An enhanced security algorithm for detecting Distributed Denial of Service attacks in Cloud Computing

by

Coster Baloyi

207081205

MASTER OF COMPUTING: INFORMATION TECHNOLOGY

in the

Department of Information Technology

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

TSHWANE UNIVERSITY OF TECHNOLOGY

Supervisor: Dr DP du Plessis

Co-Supervisor: Dr TE Mathonsi and Dr TM Tshilongamulenzhe

October 2022

Declaration

I, Coster Baloyi, declare that this research study entitled, “An enhanced security algorithm for detecting Distributed Denial of Service attacks in Cloud Computing”, is my own work, submitted to partially fulfil the requirements for a Master of Computing Degree: Information Technology, at Tshwane University of Technology, Pretoria. Furthermore, I declare that all sources cited or quoted are identified and acknowledged by a comprehensive list of references

Signature	Date
…………………….	……………………

Dedications

I would like to take this moment to dedicate this research study to my wife, the first lady, who was always by my side, encouraging me not to give up during the tough time in this research study.

Acknowledgments

First of all, I would like to take this moment to thank God for granting me the courage to conduct this study. Secondly, I would like to extend my words of gratitude to my supervisors, Dr DP du Plessis, Dr TE Mathonsi, and Dr TM Tshilongamulenzhe, for their support, warm encouragement, critical comments, and patience throughout. Their instructions and feedback were always constructive and positive. Finally, thanks go to my whole family for their support and motivation.

Publications

Parts of this research study have been submitted and accepted for publication in the following journal and conference proceedings:

Conference proceedings:

1. Coster Baloyi, Deon P. Du Plessis, Topside E. Mathonsi and Tshimangadzo M Tshilongamulenzhe, "Implementation of an Enhanced Security Algorithm for Detecting Distributed Denial of Services Attacks in Cloud Computing". Proceedings of the 2022 International Conference on Computational Science and Computational Intelligence (CSCI'22), 14 – 16 December 2022, Las Vegas, USA, Pages 954 – 958, ISBN-13: 979-8-3503-2028-2, DOI 10.1109/CSCI58124.2022. Available at: https://american-cse.org/csci2022-ieee/csci2022.zip or https://ieeexplore.ieee.org/xpl/conhome/1803739/all-proceedings.

Journal:

1. Coster Baloyi, Deon P. Du Plessis, Topside E. Mathonsi and Tshimangadzo M. Tshilongamulenzhe, "Implementation of an Enhanced Security Algorithm for Detecting Distributed Denial of Services Attacks in Cloud Computing". Submitted for publication at the Journal of Advances in Information Technology

Abstract

Cloud Computing has the benefit of offering on-demand scalable services to its customers without having to invest much in hardware infrastructure, resources, and software. Most private and public sectors are moving to the Cloud. As a result Cloud Computing has become an ideal option due to its flexibility, scalability, and cost efficiency. The existence of vulnerabilities in the network systems hosting Cloud services have raised an opportunity for attackers to launch attacks on Cloud Computing infrastructures (Shaar & Efe, 2022). Cybersecurity invaders target public and private sector's key business applications, such as webservers, financial servers, legacy servers, and other servers hosted on the cloud (Alomari et al., 2012). Cybersecurity invaders attack these business applications, exploiting Distributed Denial of Service (DDoS) attacks (Alomari et al., 2012). This research study proposed an enhanced security algorithm to detect DDoS attacks using a real-time dataset to mitigate the challenges of large amount of network traffic data transmitted in the network. The MATLAB simulation tool was employed to simulate the proposed algorithm. The proposed algorithm has achieved good results of 99.2% detection rate compared to 98.1% of the CUSUM algorithm, 80% of EWMA algorithm and the 79.9% Naïve Bayes detection rate.

Keywords: Distributed Denial of Service Attacks, Cybersecurity, Cloud Computing, and Quality of Service (QoS)

Table of Contents

List of Figures

List of Tables

List of Algorithms

Glossary

AFP	Amount of false positives
ANN	Artificial neural network
Bps	Bits per second
BS	Base station
CoA	Classification of attack
CURE	Clustering using representation
CUSUM	Cumulative Sum
DDoS	Distributed Denial of Service
DNS	Domain Name System
DoS	Denial of Service
DRFBS	Distributed random forest based on spark
DT	Decision tree
EWMA	Exponentially Weighted Moving Average
GBDT	Gradient boosting decision tree
HTTP	Hyper Text Transfer Protocol
IaaS	Infrastructure as a Service
ICMP	Internet Control Message Protocol
IDS	Intrusion detection system
IP	Internet Protocol
IT	Information Technology
JIAC	Java intelligent agents component
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LDF	Likelihood density function
MatLab	Matrix laboratory
MLP	Multi-Layer Perception
MSVM	Multiclass support vector machine
NB	Naïve Bayes
NeSSi2	Network security simulator 2

NIDS	Network Intrusion Detection System
NS-3	Network simulator 3
OMNeT	Objective modular network testbed
OPNET	Optimized network engineering tools
PaaS	Platform as a Service
PCs	Personal Computers
PoD	Ping of Death
Pps	Packets per second
QoS	Quality of Service
RF	Random Forest
RNN-IDS	Recurrent neural networks – intrusion detection system
Rps	Request per second
RTNTAD	Real-time network traffic attack detection
SaaS	Software as a Service
SDN	Software defined network
SITA	State Information Technology Agency
SLA	Service level agreement
SN	Sensor node
SQL	Structured Query language
SVM	Support vector machine
SYN ACK	Synchronize-Acknowledgement
TCP	Transmission Control Protocol
TCP SYN	Transmission Control Protocol Synchronize
UDP	User Datagram Protocol
USML	Unsupervised machine Learning
VMs	Virtual machines
WSN	Wireless sensor node
XSS	Cross-Site Scripting

Chapter 1: Introduction

1.1 Background

Cloud Computing is the imminent newcomer in the computing segment; it is assumed to be the coming generation of the Internet of Things (IoT) (Amjad, Alyas, Farooq & Tariq, 2019). Cloud Computing has the benefit of centralizing various computer services in one server. Applications and data hosted on the premises are being removed from personal computers (PCs), local servers, and desktops computers, and migrated to the private Cloud. Cloud Computing has resolved many issues regarding time, effort, and costs, by providing services at a reasonable amount and with less effort (Amjad, Alyas, Farooq & Tariq, 2019).

Due to the increasing amount of Cloud Computing adoption, customers have serious concerns about the privacy, availability, accessibility, and integrity of information, with information security being the main concern (Rani & Gangal, 2012). Information security in Cloud Computing brings uncertainties to the customers who utilize this technology (what if data stored in the Cloud is not secured?). For example, the use of a specific category of operating system such as open source in the Cloud may be a concern; and can pose a security threat, which is a security risk on its own (Teneyuca, 2011).

There are various attacks that causes severe disruption to data stored in the Cloud. There is the structured query language (SQL) injection attack, the Denial of Service (DoS) attack, the Distributed Denial of Service (DDoS), and Cross-site scripting (XSS). However, several authors have highlighted the DDoS attack as the most problematic attack due to its ability to deplete resources within the network system (Alzahrani & Hong, 2018; Bharot, Verma, Suraparaju & Gupta, 2016; Krishna & Kumar, 2018; Mahjabin, Xiao, Sun & Jiang, 2017).

DDoS attacks can be prepared so as to send simultaneous attacks to multiple systems from one source. Millions of packets can be sent to slow down the network services (Mahjabin, Xiao, Sun & Jiang, 2017). DDoS attacks can affect a network, an application, a website or online services to disrupt availability of services.

The victims that are affected by these attacks are mainly service providers, large companies, commercial services, with even government organizations being targeted (Mahjabin, Xiao, Sun & Jiang, 2017). The aim of DDoS attacks is to flood the network system with millions of malicious requests, depleting network resources such that legitimate users cannot access the network services (Sahi, Lai & Li, 2017). DDoS attacks are among the foremost troublesome issues to resolve online, in that such attacks can also target the webserver hosted on the cloud. (Sahi, Lai & Li, 2017; Alomari et al., 2012).

Recent attackers have realized the capacity to put together a set of methods through compromised accounts, social engineering, and toothless insiders from within the organization's email perimeters linked to a device in the same organization; such makes DDoS attacks more challenging (Kotey, Tchao & Gadze, 2019).

These challenges may be problematic to deal with, the aim being to disrupt and deplete the resources of the victim. In addition, other attackers may cooperate to dispatch numerous attacks to same object (Kotey, Tchao & Gadze, 2019). Furthermore, instruments for coordinating an attack may effortlessly be discovered online and involve almost no information regarding attack contents which pledge an attack to the object (Kotey, Tchao & Gadze, 2019). According to Sahi, Lai & Li, 2017, DDoS attacks at this time are the most common cybercriminal attacks after data theft.

According to British Broadcasting Corporation (BBC, 2016), in October 2016 a stream of DDoS attacks escalated, attacking the dynamic domain name system (DNS), including a huge number of Internet Protocols (IPs) (Mahjabin, Xiao, Sun & Jiang, 2017). This attack was estimated of 1.2 Tbps magnitude, affecting IoT devices (BBC, 2016) (Mahjabin, Xiao, Sun & Jiang, 2017).

DDoS attacks are grouped into three different categories, namely: application layer, capacity-based, and protocol attack (Douligeris & Mitrokotsa, 2004; Krishma & Kumar, 2018). The application layer is a type of an attack that targets an application and specific vulnerabilities in the network systems so that applications are not able to communicate and deliver content to its users. This results in the web server crashing, and is presented in request per second (Rps)

(Douligeris & Mitrokotsa, 2004; Krishma & Kumar, 2018). The capacity-based attack utilises high traffic to overwhelm the network bandwidth of the attacked site; and the magnitude of this attack is commonly presented in bits per second (Bps). This capacity-based attack method incorporates internet control message protocol (ICMP) flooding, user datagram protocol (UDP) flooding, and satirised packet flooding.

The protocol attack consumes any data, device, or other components of the environment supporting information systems such as firewalls and load balancers. This protocol attack type incorporates transmission control protocol (TCP) synchronize (SYN) flooding, divided Ping of Death (PoD), Smurf attack, and packet attack; it is presented in packets per second (Pps) (Krishma & Kumar, 2018; Mahjabin, Xiao, Sun & Jiang, 2017). This research study concentrates on only two sorts of DDoS attacks, namely, TCP SYN flooding, and UDP flooding.

TCP SYN flooding is utilized to disturb the TCP convention manipulating the three-way handshake standard, where client and server association ought to be built up first before information transmission (Krishma & Kumar, 2018). The client must send SYN communication to the server. At that moment the server may recognize this by sending SYN-ACK communication to the client. The client must then send ACK communication to the server, and the association is built up (Krishma & Kumar, 2018).

UDP is a connectionless protocol which does not have association set up before information transmission between the sender and collector (Krishma & Kumar, 2018). Likewise, UDP may not identify the packet misfortune on the data transmission and may not send any mistake communication (Krishma & Kumar, 2018). UDP flooding sends innumerable packets to sporadic objective ports on the victim's computer. This will back off the computer system and crash the victim's computer (Krishma & Kumar, 2018).

Upsetting the use of business conventions, TCP and UDP authorises the attackers to disturb the victim by causing huge sums of traffic to be sent to the casualty, causing an instant crash (Prakash et al., 2016). TCP SYN flooding is utilised to disrupt the TCP convention manipulating the three-way handshake standard.

A number of research studies were conducted in the same field using different approaches, namely, the digital signature of network phase, the anomaly-based real time prevention, the firefly algorithm for clustering, and a framework to incorporate network intrusion detection system (NIDS) in the Cloud; however, there is still a need to improve the current algorithms in order to achieve better training results (Sreeram & Vuppala, 2019; Salmen, Hernandes, Carvalho & Proenca, 2015; Prasad, Reddy & Rao, 2016; Senthilnath, Omkar & Mani, 2011; Modi et al., 2012; Chirag et al., 2012).

Salmen, Hernandes, Carvalho & Proenca, 2015, proposed an advanced signature of network segment for flow analysis; however, colossal traffic restricts the review of each packet, allowing unattended packets to go through (Sreeram & Vuppala, 2019). In building a profile in anomaly-based real time prevention (ARTP), an organization is left in an unachieved state; consequently inclined to attacks and malicious activity which is untraceable (Alzahrani & Hong, 2018; Prasad et al., 2016). Senthilnath, Omkar & Mani, 2011, utilised the firefly algorithm and k-implies for bunching purposes. Such increments the computational time intricacy and may not be relevant for detecting DDoS attacks in the application layer continuously (Sreeram & Vuppala, 2019).

This research study focuses on government organizations. This is on the grounds that the expanding patterns in network protection space trigger the need to reinforce the information technology (IT) security controls, limiting the likelihood of being a victim of cybercrime in government associations. Security breaches compel organizations to spend a great deal of money, while increasing the risk of exposure of sensitive business information. Hence, this research study proposes an enhanced information security algorithm to detect DDoS attacks, in order to improve the detection capability in Cloud Computing.

1.2 Motivation

In the year 2019, the City of Johannesburg in South Africa came to a standstill and lost over R2, 2 million due to a cyber-attack that led to all systems being shut down, including their website, e-services and billing system (Pieterse, 2021). Hence it became necessary to introduce an enhanced security algorithm to detect

DDoS attacks. Literature has indicated that, for the effectiveness of the detection of DDoS attacks for known and unknown attacks, there is a need to improve the existing algorithms such as Hadoop-based Live DDoS Detection, Naïve Bayes, and Real-Time Intrusion Detection Using Hadoop (Alzahrani & Hong, 2018), despite the change in the known attacks. This research study was designed to make a notable contribution to the telecommunication field by offering an industry-tailored solution. Thus, this research study proposed a solution that improved the detection of DDoS attacks in such a way that even the new attacks are effectively detected in real-time with low false positives and a high detection rate.

1.3 Problem Statement

Literature has indicated that other researchers have designed and implemented various security algorithms in order to detect UDP and TCP SYN flooding attacks in the Cloud Computing environment (Somasundaram & Meenakshi, 2021; Virupakshar et al., 2020). However, the existing security algorithms for monitoring and detecting DDoS attacks such as multi-layer perception (MLP), random forest (RF) algorithm, Naïve Bayes and the data mining algorithm (Awan et al., 2021; Abbas & Almhanna, 2020), are still challenged in separating malicious attacks from normal network traffic when a large amount of network traffic is being transmitted at a high speed. Thus, these security algorithms suffer from high false positives, due to poor detection accuracy. This is because these algorithms do not utilize a real-time dataset for detecting DDoS attacks in the network system. In addition, these security algorithms require more time to examine every packet being transmitted due to their computational complexity resulting to poor quality of services (QoS) in Cloud Computing.

1.4 Research Questions

The main question of this research study reads as follows: How does one develop a security algorithm that can improve detection accuracy in Cloud Computing?

Sub-questions

1. What are the existing algorithms used to detect UDP and TCP SYN flooding attacks in Cloud Computing?
2. How can the proposed security algorithm that improves the detect UDP and TCP SYN flooding attacks while reducing false positives rate in real-time be designed?
3. What is the effectiveness of the proposed security algorithm compared with the CUSUM, EWMA and Naïve Bayes algorithms?

1.5 Objectives

The main aim of this research study was to develop a security algorithm that can improve the detection accuracy in Cloud Computing.

The specific objectives of the research study are:

1. To identify and analyse the existing algorithms use to detect UDP and TCP SYN flooding attacks in Cloud Computing.
2. To design and implement an enhanced security algorithm that detects UDP and TCP SYN flooding attacks in real-time.
3. To measure the effectiveness of the proposed security algorithm against existing algorithms.

1.6 Research Methodology

The proposed algorithm was designed to minimize false positive and detection delays using machine learning techniques with real-time datasets in Cloud Computing. To reply to the research questions, the research study followed an experimental approach that was utilized together with the following research methods:

1.6.1 Literature review:

This was accomplished by referring to sources such as journal articles, conference papers, white papers, books, and other internet sources.

1.6.2 Modelling:

Mathematical models was utilized when designing the proposed security algorithm.

1.6.3 Simulation and implementation:

The effectiveness of the proposed solution was tested and measured utilizing MATLAB tool. The MATLAB simulation tool has been utilized for numerous aspects such as prototyping, modelling, exploration and visualization, simulation; development of the algorithm thus being a powerful tool.

1.7 Expected Results

This research study achieved the following:

1. An improved security algorithm was designed that can detect UDP and TCP SYN flooding attacks in Cloud Computing in real-time.
2. Publications were made through conference papers and journal articles.
3. A report was presented in the form of a dissertation.

1.8 Contribution of the Study

This research study provided the following contributions:

1. An enhanced security algorithm was designed that detects UDP and TCP SYN flooding attacks in real-time.
2. The proposed algorithm was able to respond quickly and with an early detection by 99.2% compared to 98.1% of the CUSUM algorithm, 80% of EWMA algorithm and the 79.9% Naïve Bayes.
3. The proposed algorithm improved the detection rate of UDP and TCP SYN flooding by 99.2% and reduced the false positive alarms to below 18%.

