation without the standard deviation appears to be a combination of results, since it decreased the number of false positives while nearly doubling the false negative rate (2.181%). This resulted in an overall accuracy of just 98.842%.

In terms of execution time, Figure 31 reveals that the pure ANN model was the slowest of the four models, requiring more than 2 minutes to classify traffic flow. This would be undesirable in real-time applications requiring rapid data analysis and classification. ANN variants based on data framing, on the other hand, were substantially quicker than

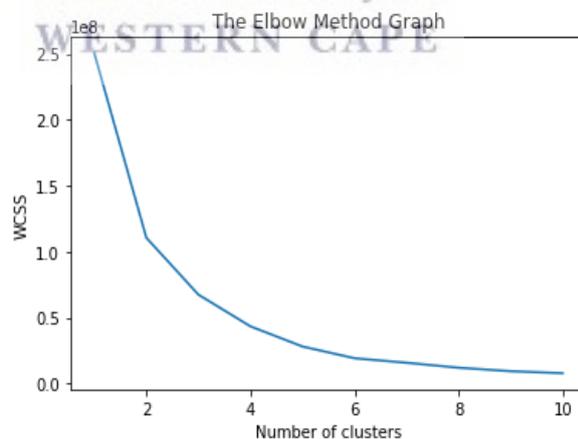
both LGR and pure ANN, taking only 11 seconds in contrast to 51 seconds and 130 seconds, respectively. This demonstrates that segmenting traffic into data frames or "windows" and processing them correspondingly may significantly reduce processing time.



**Figure 31 Average Detection Delay**

### 5.5.2 Unsupervised Learning

Though we knew the number of clusters to expect in the dataset a priori, we still ran the Elbow method [29] to verify this. Fig. 7 shows the result of the Elbow method, with k being re-confirmed to be 2. Running the K-Means classifier with K=2, resulted in an accuracy of 96.76%, with zero false positives.



**Figure 32 K-Means Elbow Method**

### 5.5.3 Semi-Supervised Learning

The outcome from the labelled output of K-Means was fed into the supervised learning models, resulting in a semi-supervised model. The performance of this hybrid combination is shown in Table 6.

**Table 6 Summary of Results for Semi-Supervised Learning Models**

	Accuracy (%)	False Positives Rate (%)	False Negatives Rate (%)	F1-Score
K-Means + LGR	100	0	0	1.0
K-Means + ANN	100	0	0	1.0
K-Means + ANN + Data framing	99.64	0	0.73	0.999
K-Means + ANN - Data framing + Standard Deviation	99.69	0	0.64	0.999

According to the table, using the K-Means classifier resulted in a considerable improvement in the performance of all models. Both LGR and conventional ANN produced flawless accuracies with no false positives or false negatives. Similarly, the accuracies of both ANN with data framing variations rose from 98.842% to 99.64% and 99.41% to 99.69%, respectively.

It is worth noting that there are no false positives in any of the four models. This is noteworthy since it suggests that semi-supervised models can properly detect all malicious attacks. Some normal internet flows, on the other hand, were nevertheless labelled as malicious. Possible factors include the dataset not being split into an equal number of data frames, thus certain data frames (particularly those at the tail end of the traffic flow) included less data than others, i.e. less than 12 data points.

### 5.5.4 Prediction

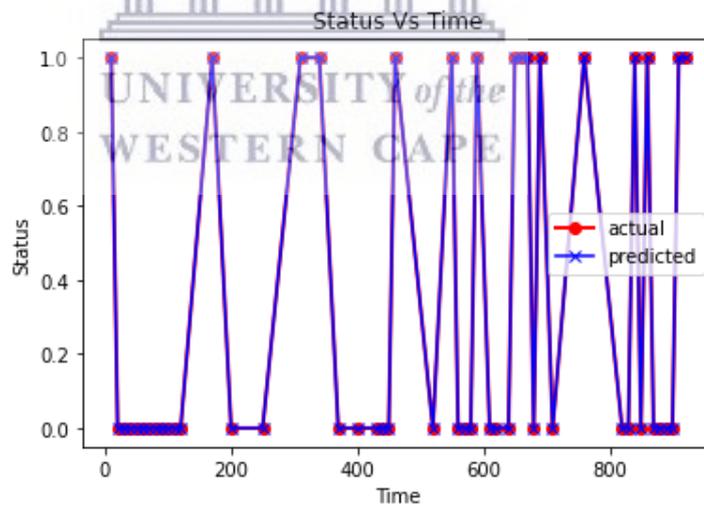
Table 7 summarizes the outcomes of three prediction models that were evaluated. LGR performed the best of the three models investigated, with a prediction accuracy of 98.6%. It was closely followed by KRR, which obtained around 98% accuracy. At 94.64%, SVR was the least accurate of the three.  $R^2$  values closer to 1 are preferable, because they represent the "closeness" of predicted values to actual values. The same pattern can be seen with  $R^2$  scores for the three models, with LGR leading with a score of around 0.94, followed by KRR at 0.91. SVR had a score of 0.76, indicating that its prediction curve differed significantly from the actual curve. Finally, RMSE values near zero are preferable

since they show lower prediction errors. LGR was the least error prone once again, with the lowest RMSE values of 0.1183, followed by KRR with a score of 0.1439. With an RMSE of 0.2314, both models were less error prone than SVR. As a result, we may infer that LGR is the strongest predictor, with KRR a close second. With such a large RMSE, SVR is a less-than-ideal predictor in our applications.

**Table 7 Summary of Results for the Prediction Models**

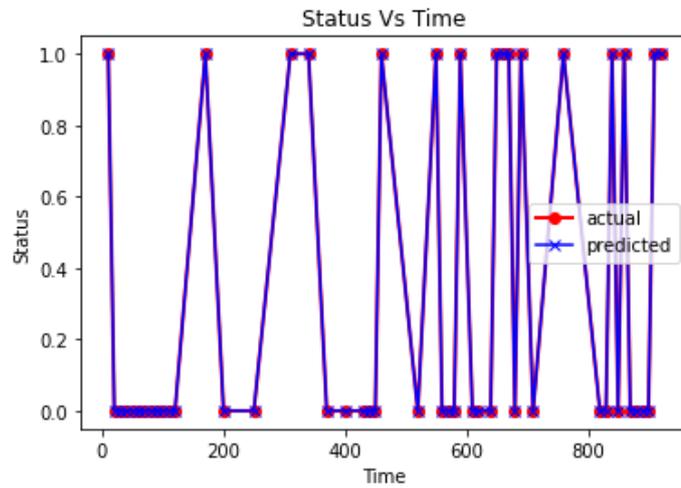
	KRR	LGR	SVR
Accuracy	97.93%	98.60%	94.64%
R <sup>2</sup>	0.9054	0.9361	0.7555
RMSE	0.1439	0.1183	0.2314

To show these findings, we created graphs comparing actual values to those predicted by the prediction models (KRR, LGR and SVR). We captured images of each model indicating its predictions for the upcoming 15 minutes (900 seconds). Furthermore, to demonstrate the scalability of the predictive models, we gathered graphical representations of predictions for the upcoming 2 to 4 hours. These graphical representations are illustrated in Figure 33 to Figure 38, where the blue lines represent predicted values and the red lines reflect actual values.

