**KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY**


# ANALYSIS AND DETECTION OF TRAFFIC FROM DOS ATTACK TOOLS USING DATA MINING

By

SETH DJANIE KOTEY

BSc. Computer Engineering

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING,

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF PHILOSOPHY IN COMPUTER INGINEERING

OCTOBER 2019

## DECLARATION

I hereby declare that this submission to the best of my knowledge contains no material previously published nor material which has been accepted for the award of a degree by this institution or any other institution, except where due acknowledge is given.



_____ _____

SETH DJANIE KOTEY Date

(STUDENT)

Certified by:

_____ _____

DR. ERIC TUTU TCHAO Date

(SUPERVISOR)

Certified by:

_____ _____

DR. EMMANUEL KOFI AKOWUAH Date
(HEAD OF DEPARTMENT)

## ABSTRACT

There has been a significant increase in the use of the internet over the past 20 years. As of June 2019, it was estimated that the number of internet users worldwide was over 4.4 billion, corresponding to about 57% of the world's population. The increase in the use and dependability of the internet has left in its trail a wide variety of vulnerabilities to defend against. One of the key security concepts that helps to guide cybersecurity policies is availability. In a computer network, a denial of service prevents users from having access to resources or services over the network. Denial of service (DoS) attacks are attacks purposely to disrupt availability of a network infrastructure. In past years, a DoS attack required a lot of skill and knowledge in networking for an attack to be launched. However, in recent years, DoS attack tools have been developed by various individuals and groups of people and are readily available on the internet for free or for a little amount of money. Such tools can be used by even the least skilled or knowledgeable attacker. This research therefore sought to develop a defence mechanism against these easy-to-use tools. Attack traffic was captured from some DoS tools and compared with benign traffic. Based on the differences between the attack traffic and benign traffic captured, a signature-based detection algorithm based on support vector machine (SVM) classifier was proposed. The algorithm was tested using the Snort IDS tool and the results were compared with some existing DoS defence schemes. Tests results from the algorithm showed the proposed defence mechanism had a high detection accuracy, low false positive rate and fast detection time.

## DEDICATION

This thesis is dedicated to the Almighty God, to my parents and to the rest of my family.

## ACKNOWLEDGMENT

I would like to express my gratitude to God Almighty for seeing me through this programme successfully. I am very grateful to my supervisor, Dr. Eric Tutu Tchao for his support and guidance throughout the programme. I would like to thank Dr. James D. Gadze for his advice and directions. I would like to thank Dr. Emmanuel Kofi Akowuah, head of department for Computer Engineering, Prof. K.O. Boateng, Dr. Selorm Klogo, Dr. Henry Nunoo Mensah and the entire Computer Engineering department for the support given during the duration of this programme.

I would like to thank my course mates, Kelvin Lartey, Kwame Asamoah Boateng, Farouk Abd-Allah, Samuel George Sefah-Nyame and Daniel Commey as well as Miss Beatrice Whyte for the many insightful discussions and contributions.

I would like to thank my parents, Mr. and Mrs. S.K. Kotey and my siblings for their unlimited support throughout this programme.

**TABLE OF CONTENTS**

**LIST OF FIGURES**

**LIST OF ABBREVIATIONS**

ARPANET…….…… Advanced Research Projects Agency Network

DARPA…….…. Defence Advanced Research Projects Agency

DDoS……..……. Distributed Denial of Service

DNS……. Domain Name System

DoS……..…….. Denial of Service

EDADT… …… Efficient Data Adapted Decision Tre

HOIC……...……...…… High Orbit Ion Cannon

HTTP……...Hypertext Transfer Protocol

URI…… Uniform Resource Identifier

ICMP……...…. Internet Control Message Protocol

IDS……...…… Intrusion Detection System

IP……..… Internet Protocol

KDD……...…… Knowledge Discovery in Databases

LOIC……...…… Low Orbit Ion Cannon

PTA-SVM…. ……...… Packet Threshold Algorithm-Support Vector Machine

RAM……...… Random Access Memory

SDN…… Software Defined Networks

SVM……..…...Support Vector Machine

TCP…… Transmission Control Protocol

UDP……. User Datagram Protocol

# CHAPTER ONE

# INTRODUCTION

## 1.1 OVERVIEW OF COMPUTER NETWORKING

Computer networks play an essential role in the modern world. Networks are very crucial for increased productivity and offer several advantages, one of which is the ability to share information and computing resources. A computer network is a connection between two or more computers for exchange of information and sharing of resources between users. Computer Networking began with development of ARPANET in the early 1970s (Sunshine,