1.9 Organization of the Dissertation

In Chapter One the research study offered a detailed background of the DDoS attacks and the impact they cause in the organizational network system. The chapter further highlighted the various types of these attacks. The chapter continued to highlight the problem that the research study has found and what it intends to achieve. The research study replies to the main research question with the main research aim and three research sub-questions guided by the three specific objectives, respectively.

In Chapter Two, the research study presents a review of literature on numerous forms of DDoS attacks available in the network. The two forms of DDoS attack, namely UDP and TCP SYN flooding attacks, were introduced as the main focus of the research study. The research study reviews in detail the numerous traditional intrusion detection method and their shortcomings. The essential purpose of such is to compare the numerous detection techniques in order to highlight the strengths and shortcomings they pose and to identify research gaps existing within literatures.

In Chapter Three the research study focuses on the research methodology, giving the details of how the research study was carried out to achieve the desired results. This research study designed the proposed algorithm by integrating three algorithms, namely, the adaptive cumulative sum (CUSUM), the exponentially weighted moving average (EWMA), and the Naïve Bayes algorithms. The newly proposed security algorithm utilised real-time simulation to detect DDoS attacks.

In Chapter Four, the research study discusses the implementation of the proposed solution using a network simulator; while in Chapter Five, the research study presents the research results. And lastly, Chapter Six presents the conclusions, the research study summary, and future work.

1.10 Chapter Summary

In this chapter the research study focuses on introducing Cloud Computing and its various attacks. The UDP and TCP SYN flooding attacks are the point of focus for this research study. The research study identified the problem and mapped the way in which this problem would be addressed. In order to solve the problem, the research study identified the main research question accompanied by sub-questions which then guide the objectives, respectively. The research study has introduced research methods that were used to design the proposed security algorithm in order solve the identified problem. In the next chapter the research study will present a comprehensive literature review and related work.

Chapter 2: Literature Review

2.1 Introduction

The paradigm was provided by Cloud Computing form's novel cybersecurity encounters such as zombie attacks (flooding attacks), DoS attacks, detection and prevention of network intrusion, among others. As a result, a number of systematic offerings have been conducted in this arena in the past (Bonguet & Bellaiche, 2017). A significant amount of work has been prepared aimed mainly on identifying vulnerabilities, threats, and new methodologies (Bonguet & Bellaiche, 2017; Kim, 2019). The techniques such as K-Mean clustering algorithms, Naïve Bayes, decision trees and Hadoop-based Live DDoS Detection were developed to address such difficulties, many of them based on machine learning (Bonguet & Bellaiche, 2017; Kim, 2019).

In addition, these security concerns may increase in the years to come due to the dynamic migration of individuals and companies to Cloud infrastructures (Bonguet & Bellaiche, 2017). This is an ongoing challenge even today − the need for centralization of various services in one server in the Cloud is a requirement for many organizations (Amjad, Alyas, Farooq & Tariq, 2019).

In 2016, it was reported that the pattern capacity of DDoS attacks was growing alarmingly compared with that reported in previous years (2007 to 2016) with regard to volume scope, as shown in Figure 2.1 (Mahjabin, Xiao, Sun & Jiang, 2017). This calls for the need to provide a noteworthy and cutting-edge expression of mitigation and prevention techniques for DDoS attacks (Mahjabin, Xiao, Sun & Jiang, 2017).

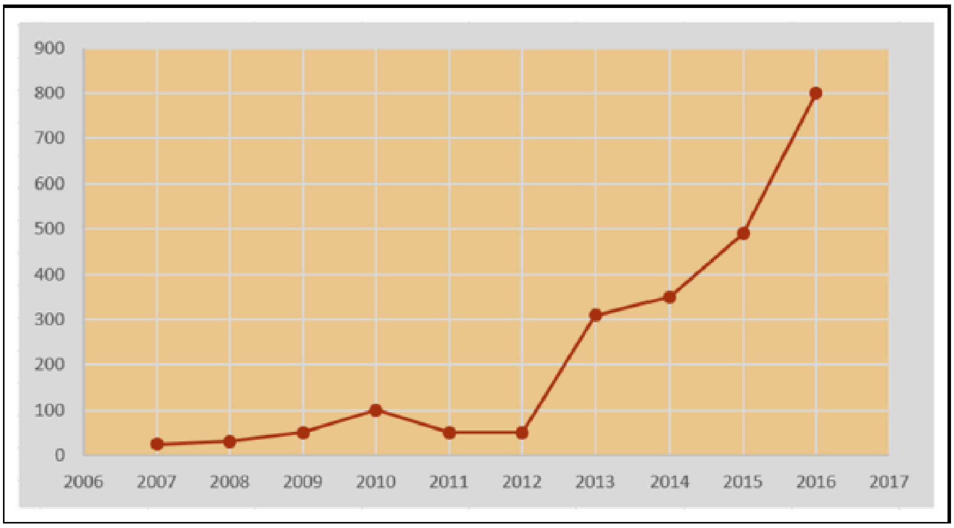

Figure 2.1: The capacity of DDoS attacks (Mahjabin, Xiao, Sun & Jiang, 2017)

In the year 2007 the capacity for DDoS attacks was about 100GB each second; 10 years later, this has increased to 800GB each second (Mahjabin, Xiao, Sun & Jiang, 2017). This increased significantly in the year 2016, triggering the need to mitigate and prevent DDoS attacks (Mahjabin, Xiao, Sun & Jiang, 2017).

Cloud Computing has eliminated challenges in the investment into training new staff, novel infrastructures, and even concerns over procuring new software licences (Amjad, Alyas, Farooq & Tariq, 2019). This comes with the benefit of increasing volume and technological capabilities off premises (Dong, Abbas & Jain, 2019).

The remaining part of this chapter is organized as follows: In Section 2.2, this research study discusses the Cloud Computing overview that includes its types.

In Section 2.3, the study presents the Cloud Computing development types. In Section 2.4, categories and effects of DDoS attacks are reviewed. In Section 2.5, the two types of DDoS attacks that the research study focuses on are defined, namely: UDP and TCP SYN flooding attacks. In Section 2.6, the research study discusses the related work. Finally, the chapter summary is presented in Section 2.7.

2.2 Cloud Computing Overview

Cloud Computing offers diverse categories of services to its users commonly known as infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) (Sen, 2013).

2.2.1 Infrastructure as a service

This type of service offers full computer resources such as networking, storage and servers in a virtualization method over the internet (Sen, 2013). This is a pay-as-you-go type of service offering. Customers using this type of service illuminate capital expenditure of deploying in-house hardware and software systems (Amjad, Alyas, Farooq & Tariq, 2019)

2.2.2 Platform as a service

This type of service only offers its customers access to a software platform (Ahmed et al., 2017). The payment model for this type of services offering is paying while using the platform (Amjad, Alyas, Farooq & Tariq, 2019).

2.2.3 Software as a service

This type of service offering provides the customers with the ability to access the services through the internet browser where applications are running online. These application are accessible through various devices which support the use of the internet browser (Sen, 2013). Service providers for SaaS manage the backend infrastructure for smooth running of applications (Dong et al., 2019).

2.3 Cloud Computing Development Types

Cloud Computing supports four development frameworks, namely (Dong et al., 2019):

2.3.1 Public Cloud

This is a type of development framework in which a Cloud service provider offers services freely to its customers (Amjad, Alyas, Farooq & Tariq, 2019; Dong et al., 2019). This will mean that the customers will have control over security, network, and data which enhance productivity and effectiveness in organizations (Sen, 2013).

2.3.2 Private Cloud

This is a type of development framework in which the service is owned by one client (Ahmed et al., 2017). Customers utilizing this service will have control over performance, security, and reliability (Amjad, Alyas, Farooq & Tariq, 2019; Dong et al., 2019). The building and the management of the private Cloud is accomplished by the external service provider. However, it is also referred to as an internal Cloud, in that the organization which utilizes this option has to have many customers (Amjad, Alyas, Farooq & Tariq, 2019)

2.3.3 Hybrid Cloud

This is a type of development framework in which the public and the private Cloud are combined to form Hybrid Cloud (Sen, 2013). The properties of Hybrid Cloud are for both public and private Cloud (Amjad, Alyas, Farooq & Tariq, 2019). Organizations have the flexibility of storing their basic applications and data in the private Cloud while outsourcing others to the public Cloud (Dong et al., 2019).

2.3.4 Community Cloud

This is a type of development framework in which a specific community of customers with mutual anxieties utilizes a shared infrastructure service (Amjad et al., 2019). One or more businesses within a community can thus manage or operate the community Cloud (Amjad, Alyas, Farooq & Tariq, 2019).

This particular research study focuses mainly on the private Cloud. Most government organizations in South Africa are adopting the private Cloud, benefiting by collaborating with the State Information Technology Agency (SITA) which is the current private Cloud service provider for government organizations. This move to the Cloud raises a need to review challenges in the

Cloud Computing. The greatest concern is security, there being increasing vulnerabilities in the network systems. The DDoS attacks are of the most frequent and serious attacks online today.

2.4 Categories and Effects of DDoS Attacks

The cybersecurity issues in Cloud Computing include man-in-the-middle attacks and DDoS attacks. The essentials of a DDoS attack are deliberated. An attacker uses several zombies to increase the effect of the attack from the victim's side by utilizing one or more devices to launch numerous attacks (Krishma & Kumar, 2018). The diagram below demonstrates an attack in the Cloud Computing, as shown in Figure 2.2.

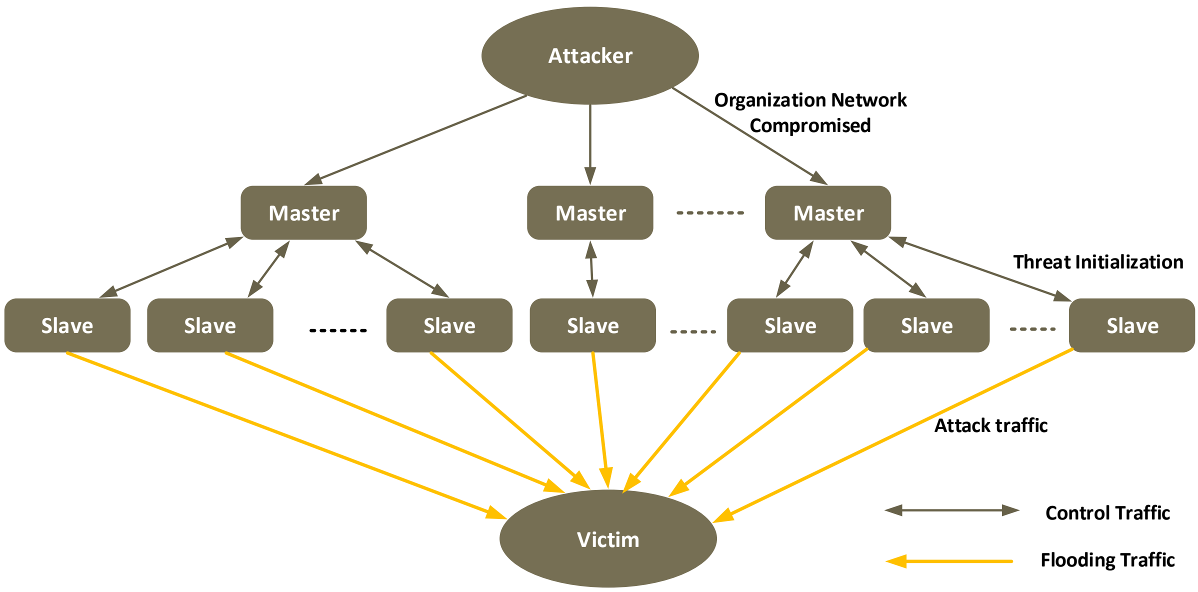

Figure 2.2: DDoS attacks in Cloud Computing (Bhaya & Manaa, 2014; Mahjabin, Xiao, Sun & Jiang, 2017; Shaar & Efe, 2022)

The attack is achieved in three stages. In the first stage an attacker invests a great deal of time setting up the handlers, sometimes called masters. The masters are the controllers of all other machines known as slaves or zombies as shown in Figure 2.2 (Mahjabin, Xiao, Sun & Jiang, 2017). The main aim of the master-slave set-up is to scan for machines with security vulnerabilities in order to exploit them.

The second stage begins when a sufficient number of machines, also known as victims, is found through the scan (Mahjabin, Xiao, Sun & Jiang, 2017; Shaar & Efe, 2022). The attacker feeds the masters with the malicious commands and codes which are further sent to all slaves to set up the attack. This type of attack is known as a botnet attack (Mahjabin, Xiao, Sun & Jiang, 2017; Shaar & Efe, 2022).

In the final stage, the attacker commands the slaves to launch attacks on its victims, overwhelming the resources of the victim, and flooding the network with multiple packets (Bhaya & Manaa, 2014; Mahjabin, Xiao, Sun & Jiang, 2017). These attacks disrupt the network and may result in an outage. There are three classes of DDoS attacks, namely, protocol attack, application-layer attack, and capacity-based attack (Douligeris & Mitrokotsa, 2004; Krishma & Kumar, 2018).

2.4.1 Capacity-based attack

This type of attack may flood the information-transfer capacity of the victim site. Such is assessed in bits per second. This attack type incorporates ICMP flooding, UDP flooding, and other satirized packet flooding (Douligeris & Mitrokotsa, 2004; Krishma & Kumar, 2018).

2.4.2 Protocol attack

The protocol attack hinders the real server's assets, estimated in packets every second. This type of attack integrates TCP SYN flooding, divided SQL injection, Smurf attack, and Ping of Death (Douligeris & Mitrokotsa, 2004; Krishma & Kumar, 2018).

2.4.3 Application layer attack

Another class of DDoS attack is the application layer attack. Here the web server may crash; estimated in demands per second (Amjad, Alyas, Farooq & Tariq, 2019). There are various types of DDoS attack available online, such as Ping of Death, zero-day, hypertext transfer protocol (HTTP) flooding and many more. This research study concentrates on two sorts of DDoS attack: UDP and TCP SYN flooding attacks.

2.5 UDP and TCP SYN Flooding Attacks

2.5.1 UDP Flood

This is a protocol that is connectionless, in which there is no association set-up before information transmission between the sender and collector (Krishma & Kumar, 2018; Mahjabin, Xiao, Sun & Jiang, 2017). UDP flooding may not recognize untoward packets amid the data transmission and may not direct any error notification (Amjad, Alyas, Farooq & Tariq, 2019; Krishma & Kumar, 2018; Mahjabin, Xiao, Sun & Jiang, 2017).

The discrepancy in transmission speed for TCP and UDP favours the UDP. In any case, UDP packets may be misused by perpetrators to dispatch UDP flooding outbreaks, for example, tall transfer speed outbreaks (Krishma & Kumar, 2018). UDP flooding may be pushed through, sending countless packets to irregular goal ports on the target's computer. The flooding may back off the computer framework and cause severe impact as a result of an outage (Krishma & Kumar, 2018; Mahjabin, Xiao, Sun & Jiang, 2017).

The UDP packet may identify the application type waiting at the destination port by responding with an ICMP packet, as shown in Figure 2.3 (Mahjabin, Xiao, Sun & Jiang, 2017).

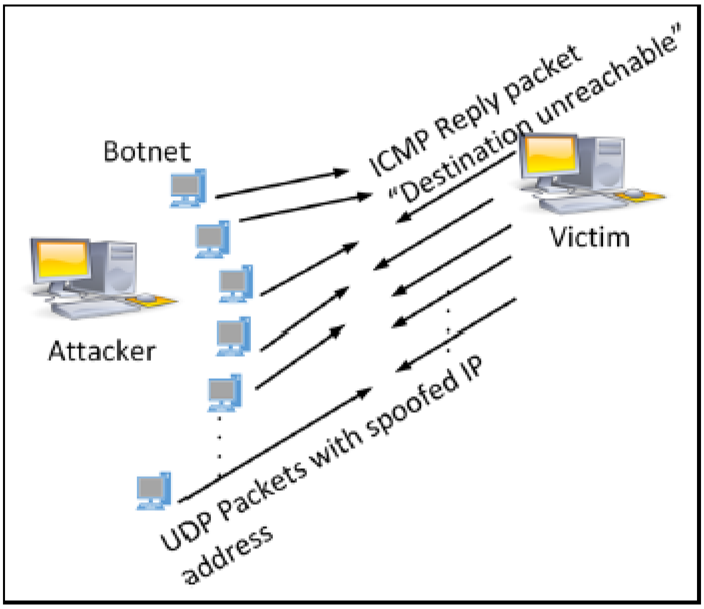

Figure 2.3: An attack using UDP flooding (Mahjabin, Xiao, Sun & Jiang, 2017)

2.5.2 TCP SYN flooding

In the TCP SYN flood connection, the customer and server association ought to be built up initially before data transmission. This is referred to as the TCP three-way handshake (Amjad et al., 2019; Gkounis, 2014; Krishma & Kumar, 2018; Virupakshar et al, 2020). A SYN communication may have to be sent to the server by the customer. At that point the server may identify itself through transferring SYN-ACK communication to the customer. The customer may thereafter have to send ACK communication to the server so that an association may be built up (Amjad et al., 2019; Krishma & Kumar, 2018; Virupakshar et al, 2020).