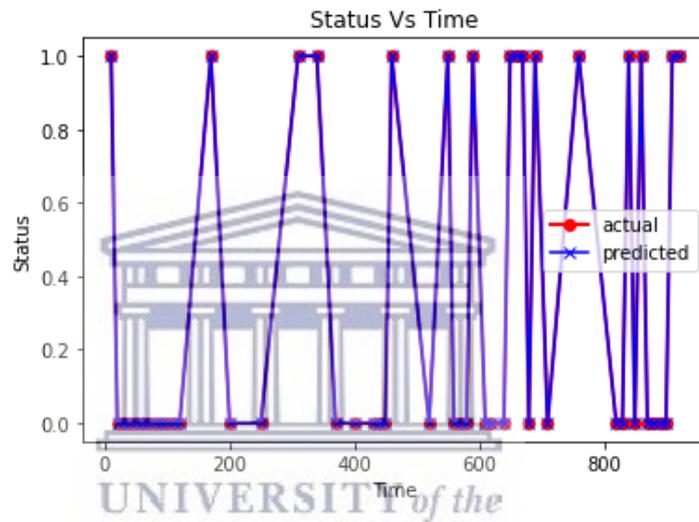


**Figure 33 KRR predictions for upcoming 15 minutes**

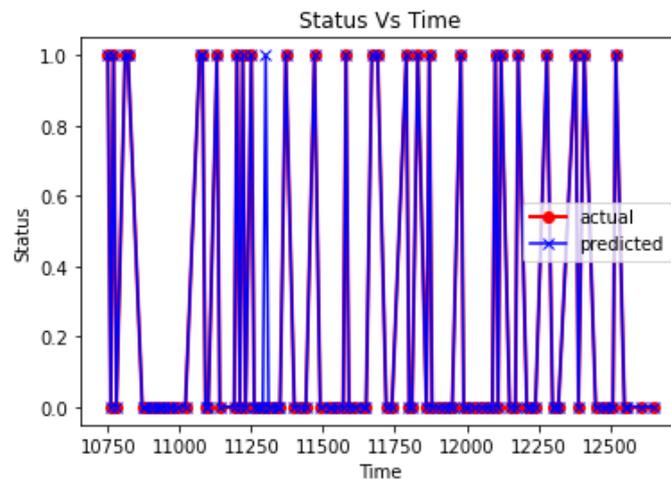
Figure 33 to Figure 35 show status vs time charts for KRR, LGR, and SVR, respectively. Status values are binary and may only be 0 or 1, with 0 indicating that no attack is expected and 1 indicating that an attack may occur. For the 15-minute time window, matching graphs were observed for all three models.



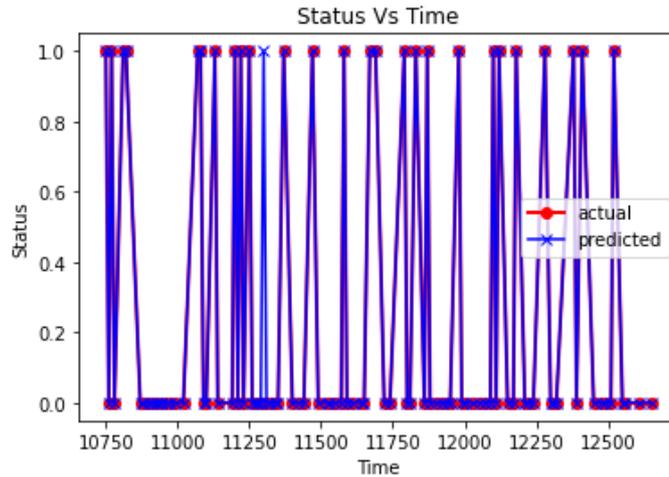
**Figure 34 LGR predictions for upcoming 15 minutes**



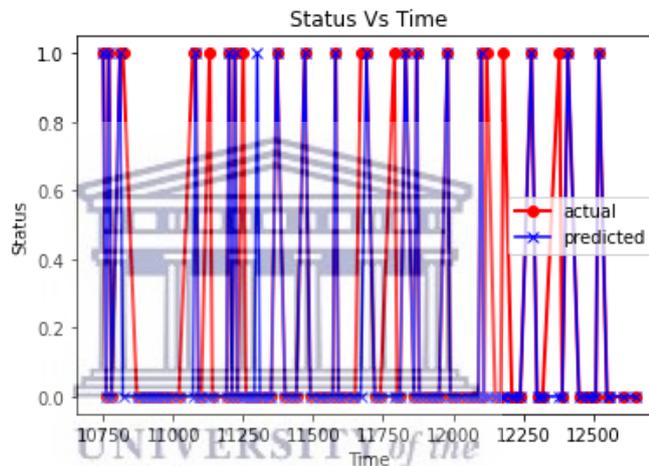
**Figure 35 SVR predictions for upcoming 15 minutes**



**Figure 36 KRR predictions for upcoming 3 to 4 hours**



**Figure 37 LGR predictions for upcoming 3 to 4 hours**



**Figure 38 SVR predictions for upcoming 3 to 4 hours**

Figure 36 to Figure 38 depict the predictions of the three models over the next 3–4 hours. KRR and LGR exhibited graphs that were identical, with both models incorrectly forecasting the appearance of an attack around the 11,260th second point. The outcomes of SVR's predictions for the same time are presented in Figure 38, with SVR predicting incorrectly on ten separate instances. Surprisingly, SVR, like KRR and LGR, incorrectly forecasted an impending attack at the same 11,260th second mark.

Overall, these findings suggest that for our use case, LGR and KRR are stronger prediction models than SVR. With attack prediction accuracies of around 98 % for both the LGR and KRR models, regression models can be utilized to forecast potential DDoS attacks. Both LGR and KRR made erroneous forecasts in cases when they anticipated attacks would occur when none did. These incorrect predictions or false alarms, while resulting in the premature deployment of protective measures, are better than the alternative. In the other instance, as shown with SVR, the model provides a false sense of security by

forecasting that no attack would occur while there are several potential threats. As a result, we see KRR and LGR's incorrect forecasts as "erring on the side of caution".

### 5.5.5 Summary

Five machine learning algorithms were investigated in this section of the study for modelling DDoS attacks in TCP/IP networks. Logistic Regression (LGR), Artificial Neural Networks (ANN), K-Means, Kernel Ridge Regression (KRR), and Support Vector Regression (SVR) were deployed. Two supervised ML classifiers, LGR and ANN, were applied to differentiate between normal and attack traffic. LGR had a classification accuracy of 99.19%, a false positive rate of 1.62%, and an average detection delay of 51 seconds after the start of an attack.

The ANN model, on the other hand, exhibited higher accuracy (99.41%) and a lower false positive rate (0.67%), although it was relatively inefficient (130 seconds). We also investigated the K-Means unsupervised ML model, which had a classification accuracy of 96.76%. Finally, we created semi-supervised ML models by incorporating K-Means, ANN, and LGR. These combinations had a classification (detection) accuracy of 100 % with zero false positives.

We also investigated the use of regression models to assist network managers in transitioning from a reactive to a proactive approach to network management. The ability of Logistic, Kernel Ridge, and Support Vector Regression models (LGR, KRR, and SVR) to properly predict an attack before it occurs was explored. LGR had the highest prediction accuracy at 98.6 %, followed by KRR at 97.9 %, and SVR had the lowest at 94.64 %. The  $R^2$  values for the LGR and KRR were 0.94 and 0.91, respectively, indicating closeness to real values, while their RMSE values were 0.12 and 0.14, respectively. For these criteria, SVR was far off the mark. In essence, LGR and KRR are both capable of predicting impending dangers, with LGR marginally outperforming KRR.

## 5.6 1D Convolutional Neural Network Technique

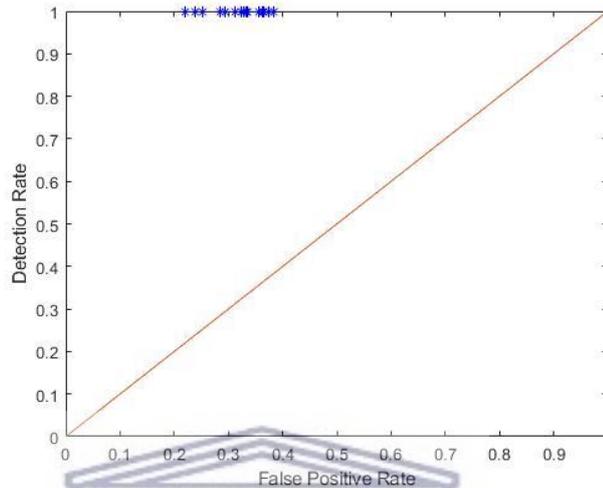
The deep learning architecture adopted for this study is a variation of the CNN that is designed for sequence modelling problems and it is called TCN. This architecture utilizes casual convolutions and dilated convolutions to be adaptive for sequential data. As described in 4.7, for the purposes of this study, the TCN architecture was designed with four residual blocks of 1D convolutional layers and a fully connected convolutional layer. A residual block contained three input channels and each dilated causal convolution layer comprised 175 size 3 filters, and a dropout factor of 0.05. The training parameters of the study were set up such that each custom loop would train for 30 epochs with a one mini-batch size. The initial learning rate was set to 0.001 and the learning rate drop-factor was set to 0.01 for every 12 epochs, and the gradient threshold was set to one.

In order to improve the accuracy, the experiments further incorporated the sliding window method that was described in 4.4.2. The sliding window method was expanded to use lagged features. Lag feature is a classical time series technique that reconstructs a time series dataset given a sequence of numbers to seem more like a supervised learning problem by treating previous time steps as input features. This method is widely used in literature including authors in [145-147]. For the purposes of the 1D CNN

experiments, multiple sets of experiments were conducted using a sliding window width size from zero to fifteen.