1989). Before that, there existed terminals for remote job entries connected to a mainframe. Networking in which individual computers were viewed as equal peers to for resource sharing only began with the ARPANET design. The aim of a computer network is to ensure local resources of any computer in the network are made available to authorised users remotely without a significant degradation in performance of the host computer. Data, information, software and hardware resources can be shared over a network. Networks help to eliminate the distance limitation between users sharing these resources. Networks have also introduced the ability to access a large store of information accumulated over many years through the internet (Das & Chaudhuri, 2017). The internet is a global system of interconnected networks. The internet has drastically changed communication in modern civilisation (Bijalwan, Wazid, Pilli, & Joshi, 2015) (Kolahi, Treseangrat, & Sarrafpour, Analysis of UDP DDoS flood cyber attack and defense mechanisms on Web Server with Linux Ubuntu 13, 2015). Several social media services have become available to connect with known as well as preciously unknown people. File sharing, video conferencing, ecommerce services among others have increased productivity (Hannula, 2011).

There has been a significant increase in the use of the internet over the past 20 years. As of

June 2019, it was estimated that the number of internet users worldwide was over 4.4 billion, corresponding to about 57% of the world's population (Miniwatts Marketing Group, 2019). The increase in the use and dependability of the internet has left in its trail a wide variety of vulnerabilities to defend against (Kolahi, Alghalbi, Alotaibi, Ahmed, & Lad, 2014). One of the key security concepts that helps to guide cybersecurity policies is availability (Dua & Du, 2016). Data availability is the guarantee of reliable and constant access to data by authorised people. It is important for every business or organisation to ensure computer resources are readily accessible to authorised users over the network or internet at all times. When the computer network or internet is inaccessible, information availability is affected and significantly impacts users' productivity.

## 1.2 BACKGROUND

Computer networks are an important part of communication in the modern world. The ability to share files, information and computer resources over a network conveniently and across large distances have made it a vital component for businesses as well as individual users. With the convenience of networks comes certain challenges that must be addressed to ensure security is maintained. Availability is one of the major components of a secure system.

Denial of service is a disruption in the provision of a service to users. In a computer network, a denial of service prevents users from having access to resources or services over the network. Some causes of denial of service are genuine problems, for example, an unplanned ability to handle numerous requests (Purwanto, Kuspriyanto, Hendrawan, & Rahardjo, 2015). This is known as a flash crowd. This normally occurs when there is a much-anticipated new feature available and many end users try to access it around the same time to have a firsthand experience of the feature. This, though a legitimate attempt by users, could cripple servers if unplanned for. Other causes of denial of service are malicious attempts by attackers to cripple a network or service.

Denial of service (DoS) attacks are attacks purposely to disrupt availability of a network infrastructure (Grance, Kent, & Kim, 2004) (Holl, 2015). Distributed denial of service (DDoS) attacks are coordinated DoS attacks launched from different locations at the same time with the same target (Amazon Web Services, 2016). DoS attacks are perpetrated by a user using a single