Nonetheless, the typical TCP three-way handshake may be transformed into a TCP SYN flooding once the attacker transfers rehashed SYN packets to the server arbitrary port by utilizing a phony IP address (Krishma & Kumar, 2018; Virupakshar et al, 2020). The server may be confronted by some issues which may include trouble shutting the “link remains open” and reliably gaining an extensive amount of SYN packets. However, there may not be a reaction formed to authenticate the customer; this may result in the server being crashed (Amjad et al., 2019; Krishma & Kumar, 2018).

In addition to this, attackers send a SYN packet with a spoofed IP address that is not genuine. The server may respond with SYN + ACK packets, while the attacker certainly has not received the ACK packets from the attacker as shown in Figure 2.4 (Mahjabin, Xiao, Sun & Jiang, 2017).

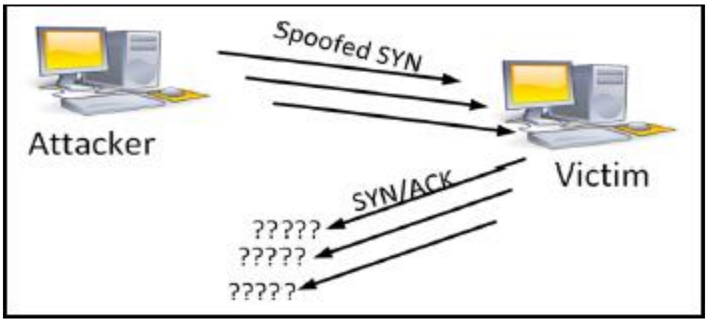

Figure 2.4: An attack for TCP SYN flooding (Mahjabin, Xiao, Sun & Jiang, 2017)

2.6 Related Work

In order to ensure that DDoS attacks in the network system are detected effectively and accurately with few false positives, a number of studies were proposed over the past years. However, many of these studies did not focus on utilizing a real-time dataset in order to improve the detection accuracy of DDoS attacks in Cloud Computing.

Zekri et al. (2017) proposed an IDS signature-based scheme that utilized a C4.5 algorithm in order to improve the detection accuracy of DDoS attacks. The C4.5 algorithm and signature detection scheme were used to create a decision tree to provide effective and automatic detection of DDoS attacks. Due to constraints in resources, an OpenStack Juno simulation tool was used to simulate the results in a virtualized environment in which a virtual LAN and a list of VMs were utilized. The simulation results showed that the C4.5 algorithm improves the detection accuracy to 98.8% in 0.58 second compared with Naïve Bayesian in which the detection accuracy was 91.4% in 1.25 second. The limitation of the scheme is that it did not use the real-time dataset, thus it is unable to detect unknown attacks in Cloud Computing.

Awan et al. (2021) proposed a machine learning (ML) approach that involved two mechanisms, namely: MLP and RF, in order to detect the DDoS attacks in real time. The researchers utilized the big data models Spark and Scikit ML for evaluating the performance on Google Colab library. Databricks Community Editions was deployed to evaluate their solution. Their study included the Apache Spark tool, a big data tool which trains and tests the time measured per minute. Their proposed solution achieved 99.5% detection accuracy, thus their solution could detect attacks in real time within a few milliseconds. The real-time dataset utilised was Dos Dataset which operates in the application layer. The limitation of their scheme is that it is unable to examine every packet when a large amount of traffic is being transmitted athigh speed in Cloud Computing.

Wang et al. (2014) proposed an IDS anomaly-based scheme solution that depends on the DDoS attack mitigation architecture utilizing software-defined networking (DaMask). Challenges of security in Cloud Computing are addressed, demonstrating that a DDoS attack defence can be extremely efficient and effective. DaMask comprises three layers, the network switches, the controller and an application which enhances the defence mechanism against DDoS attacks. Wang et al.'s (2014) study shows that the successful implementation of a software-defined network (SDN) scheme enables enterprises to defend themselves against DDoS attacks. The above-mentioned study added the use of the Snort to enhance performance, providing a positive online test process. The cloud service Amazon EC2 was used for simulation. The results of the study showed that the scheme works well under new challenges on the network, while reducing computational and communication overheads. The concern with this approach is that the study did not use the real-time dataset to deal effectively with both new and old attacks, thus avoiding further DDoS attacks in a complex network environment.

Abbas & Almhanna (2021) proposed an IDS anomaly-based scheme that utilised a data-mining algorithm in order to detect DDoS attacks in the network system. The scheme was divided into four segments. The first segment is pre-processing; the second segment is the anomaly detection model in which classification of various features in the training step is achieved using the Naïve Bayes (NB) algorithm; and extraction of data patterns is accomplished per a random forest (RF) algorithm, compareng the results. In the third segment the results were tested utilizing the trained dataset. The fourth segment offered the collection of system performance evaluation metrics such as detection rate, accuracy, precision and false alarms. Their study utilized the MIX dataset and further combined PORTMAP and LDAP datasets. Their accuracy in detection was 99.98% with the detection rate of 100%, and no false alarms were detected. The limitation with their scheme is that the dataset used is not real time, and the proposed RF and NB systems were not online.

Tuan et al. (2019) proposed an experimental performance analysis designed by integrating machine-learning techniques, namely, the support vector machine (SVM), the Naïve Bayes (NB), the unsupervised learning (USML), the artificial neural network (ANN) and the decision tree (DT) in order to solve the challenges of DDoS attacks in the network system. Their study utilized the UNBS-NB15 and KDD99 datasets. The confusion matrix was deployed to assess and compare the outcome of the algorithm’s performance. The results showed a 84.32 % SVM, 94.43% DT, 71.63% NB, 64% ANN and 94.8% USML when the experimental analysis was conducted using the UNBS-NB15 dataset. A similar analysis for performance was conducted using the KDD99 dataset. The results were 91.6% SVM, 93.3% DT, 96.7% NB, 97.4% ANN and 98.1% USML. The highest accuracy classifier was the USML in both performance and experimental analysis. The limitation with this study is that the datasets are not given in real time, and that the accuracy can be further improved. Hence here is a need to introduce a new approach necessary to handle the large amount of data traffic using a real time dataset.

Bhaya & Manaa (2017) proposed an unsupervised data-mining scheme known as clustering using representation (CURE) in order to improve the DDoS attacks detection in the network system as an intrusion detection system. Their study utilized the framework of entropy for windowing the packets as they come, and a data-mining scheme using CURE to detect DDoS attacks. Their study utilized the CAIDA2007, CAIDA2008 and DARPA2000 datasets. Their results showed a 96.29% detection rate, a zero per cent false positive, and more than 99% accuracy. The limitation to this approach is that it requires more time to train the datasets.

Zhang et al. (2018) proposed a distributed random forest based on spark (DRFBS) technique in order to handle high-speed traffic data in real time in a network system. The DRFBS technique comprises three segments, namely: a segment that captures data through NetFlow, a data pre-processing segment, and an intrusion detection based on classification. Their study was achieved by comparing the techniques such as Adaboost, multiclass support vector machine (MSVM), gradient boosting decision tree (GBDT) and random forest (RF) techniques. Their study utilized the CICIDS2017 dataset to perform real-time detection of DDoS attacks. The simulation of their techniques was achieved through the Kafka, Spark and Logstash. Their results showed that the DRFBS technique achieved 96.4% of the precision, 96.9% recall and a 0.01second detection time, the RF technique 97.9% of the precision, 97.6 recall, and a 1.10 second-detection time. The GBDT technique achieved 98.2% precision, 97.4 recall, and a 5.39 second detection time. The Adaboost technique achieved 88.7% precision, 95.1 recall and a 19.56 second detection time. MSVM technique achieved 94.5% precision, 84.5 recall and a 2600.36 second detection time, as shown below:

Table I: Summary of the Results of the Techniques Employed (Zhang et al., 2018)

Technique	**DRFBS**	**RF**	**GBDT**	**Adaboost**	**MSVM**
Precision (%)	96.4	97.9	98.2	88.7	94.5
Recall (%)	96.6	97.6	97.8	95.1	84.5
Detection Time (s)	00.1	1.10	5.39	19.56	2600.36

Their study was relevant as it proved that their proposed approach has satisfactory accuracy and efficiency. The limitation to this approach is the probability of success (96% precision rate) for DRFBS technique which still requires enhancement.

Yin et al. (2017) proposed a deep learning technique for intrusion detection utilizing the recurrent neural networks (RNN-IDS) in order to explore how to ncorporate model intrusion detection systems into the network system. The study utilized the NSL-KDD dataset. The training of data in the RNN-IDS model consists of two sections, namely: forward propagation, which is used for calculating the values, and back propagation, which passes on the outstanding data. The results showed that the RNN-IDS has a good accuracy rate of 97.09%

on the testing dataset, higher than the detection rate on the NSL-KDD dataset which was only 94.1%. The study has demonstrated that RNN-IDS has a good modelling capability and a high accuracy detection rate. The limitation of the study is that the detection rate on the NSL dataset is only 94.1% which still requires enhancement and more training time.

Abdulrahman & Ibrahem (2018) proposed a hosted-based intrusion detection technique for handling numerical data utilizing the following techniques, namely: Naïve Bayes (NB), C5.0, support vector machine (SVM) and random forest (RF) in order to select the best technique for intrusion detection in the network system. Their study utilized the CICIDS2017 dataset with 225,711 samples for training and testing. The simulation of their study was conducted per R studio software. Of the four techniques evaluated, the results showed that the C5.0 technique achieved 86% accuracy, 86% recall, 99% precision, an 81% detection rate and 0.46% false positives. The RF technique achieved 86% accuracy, 86% recall, 99% precision, an 81% detection rate and 0.50 false positives. The NB technique achieved 80% accuracy, 90% recall, 86% precision, a 73% detection rate and 64% false positive. The SVM technique achieved 80% accuracy, 92% recall, 86% precision, a 73% detection rate and 75% false positives, as shown below.

Table II: Summary of the Performance Results (Abdulrahman & Ibrahem, 2018)

Technique	C5.0	RF	NB	SVM
Accuracy (%)	86	86	79	79
Recall (%)	86	86	90	92
Precision (%)	99	99	86	84
Detection Rate (%)	81	81	73	75
False positives (%)	0.46	0.50	64	75

The 86% accuracy for both the C5.0 technique and the RF technique supersedes the others with an accuracy average as shown in Table II. The limitation to this study is that the 86% accuracy for both C5.0 and RF techniques still requires enhancement.

Table III: Summary of the Related Work, Advantages, and Research Gap

Summary of the related work	Advantages of the proposed solution	Limitations of the solution
Zekri et al. (2017) proposed the C4.5 algorithm	• Offers tall accurateness in discovering attacks that are well-known.	• Unknown attacks may not be detected

Awan et al. (2021) proposed the MLP and RF algorithms	• Real-time DDoS attack detections	• High amount of traffic may limit the examination of every packet
Wang et al. (2014) proposed the DaMask	• Improves accuracy the more it is utilized	• Ineffective when working with a large amount of data
Abbas & Almhanna (2021) proposed a data mining algorithm	• Has best accuracy and detection results	• The dataset is not real time and the proposed system is not online
Tuan et al. (2019) proposed the machine learning algorithms : SVM, NB, USML, ANN and Decision Tree	• Expanded adaptability in addressing complex challenges	• Effectiveness is uncertain while using real-time dataset
Bhaya & Manna (2017) proposed a data mining algorithm using CURE	• improved accuracy and detection rate	• Requires more time
Zhang et al. (2018) proposed an RF algorithm	• Shorter detection time	• Precision can still be improved
Yin et al. (2017) proposed a deep learning technique	• Good modelling capability and a high accuracy detection rate	• Only 97.09 per cent of the detection rate which can be improved.
Abdulrahman & Ibrahem (2018) proposed C5.0 and RF algorithms	• Better compared with other machine learning	• High false positive rate and requires enhancement

2.7 Chapter Summary

In this chapter the research study started by offering an introduction to the issues delivered by DDoS attacks in Cloud Computing in Section 2.1. In Section 2.2, the research study presented a Cloud Computing overview, and development framework types in Section 2.3. In Section 2.4, the research study presented the categories and effects of DDoS attacks. In Section 2.5, the research study presented the two DDoS attack categories that the study is focusing on, namely: UDP and TCP SYN flooding attacks. In Section 2.6, the research study offers the related work and provides the research gap. In the next chapter the research study will present the proposed security algorithm to detect DDoS attacks, even when there is little variation in the known attacks.

Chapter 3: Research Methodology and System Architecture

3.1 Introduction

This research study proposed an enhanced security algorithm to detect DDoS attacks, using a real-time dataset to mitigate the challenges of large amounts of network traffic data transmitted in the network. The proposed algorithm has the capability of detecting DDoS attacks as soon as they enter the network. The DDoS attacks are effectively detected in real time with low false positives and a high detection rate in Cloud Computing. In Chapter 2, Section 2.6, this research study presented the literature review based on various categories of DDoS attack and their shortcomings.

Therefore, this chapter concentrates on developing an algorithm that will address the shortcomings identified in Section 2.6. The proposed real-time network traffic attacks detection (RTNTAD) algorithm has the capability of detecting DDoS attacks in a large amount of network traffic data being transmitted in the network. Thus, the proposed RTNTAD reduced false positives and detection delays while improving the detection accuracy.

The research study starts by proving the research methodology used in Section 3.2, the study describe the proposed system architecture of the Cloud Computing environment with various network segments to of attacks in Section 3.3. In Section 3.4, the research study discusses in detail the algorithms integrated during the design of the proposed algorithm, and the reason for adopting such algorithms. The designed algorithm is presented in Section 3.5. In Section 3.6, the research study provides the summary of how the designed algorithm works step by step. In Section 3.7, the simulation tool is discussed. In Section 3.8, the research study provides the chapter summary.

3.2 Research Method

In this research study a proposed security algorithm was designed to improve the detection of DDoS attacks in real-time in cloud computing while minimizing false positives detections. Literature review, modelling and simulation was utilized to reply to the research question. The research study utilised MatLab for simulation.

3.3 Proposed System Architecture

In this research study a Cloud Computing Wireless Sensor Node (WSN) system architecture is introduced. A WSN can be defined as a network that is built out of a number of sensor nodes that communicate with one another through a wireless radio device (Akyildiz et al., 2002; Loannis et al., 2007; Ying, 2014). The nodes that form a typical Cloud Computing WSN are the sensor nodes and the base station (BS). The base station, also referred to as the sink node, is a node that, due to its role as a controlling node, behaves as the interface between the sensor nodes and the slaves of the network. The sensor nodes (SNs), also referred to simply as nodes are devices which are small in nature and are capable of detecting the surrounding environment.

The assumption is that there are M nodes in the network system. The sensor nodes produce sensor information and combine the information packets. The BS allot the information from the SNs from time to time. There are a number of malicious nodes in the network. The assumption is that the number of malicious nodes is $x(0 < x \ll M)$ which assess all the packets passing through the network. The DDoS attacks threatens the routing layer of the WSN. The system architecture reflected below represents the challenges of such attacks in the Cloud Computing WSN.

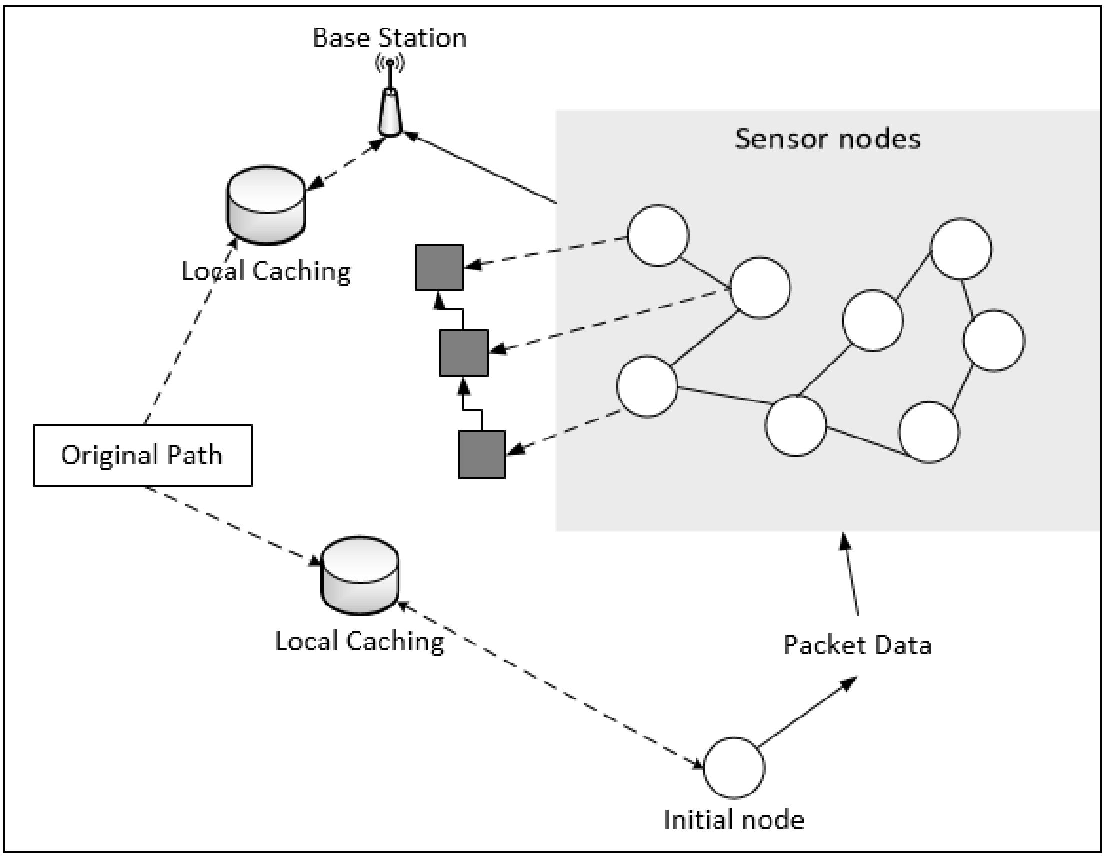

Figure 3.1: The proposed system architecture (Ying, 2014)

The architecture in Figure 3.1 represents the communication of data in Cloud Computing which is a method of packet data relay from the initial node to the base station (Ying, 2014). In order to determine whether there was an attack or not, the packet will have to reach the base station point successfully, confirming that there were no malicious nodes on the path. In the event where the packet does not reach the base station then the node was malicious, confirming and attack occurred. This method is therefore utilized to enhance the quality of the succeeding information packets, implementing intrusion detection (Ying, 2014).