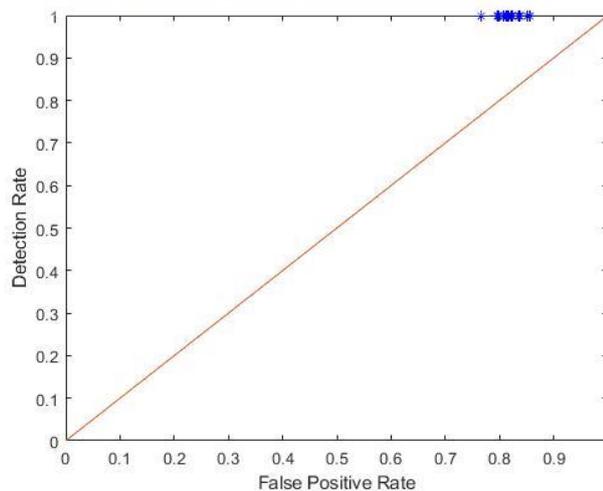
### 5.6.1 False Positive Rate and Detection Rate

In these experiments we seek to examine the algorithm's performance for detecting low-rate and high-rate DDoS attacks. We measure its performance based on the ROC curves, viz. the false positive rate and the detection rate. Figure 39 and Figure 40 indicate the ROC curves for low-rate and high-rate attacks, respectively.



**Figure 39 ROC Curves for Low Rate Attacks**

The results for the experiment that was simulating low-rate DDoS attacks are depicted by Figure 39. Each point on the graph represents a value of the sliding window width sizes from 0 to 15. The algorithm was able to obtain a 100% detection rate for some experiment and had an average false positive rate of 31.78%, while the minimum false positive rate was 22.09%.



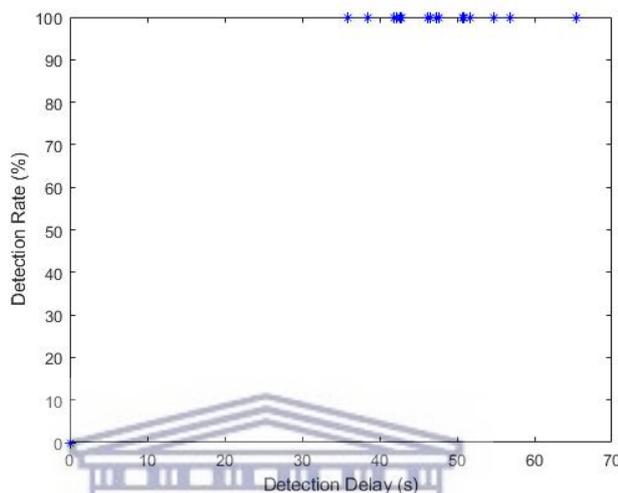
**Figure 40 ROC Curves for High Rate Attacks**

Similarly, for high-rate DDoS attacks, the simulations results are represented in Figure 40. For some sliding window width size, the algorithm was able to obtain a 100%

detection rate, however, this was also accompanied by a higher average false positive rate of 76.52%.

### 5.6.2 Detection Rate and Detection Delay

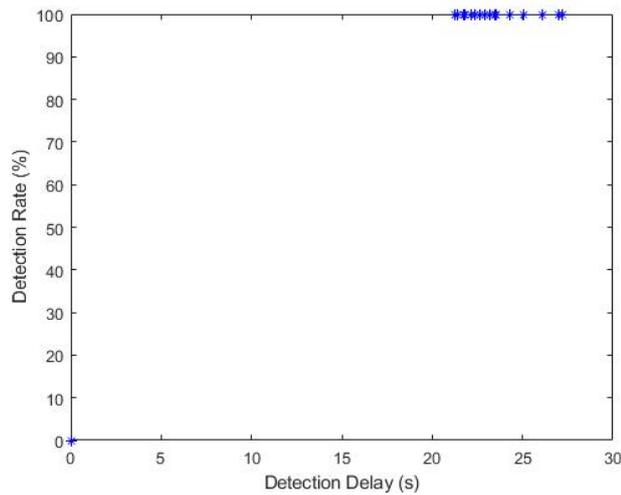
The next set of experiments seeks to examine the algorithm's detection rate and detection delay performance for both low-rate and high-rate DDoS attacks. In this section of the simulations, multiple sets of experiments were conducted using the sliding window width size from zero to fifteen. Figure 41 and Figure 42 are visual representations of the results from this set of experiments.



**Figure 41 Graph displaying the trade-off between Detection Rate and average Detection Delay for low-rate attacks.**

For low-rate DDoS attacks, the algorithm was able to obtain a 100% detection rate for some experiments. The best detection delay performance was 35.92 seconds while the maximum was 65.48 seconds. The algorithm further displayed an average detection delay performance of 47.62 seconds. This means that for a low-rate DDoS attack, on average, it takes the algorithm 47.62 seconds to detect an attack from its onset.

For high-rate DDoS attacks, the algorithm also managed to obtain a 100% detection rate performance for some experiments. The minimum detection delay performance was 21.28 seconds while the maximum detection delay was 27.23 seconds. The average detection delay performance for the algorithm was 23.51 seconds. For high-rate DDoS attacks, on average, it took the algorithm 23.51 seconds to accurately detect an attack from the onset of that attack.



**Figure 42 Graph displaying the trade-off between Detection Rate and average Detection Delay for high rate attacks.**

### 5.6.3 Summary

The CNN algorithm presented an improved performance for the experiments simulating low-rate DDoS attacks. The CNN had a 100% detection rate for some simulations, an average false positive rate of 31.78% while the minimum false positive rate was 22.09%. From the onset of a low-rate DDoS attack, the CNN was able to detect the attack, in a minimum time of 35.92 seconds and 47.62 seconds on average.

The experiments that simulate high rate DDoS attack displayed a higher average false positive rate of 76.52%. This signifies that for high-rate DDoS attacks, the CNN has a higher likelihood of classifying a normal network event as an attack. For these simulations, the CNN had an improved average detection delay of 23.51 seconds and a faster detection delay time of 21.28 seconds.

**Table 8 1-D CNN Overall Performance**

	Accuracy (%)	False Positive Rate (%)	False Negative Rate (%)	F1-Score
1-D CNN	99.662	2.253	0.326	0.998

### 5.7 Discussions

The simulation experiments conducted for these studies were aimed to evaluate the performance of the following algorithms: CUSUM, EWMA, LGR, ANN, K-means, KRR, SVR and the 1D CNN algorithms for detection of low-rate and high-rate DDoS attacks.

For low-rate DDoS attack simulation experiments, the CUSUM algorithm performance can be categorised as poor. A rise in detection accuracy causes a rise in the false positives rate, therefore this increases the likelihood that the CUSUM algorithm may flag

each low-rate data point as an attack occurrence. The poor performance is also witnessed in the lengthy average detection delay for those accurately classified attacks. The average detection delay for the simulation experiment was more than sixty seconds from the start of the attack. The performance of the CUSUM algorithm for low rate is graphically displayed in Figure 21 and Figure 23.

The CUSUM detection algorithm's performance against low-rate attacks was not satisfactory. The ROC curves revealed that the algorithm was not able to attain a 100% detection rate for the varied set of low-rate DDoS attack simulations, and the highest detection rate for the simulations was 97.52%. However, the algorithm managed to maintain an average false positive rate of 3.75%, which is relatively low, for all low-rate DDoS attack simulations. The algorithm further attained a best detection delay that was a little under 80 seconds for low rate attacks.

For high-rate DDoS attack simulations, the CUSUM algorithm attained a 100% detection rate on some of the simulations and at the same time maintained a relatively low average false positive rate of 6.01%. For simulations with 100% detection rate, the algorithm had the lowest detection delay under 27 seconds from the onset of an attack.

A similar performance was also witnessed for low-rate simulation experiments for the EWMA algorithm. The EWMA algorithm detection rate was also accompanied by an increase in false positive rates, and this performance is graphically represented by Figure 27. Therefore, for low-rate attacks, the EWMA algorithm has a high likelihood of classifying a random network occurrence as a low-rate attack. As an illustration, for values of  $k = 1$ , the EWMA was able to achieve 100% detection rate with an average detection delay of 36.9 seconds; however, the high detection accuracy and impressive detection delay was also accompanied by a 99.17% false positive rate.

In the simulations for low-rate DDoS attacks, the EWMA detection algorithm's performance was also not satisfactory. DDoS attack simulations with higher accuracy rates were also accompanied by an increased false positive rate. This means that the chances that the detector algorithm will classify a random network occurrence as an attack becomes equally classifying as a normal network occurrence. This is depicted by the performance point at the top right corner of Figure 27, for a 100% detection rate, the classifier also experienced a 91.7% false positive rate. The detection delay performance for a 100% detection rate was slightly faster at 36.9 seconds from the start of the attack.

For experiments simulating high-rate DDoS attacks, the performance of the EWMA algorithm was an improvement from the experiments simulating low-rate DDoS attacks. For high-rate attack simulations, the algorithm also experienced a high false positive rate, ranging from 34.71% to 99.17%. For the simulation setup in these experiments, the algorithm was able to attain a 100% detection rate, with a 34.71% false positive rate and a detection delay of 48.67 seconds for high-rate DDoS attacks.

For all experiments, CUSUM achieved an overall accuracy of 86.39%, however, it had a high false positive rate of 80.15%. The EWMA had a lower accuracy performance of 78.23% and a false positive rate of 29.58%. The performance of both the CUSUM and EWMA were generally not satisfactory for a DDoS application that requires detection with high accuracy and low detection latency.

Machine learning techniques for identifying DDoS attacks were introduced and enhanced performance. The LGR and ANN algorithms were investigated for their efficacy to classify both high and low-rate DDoS attacks. When tasked with distinguishing between normal and malicious traffic data, the LGR algorithm achieved 99.192% accuracy. LGR also exhibited a 1.62 % false positive rate and a detection delay of 51 seconds on average. The ANN, on the other hand, performed better, with an accuracy of 99.414 % and a false positive rate of 0.670 %. The average detection delay for the ANN was 131 seconds. The following series of tests sought to investigate whether data framing would enhance the ANN algorithm's performance. There was certainly an improvement in terms of average detection delay, with the ANN model with data framing improving to an average detection delay of 11 seconds, but there was a performance trade-off. The ANN model's accuracy was 98.842 %, with a false positive rate of 0.130 %. The following ANN experimental model enhanced accuracy to 99.405 % and the false positive rate to 0.207 % by using data framing and standard deviation in the design.

**Table 9 Summary of All Algorithm Performance**

Algorithm	Accuracy (%)	False Positive Rate (%)	False Negative Rate (%)	F1-Score
CUSUM	86.392	80.149	3.126	0.92
EWMA	78.225	29.583	19.601	0.85
LGR	99.192	1.622	0	0.999
ANN	99.414	0.670	0	0.999
ANN - Data framing	98.842	0.130	2.181	0.999
ANN - Data framing + Standard Deviation	99.405	0.297	0.957	0.999
K-Means + LGR	100	0	0	1.0
K-Means + ANN	100	0	0	1.0
K-Means + ANN + Data framing	99.64	0	0.73	0.999
K-Means + ANN - Data framing + Standard Deviation	99.69	0	0.64	0.999
1-D CNN	99.662	2.253	0.326	0.998

Further experiments were carried out to evaluate the feasibility of building a hybrid model composed of supervised and unsupervised machine learning models. The K-Means model was used to distinguish between attack traffic and normal network data, and it obtained a 99.76 % accuracy rate. The K-Means classifications were further applied to the same LGR model, ANN model and the variations. The findings suggest that using the unsupervised K-means method improved both the LGR and the ANN model's performance. The detection accuracy of the ANN models with variations increased from 98.842 % to 99.64 % and 99.41 % to 99.69 %.

It is also crucial to make the transition from reactive to proactive network management. As a result, we went a step further and assessed the effectiveness of machine learning algorithms in predicting the occurrence of a network assault before it occurred. The challenge was determined to be better suited for the KRR, LGR, and SVR algorithms. The LGR had a greater accuracy rate of 98.60 %, a higher  $R^2$  of 0.9361, and a more favourable lower RMSE of 0.1183 than the other two models. The summary of results is presented in Table 7. Though the model included some false alarms that may result in defence measures being prematurely deployed when there was no assault, it would be a safer proactive network management technique than being reactive to DDoS attacks, which might bring down the entire network system.

The CNN algorithm presented an improved performance for the experiments simulating low-rate DDoS attacks. The CNN had a 100% detection rate for all simulations, an average false positive rate of 31.78% while the minimum false positive rate was 22.09%. From the onset of a low-rate DDoS attack, the CNN was able to detect the attack, in a minimum time of 35.92 seconds and 47.62 seconds on average.

The experiments that simulate high-rate DDoS attacks displayed a higher average false positive rate of 76.52%. This signifies that for high-rate DDoS attacks, the CNN has a higher likelihood to classify a normal network event as an attack. For these simulations, the CNN had an improved average detection delay of 23.51 seconds and a faster detection delay time of 21.28 seconds.

## 5.8 Summary

In this chapter of the thesis, we analysed the performance of CUSUM, EWMA and CNN algorithms in an experiment that simulates a set of high-rate and low-rate DDoS flooding attacks on a network information system. The details of the experiment setup are discussed in detail in Chapter 4: The performance of the CUSUM, EWMA, LGR, K-Means, KRR, ANN, SVR and the 1D CNN was analysed in terms of accuracy and efficiency. It was found that the CUSUM and EWMA algorithm performance is not satisfactory for low-rate DDoS attacks, whereas these algorithms performed well for high-rate DDoS attacks. The opposite performance was observed for the CNN algorithm. The CNN performed well for low-rate DDoS attacks simulations, while a lower performance was displayed for the high-rate DDoS attacks simulation.

The chapter that follows will summarise the experiment simulations and highlight the key findings for the research.

## Chapter 6: Conclusion

---

This final chapter will present a concluding and overall view of the work presented in this thesis. The chapter will present a summary of contributions to the body of knowledge and recommendations for future work.

### 6.1 Summary of Contributions

The main research question of this thesis as described in *chapter 1* was “**How can we detect low-rate and high-rate DDoS attacks with accuracy and with minimal detection delay?**” (RQ1). In order to thoroughly address the main question, a sequence of four extra questions were also raised.

To answer this issue, we performed a background investigation into how attackers leverage susceptible machines, including IoT applications, to launch a DDoS attack. The mechanism of execution for DDoS attacks was investigated, as well as common defence mechanisms deployed at various locations throughout the network and the sorts of defence measures employed. This is detailed in the thesis's Chapter 2. A literature survey of the types of detection algorithms designed to detect DDoS attacks was also undertaken in the research thesis. We identified three generally used algorithms in the class of statistical change point approaches, machine learning, and deep learning techniques in Chapter 3 of the thesis, which were employed for simulations in this thesis.

To expand on RQ1, we pose additional sub-research questions RQ2-RQ5. These questions were addressed by the various sections of the study, and they are summarized in descending order in the next section.

For RQ5, we asked “**How do we model the characteristics of DDoS attacks so that we can simulate and generate practical attack traffic datasets?**”. To answer this question, we adopted the work of the authors in [138, 164-167] and created a sliding time window framework that accepts incoming IP traffic packets and divides them into specified attack periods. We calculated packet intensity and packet statistics adaptively in each attack window. Adaptive attacks were created in each sliding time window depending on the accompanying sliding window's determined statistics. The intricacies of modelling a TCP SYN attack were detailed in further depth in section 4.4.1 of the study, while section 4.4.2 of the study revealed how synthetic DDoS attacks were constructed for our study. Figure 14 in Chapter 4 of the thesis illustrates this, and this modelling framework was extended for the CUSUM and EWMA algorithms.

There were several modifications of the modelling methodologies depending on the algorithm design and parameters for the other algorithms employed in this study. For example, the machine learning models used in this work included supervised, semi-supervised, unsupervised, and prediction models. Figures 15-18 show the adaption of this modelling. Each is distinct in their technique of modelling attributes.

For RQ4, we asked “**What are the key statistical features of a DDoS attack?**”; and for RQ3, we asked “**How can the onset of a DDoS attack be identified on the basis of simple features of the source IP address?**”. RQ4 and RQ3 are inextricably related with RQ5 in the sense that, in order to simulate DDoS assaults, one must first understand their statistical features. To answer these questions, we looked into the relevant literature,

and the authors' work in [92, 160-162] established the foundations and background context for understanding statistical features of DDoS assaults in preparation for the modelling task. Sections 4.4.1-4.4.2 go into detail into the statistical features and modelling of DDoS attacks. We also discovered that other studies employed different combinations of IP packet header information, however employing additional features always adds further dimensions to modelling a solution, as outlined in section 4.3 of the paper. In our case, we employed an adaptive sliding window to determine the initiation of an attack by calculating IP packet intensity and packet characteristics. When modelling an attack, the use of this method minimizes processing overheads and the curse of dimensionality.