The assumption is that if the packet data from the initial node point successfully arrives at the base station point, the route path is likely to be secured. This can be presented as follows (Ying, 2014):

(1) The initial point node (N) directs packet to the base station(P), while N appends an empty list (M) to each packet data. When a node in the sensor (I_k) collects a data packet regarded as a normal node, it should add (g_k) to its identity with M.

(2) The possibility is that the malicious nodes may disguise themselves by taking the same action.

(3) When the packet data arrive, P extracts $M = \{g_1, g_2, \dots, g_n\}$ (here g_i refers to the identity of a relay sensor node (I_i) from the packet data and keeps it in its local caching. Here M is referred to as an original path.

(4) P adds M to a notification packet and sends the packet to N. The sensor nodes in M are used as the relay nodes.

(5) When a relay sensor node (I_j) collects the warning packet, if its distinctiveness is g_j it utilizes M, then it extracts a sub-route path M_j = {g_j +1, g_j+2,..., g_n} from M and keeps it in its local caching. I_j extracts its next-hop node (I_j−1) with distinctiveness being g_j−1 from M, forwarding the packet to it.

(6) On the arrival of the warning packet, N extracts M from the packet and holds it in its local caching.

3.4 Overview of the Integrated Algorithms

In this section the research study discusses the algorithms that were integrated when designing the proposed algorithm.

3.4.1 Adaptive cumulative sum (CUSUM) algorithm

A number of studies considered the use of CUSUM algorithm in detecting DDoS attacks. Many of them have received good success of accuracy detection rate (Babic et al., 2021; Olufowobi et al., 2019; Sklavounos et al., 2017). This is one of the reasons for the research study considering this algorithm. Furthermore, the CUSUM algorithm is utilized in the detection of TCP SYN flooding attacks, which is one of the common DDoS attacks (Babic et al., 2021).

The CUSUM algorithm is preferred for measuring any deviation from a predetermined value. The CUSUM algorithm can detect unknown attacks effectively, and at a rapid rate. Li et al. (2007) have improved the CUSUM algorithm with the entropy detection algorithm to have more accuracy in the detection by integrating a variation detection and cumulative sum to their entropy algorithm. This aggregates the entropy based on rules; in this way it becomes more effective. Hence, the need to integrate it with EWMA and Naïve Bayes to enhance the detection accuracy in this research study.

The adaptive CUSUM algorithm is deployed to resolve the issues of unidentified parameters that differ over time. The main idea was to assess the parameters on an ongoing in real-time using CUSUM testing instantly, regardless of the accuracy. The important steps in the algorithm use are to assess the attacks and the time of the attack efficiently from the measured sample of the network instances.

We follow a real-time approach using an adaptive CUSUM algorithm that models traffic transmitted in the network system (Olufowobi et al., 2019). Therefore, let P $=\{P_{1,}P_2 \dots \dots, P_n\}$ be a random set of packets detected in sequence. Package P characterises the data frame in the network system. Packet P is the dominating packet at first; and each P_1 follows a likelihood density function (LDF), $d(P_1, \varphi)$ based on the determining parameter φ. The mentioned parameters are to be known as mean $\emptyset$ and unknown θ^2. This packet may contain a change that take place suddenly at time $\widetilde{l_c}$ that is demonstrated by an instant change to the value of φ = φ_0 at time $\widetilde{l_c}$ before-change and may be φ = φ_1 at time $\widetilde{l_c}$ after-change.

When a change take place, an alert should be sent immediately for an action to be applied with fewer false positives.

In the algorithm-detection step, a decision has to be taken between two hypotheses H_0 and H_1 from the observed packets P. The immediate log-probability proportion test is utilized to choose the hypotheses; for example, the testing of a change in signal, is presented by Equation (3.1):

$$G_i = \ln\left(\frac{d(p_{\_i}, \varphi_1)}{d(p_{\_\mathrm{i}}, \varphi_0)}\right) \tag{3.1}$$

Furthermore, the cumulative sum from 0 to *n* is presented by Equation (3.2):

$$G_n = \sum_{i=0}^{n} G_i \tag{3.2}$$

The decision function h_n is presented by Equation (3.3), and the change time estimate l_C by Equation (3.4):

$$h_n = G_n - \min_{1 \le l_c \le n} G_{l_c - 1} \tag{3.3}$$

$$\tilde{l_c} = \frac{min}{1 \le t_c \le n} G_{l_{c-1}} \tag{3.4}$$

The above equations (3.1), (3.2) and (3.4) represent the CUSUM algorithm. Moreover, for the real-time detection, the below equation (3.5) was used:

$$G_n = G_{n-1} + G_n \tag{3.5}$$

A decision function h_n is compared with a positive threshold as presented in Equation (3.6):

$$h_n = \{h_{n-1} + G_n\}^+ \tag{3.6}$$

Once the sudden change is detected, the equation (3.4) is utilised to assess the estimated change time l_c from the samples measured $(P_{1,}P_2 \dots\dots, P_n)$. The recurrence of each packet occasion P_1 in the unique window ω_i is maintained. The research study calculates the CUSUM G_n as demonstrated in Equation 3.5, by calculating the immediate log-probability proportion as presented in Equation (3.7):

$$G_i = \frac{\emptyset_{P_1} - \emptyset_{P_0}}{\theta_P^2}\left(P_i - \frac{\emptyset_{P_1} + \emptyset_{P_0}}{2}\right) \tag{3.7}$$

When an abnormal event is detected, and there is a shift in the process, the algorithm terminates and signals an alarm. The adaptive CUSUM algorithm is presented in Algorithm 3.1.

```
Initialization
    Set the detection threshold k > 0
    G_{-1} = h_{n-1} = 0    // G_{-1} represent change from observed packet
    n = 0
End
While the algorithm is not stopped do
    Measure the current sample x[n]
        G_i = ln( d(p_i,φ_1) / d(p_i,φ_0) ) // This instantaneous log-probability ratio
        G_n = Σ_{i=0}^{n} G_i // The cumulative sum from 0 to n
        h_n = G_n − min_{1≤l_c≤n} G_{l_c−1}   // Decision function h_n
        l̃_c = min_{1≤t_c≤n} G_{l_{c−1}}    // log-probability ratio
        G_n = G_{n−1} + G_n   // real-time detection of change
        h_n = {h_{n−1} + G_n}^+    // decision function G_n compared to a positive
    threshold
```

$G_i = \frac{\emptyset_{P_1} - \emptyset_{P_0}}{\theta_P^2}\left(P_i - \frac{\emptyset_{P_1} + \emptyset_{P_0}}{2}\right)$ // calculating log-probability proportion
If $h_n[n] > k > 0$ $then$
$l_d \leftarrow n$
$\widetilde{l_c} = \min_{1 \le l_c \le n} G_{l_c - 1}$
Stop or reset the algorithm
End
$n = n + 1$
End

Algorithm 3.1: Adaptive CUSUM algorithm

The description of the Adaptive CUSUM algorithm is as follows:

Step 1: The primary objective of this algorithm is to predict parameters in a constant structure with the CUSUM testing algorithm starting promptly, nevertheless with predicted precision.

Step 2: The prediction continues along with the process. This process can lead to more precision prediction.

Steps 3: Then the CUSUM algorithm models the packets transmitted in the system utilizing a change-point prediction system. An anomaly can be modelled utilizing two hypotheses, H_0 and H_1 which represent that the packet is deemed malicious until proven not malicious, that align with 0; and h threshold values. $\varphi = \varphi_0$ represent packet change before, while $\varphi = \varphi_1$indicates change after.

Step 4: The algorithm has to decide between the two hypotheses and predict the duration of the change efficiently using the results measured from the instance. This is referred to as the prediction and detection, respectively.

Step 5: In the event where an anomaly is detected an alarm signal should be sent immediately to allow remediation actions to take place.

3.4.2 EWMA

The EWMA was first introduced to analyse if the given value of the parameter at the point it is observed exceeds a particular threshold at given time interval (Machaka et al., 2016). Machaka et al. (2016) used this solution to present how, by means of testing and describing, the various parameters affect performance detection (Babic et al., 2021). In their description, the connection between false positives and detection rate is also clearly stipulated. The conclusion was that the EWMA may fit well on the high-rate attacks and not very well on the low-rate attacks (Babic et al., 2021).

Based on the conclusion above this study utilizes the EWMA algorithm during the design of the proposed RTNTAD. The EWMA and CUSUM were previously utilized for the shifts of the mean process of the TCP SYN flooding attacks, obtaining a better outcome (Babic et al., 2021; Sklavounos et al., 2017).

The feature of the EWMA that implies that the new data supersedes the older data is important, and is represented using Equation 3.8:

$$\bar{\vartheta}_n = \tau\bar{\vartheta}_n - 1 + (1 - \tau) \tag{3.8}$$

The constant factor τ, a number between 0 and 1, represents the level of weighing decrease. If $\tau = 1$, only new data will be considered, If $\tau = 0$, the opposite holds true, in which old data will be important.

The equation $\vartheta_n = 0$ represents a process mean predicted in real time. The EWMA algorithm is presented in Algorithm 3.2.

---

1. Initialize
2. $\vartheta_n = 0$ //process mean predicted in real time
3. $\tau = 0$ // factor of EWMA
4. End
5. While the algorithm is not stopped do
6. Measure the current sample $x[m]$
7. $\bar{\vartheta}_n = \tau\bar{\vartheta}_n - 1 + (1 - \tau)//$ implies that the new data supersede the older data, equation 3.8
8. If $\tau = 1$ then
9. // only new data may be considered
10. Else
11. If $\tau = 0$
12. //therefore the older data are utmost imperative
13. End else if
14. End

Algorithm 3.2: EWMA Algorithm

The above EWMA algorithm assisted in providing the feature that implies that the new data superseding the older data which is an utmost imperative for this study. This is because an attack can initiate and halt immediately, therefore new data is important (Babic et al., 2021).

The EWMA algorithm in its original state may yield higher numbers of false positives. However, with slight performance fine-tuning and moderation it may produce better results (Babic et al., 2021). In this study the original EWMA algorithm was integrated with CUSUM and Naïve Bayes to enhance performance.

3.4.3 Naïve Bayes

The Naïve Bayes could be a conditional likelihood of attack technique, which allots the class mark to specified information based on the attack likelihood (Barki et al., 2016). Since Naïve Bayes utilizes the likelihood technique, the precision of the classification is greater; however, the classification takes longer as the training information increases. Hence, the need to integrate the algorithm with CUSUM and EWMA to enhance performance (Barki et al., 2016). Furthermore, this type of machine learning technique is capable of detecting intrusions in real time utilizing the Naïve Bayes classifier (Alzahrani & Hong, 2018). For anomaly or normal, Naïve Bayes utilizes the classifier to classify the approaching requests (Barki et al., 2016).

This algorithm allots the class label to given datasets based on the attack likelihood (Barki et al., 2016).

$L(M_k|y_1, \ldots\ldots y_n)$. This is for each of K potential classes and end results presented by Equation 3.9.

$$L(M_k|y = \frac{L(M_k)L(y|M_k)}{\sum_{i=1}^{k} L(y|M_i)L(M_i)} \tag{3.9}$$

The Naïve Bayes algorithm consumes more time for the classification as training data increases and therefore, works well when attributes are independent on one another (Barki et al., 2016). This algorithm cannot therefore be utilized on its original state; this study only utilized the step in which the obtained value is classified based on the probability and value along with the class label appended in the training dataset. We are making use of a dataset with six features that has three classes and 700 samples utilizing ‘classifier’ function. Below is the Naïve Bayes algorithm presented using Algorithm 3.3

---

1. Training a naive Bayes classifier when controller starts
2. Count = Number_ hosts per seconds // the number of host at a given time per second
3. Then obtained value is classified by Naïve Bayes based on the probability. This value, along with class label, is appended in the training dataset
4. if packet is classified as anomaly then
5. warning is given to the user
6. else
7. entry is made in the table of a switch and sent through the port to a destination
8. end if
9. end

Algorithm 3.3: Naïve Bayes algorithm

3.5 Proposed Real-Time Network Traffic Attacks Detection Algorithm

The RTNTAD algorithm is the result of the integration of the three existing algorithms, namely: the adaptive CUSUM algorithm, the EWMA algorithm, and the Naïve Bayes algorithm, in order to improve the detection accuracy of DDoS attacks in Cloud Computing. The newly designed RTNTAD improved the detection rate of DDoS attacks, presenting low false positives. Furthermore, the proposed algorithm has the capability of detecting unknown attacks. The proposed algorithm is presented using Algorithm 3.4.

---

1. Initialization
2. Set the detection threshold $k > 0$
3. $G_{-1} = h_{n-1} = 0$
4. $n = 0$

5. $\vartheta_n = 0$ //process mean predicted in observation real time
6. $\tau = 0$ // factor of EWMA
7. End
8. While the algorithm is not stopped do
9. Measure the current sample $x[n]$
10. $G_i = \ln\left(\frac{d(p\_i,\varphi_1)}{d(p\_i,\varphi_0)}\right)$ // This utilised to make a choice between the hypotheses
11. $G_n = \sum_{i=0}^{n} G_i$ // The cumulative sum from 0 to n
12. $h_n = G_n - \underset{1\le l_c \le n}{min} G_{l_c - 1}$ // Decision function h_n
13. $\widetilde{l_c} = \frac{min}{1\le t_c \le n} G_{l_{c-1}}$ // log-probability ratio
14. $G_n = G_{n-1} + G_n$ // The equations may be rewritten in a recursive form for real-time detection of change
15. $G_n = \{G_{n-1} + S_n\}^+$ // The decision function G_n may be compared to a positive threshold
16. $h_n = \{h_{n-1} + G_n\}^+$ // decision function G_n compared to a positive threshold
17. $G_i = \frac{\emptyset_{P_1} - \emptyset_{P_0}}{\theta_P^2}\left(P_i - \frac{\emptyset_{P_1} + \emptyset_{P_0}}{2}\right)$ // Calculating log-probability proportion
18. $\bar{\vartheta}_n = \tau\bar{\vartheta}_n - 1 + (1-\tau)$//implies that the new data supersede the older data
19. $L(M_k|y = \frac{L(M_k)L(y|M_k)}{\sum_{i=1}^{k} L(y|M_i)L(M_i)}$ // allots the class label to given dataset based on the likelihood
20. If $h_n[n] > k > 0$ $then$
21. $l_d \leftarrow n$
22. $\widetilde{l_c} = \underset{1\le l_c \le n}{min} G_{l_c - 1}$
23. Stop or reset the algorithm
24. End
25. If $\tau = 1$ then
26. // only new data may be considered
27. Else
28. If $\tau = 0$
29. //therefore the older data are utmost imperative
30. End
31. End
32. End
33. End
34. $n = n + 1$
35. End

Algorithm 3.4: RTNTAD Algorithm

3.6 Implemented Algorithm Summary

The proposed algorithm was designed by integrating the adaptive CUSUM algorithm, the EWMA algorithm, and the Naïve Bayes algorithm in order to detect DDoS attacks effectively with minimum false positives and detection delays. Thus, the proposed algorithm improves the detection accuracy of DDoS attacks in Cloud Computing. The values of the combined algorithms into the RTNTAD algorithm when calculated will trigger alarms accordingly in the event of malicious traffic detected to ensure remedial actions. Below is the summary of how the designed algorithm operates:

Step 1: The RTNTAD algorithm starts by initializing all parameter attributes used in order to enable execution of these attributes to achieve the effectiveness and accuracy in the detection of DDoS attacks with low false positives rate.

Step 2: The prediction continues as new packets comes through, leading to more precision prediction utilising a decision faction to detect anomaly packets If a change from the known value occurs, then an attack can be triggered.

Step 3: The equation (3.8) from the RTNTAD algorithm ensures that the new data supersede the older data to help with the accuracy and reduction of the false positives detection which is the EWMA feature.