For **RQ2**, we asked “*How does the performance of statistics-based techniques, machine learning techniques compare to modern deep learning convolutional neural network techniques when detecting low-rate and high-rate DDoS attacks*”. To answer this question, we ran simulations of each technique and compared their performance. The design model for statistics techniques was described in depth in section 4.5, and the findings were presented in section 5.4. The design framework for machine learning techniques is explained in part 4.6, and the simulation results are detailed in section 5.5. Similarly, for deep learning techniques, we employed the 1D CNN algorithm, which was detailed in section 4.7, and the simulation results were explained in section 5.6. Section 5.7 compares the performance of these strategies in an attempt to address **RQ2**.

In summary, the work conducted in this study for **RQ2-RQ5** was designed to answer the core research question, **RQ1**. We created a DDoS simulation framework that generates synthetic DDoS attacks in order to simulate low and high-rate DDoS attacks. To detect the onset of an attack, the system employed incoming IP address statistic entropy characteristics, as well as statistic-based change point detection approaches, machine learning, and deep learning techniques. The findings indicate that machine learning approaches are well suited for DDoS detection. The use of ANN in conjunction with a sliding window and data framing resulted in a more accurate DDoS detection performance.



A significant part of the work presented in **Chapter 2:** and **Chapter 3:** of this thesis was published in the following article and book chapter:

- Machaka, P., & Nelwamondo, F. (2016). Data Mining Techniques for Distributed Denial of Service Attacks Detection in the Internet of Things: A Research Survey. In O. Isafiade, & A. Bagula (Ed.), Data Mining Trends and Applications in Criminal Science and Investigations (pp. 275-334). IGI Global. <http://doi:10.4018/978-1-5225-0463-4.ch010>

A significant part of the work presented in **Chapter 4:** was further published in the following research article and book chapter:

- Machaka P., Bagula A. (2021) Statistical Properties and Modelling of DDoS Attacks. In: Vinh P.C., Rakib A. (eds) Context-Aware Systems and Applications, and Nature of Computation and Communication. ICCASA 2020, ICTCC 2020. Lecture Notes of the Institute for Computer Sciences, Social Informatics and

- P. Machaka, A. Bagula and F. Nelwamondo, "Using Exponentially Weighted moving Average algorithms to Defend against DDoS attacks," 2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2016, pp. 1-6, doi: 10.1109/RoboMech.2016.7813157
- Machaka P., McDonald A., Nelwamondo F., Bagula A. (2016) Using the Cumulative Sum Algorithm Against Distributed Denial of Service Attacks in Internet of Things. In: Vinh P., Alagar V. (eds) Context-Aware Systems and Applications. ICCASA 2015. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 165. Springer, Cham. [https://doi.org/10.1007/978-3-319-29236-6\\_7](https://doi.org/10.1007/978-3-319-29236-6_7)
- Machaka, P., Ajayi, O., Maluleke, H., Kahenga, F., Bagula, A. and Kyamakya, K., 2021. Modelling DDoS Attacks in IoT Networks using Machine Learning. arXiv e-prints, pp.arXiv-2112 (submitted for review).

## 6.2 Future Work

The work presented in this research was a step in the right direction and demonstrated advancement in the field of DDoS attack detection; nevertheless, it was faced with challenges that will be addressed in the future.

- The study has mostly focused on identifying TCP SYN DDoS attacks. Nevertheless, future work should incorporate other forms of attacks, such as the HTTP flooding attacks, UDP flooding attacks, and ICMP flooding attacks discussed in section 2.4.6. This would facilitate the development of an efficient and heterogeneous testbed for DDoS attacks and simulations.
- Some security scientific institutions have released network intrusion datasets to aid in the evaluation of intrusion detection algorithms for both known and unknown threats. They have aided researchers in developing their own testbed for collecting, pre-processing, extracting various sorts of characteristics, and building an unbiased dataset. Nevertheless, the quality of the dataset against which the system is compared might limit the benchmark datasets. It is incredibly difficult to create an unbiased, realistic, and complete dataset. More specifically, benchmark datasets include a combination of IPv4 and IPv6 IP addresses, as well as network datasets that contain IoT devices. For future work in the creation of DDoS attack detection, this approach will assure resilience, scalability, and high performance.
- Reducing false positives: The work presented revealed that the detection algorithms under investigation suffer from a high false positive rate. In an ideal world a detection algorithm should avoid a high false positive rate. In reality, it is very hard for a detection algorithm to completely avoid false positives, however, it should aim to keep false positives at a very low rate. This was a major challenge for the detection algorithms in this investigation. Therefore, it is very

important that for the future of this research we investigate and design detection algorithms that can reduce false positive rates to a very low rate.

- Deep Learning for DDoS Detection: The research work that has been developed so far on deep learning has mostly focused on applications for processing images, video, text and sound. The research work has shown great potential for deep learning and its applications. However, very little research is available on intrusion detection, and more so, very little on DDoS attack detection techniques. The future direction would be to investigate and design accurate and efficient detection techniques that take advantage of the deep learning algorithms.
- Computational overheads: from the simulations we learned that building a model that can then be used for detection requires a lot of computational resources and it is time consuming. However, once the model is built, it can be implemented on an already existing system. For future research, it would help the researcher to build a model that can be improved upon using transfer learning for deep learning models.



## References

---

1. L. Garber, "Denial-of-service attacks rip the Internet," *Computer*, 4 2000, pp. 12-17.
2. S.T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *Communications Surveys & Tutorials*, IEEE, Vol. 15, 4 2013, pp. 2046-2069.
3. N.Z. Bawany, J.A. Shamsi, and K. Salah, "DDoS attack detection and mitigation using SDN: methods, practices, and solutions," *Arabian Journal for Science and Engineering*, Vol. 42, 2 2017, pp. 425-441.
4. D.K. Bhattacharyya and J.K. Kalita, *Network Anomaly Detection: A Machine Learning Perspective*, CRC Press, 2013.
5. K.R. Choo, "The cyber threat landscape: Challenges and future research directions," *Computers & Security*, Vol. 30, 8 2011, pp. 719-731.
6. K. Ashton, "That 'internet of things' thing," *RFID Journal*, Vol. 22, 7 2009, pp. 97-114.
7. I. Strategy and P. Unit, "ITU Internet Reports 2005: The internet of things," Geneva: International Telecommunication Union (ITU) 2005.
8. O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, I.S. Jubert, M. Mazura, M. Harrison, and M. Eisenhauer, "Internet of things strategic research roadmap," O.Vermesan, P.Friess, P.Guillemin, S.Gusmeroli, H.Sundmaeker, A.Bassi, et al., *Internet of Things: Global Technological and Societal Trends*, Vol. 1 2011, pp. 9-52.
9. D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, Vol. 1 2011, pp. 14.
10. H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, "Vision and challenges for realising the Internet of Things," 2010.
11. A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, "A survey on facilities for experimental internet of things research," *Communications Magazine*, IEEE, Vol. 49, 11 2011, pp. 58-67.
12. J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, Vol. 29, 7 2013, pp. 1645-1660.
13. D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks*, Vol. 10, 7 2012, pp. 1497-1516.

14. C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context Aware Computing for The Internet of Things: A Survey," *Communications Surveys & Tutorials*, IEEE, Vol. 16, 1 2014, pp. 414-454.
15. A.Z. Alkar and U. Buhur, "An Internet based wireless home automation system for multifunctional devices," *Consumer Electronics, IEEE Transactions on*, Vol. 51, 4 2005, pp. 1169-1174.
16. M.M. Baig, H. Gholamhosseini, and M.J. Connolly, "A comprehensive survey of wearable and wireless ECG monitoring systems for older adults," *Medical & biological engineering & computing*, Vol. 51, 5 2013, pp. 485-495.
17. B. Latré, B. Braem, I. Moerman, C. Blondia, and P. Demeester, "A survey on wireless body area networks," *Wireless Networks*, Vol. 17, 1 2011, pp. 1-18.
18. M. Yun and B. Yuxin, "Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid," 2010, pp. 69-72.
19. Z. Liqiang, Y. Shouyi, L. Leibo, Z. Zhen, and W. Shaojun., "A Crop Monitoring System Based on Wireless Sensor Network," *Procedia Environmental Sciences*, Vol. 11, Part B 2011, pp. 558-565.
20. F.J. Oppermann, C.A. Boano, and K. Römer, "A decade of wireless sensing applications: Survey and taxonomy," *The Art of Wireless Sensor Networks*, Springer, 2014, pp. 11-50.
21. M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," 1996, pp. 164-173.
22. Forrester Consulting, "The trends and Changing Landscape of DDoS Threats and Protection," 2009.
23. C. Douligeris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: classification and state-of-the-art," *Computer Networks*, Vol. 44, 5 2004, pp. 643-666.
24. C.V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Computers & Security*, Vol. 29, 1 2010, pp. 124-140.
25. I. Prolexic Technologies, "Prolexic Quarterly Global DDoS Attack Report (Q4 2012)," Vol. 2014, October 2012.
26. J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, Vol. 34, 2 2004, pp. 39-53.
27. P. Liu, W. Zang, and M. Yu, "Incentive-based modeling and inference of attacker intent, objectives, and strategies," *ACM Transactions on Information and System Security (TISSEC)*, Vol. 8, 1 2005, pp. 78-118.

28. V. Anand, "Intrusion Detection: Tools, Techniques and Strategies," 2014, pp. 69-73.
29. K.V. Kumar, "Distributed Denial of Service (DDoS) Attack, Networks, Tools and DEFENSE." International Journal of Applied Engineering Research, Vol. 10, 8 2015.
30. A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," Managing Cyber Threats, Springer, 2005, pp. 19-78.
31. B. Nagpal, P. Sharma, N. Chauhan, and A. Panesar, "DDoS tools: Classification, analysis and comparison," 2015, pp. 342-346.
32. R.C. Prathibha, "A Comparative Study of Defense Mechanisms against SYN Flooding Attack," International Journal of Computer Applications, Vol. 98, 18 2014.
33. T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of network-based defense mechanisms countering the DoS and DDoS problems," ACM Computing Surveys (CSUR), Vol. 39, 1 2007, pp. 3.
34. M.H. Bhuyan, H.J. Kashyap, D.K. Bhattacharyya, and J.K. Kalita, "Detecting distributed denial of service attacks: methods, tools and future directions," The Computer Journal 2013, pp. bxt031.
35. O. Kupreev, E. Badovskaya, and A. Gutnikov, "Kaspersky Report: DDoS attacks in Q2 2020," Vol. 2020, August 10 2020.
36. J. Mirkovic and P. Reiher, "D-WARD: a source-end defense against flooding denial-of-service attacks," Dependable and Secure Computing, IEEE Transactions on, Vol. 2, 3 2005, pp. 216-232.
37. E. Alomari, S. Manickam, B.B. Gupta, S. Karuppayah, and R. Alfari, "Botnet-based distributed denial of service (DDoS) attacks on web servers: classification and art," arXiv preprint arXiv:1208.0403 2012.
38. Z. Chen, Z. Chen, and A. Delis, "An inline detection and prevention framework for distributed denial of service attacks," The Computer Journal, Vol. 50, 1 2007, pp. 7-40.
39. B.B. Gupta, R.C. Joshi, and M. Misra, "Defending against distributed denial of service attacks: issues and challenges," Information Security Journal: A Global Perspective, Vol. 18, 5 2009, pp. 224-247.
40. A. Mishra, B.B. Gupta, and R.C. Joshi, "A comparative study of distributed denial of service attacks, intrusion tolerance and mitigation techniques," 2011, pp. 286-289.
41. I.K. Somal and S. Virk, "Classification of Distributed Denial of Service Attacks—Architecture, Taxonomy and Tools," 2014.
42. P.J. Criscuolo, "No title," Distributed Denial of Service: Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht CIAC-2319 2000.

43. D. Dittrich, "The DoS Project's 'trinoo' distributed denial of service attack tool," 1999.
44. D. Dittrich, "The 'Tribe Flood Network' distributed denial of service attack tool," University of Washington, Vol. 10 1999.
45. J. Barlow and W. Thrower, "TFN2K—an analysis," Axent Security Team 2000.
46. D. Dittrich, G. Weaver, S. Dietrich, and N. Long, "The 'mstream' distributed denial of service attack tool," URL <http://staff.washington.edu/dittrich/misc/mstream.analysis.txt>, Vol. 3 2000.
47. S. Dietrich, N. Long, and D. Dittrich, "Analyzing Distributed Denial of Service Tools: The Shaft Case." 2000, pp. 329-339.
48. B. Hancock, "Trinity v3, a DDoS tool, hits the streets," Computers & Security, Vol. 19, 7 2000, pp. 574.
49. A.M. Batishchev, "'Low orbit ion cannon (loic)," This is an electronic document. Available: "<http://sourceforge.net/projects/loic/>". Date of publication: [January 29, 2012]. Date retrieved: September, Vol. 6 2012.
50. P. Ferguson, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing," 2000.
51. X. Geng and A.B. Whinston, "Defeating distributed denial of service attacks," IT Professional, Vol. 2, 4 2000, pp. 36-42.
52. C. Jin, H. Wang, and K.G. Shin, "Hop-count filtering: an effective defense against spoofed DDoS traffic," 2003, pp. 30-41.
53. K. Park and H. Lee, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets," Vol. 31, 4 2001, pp. 15-26.
54. T. Peng, C. Leckie, and K. Ramamohanarao, "Protection from distributed denial of service attacks using history-based IP filtering," Vol. 1 2003, pp. 482-486.
55. H. Wang, C. Jin, and K.G. Shin, "Defense against spoofed IP traffic using hop-count filtering," IEEE/ACM Transactions on Networking (ToN), Vol. 15, 1 2007, pp. 40-53.
56. N. Ye, S. Vilbert, and Q. Chen, "Computer intrusion detection through EWMA for autocorrelated and uncorrelated data," Reliability, IEEE Transactions on, Vol. 52, 1 2003, pp. 75-82.
57. H. Burch and B. Cheswick, "Tracing Anonymous Packets to Their Approximate Source." 2000, pp. 319-327.
58. J. Mirković, G. Prier, and P. Reiher, "Attacking DDoS at the source," 2002, pp. 312-321.

59. G. Zhang and M. Parashar, "Cooperative defence against ddos attacks," *Journal of Research and Practice in Information Technology*, Vol. 38, 1 2006, pp. 69-84.
60. G. Badishi, A. Herzberg, I. Keidar, O. Romanov, and A. Yachin, "Denial of service? leave it to Beaver," Project supported by Israeli Ministry of Science 2006, pp. 3-14.
61. A.D. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure overlay services," Vol. 32, 4 2002, pp. 61-72.
62. E. Shi, I. Stoica, D.G. Andersen, and A. Perrig, "OverDoSe: A generic DDoS protection service using an overlay network," *Computer Science Department* 2006, pp. 76.
63. H. Beitollahi and G. Deconinck, "Analyzing well-known countermeasures against distributed denial of service attacks," *Computer Communications*, Vol. 35, 11 2012, pp. 1312-1332.
64. R. Mahajan, S.M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *ACM SIGCOMM Computer Communication Review*, Vol. 32, 3 2002, pp. 62-73.
65. R. Mahajan, S.M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Aggregate-based congestion control," ICSI Center for Internet Research (ICIR) AT&T Labs—Research 2002.
66. Haining Wang, Danlu Zhang, and K.G. Shin, "Detecting SYN flooding attacks," Vol. 3 2002, pp. 1530-1539.
67. M. Hosaagrahara and H. Sethu, "Max-min fair scheduling in input-queued switches," *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 19, 4 2008, pp. 462-475.
68. D.K. Yau, J. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," *IEEE/ACM Transactions on Networking (TON)*, Vol. 13, 1 2005, pp. 29-42.
69. S.M. Bellovin, M. Leech, and T. Taylor, "ICMP traceback messages," 2003.
70. M. Sung and J. Xu, "IP traceback-based intelligent packet filtering: a novel technique for defending against Internet DDoS attacks," *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 14, 9 2003, pp. 861-872.
71. R. Stone, "CenterTrack: An IP Overlay Network for Tracking DoS Floods.," Vol. 21 2000, pp. 114.
72. M. Adler, "Trade-offs in probabilistic packet marking for IP traceback," *Journal of the ACM (JACM)*, Vol. 52, 2 2005, pp. 217-244.