Steps 4: The equation models the packets transmitted in the system utilizing a change-point prediction system. An anomaly can be modelled utilizing two hypotheses, φ_0 and φ_1 that goes with 0 and h threshold values. φ_0 represents the packet change before transmission while $\varphi_1$is the change after transmission.

Step 5: The algorithm has to decide between the two hypotheses, predicting the duration of the change efficiently from the results measured from the instance. This is referred to as the prediction and detection, respectively.

Step 6: The proposed algorithm uses Equation 3.9 which is a Naïve Bayes feature to improve classification of nodes by allotting the class label on a given dataset, based on the likelihood of improving detection accuracy and effectiveness.

Step 7: In the event of an anomaly being detected, an alarm signal is sent immediately to allow remediation actions to take place.

3.7 Simulation Tool

This research study utilized MATLAB simulation tool in order to evaluate the performance of the proposed algorithm. MATLAB is a simulation software that helps in predicting the behaviour of a system. MATLAB R2017a version 9.2.0.538062 which was released in February 23, 2017, was used in this research study. This version is a complete package comprising modules such as StateFlow, SimEvents, and Simulink. The PC specifications of some 2.6GHz (4 cores) processor Intel (R) and 8GB RAM are utilized. MATLAB has been utilized by engineers and scientists worldwide for a range of applications including deep learning and machine learning, signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology.

3.8 Chapter Summary

This chapter presented the proposed algorithm which has been designed to detect DDoS attacks in Cloud Computing. The proposed algorithm has been designed to improve the detection accuracy with minimal false positives. This chapter introduced the algorithms that were integrated when designing the proposed RTNTAD algorithm. The study then described the RTNTAD algorithm in detail and finally introduced the MATLAB simulation tool which was deployed in this study.

The next chapter will focus on the system implementation using the MATLAB simulation tool to simulate the algorithm that the study has designed in this chapter. The MATLAB simulation tool will be compared with other simulation tools available. Thereafter, the research study presents reasons for MATLAB being selected over other available simulation tools.

Chapter 4: System Implementation

4.1 Introduction

This research study designed the RTNTAD algorithm in order to detect DDoS attacks in real-time. When a malicious traffic has been detected, it is immediately halted and an alarm signal of such attack is sent. The study utilized the MATLAB simulation tool to evaluate the performance of the proposed algorithm in Cloud Computing. The MATLAB application version R2017a was installed in a Windows 10 Enterprise machine that runs 2.60GHz processor with 8192MB RAM.

The rest of the chapter is structured as follows: In section 4.2, the study provides the details of the simulation tools available online, making a comparison between these tools. In Section 4.3, the study gives an overview of the selected simulation tool for the designed algorithm. In Section 4.5, the chapter is concluded.

4.2 Overview of Network Simulation Tools

Because technology improves day by day and there is extreme growth of data traffic in the cloud, making it is exposed to DDoS attacks. The study conducts a simulation of DDoS attacks in order to determine the defense mechanism of DDoS attacks in the network (Valdiviezo, 2014). This section provides the overview of various network simulation tools in order to compare and to provide their key futures.

4.2.1 NeSSi2

NeSSi2 is an agent-based simulation tool built on the Java intelligent agents component ware (JIAC) framework (Hirsch, Konnerth & Hebler, 2009; Valdiviezo, 2014). NeSSi2 provides telecommunication simulation capabilities that support evaluation of security solutions such as IDS, routers, servers and clients (Grunewald, Lutzernberger, Chinnow, Bye, Bsufka & Albayrak, 2011; Valdiviezo, 2014). NeSSi2 can simulate special common attacks. Botnet-based DDoS attacks and worm-spread events are the only two supported examples for NeSSi2, hence, not utilized for this study.

NeSSi2 is structured into three components: the graphical frontend, the agent-based simulation backend, and the result database (Grunewald, Lutzernberger, Chinnow, Bye, Bsufka & Albayrak, 2011; Schmidt, Bye, Chinnow, Bsufka, Camtepe & Chinnow, 2009).

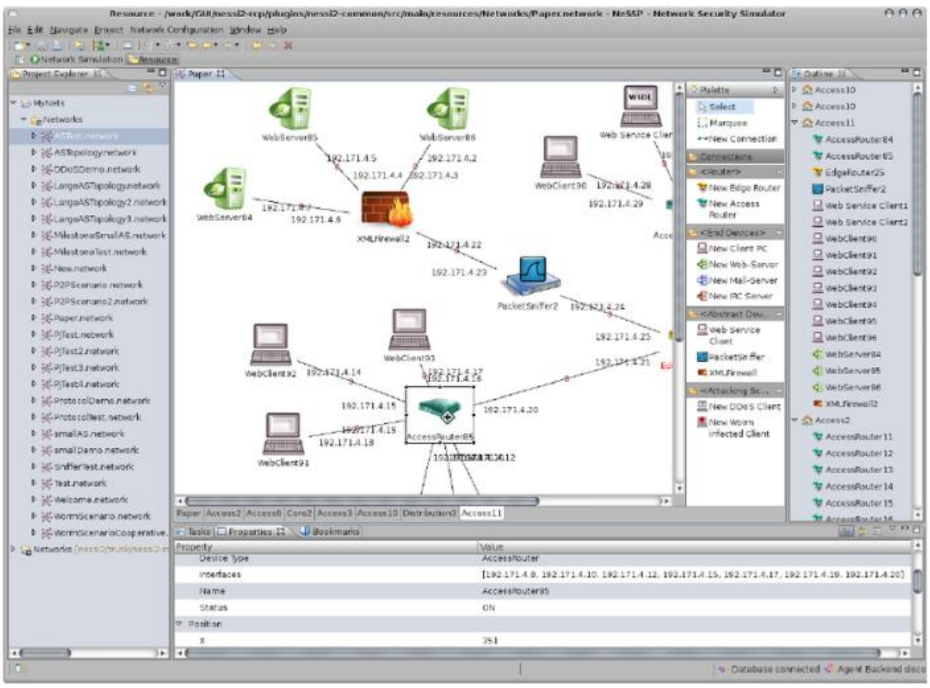

Figure 4.1: Graphical frontend of the NeSSi2 (Grunewald, Lutzernberger, Chinnow, Bye, Bsufka & Albayrak, 2011)

The architecture in Figure 4.1 shows the frontend of the NeSSi2 that permits the creation and edition of the necessary network simulation components (Grunewald, Lutzernberger, Chinnow, Bye, Bsufka & Albayrak, 2011). The backend carries out diverse agent roles tasks of parallel simulation execution. The resultant database stores simulation results based on the configurations specified in the GUI during the simulation process (Grunewald, Lutzernberger, Chinnow, Bye, Bsufka & Albayrak, 2011).

4.2.2 NS-3

The NS-3 is an open-source software application publicly available for use, research, and development (Valdiviezo, 2014). The NS-3 tool is licensed under the GNU GPLv2 licence; while developed in C++ and built as a library which can either be statically or dynamically integrated with the C++ main programme (Valdiviezo, 2014). The NS-3 is a discrete-event network simulator and internally represents simulation durations as 64-bit signed integers. The limitation of NS-3 is that it still suffers from deficiency of credibility (Kabir, Islam, Hossain & Hossain, 2014), hence, it is not utilized in this study. Figure 4.2 is the graphical interface of the NS-3

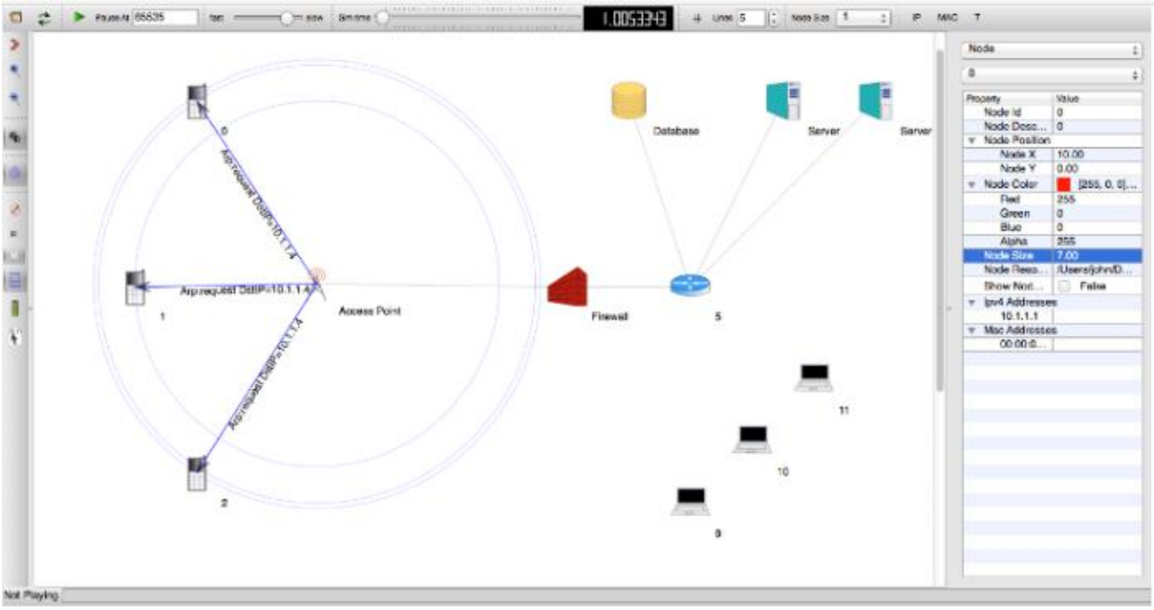

Figure 4.2: NS-3 graphical user interface (Qayyum & Ejaz, 2020)

4.2.3 OPNET

Optimized network engineering tools (OPNET) is a discrete event simulation tool that is very helpful while working with a complex network consisting of many devices with traffic streams (Kabir, Islam, Hossain & Hossain, 2014; Prokkola, 2006). OPNET can also be considered where a slight change is critical. OPNET is developed on C++ and C development language, and comprises a high-level user interface. The hierarchical structure model is structured in three domains, namely: the network, the node and the process domain. One of the best qualities of the OPNET simulation tool is that it has a rapid discrete event simulation engine (Islam & Hossain, 2014). The limitation of OPNET is its inability to allow many nodes in a single connected device (Kabir, Islam, Hossain & Hossain, 2014). Hence, it is not utilized for this study. Figure 4.3 is the graphical interface representation of OPNET.

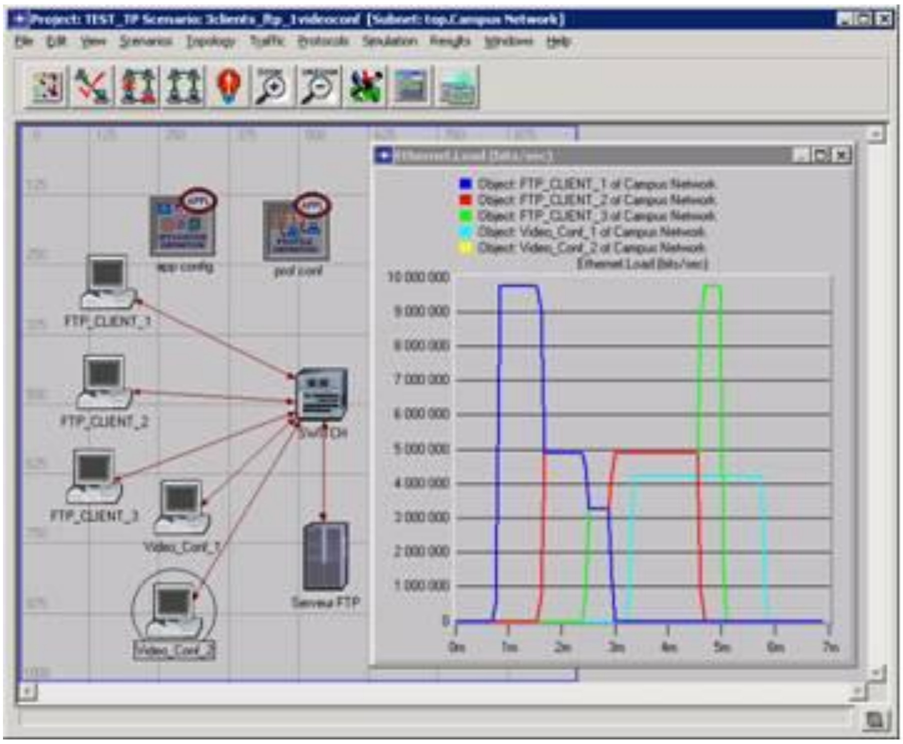

Figure 4.3: OPNET graphical user interface (Pan & Jain, 2008)

4.2.4 OMNeT++

OMNeT++ is a discrete event simulation tool that offers message exchange between elements simulated, trial visualization, interaction with the handler, and debugging the states of objects and classifications of model events (Kotenko, Konovalov & Shorov, 2010). OMNeT++ offers a special language to define the connection between elements simulated and to permit trial parameters. The individual participants involved in message exchange are defined as C++ classes. OMNeT++ does not require conversion of configuration scripts into binary code (Kotenko et al., 2010). OMNeT++ combines low- and high-level languages in order to provide high performance with adequate flexible configuration mechanism required when conducting numerous trials. OMNeT++ message processing is achieved per C++ language and characterized by low overheads. The limitation of the OMNeT++ simulation tool is that it lacks quality analysis and management of the performance (Kabir, Islam, Hossain & Hossain, 2014). Hence, it is not utilized for this study. Figure 4.4 is the graphical interface representation of OMNeT++.

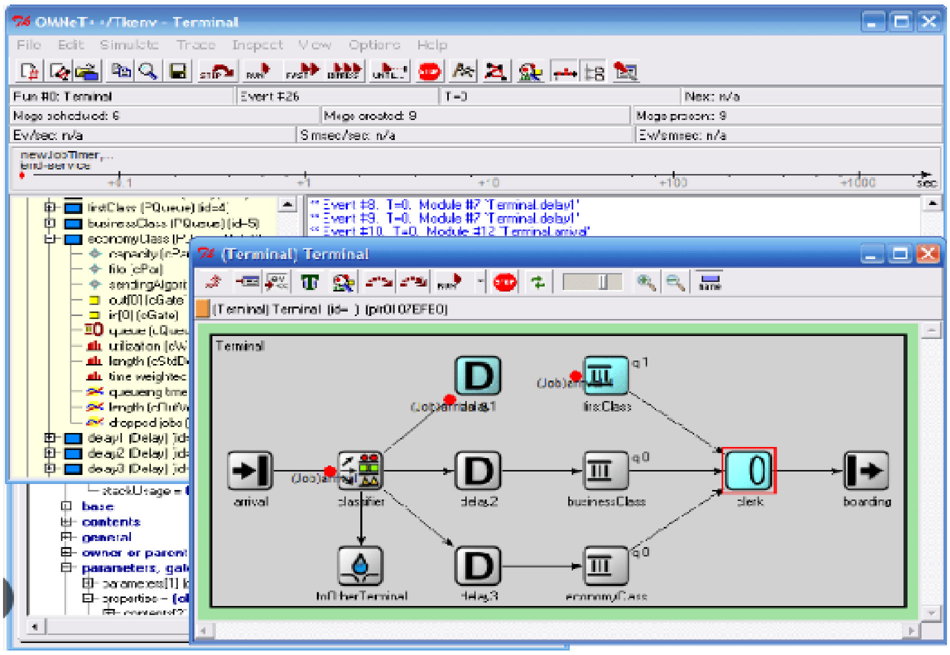

Figure 4.4: OMNeT++ graphical user interface (Bachmeier, Jaeger & Holzinger, 2020)

4.2.5 MatLab

MatLab is a network simulation tool that is easier to model than any other simulation tool (Bala & Osais, 2013). MatLab application version R2017a version 8.0.0.783, released in 2017, is a complete package with all classes such as StateFlow, SimEvents, and Simulink (Bala & Osais, 2013). MatLab offers a complete enhanced development environment for creating sophisticated experiments and user interfaces to perform algorithms and models, in order to visualize and explore results. MatLab user interface is presented in Figure 4.5.

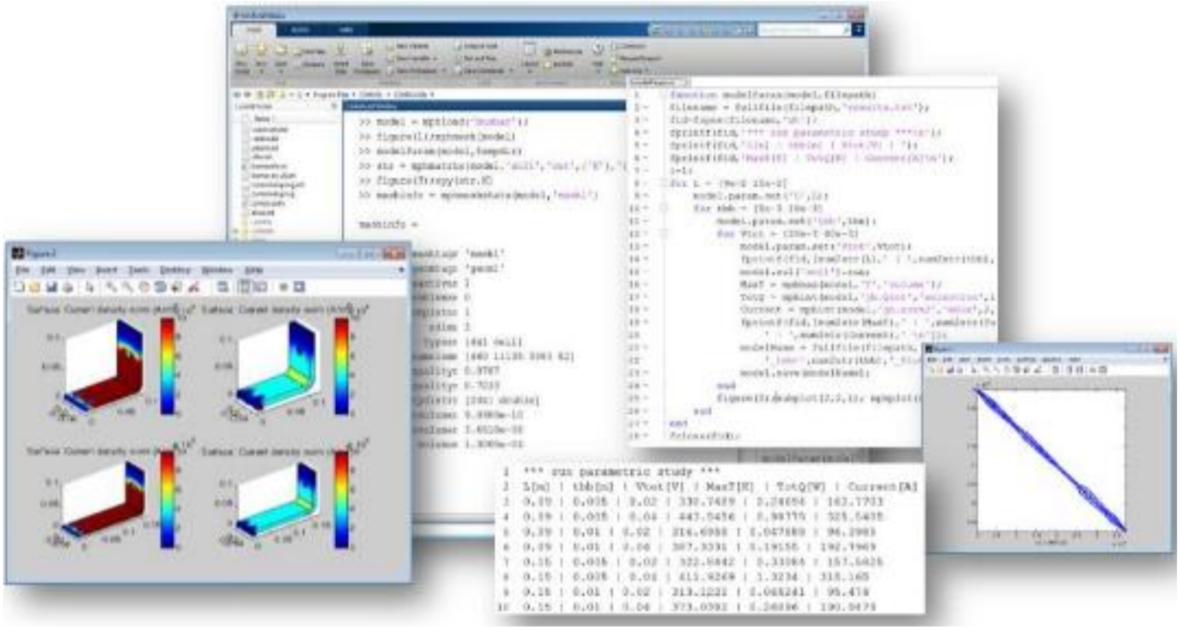

Figure 4.5: MatLab application window

4.2.6 Overview of Simulation Tools Comparison

The summary of the comparison between simulations tools discussed in this study, namely: NeSSi2, NS-3, Net Stress, OMNeT++ and MatLab, are being presented in Table IV.

Table IV: The Comparison of Simulation Tools

Features	NeSSi2	NS-3	OPNET	OMNeT++	MatLab

Language	JAIC	C++ and Python	C++ and C	C++	MatLab
Licence type	Open source	Open source	Closed source	Open source	Closed source
GUI Supported	Yes	Yes	Yes	Yes	Yes
OS Supported	Windows 7, Mac OS	Windows Vista, Windows 7 and XP, Liux, Mac OS	Windows NT, Windows XP and Windows Vista	Linux, Mac OS, Windows XP or later	Windows, Mac OS, Linux
Ease of Use	Easy	Hard	Easy	Easy	Easy
Availability Cost	Free of cost	Free of cost	costly	Free of cost	Costly
Documentation Support	Good	Excellent	Good	Excellent	Excellent
Scalability	Highly-scalable	Limited	Highly-scalable	Scalable	Highly-scalable
Simulation Event Type	Discrete event	Discrete event	Discrete event	Discrete event	Discrete event

4.3 MatLab Overview

MatLab stands for Matrix Laboratory (Naim, Zaini, Sarnin & Yaácob, 2017). MatLab is a high-performance simulation tool utilized by technical computing. Its best quality is its integration of numerical computation, programming, and the visualization environment (Naim, Zaini, Sarnin & Yaácob, 2017; Houcque, 2005). MatLab offers a unique built-in routine that supports a wide variety of computations. Through its modern design it offers an easy-to-use graphical interface that provides the visualization of results as rapidly as possible (Houcque, 2005; Naim, Zaini, Sarnin & Yaácob, 2017).

4.3.1 MatLab Software

In order to simulate DDoS attacks in the network in real time, a MatLab software application version R2017a is installed in a Windows 10 enterprise machine that runs a 2.60GHz processor with 8192MB RAM. MatLab software has a programming environment that is C-like (Takanashi, Adachi, 2010). Table V provides MatLab's unique features.

Table V: Three Unique Features of MatLab

Feature	**Description**
Storage allocation is automatic	Assigned variables in Matlab must be declared before assigning them.
Variable arguments lists with functions	Collection of functions is large in MatLab. The functions collect zero or more input argument and return zero or more output arguments.
Arithmetic and complex arrays	The data type used is the multi-dimensional array of complex numbers. The special cases are scalars, vectors, and matrices. Floating-point arithmetic is used to perform all computation in MatLab; and when the data is complex, a complex arithmetic is used by default.

4.3.2 MatLab Classes and Objects

MatLab has classes and objects that enable you to add new operations and data types. The class describes a variable structure and specifies the type of the function and operations that can be applied to the variable. An object is a class instance. In the MatLab environment you can add classes by specifying the structure which provides data storage of an object; and constructing a class directory storing M-files which operate on the object. The M-files store the methods for the class. The architecture in Figure 4.6 represents fourteen fundamental data types in MatLab.

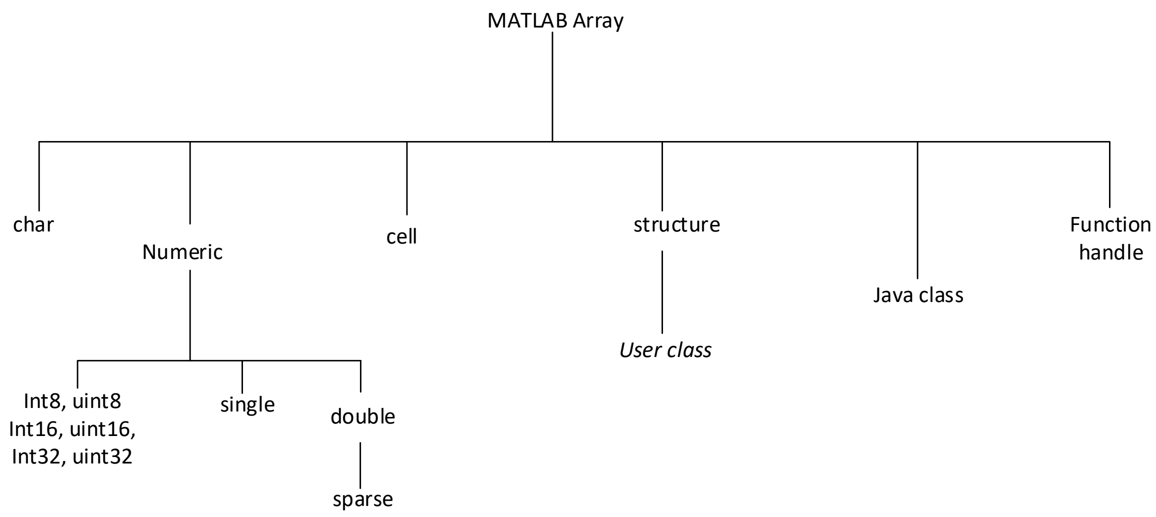

Figure 4.6: MatLab data class architecture

The architecture in Figure 4.6 demonstrate the user class that inherits from the structure class. Designing a new class in MatLab depends on the choice of the data structure that should be utilized by the class. The MatLab structure stores objects.

4.4 RTNTAD Algorithm Simulation Details

In order to demonstrate that the designed RTNTAD algorithm can improve DDoS attack detection accuracy in a Cloud Computing network, simulation is performed. The machine runs a Windows 10 enterprise operating system with a

2.60GHz processor and 8192MB RAM. The study employs the Canadian Institute for Cyber Security Intrusion Detection System dataset (CICIDS 2017) which holds the latest attributes of the recent DDoS attacks (Abdulrahman & Ibrahem, 2018; Zhang et al., 2018). The CICIDS 2017 dataset holds more than eighty intrusion-based classified network-traffic features (Abdulrahman & Ibrahem, 2018; Zhang et al., 2018). Due to the over eighty network-traffic features, it becomes impractical to calculate all these feature parameters for each flow in a high-speed network environment in avoiding computational complexity (Zhang et al., 2018). The data in Table VI illustrates the DDoS attack model.

Table VI: DDoS attack features list (Zhang et al., 2018)

DDoS Attack Features	**Feature Description**	
	src_IP	Source IP
Basic Features	src_port	Source Port
	dst_IP	Destination IP
	dst_port	Destination Port
	Pro	Layer 4 protocol (TCP, UDP, ICMP, etc.)
Time Features	duration	Flow duration
	flowiat	Flow inter-arrival time, i.e., the time between two packets sent in either direction (mean, min, max, std)
	bp_psec	Backward packets per second
Packet Features	fp_len	Forward packet length (tot, min, max)
	bp_len	Backward packet length (total, mean, min, max)
	ap_size	Average packet size
Byte	iw_bytes	Initial window bytes (forward, backward)
Features	sf_bytes	Subflow bytes (forward, backward)

The data in table VI illustrates the model of DDoS attacks and includes both the continuous data and discrete data. The DDoS attacks features such as number of bytes, duration, length; and the number of packets are calculated (Abdulrahman & Ibrahem, 2018). The flow has the following labelled, namely, src_IP, src_port, dst_IP, dst_port, Pro, duration, and flowiat for each of the network traffic features (Abdulrahman & Ibrahem, 2018). The RTNTAD algorithm has a higher detection and classification capability for accuracy and runtime in a high-speed network environment in real time.

4.4.1 Artificial Neural Network Architecture

In 1943 Walter Pitts and Warren McCulloch proposed the initial artificial neuron network (ANN) which was the Linear Threshold Unit or Threshold Logic Unit (TLU) (Akhtar, 2016). The model was a systematic model in the brain of the nerve network. In their model the neuron was representing the binary processing unit. The neuron will operate the firing or not firing its output signal which was guided by activation function or threshold logic (Akhtar, 2016). The neuron will have a number of inputs while only computing a single output. For each input there will be a weight and in order to compute if the neuron can fire the output signal, the sum of the multiplication between each input and weight is computed (Akhtar, 2016). The sum value from the beginning is used to compute the activation function as shown in Figure 4.7.

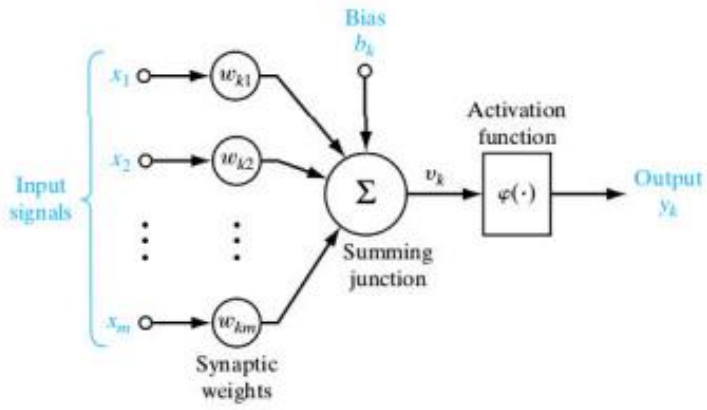

Figure 4.7 Artificial Neuron Network Architecture (Akhtar, 2016)

The architecture of the ANN in Figure 4.7 represent the single input neuron. The input scalar x is multiplied by weight w to form wp, the input 1, is multiplied by a bais b. The output net then goes in to the activation function φ provide the neuron output scalar a.

4.4.2 Experimental Set-up

An ANN for a DDoS attacks simulation model is set-up in MatLab application version R2017a. The experimental set-up input layer has 41 input neurons that describe 41 features in the CICIDS 2017 dataset (Mukhopadhayay, Polle & Naskar, 2014). The study utilised 12 neurons for the hidden layer together with bias b and weight w for classifying the DDoS attacks, as shown in Figure 4.8.

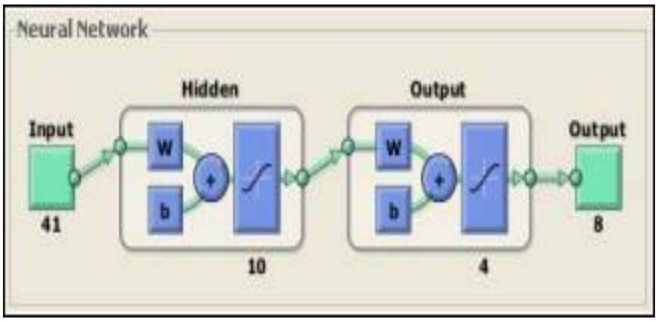

Figure 4.8: MatLab ANN experimental set-up (Mukhopadhayay et al., 2014)

4.5 Chapter Summary

This chapter focused on the system implementation using a MatLab simulation tool to simulate the results of the algorithm that the study has designed. The MatLab simulation tool is compared with other simulation tools available online. Thereafter, the research study presents reasons for MatLab being selected over other available simulation tools.

The next chapter will focus on the results and the analysis of the simulated RTNTAD algorithm.

Chapter 5: Results and Discussion

5.1 Introduction

This study performed simulation experiments in order to assess the effectiveness of the designed RTNTAD algorithm. The proposed RTNTAD algorithm was developed to improve the detection accuracy of DDoS attacks in Cloud Computing. Simulation is another way of modelling, as well as a preferred approach to understanding how the proposed algorithm will perform in the real-world network environment (Stancic, Seljan, Cetinic & Sankovic, 2007). The employment of simulation addresses the natural method of "doing while learning" (Stancic et al., 2007). Computer simulation is the discipline of planning a model of a real hypothetical framework, executing it on a computerized environment in order to analyse the execution outcome (Fishwick, 1995; Stancic et al., 2007). Hence, a simulation was used in this study.

The rest of the chapter is structured as follows: In Section 5.2, the study presented the computer simulation evaluation. In Section 5.3, the study presents the results and analysis of the simulation. In Section 5.4, the chapter is concluded.

5.2 Computer Simulation Evaluation

In order to access the effectiveness of the proposed RTNTAD algorithm the study performed evaluations through MatLab simulator version 2017a. The CICIDS 2007 dataset was utilised in MatLab in order to train the ANN function and the target value. The ANN is useful as it is a viable method of increasing the performance of the IDS framework which depends on training and learning processes that can be utilized for detection of anomalies and abuse in the network (Norwahidayah, Farahah, Amirah, Liyana & Suhana, 2021). ANN has two classes of learning process, namely: a supervised training approach that trains the input and output patterns; as well as the unsupervised training approach that only trains the input pattern (Norwahidayah, Farahah, Amirah, Liyana & Suhana, 2021). This study utilized the supervised training approach.

During the experimental set-up, the input layer has 41 input neurons that describe 41 features in the CICIDS 2017 dataset. The study utilized 12 neurons for the hidden layer together with bias *b* and weight *w* for classifying the DDoS attacks. At the point at which the network traffic arrives in the network, which is the input, the output will be generated. The hidden layer is found where the weight *w* can be modified to gain the accurate output.

5.2.1 Collection and processing of CICIDS 2017 dataset

The initial step is collection and processing the dataset to be utilised for training ANN. The training will stop while there is a decrease in training errors and performance start to deteriorate. The dataset for training is labelled as benign and other types of attacks. To perform the first training, the study changes the training dataset through labelling each row as Benign and attack. The first training is performed while other feature of the dataset are normalised. The abnormalities and the normalities are classified by through training. In this study, only 4 critical dataset features are considered; and 41 features are utilized to train ANN. The available attributes in the dataset are changed over to a double-data type for the attributes to be compatible with the ANN toolbox of MatLab, on the grounds that it only support data in integer format. This includes a protocol types feature having values such as TCP is 2 and UDP is 3.

For testing purpose the study processes each row of the dataset by node 1 and node 2 which is added in the output of node 1 and node 2 to the features of each row. Finally the study process the output results by node 3 that is classified as benign or a specific attack type. The data in table VII shows the parameter settings that are utilized in this study.

Table VII: Dataset Parameters Settings

Parameter	Values
Number of Training Samples	41
Mass	15
Maximum of Iteration	50
Inertia Weight	2
Minimum Weight	0.5
Maximum Weight	0.9
Objective Function	Maximization

5.2.2 Process Flow

The CICIDS 2017 dataset is used to train the ANN that provides the accuracy output of the system detection, in order to plot and observe it as a result of the system. Figure 5.1 demonstrates the flow of the process.

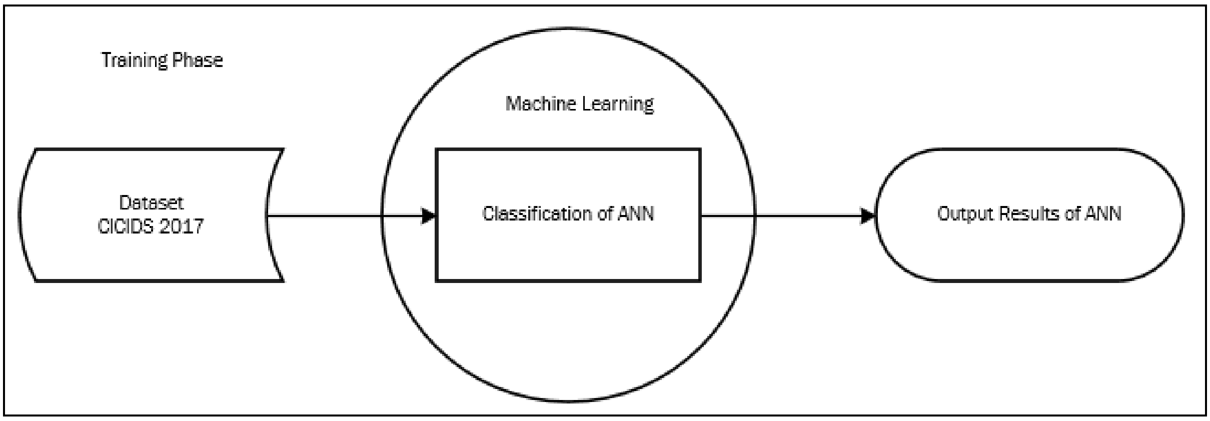

Figure 5.1: Flow of process

5.3 Simulation Results and Analysis

The details of the simulation were provided in Chapter 4, Section 4.4; and further elaborated in Section 5.2 in order to access the effectiveness of the designed RTNTAD algorithm. This study presented the simulation results based on an average of 30 simulations, focusing mainly on the following parameters:

The first parameter is the detection rate, which is the effectiveness of the accuracy in detection. The second parameter is the classification of an attack, (CoA) which is the way to detect whether the traffic contains malicious traffic and can be classified as an attack with a notification signal raised. The third parameter is the network connectivity which also affects performance of the algorithms. The final parameter is the number of false positives (AFP) which appear if the algorithm wrongful classifies the traffic as an attack.