73. D.X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," Vol. 2 2001, pp. 878-886.
74. D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback," ACM Transactions on Information and System Security (TISSEC), Vol. 5, 2 2002, pp. 119-137.
75. B. Al-Duwairi and M. Govindarasu, "Novel hybrid schemes employing packet marking and logging for IP traceback," Parallel and Distributed Systems, IEEE Transactions on, Vol. 17, 5 2006, pp. 403-418.
76. R. Chen, J. Park, and R. Marchany, "TRACK: A novel approach for defending against distributed denial-of-service attacks," Technical Report TR ECE—06-02, Dept. of Electrical and Computer Engineering, Virginia Tech 2006.
77. A. John and T. Sivakumar, "Ddos: Survey of traceback methods," International Journal of Recent Trends in Engineering, Vol. 1, 2 2009, pp. 241-245.
78. S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," Vol. 30, 4 2000, pp. 295-306.
79. R. Chen and J. Park, "Attack Diagnosis: Throttling distributed denial-of-service attacks close to the attack sources," 2005, pp. 275-280.
80. R. Chen, J. Park, and R. Marchany, "NISp1-05: RIM: Router Interface Marking for IP Traceback," 2006, pp. 1-5.
81. X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: Network-layer DoS defense against multimillion-node botnets," ACM SIGCOMM Computer Communication Review, Vol. 38, 4 2008, pp. 195-206.
82. A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," 2003, pp. 93-107.
83. K. Argyraki and D.R. Cheriton, "Scalable network-layer defense against internet bandwidth-flooding attacks," IEEE/ACM Transactions on Networking (TON), Vol. 17, 4 2009, pp. 1284-1297.
84. J. Mirkovic, M. Robinson, and P. Reiher, "Alliance formation for DDoS defense," 2003, pp. 11-18.
85. J.B. Cabrera, L. Lewis, X. Qin, W. Lee, and R.K. Mehra, "Proactive intrusion detection and distributed denial of service attacks—a case study in security management," Journal of Network and Systems Management, Vol. 10, 2 2002, pp. 225-254.
86. J. Park and M. Kim, "Design and implementation of an SNMP-based traffic flooding attack detection system," Challenges for Next Generation Network Operations and Service Management, Springer, 2008, pp. 380-389.

87. X. Qin, W. Lee, L. Lewis, and J.B. Cabrera, "Integrating intrusion detection and network management," 2002, pp. 329-344.
88. W.W. Streilein, D.J. Fried, and R.K. Cunningham, "Detecting flood-based denial-of-service attacks with snmp/rmon," 2003.
89. J. Yu, H. Lee, M. Kim, and D. Park, "Traffic flooding attack detection with SNMP MIB using SVM," *Computer Communications*, Vol. 31, 17 2008, pp. 4212-4219.
90. H. Liao, C. Richard Lin, Y. Lin, and K. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, Vol. 36, 1 2013, pp. 16-24.
91. A.G. Tartakovsky, B.L. Rozovskii, R.B. Blazek, and H. Kim, "A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods," *Signal Processing, IEEE Transactions on*, Vol. 54, 9 2006, pp. 3372-3382.
92. A.G. Tartakovsky, A.S. Polunchenko, and G. Sokolov, "Efficient computer network anomaly detection by changepoint detection methods," *Selected Topics in Signal Processing, IEEE Journal of*, Vol. 7, 1 2013, pp. 4-11.
93. A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM computer communication review*, Vol. 35, 4 2005, pp. 217-228.
94. C.E. Shannon, "Communication theory of secrecy systems," *Bell system technical journal*, Vol. 28, 4 1949, pp. 656-715.
95. C. Tsallis, "Possible generalization of Boltzmann-Gibbs statistics," *Journal of statistical physics*, Vol. 52, 1-2 1988, pp. 479-487.
96. A. Rényi, "On measures of entropy and information," 1961.
97. K. Li, W. Zhou, S. Yu, and B. Dai, "Effective DDoS attacks detection using generalized entropy metric," 2009, pp. 266-280.
98. Z. Jian-Qi, F. Feng, Y. Ke-xin, and L. Yan-Heng, "Dynamic entropy based DoS attack detection method," *Computers & Electrical Engineering*, Vol. 39, 7 2013, pp. 2243-2251.
99. Y. Gu, A. McCallum, and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," 2005, pp. 32.
100. X. Ma and Y. Chen, "DDoS detection method based on chaos analysis of network traffic entropy," *IEEE Communications Letters*, Vol. 18, 1 2013, pp. 114-117.
101. J. Zhang, Z. Qin, L. Ou, P. Jiang, J. Liu, and A.X. Liu, "An advanced entropy-based DDOS detection scheme," Vol. 2 2010, pp. V2-71.

102. Y. Xiang, K. Li, and W. Zhou, "Low-rate DDoS attacks detection and traceback by using new information metrics," *IEEE transactions on information forensics and security*, Vol. 6, 2 2011, pp. 426-437.
103. M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita, "E-LDAT: a lightweight system for DDoS flooding attack detection and IP traceback using extended entropy metric," *Security and Communication Networks*, Vol. 9, 16 2016, pp. 3251-3270.
104. S.M. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against SDN controllers," 2015, pp. 77-81.
105. M. Javed, A.B. Ashfaq, M.Z. Shafiq, and S.A. Khayam, "On the Inefficient Use of Entropy for Anomaly Detection." 2009, pp. 369-370.
106. P. Fiadino, A. D'Alconzo, M. Schiavone, and P. Casas, "Challenging entropy-based anomaly detection and diagnosis in cellular networks," 2015, pp. 87-88.
107. G.V. Moustakides, A.S. Polunchenko, and A.G. Tartakovsky, "A NUMERICAL APPROACH TO PERFORMANCE ANALYSIS OF QUICKEST CHANGE-POINT DETECTION PROCEDURES," *Statistica Sinica*, Vol. 21, 2 2011, pp. 571-596.
108. M. Basseville and I.V. Nikiforov, *Detection of abrupt changes: theory and application*, Prentice Hall Englewood Cliffs, 1993.
109. H.V. Poor and O. Hadjiladis, *Quickest detection*, Cambridge University Press Cambridge, 2009.
110. G. Carl, G. Kesidis, R.R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *Internet Computing, IEEE*, Vol. 10, 1 2006, pp. 82-89.
111. E.S. Page, "Continuous inspection schemes," *Biometrika* 1954, pp. 100-115.
112. V.A. Siris and F. Papagalou, "Application of anomaly detection algorithms for detecting SYN flooding attacks," *Computer Communications*, Vol. 29, 9 2006, pp. 1433-1442.
113. C. Bo, B. Fang, and X. Yun, "A new approach for early detection of internet worms based on connection degree," Vol. 4 2005, pp. 2424-2430.
114. S.W. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, Vol. 1, 3 1959, pp. 239-250.
115. N. Ye, C. Borrer, and Y. Zhang, "EWMA techniques for computer intrusion detection through anomalous changes in event intensity," *Quality and Reliability Engineering International*, Vol. 18, 6 2002, pp. 443-451.
116. G. Münz and G. Carle, "Application of forecasting techniques and control charts for traffic anomaly detection," 2008.

117. K. Singh, P. Singh, and K. Kumar, "Application layer HTTP-GET flood DDoS attacks: Research landscape and challenges," *Computers & Security*, Vol. 65 2017, pp. 344-372.
118. S. Hosseini and M. Azizi, "The hybrid technique for DDoS detection with supervised learning algorithms," *Computer Networks*, Vol. 158 2019, pp. 35-45.
119. C.A. Catania and C.G. Garino, "Automatic network intrusion detection: Current techniques and open issues," *Computers & Electrical Engineering*, Vol. 38, 5 2012, pp. 1062-1072.
120. W. Lee and S.J. Stolfo, "Data mining approaches for intrusion detection," 1998.
121. F. Rosenblatt, *The perceptron, a perceiving and recognizing automaton Project Para*, Cornell Aeronautical Laboratory, 1957.
122. I. Arel, D. C. Rose, and T. P. Karnowski, "Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier]," - *IEEE Computational Intelligence Magazine*, Vol. 5, 4 2010, pp. 13-18.
123. L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion Detection with Unlabeled Data Using Clustering," 2001.
124. V. Akilandeswari and S.M. Shalinie, "Probabilistic Neural Network based attack traffic classification," 2012, pp. 1-8.
125. C. Siaterlis and V. Maglaris, "Detecting incoming and outgoing DDoS attacks at the edge using a single set of network characteristics," 2005, pp. 469-475.
126. O. Ali and P. Cota, "Towards DoS/DDoS Attack Detection Using Artificial Neural Networks," 2018, pp. 229-234.
127. Y.N. Soe, P.I. Santosa, and R. Hartanto, "Ddos attack detection based on simple ann with smote for iot environment," 2019, pp. 1-5.
128. M. Wang, Y. Lu, and J. Qin, "A dynamic MLP-based DDoS attack detection method using feature selection and feedback," *Computers & Security*, Vol. 88 2020, pp. 101645.
129. C. Ioannou and V. Vassiliou, "Classifying security attacks in IoT networks using supervised learning," 2019, pp. 652-658.
130. P. Chaudhary and B.B. Gupta, "Ddos detection framework in resource constrained internet of things domain," 2019, pp. 675-678.
131. R.T. Kokila, S.T. Selvi, and K. Govindarajan, "DDoS detection and analysis in SDN-based environment using support vector machine classifier," 2014, pp. 205-210.
132. K. Wehbi, L. Hong, T. Al-salah, and A.A. Bhutta, "A survey on machine learning based detection on DDoS attacks for IoT systems," 2019, pp. 1-6.