5.3.1 The detection rate

The detection rate of the proposed RTNTAD algorithm was compared with that of the CUSUM, EWMA, and the Naïve Bayes algorithms, as described in Figure 5.1. The results were analysed when the amount of traffic was set from 4 to 40 in order to observe the performance of the detection accuracy for each algorithm. The proposed RTNTAD algorithm is not constrained by the amount of new traffic coming in to the network, as shown in Figure 5.2.

Figure 5.2: Detection rate comparison

The detection rate of the RTNTAD algorithm has shown good results of 99.2% compared with the CUSUM algorithm that came second with 98.1%, EWMA with 80%, and Naïve Bayes with 79.9%. The observation is conducted in real time with a change in traffic load to test large amounts of traffic capability, and at high speed. The RTNTAD has the capability of detecting a shift in the values of the network traffic; and is easy to deploy in real time.

5.3.2 The classification of DDoS attacks

The classification of network traffic was conducted in order to determine whether the coming traffic contains malicious items. This was observed in 40 iterations. The RTNATD came first in the detection process, as shown in Figure 5.3.

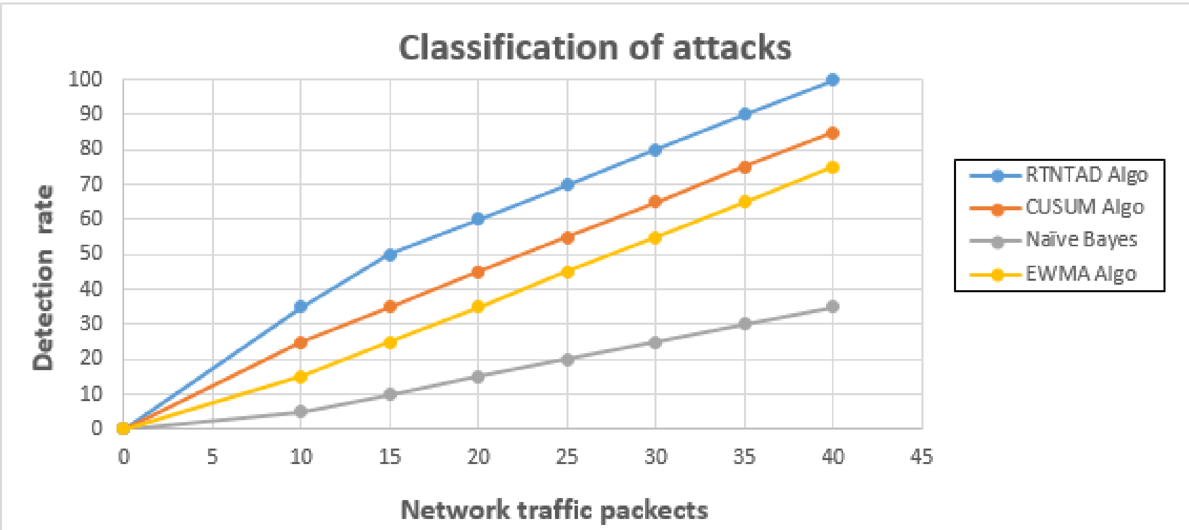

Figure 5.3: Traffic classification comparison

As reflected in Figure 5.3, the RTNTAD algorithm has shown good performance over other algorithms at 99.5%. The CUSUM algorithm came second at 85.6% below the RTNTAD algorithm. The EWMA scored 70%, and Naïve Bayes came last with an average of 39%. The RTNTAD has combined the capability of unknown detection with that of the EWMA which uses current and previous data traffic to enhance classification. Therefore, the classification of the RTNTAD becomes more effective than all other algorithms.

5.3.3 Network Connectivity

Network connectivity issues can affect performance. Therefore a simulation was conducted to determine the efficiency of the performance of the algorithms through network connectivity throughput limitations, as shown in Figure 5.4. The connectivity time out was analysed when the throughput changes from 5 to 35, and the threshold delay was set to 15.

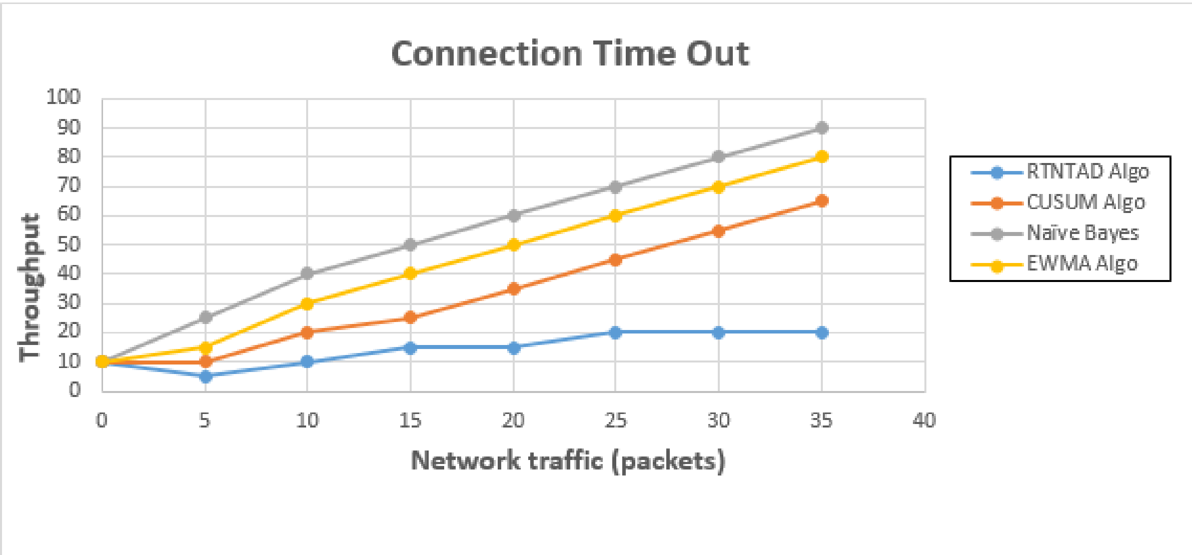

Figure 5.4: Connection time out

The presented results can lead to the interpretation that the proposed RTNTAD algorithm performed better under various network limitations by 0.9%. The amount of connection time out was minimal compared with the other three algorithms 5.9% for CUSUM, 7.9% for EWMA and 8.9% for Naïve Bayes. This includes the speed of the network, the traffic volume per second, and the session time out. Therefore the RTNTAD has improved the memory control as a result of being able to handle more than one session at a time with large amount of traffic, unlike each one managed individually.

5.3.4 Number of false positives

The number of false positives comes when the algorithm wrongly classifies traffic as malicious attacks. The results were analysed and presented as shown in Figure 5.5. Evaluations were conducted to determine whether the proposed algorithm is capable of improving the detection of false positives.

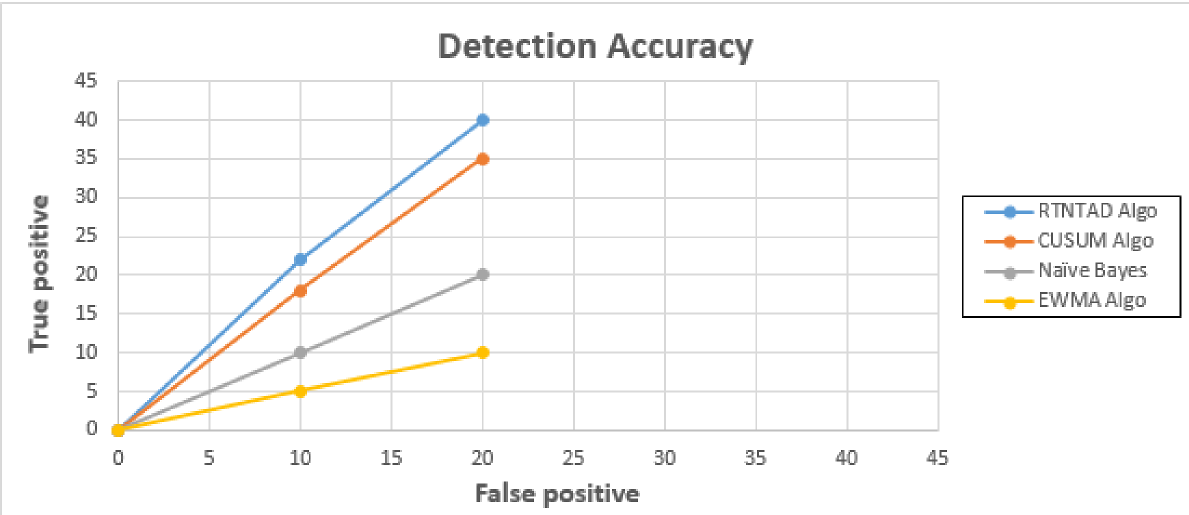

Figure 5.5: Amount of false positive detection

Based on the results as shown in Figure 5.5, it is safe to conclude that the proposed RTNTAD algorithm improved the detection of false positives. The RTNTAD algorithm achieved over 80% true positives when the simulation was repeated 30 times, with less than 18% false positives while the other algorithm was still showing a high number of false positives. The capability of the RTNTAD algorithm to use the current and previous data traffic has increased the classification feature, unlike when the algorithms are disjointed. The summary of the simulation results is illustrated in Table VIII.

Table VIII: Simulation Results

Parameter	RTNTAD	CUSUM	EWMA	Naive Bayes
Detection rate (%)	99.2	98.1	80	79.9
Classification of Attacks (%)	99.5	85.6	70	39
Network Connectivity Timeout (%)	0.9	5.9	7.9	8.9
False Positive (%)	15	39	47	52

5.4 Chapter Summary

This chapter presented the simulation results of the study as described in 4.4 of Chapter 4 and 5.2 of Chapter 5. The proposed RTNTAD algorithm has proved more effective over the CUSUM, the Naïve Bayes and EWMA algorithms in all the parameters that were used in the simulations for the algorithms.

The next chapter will focus on the executive conclusion of the study which will include the challenges and the future work.

Chapter 6: Conclusion and Future Work

6.1 Introduction

This research study proposed a RTNTAD algorithm by integrating the adaptive CUSUM, EWMA and the Naïve Bayes algorithms in order to improve the detection of DDoS attacks in real time in the network system. The proposed RTNTAD algorithm is capable of detecting DDoS attacks as soon as they enter the network, reducing the amount of false positive detections. Thus, the proposed RTNTAD algorithm has improved the detection accuracy and QoS.

The rest of the chapter is structured as follows: In Section 6.2, the study focuses on how the objectives of the study were achieved by providing more information for each objective. In Section 6.3, the study provides the summary of the whole study. In Section 6.4, the study discusses the challenges experienced during the research study. In Section 6.5, the study discusses the future work; and lastly, the conclusion of the chapter is presented in Section 6.6.

6.2 Objectives Achieved

The outcome of the study led to the interpretation that all the objectives of the study stipulated in Chapter 1 Section 1.5 have been met, as described below:

6.2.1 To identify and analyse the existing algorithms used to detect UDP and TCP SYN flooding attacks in Cloud Computing.

In Chapter 2, Section 2.6, the study discussed the related work which has assisted in achieving this objective through the discussion of various schemes that other researchers have proposed and implemented previously. The review of literature has indicated that various schemes such as the data mining algorithm, the C4.5 algorithm, the signature detection scheme, the MLP and the RF algorithm were developed in order to improve the detection of DDoS attacks in a network system.

The data-mining algorithm was divided into four segments: the pre-processing, anomaly detection, and simulation of the results for detection of DDoS attacks in the network. The C4.5 and the signature-detection algorithms were used to create a decision tree in order to provide effective and automatic detection of DDoS attacks. The MLP and RF were used to detect the DDoS attacks in real time.

The limitation of the data-mining algorithm is the dataset used which is not in real time. The C4.5 and signature-detection algorithm limitation is that it cannot detect unknown attacks and does not use a real-time dataset. The drawback of the MLP and RF algorithms is that they are unable to examine every packet when large amounts of traffic are transmitted at high speed.

6.2.2 To design and implement an enhanced security algorithm that detects UDP and TCP SYN flooding attacks in real time.

In Chapter 3, Section 3.4, the study proposed a RTNTAD algorithm by integrating the adaptive CUSUM, the Naïve Bayes and the EWMA algorithms to detect the DDoS attacks in real time. The algorithm has proven to be effective over existing algorithms.

6.2.3 To measure the effectiveness of the proposed security algorithm against existing algorithms.

In Chapter 5, Section 5.3, the study presented the simulation results and analysis to measure the effectiveness of the proposed algorithm against existing algorithms. The results showed that the proposed RTNTAD algorithm outperformed the existing algorithms with an average DDoS detection rate of 99.2%, classification of malicious traffic of 99.8%, connectivity time out of 0.9%, and 80 % of true positives.

6.3 Study Summary

This study proposed a RTNTAD algorithm which has effectively improved the detection of DDoS attacks in the network system. The algorithm is capable of detecting the attacks as soon as they arrive in the network. The proposed RTNTAD algorithm was not negatively impacted by network connectivity issues. As a result, it had minimal connectivity time outs. Thus, the proposed RTNTAD algorithm resolved the challenges of detecting DDoS attacks when a large amount of traffic is being transmitted at high speed; and as a result, QoS was improved.

6.4 Challenges Experienced during the Study

The MatLab simulation tool requires sufficient knowledge; and as a result, more time was required to enhance the knowledge in order to simulate the results. Thus, I had to work late at night, early in mornings, and on weekends in order to simulate the proposed algorithm using MatLab.

6.5 Future Work

QoS is one aspect that is key in any service offering, ensuring that the service being offered meets the needs of the customer based on the service level agreement (SLA). Much has to be achieved around latency which will further improve QoS; and therefore, the study must focus on latency for future work.

6.6 Chapter Summary

This chapter presented the executive conclusion of the study. The chapter provides a summary of the whole study, including details of how the objectives of the study were met. This chapter introduced latency as a topic in future work, including the challenges encountered during the study. The proposed RTNTAD algorithm has proved to be efficient over the adaptive CUSUM, the EWMA and Naïve Bayes algorithm, by meeting all objectives of the study. Therefore the study objective was successfully accomplished.

References

Abbas, S.A. Almhanna, M.S., 2021. Distributed Denial of Service Attacks Detation System by Machine Learning Based on Dimensionality Reduction. Journal of Physics: Conference Series.

Abdulrahman, A.A. & Ibrahem, M. K., 2018. Evaluation of DDoS Attacks Detection in a CICIDS2017 Dataset Based on Classification Algorithms. Iraqi Journal of Information and Communications Technology ((IJICT), Vol. 1, No. 3, pp. 49-55.

Ahmed, M. & Hossain, M. A., 2014. Cloud Computing and Security Issues in the Cloud. International Journal of Network Security & Its Applications (IJNSA), Vol. 6, No.1, pp 25-36.

Akthar, S., 2016. A Study on Neural Network Architectures. Computer Engineering and Intelligent Systems, Vol.7, No.9, pp.1-7.

Akyildiz, F., SU, W., Sankarasubramaniam, Y. & Cayirci, E., 2002. Wireless sensor networks: a survey. Computer Networks, Vol. 38, No. 4, pp. 393–422.

Alanazi, S., Karuppayah, S., Anbar, M. & Al-Ani, A. K., 2019. Detection Techniques for DDoS Attacks in Cloud Environment: Review Paper. Intelligent and Interactive Computing, pp. 337-354.

Alkasassbeh, M., Hassanat, A. B., Almseidin, M. & Naymat, G. A., 2016. Detecting Distributed Denial of Service Attacks Using Data Mining Techniques. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 7, No.1, pp. 436-445.

Alomari, E., Gupta, B. B. & Karuppayah, S., 2012. Botinet-based Distributed Denial of Service (DDoS) Attacks on Web Servers: Classification and Art. International Journal of Computer Applications Vol. 49, No. 7, pp. 24-32.

Alzahrani, S. & Hong, L., 2018. A Survey of Cloud Computing Detection Techniques against DDoS Attacks. Journal of Information Security, Vol. 9, No. 1, pp. 45-69.

Amjad, A., Alyas, T., Farooq, U. & Tariq, M. A., 2019. Detection and Mitigation of DDoS attack in Cloud Computing using machine learning algorithm. EAI Endorsed Transactions on Scalable Information Systems, Vol. 6, No. 23, pp 1-8.

Ankita, P. & Fenil, K., (2015). Survey on DDoS Attack Detection and Prevention in Clouds. International Journal of Engineering Technology , Management , and Applied Sciences, Vol. 3, No. 1, pp 43-47.

Anon., n.d. Exploit Database. ImageMagick 6.9.3-9 / 7.0.1-0//Multiple Vulnerabilities (ImageTragick). [Online] Available at: https://www.exploit-db.com/exploits/39767/ [Accessed 26 February 2019].

Arora, S. & Dalal, S., 2019. DDoS Attacks Simulation in Cloud Computing Environment. Issues in Information System, International Journal of Innovative Technology and Exploring Engineering (IJITEE), Vol. 9, No. 1, pp. 414-417.

Astani, M. & Ready, K. J., 2016. Trends and Preventive Strategies for Mitigating Cybersecurity Breaches in organizations. Issues in Information System, Vol. 17, No. 2, pp. 208-214.

Awan, M.J., Farooq, U., Babar, H.M.A., Yasin, A., Nobanee, H., Hussain, M., Hakeem, O. & Zain, A.M., 2021. Real-Time DDoS Attack Detection System Using Big Data Approach. Sustainability 2021, Vol. 13, No. 19, pp. 1-19.

Babic, I., Miljkovic, A., Cabarkapa, M., Nikolic, V., Dordevic, A., Randelovic, M. & Randelovic, D., 2021. Triple Modular Redundancy Optimization for Threshold Determination in Intrusion Detection Systems. Symmetry 2021, Vol. 13, No. 557, pp. 1-20.

Bachmeier, S., Jaeger, B. & Holzinger, K., 2020. Network Simulation with OMNet++. Seminar IITM, Vol. 20, No. 08, pp.37-41.

Bala, A. & Osais, Y., 2013. Modelling and simulation of DDOS Attack using SimEvents. International Journal of Scientific Research in Network Security and Communication, Vol.1, No. 1, pp 2321–3256

Barki, L., Shidling, A., Meti, N., Narayan, D.G., Mulla, M.M., 2016. Detection of Distributed Denial of Service Attacks in Software Defined Networks. Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1-6.

BBC, 2016. ‘‘Smart’’ home devices used as weapons in website attack. Available at: https://www.bbc.com/news/technology-37738823

Bharot, N., Verma, P., Suraparaju, V. & Gupta, S., 2016. Mitigating Distributed Denial of Service Attack in Cloud Computing Environment using Threshold based Technique. Indian Journal of Science and Technology, Vol. 9, No. 38, pp. 1-7.

Bhaya, W. & EbadyManaa, M., 2017. DDoS attack detection approach using an efficient cluster analysis in large data scale. Proceedings of 2017 Annual Conference on New Trends in Information &Communications Technology Applications (NTICT), pp. 168-173.

Bindra, N. & Sood, M., 2019. Detecting DDoS Attacks Using Machine Learning Techniques and Contemporary Intrusion Detection Dataset. Automatic Control and Computer Sciences, Vol. 53, No. 5, pp. 419–428.

Bonguet, A. & Bellaiche, M., 2017. A Survey of Denial-of-Service and Distributed Denial of Service Attacks and Defenses in Cloud Computing. Computer Engineering and Engineering Software, Vol. 9, No. 43, pp. 1-19.

Chavan, S., Shah, K., Dave, N., Mukherjee, S., Abraham, A. & Sanyal, S., 2004. Adaptive neuro-fuzzy intrusion detection systems. IEEE international conference on information technology, Vol. 1, No. 10, pp. 70-74.

Chirag , N.M., Dhiren, R.P., Avi, P. & Muttukrishman, R., 2012. Integrating Signature Apriori based Network Intrusion DetectionSystem (NIDS) in Cloud Computing. 2nd International Conference on Communication, Computing & Security, Vol. 6, No. 1, pp. 905-912.

David, J. & Thomas, C., 2015. DDoS Attack Detection using Fast Entropy Approach on Flow Based Network Traffic. 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15),Vol. 50, No. 15, pp. 30-36.

Dong, S., Abbas, K. & Jain, R., 2019. A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing. IEEE Access, Vol. 7, No. 10, pp. 80813-80828.

Douligeris, C. & Mitrokotsa, A., 2004. DDOS Attacks and Defense Mechanisms: a Classification. IEEE conference paper, Vol. 44, No. 5, pp. 643-66.

Fishwick, P.A., 1995. Computer Simulation: The Art and Science of Digital World Construction. pp. 1-9.

Girma, A. & Wang, P., 2018. An Efficient Hybrid Model For Detecting Distributed Denial of Service (DDoS) Attacks in Cloud Computing Using Multivariate Correlation and Data Mining Clustering Techniques. Issues in Information Systems, Vol. 19, No. 2, pp. 1-12.

Girma, A., Abayomi, K. & Garuba, M., 2016. The Design, Data Flow Architecture, and Methodologies for a Newly Researched Comprehensive Hybrid Model for the Detection of DDoS Attacks on Cloud Computing Environment. Springer International Publishing Switzerland, Vol. 448, No.10, pp. 377-387.

Gkounis, D., 2014. Cross-domain DoS link-coding attack detection and mitigation using SDN principles. MSc Th., ETH, pp. 1-49.

Gong, R. H., Zulkernine, M. & Abolmaesumi, P., 2005. A software implementation of a genetic algorithm based approach to network intrusion detection. Sixth International Conference on Software Engineering, Vol.9, No. 10, pp. 246-253.

Grunewald, D., Lutzernberger, M.,Chinnow, J., Bye, R., Bsufka, K. & Albayrak, S., 2011. Agent-based Network Security Simulation (Demonstration), International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org), Vol. 2, No. 6, pp. 1325-1326.

Gupta, B. B., Joshi, C. R. & Misra, M., 2011. ANN Based Scheme to Predict Number of Zombies in a DDoS Attack. International Journal of Network Security, Vol. 13, No. 3, pp. 216-225.

Gupta, B. B., Joshi, R. C. & Misra, M., 2008. An ISP level solution to combat DDoS Attachs Using Combined Statistical Based Approach. International Journal of Information Assurance and Security (JIAS), Vol. 3, No. 2, pp. 102-110.

Gurina, A. & Eliseev, V., 2019. Anomaly-Based Method for Detecting Multiple Classes of Network Attacks. information, Vol. 10, pp. 2-84.

Hameed, S & Ali, U., 2016. Efficacy of Live DDoS Detection with Hadoop. IEEE/IFIP Network Operations and Management Symposium, pp. 488-494.

Hirsch, B., Konnerth, T. & Hebler, A., 2009. Merging Agents and Services — the JIAC Agent Platform, Multi-Agent Programming, pp. 159-185.

Houque, D., 2005. Introduction to MATLAB for Engineering Students, School of Engineering and Applied Science, pp. 1-64

Joshi, R. C., Gupta, B. B. & Misra, M., 2009. An Efficient Analytical Solution to Thwart DDoS Attacks in Public Domain. International Conference on Advances in Computing, Communication and Control (ICAC3'09), p. 504-509.

Kabir, M.H., Islam, S., Hossain, S. & Hossain, M., 2014. Detail Comparison of Network Simulator , International Journal of Scientific & Engineering Research, Vol. 5, No. 10, pp. 203-218.

Korad, C. et al., 2016. Detection of Distributed Denial of Service Attack with Hadoop on Live Network. International Journal of Innovative Research in Computer and Communication Engineering, pp 92-98.

Kotenko, I., Konovalov, A. & Shorov, A., 2010. Agent-Based Modeling and Simulation of Botnets and Botnet Defense. Conference on Cyber Conflict Proceedings, pp. 22-44.

Kotey, S. D., Tchao, E. T. & Gadze, J. D., 2019. On Distributed Denial of Service Current Defense Schemes. Vol. 7, No. 19, pp. 1-24.

Krishma, U. L. & Kumar, M. M. V. M., 2018. Detecting Distributed Denial-of-Service Flooding Attacks using Detection and Defense Algorithm. International Journal of Creative Research Thoughts, Vol. 6, No. 1, pp. 1495-1503.

Li, L., Zhou, J. & Xiao, N., 2007. DDoS Attack Detection Algorithms Based on Entropy Computing. Springer-Verlag Berlin Heidelberg, pp. 452-466.

Loannis, K., Dimitriou, T. & Freiling, F.C., 2007. Towards Introsion Detection in Wireless Sensor Networks, EuropeanWireless Conference, pp. 1-8.

Lonea, A.M., Popescu, D.E., Prostean, O & Tianfield, H., 2013. Evaluation of experiments on detecting distributed denial of service (DDoS) attacks in eucalyptus private cloud. Adv.Intell. Syst. Comput., vol. 167, No. 1, pp. 367–379, 2013.

Machaka, P., Bugula, A. & Nelwamondo, F. 2016. Using exponentially weighted moving average algorithm to defend against DDoS attacks, International Conference (PRASA-RobMech), pp. 1–6.

Mahjabin, T., Xiao, Y., Sun, G. & Jiang, W., 2017. A Survey of distributed denial-of-service attack, prevention, and mitigation techniques. International Journal of Distributed Sensor Networks, Vol. 13. No. 12, pp. 1-33.

Masdari, M. & Jalali, M., 2016. A survey and taxonomy of DoS attacks in Cloud Computing. Security Comm. Networks, Vol. 9, No. 16, pp. 3724-3757.

Modi, C. et al., 2013. A Survey on Security Issues and Solutions at Different Layers of Cloud Computing. The Journal of Supercomputing, Vol. 63, No. 2, pp. 561-592.

Mukhopadhayay, I., Polle, S. & Naskar, P., 2013. Simulation of Denial of Service (DoS) Attack using Matlab and Xilinx. IOSR Journal of Computer Engineering (IOSR-JCE). Vol. 16, No. 3, pp. 119-125.

Naim, N.F., Zaini, H.I.M., Sarnin, S.S. & Yaácob, N., 2017. Interactive Learning Software for Electrical Engineering Subjects Using MATLAB and its GUI, Journal of Fundamental and Applied Sciences, Vol. 9, No. 5, pp. 260-276.

Neghina, D. & Scarlat, E., 2013. Managing Information Technology security in the context of cyber crime trends. International Journal of Computers, Communications & Control (IJCCC), Vol. 8, No. 1, pp. 97-104.

Norwahidayah, S., Farahah, N.N., Amirah, A., Liyana, N. & Suhana, N., 2021. Performances of Artificial Neural Network (ANN) and Particle Swarm Optimization (PSO) Using KDD Cup '99 Dataset in Intrusion Detection System (IDS). Journal of Physics: Conference Series, pp.1-8.

Olufowobi, H., Ezeobi, U., Muhati, E., Robinson, G., Young, C., Zambreno, J. & Bloom, G., 2019. Anomaly Detection Approach Using Adaptive Cumulative Sum Algorithm for Controller Area Network . Electrical and Computer Engineering Conference, pp. 25-30.

Pan, J. & Jain, J., 2016. A Survey of Network Simulation Tools: Current Status and Future Developments, pp. 1-13.

Pieterse, H., 2021. The cyber threat landscape in South Africa: A 10-year review, The African Journal of Information and Communication (AJIC), Vol. 28, No. 28, pp. 1-21.

Prakash, A., Satish, M., Bharav, T. S. & Bhalaji, N., 2016. Detection and Mitigation of Denial of Service Attacks Using Stratified Architecture. Procedia Computer Science, Vol. 87, No.1, pp. 275-280.

Prasad, K.M., Reddy, A.R.M. & Rao, K.V., 2016. Anomaly based Real Time Prevention of under Rated App-DDOS Attacks on Web: An Experiential Metrics based Machine Learning Approach, Indian Journal of Science and Technology, Vol. 9, No. 27, pp. 1-10.

Qayyum, R. & Ejaz, H., 2020. A Comparative Study of Location Based Services Simulators, International Journal of Computer Engineering in Research Trends, Vol. 7, No. 11, pp. 1-12.

Rani, S., & Gangal, A., 2012. Security Issues of Banking Adopting The Application Cloud Computing Stratified Architecture. International Journal of Information Technology and Knowledge Management, Vol. 5, No. 2, pp. 243-246.

Sahi, A., LAI, D. & LI, Y., 2017. An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment. IEEE Access, Vol, 5. No, 1, pp. 6036-6048.

Salmen, F., Hernandes, P.R.G., Carvalho, L.F. & Proenca, M.L., 2015. Using Firefly and Genetic Metaheuristics for Anomaly Detection based on Network Flows, AICT 2015 : The Eleventh Advanced International Conference on Telecommunications, pp. 113-118.

Sen, J., 2013. Security and Privacy Issues in Cloud Computing, Innovation Labs, pp. 1-42.

Senthilnath, J., Omkar, S.N. & Mani, V., 2011. Clustering using firefly algorithm: Performance study, Indian Institute of Science, Swarm and Evolutionary Computation Vol. 1, No, 3, pp.164-171.

Shaar, F. & Efe, A., 2022. DDoS Attacks and Impacts on Various Cloud Computing Components. International Journal of Information Security Science, Vol. 7, No. 1, pp. 26-48.

Shekhar, S. & Neeraj Varshney, 2020. Enhanced Artificial Neural Network with Particle Swarm Optimization Algorithm for Detecting Distributed Denial of Service in Cloud. ISSN: 0193 - 4120, Vol. 82, No. 1, pp. 101-108.

Sklavounos, D., Edoh, A. & Plytas, M., 2017. A Statistical Approach Based on EWMA and CUSUM Control Charts for R2L Intrusion Detection. In Proceedings of the Cybersecurity and Cyberforensics Conference (CCC), pp. 25–30.

Somasundaram, A. & Meenakshi, V. S., 2021. DDoS Mitihation In Cloud Computing Environment By Dynamic Resource Scaling With Elastic Balancing. Turkish Journal of Computer and Mathematics Education, Vol. 12, No. 11, pp. 3346-3362.

Sreeram, I. & Vuppala, V. P. K., 2019. HTTP flood attack detection in application layer using machine learning metrics and bio inspired bat algorithm. Applied Computing and Informatics, Vol. 15, No. 1, pp. 59-66.

Stancic, H., Seljan, S., Centinic, A. & Sankovic, D., 2007. Simulation Models in Education, Digital Information and Heritage, pp. 469-481.

Sureh, M. & Anitha, R., 2011. Evaluating Machine Learning Algorithms for Detecting DDoS Attacks, International Conference on Network Security and Applications, Vol. 196, No. 1, pp. 441-452.

Tatanashi, H. & Adachi, S., 2010. Graphical User Interfaceof System Identification Toolbox for MATLAB, Open Access, pp. 106-118.

Teneyuca, D., 2011. The illusion of inclusion. Information Security Technical Report, Internet Cloud security, Vol. 16, No. 3, pp. 102-107.

Tillapart, P., Thammarojsakul, S. & Santiprabhob, P., 2005. An Approach to Hybrid Clustering and Routing in Wireless Sensor Networks. IEEE Aerospace Conference, pp. 1-8.

Tuan, T.A., Long, H.V., Son, L.H., Kumar, R., Priyadarshini, I. & Son, N.T.K., 2019. Performance evaluation of Botnet DDoS attack detection using machine learning. Evolutionary Intelligence. Vol. 13, No. 8, pp. 283-294.

Valdiviezo, L.M., 2014. Simulation Models for the Evaluation of Detection and Defense Protocols against Cyber Attacks Preparation of Doctoral Consortium Contributions, International Conference on Simulation and Modeling, Vol. 17, No. 1, pp. 42-47.

Veetil, S. & Gao, Q., 2014. Real-Time Network Intrusion Detection using Hadoop-Based Bayesian Classifier. Emerging Trends in ICT Security, pp 281-299.

Virupakshar, K.B., Asundi, M., Channal, K., Shettar, P., Patil, S. & Narayan, D.G., 2020. Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based

Private Cloud. International Conference on Computational Intelligence and Data Science (ICCIDS 2019),pp. 2297-2307.

Wang, B., Zheng, B., Lou, W. & Hou, T. Y., 2014. DDoS Attack Protection in the Era of Cloud Computing and Software-Defined Networking. IEEE 22nd International Conference on Network Protocols, Vol. 81, No, 1, pp. 624-629.

Wany, B., Zheng, Y., Lou, W., & Hou, Y.T., 2015. DDoS attack protection in the era of cloud computing and Software-Defined Networking. International Conference on Network Protocols, Vol. 81, No. 1, pp. 308–319.

Yin, C., Zhu, Y., Fei, J. & He, X., 2017. A Deep Learning Approach for Detection Using Recurrent Neural Networks, IEEE Acess, Vol. 5, No. 1, pp. 21954-21961.

Ying, B., 2014. CUSUM-Based Intrusion Detection Mechanism for Wireless Sensor Networks, Journal of Electrical and Computer Engineering, Vol. 2, No. 1, pp. 1-6.

Zekri, M., Kafhali, S. E., Aboutabit, N. & Saadi, Y., 2017. DDoS Attack Detection Using Machine Learning Techniques in Cloud Computing Environment. International Conference of Cloud Computing Technologies and Applications (CloudTech), pp. 1-7.

Zhang, H., Dai, S., Li, Y. & Zhang, W., 2018. Real-time Distributed-Random-Forest-Based Network Intrusion Detection System Using Apache Spark. IEEE 37th International Performance Computing and Communications Conference (IPCCC), pp.1-7.