133. H. Polat, O. Polat, and A. Cetin, "Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models," *Sustainability*, Vol. 12, 3 2020, pp. 1035.
134. T. Aytaç, M.A. Aydın, and A.H. Zaim, "Detection DDOS attacks using machine learning methods," 2020.
135. Ö Tonkal, H. Polat, E. Başaran, Z. Cömert, and R. Kocaoğlu, "Machine Learning Approach Equipped with Neighbourhood Component Analysis for DDoS Attack Detection in Software-Defined Networking," *Electronics*, Vol. 10, 11 2021, pp. 1227.
136. A. Churcher, R. Ullah, J. Ahmad, F. Masood, M. Gogate, F. Alqahtani, B. Nour, and W.J. Buchanan, "An experimental analysis of attack classification using machine learning in iot networks," *Sensors*, Vol. 21, 2 2021, pp. 446.
137. C. Tsai, A.Y. Chang, and H. Ming-Szu, "Early Warning System for DDoS Attacking Based on Multilayer Deployment of Time Delay Neural Network," 2010, pp. 704-707.
138. L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," Vol. 1 2003, pp. 303-314.
139. S. Theodoridis, "Neural networks and deep learning," *Machine Learning 2015*, pp. 875-936.
140. G. Öke and G. Loukas, "A denial of service detector based on maximum likelihood detection and the random neural network," *The Computer Journal*, Vol. 50, 6 2007, pp. 717-727.
141. G. Oke, G. Loukas, and E. Gelenbe, "Detecting denial of service attacks with bayesian classifiers and the random neural network," 2007, pp. 1-6.
142. A.A. Al Islam and T. Sabrina, "Detection of various denial of service and Distributed Denial of Service attacks using RNN ensemble," 2009, pp. 603-608.
143. Z.A. Baig and K. Salah, "Multi-agent pattern recognition mechanism for detecting distributed denial of service attacks," *IET information security*, Vol. 4, 4 2010, pp. 333-343.
144. A. Saied, R.E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using Artificial Neural Networks," *Neurocomputing 2015*.
145. C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, Vol. 5 2017, pp. 21954-21961.
146. X. Yuan, C. Li, and X. Li, "DeepDefense: identifying DDoS attack via deep learning," 2017, pp. 1-8.

147. R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, "LUCID: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection," *IEEE Transactions on Network and Service Management* 2020.
148. M. Roopak, G.Y. Tian, and J. Chambers, "Deep learning models for cyber security in IoT networks," 2019, pp. 452.
149. E. Ahmed, G. Mohay, A. Tickle, and S. Bhatia, "Use of ip addresses for high rate flooding attack detection," 2010, pp. 124-135.
150. P. Gogoi, M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita, "Packet and flow based network intrusion dataset," 2012, pp. 322-334.
151. T. Peng, C. Leckie, and K. Ramamohanarao, "Proactively detecting distributed denial of service attacks using source IP address monitoring," 2004, pp. 771-782.
152. J. Cheng, J. Yin, Y. Liu, Z. Cai, and M. Li, "DDoS attack detection algorithm using IP address features," *Frontiers in Algorithmics*, Springer, 2009, pp. 207-215.
153. P. Barford and D. Plonka, "Characteristics of network traffic flow anomalies," 2001, pp. 69-73.
154. H.H. Takada and A. Anzaloni, "Protecting servers against DDoS attacks with improved source IP address monitoring scheme," 2006, pp. 6 pp.-159.
155. T.M. Gil, "MULTOPS: A data structure for denial-of-service attack detection," *Frontiers in Algorithmics* 2009, pp. 207-215.
156. M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita, "Network traffic anomaly detection techniques and systems," *Network Traffic Anomaly Detection and Prevention*, Springer, 2017, pp. 115-169.
157. A. Wagner and B. Plattner, "Entropy based worm and anomaly detection in fast IP networks," 2005, pp. 172-177.
158. Q. Le, M. Zhanikeev, and Y. Tanaka, "Methods of distinguishing flash crowds from spoofed DoS attacks," 2007, pp. 167-173.
159. S. Bhatia, D. Schmidt, G. Mohay, and A. Tickle, "A framework for generating realistic traffic for Distributed Denial-of-Service attacks and Flash Events," *Computers & Security*, Vol. 40 2014, pp. 95-107.
160. M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection," *Pattern Recognition Letters*, Vol. 51 2015, pp. 1-7.
161. D. Erhan and E. Anarim, "Statistical Properties of DDoS Attacks," 2019, pp. 1238-1242.

162. S. Bai, J.Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," arXiv preprint arXiv:1803.01271 2018.
163. P. Machaka and F. Nelwamondo, "Data Mining Techniques for Distributed Denial of Service Attacks Detection in the Internet of Things: A Research Survey," Data Mining Trends and Applications in Criminal Science and Investigations, IGI Global, 2016, pp. 275-334.
164. R. Dahlhaus, "Locally stationary processes, Chapter in Time Series Analysis: Methods and Applications, Vol. 30," 2012.
165. D. Hu, P. Hong, and Y. Chen, "FADM: DDoS flooding attack detection and mitigation system in software-defined networking," 2017, pp. 1-7.
166. W.U. Qing-Tao and S. Zhi-qing, "Detecting DDOS attacks against web server using time series analysis," Wuhan University Journal of Natural Sciences, Vol. 11, 1 2006, pp. 175-180.
167. E. Ahmed, A. Clark, and G. Mohay, "A novel sliding window based change detection algorithm for asymmetric traffic," 2008, pp. 168-175.
168. L. Morissette and S. Chartier, "The k-means clustering technique: General considerations and implementation in Mathematica," Tutorials in Quantitative Methods for Psychology, Vol. 9, 1 2013, pp. 15-24.
169. H. Wang, X. Liu, J. Lai, and Y. Liang, "Network security situation awareness based on heterogeneous multi-sensor data fusion and neural network," 2007, pp. 352-359.
170. C. Meng, Y. Lv, L. You, and Y. Yue, "Intrusion Detection Method Based on Improved K-Means Algorithm," , Vol. 1302, 3 2019, pp. 032011.
171. H. Drucker, C.J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," Advances in neural information processing systems, Vol. 9 1997, pp. 155-161.
172. V. Vovk, "Kernel ridge regression," Empirical inference, Springer, 2013, pp. 105-116.
173. I.H. Witten, E. Frank, M.A. Hall, and C.J. Pal, "Extending instance-based and linear models," Data Mining: Practical Machine Learning Tools and Techniques, Elsevier, 2017.
174. A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, Vol. 25 2012, pp. 1097-1105.

175. O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," 2012, pp. 4277-4280.
176. H.I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, Vol. 33, 4 2019, pp. 917-963.
177. W. Chen, H. Zhang, X. Zhou, and Y. Weng, "Intrusion Detection for Modern DDos Attacks Classification Based on Convolutional Neural Networks," 2021, pp. 45-60.
178. E. McCullough, R. Iqbal, and A. Katangur, "Analysis of Machine Learning Techniques for Lightweight DDos Attack Detection on IoT Networks," 2020, pp. 96-110.
179. S. Kiranyaz, T. Ince, R. Hamila, and M. Gabbouj, "Convolutional neural networks for patient-specific ECG classification," 2015, pp. 2608-2611.
180. S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D.J. Inman, "1D convolutional neural networks and applications: A survey," *Mechanical Systems and Signal Processing*, Vol. 151 2021, pp. 107398.
181. F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," arXiv preprint arXiv:1511.07122 2015.