machine connected to one network sending bogus traffic to a victim machine to try to overwhelm it and cause a disruption of services offered by the victim machine or a total shutdown of the victim machine (Merouane, 2017). In 2018, a 1.3Tbps (Newman, Lily Hay, 2018) attack and another 1.7Tbps (Morales, Carlos, 2018) attack were reported within days of each other. DDoS attacks could be engineered by one or more attackers, harnessing the power of numerous compromised devices connected online, known as zombies or collectively known as a botnet, in a bid to overwhelm a victim machine with bogus traffic. DoS attacks could last for even months, making them a menace to provision of online services. Though most DoS attacks are not perpetrated for financial gain, some attackers request a ransom to end the attack. DoS and DDoS attacks are mostly carried out to disrupt service, but not to gain information, unlike other security attacks. There are number of reasons for an attacker to perform such an attack. A typical one is for revenge on the victim for a perceived injustice done to the attacker by the victim. Some people also perform these attacks as a challenge to showcase their abilities (Rajkumar & Jitendra, 2013). For whichever reason the attack is performed, the effect of the attack, if successful, causes a lot of havoc.

In past years, a DoS attack required a lot of skill and knowledge in networking for an attack to be launched. However, in recent years, DoS attack tools have been developed by various individuals and groups of people and are readily available on the internet for free or for a little amount of money (Imperva Inc., 2019). Such tools can be used by even the least skilled or knowledgeable attacker. Any newbie can launch a high-level DoS or even DDoS attack with little to zero knowledge about networks and attack scripts. These tools have made it easy for attackers to launch DoS attacks and has resulted in an increase in the number of DoS attacks in recent years.

## 1.3 RESEARCH PROBLEM

Various efforts have been made to defend against Denial of Service attacks. These efforts, varying in approach, have been met with varying degrees of success. Successful DoS attacks can be expensive to the victim with losses capable of reaching $100,000 per hour (Matthews, Tim, 2014). The reputation of organisations, especially those offering e-commerce services can be damaged. The development and ready availability of DoS tools has made it easy for even the most unskilled person to launch a disruptive DoS attack. This has resulted in an increase in the number of DoS attacks. Attacks using these tools are still successful, even with the development of DoS defence mechanisms. This is because there are no defence mechanisms designed purposely to defend against attacks from these attack tools. Knowledge of traffic patterns of current DoS attack tools is also not available. There is therefore the need to analyse the current DoS tools in use and develop methods to neutralise their effects.

## 1.4 RESEARCH OBJECTIVES

This research seeks to develop a DoS detection scheme to detect attacks from unsophisticated DoS tools. The study therefore seeks to achieve the following specific objectives:

1. capture and analyze traffic generated from current dos attack tools and determine the differences between traffic packet contents of benign traffic and attack traffic using data mining.

2. propose a dos detection scheme based on the disparities between benign traffic packets and attack traffic packets.

3. test the proposed detection scheme and compare results with existing detection schemes.

## 1.5 RESEARCH JUSTIFICATION

Distributed denial of service attack is a common yet effective way of disrupting the service of a network-based system. With availability of service being a major selling point of any network-based system, network outage is not desirable for any system of this sort. Every network-based system requires an active and reliable connection to the network and without this, the purpose for network-based systems is defeated. Development of unsophisticated DoS tools and the ease of access of these tools by even unskilled users has given rise to the number of DoS attacks being launched. Developing a defence mechanism to detect these attacks will reduce the number of successful DoS attacks from these tools. This will in effect reduce costs to victims as a result of these DoS attacks as well as ensure availability of network resources.

## 1.6 THESIS ORGANIZATION

The rest of the thesis is organized as follows: Literature review and related works are presented in Chapter 2 and the algorithm design is presented in Chapter 3. Chapter 4 presents the results from testing the algorithm and comparisons with existing models proposed in literature. Conclusions, recommendations and future directions are presented in Chapter 5.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.1 INTRODUCTION

A denial of service (DoS) attack is a computer network attack in which an attacker seeks to overload a system with data over the network to prevent legitimate requests for service to be processed. DoS attacks are launched to cripple the target system for as long as possible. DoS attacks involve just one attacker, but are capable of causing enough damage if not detected early. Distributed denial of service (DDoS) attacks launched from multiple sources simultaneously for a larger attack size. DDoS attacks can essentially be viewed as multiple DoS attacks from different sources targeting the same victim at the same time. This means being able to defend these individual DoS attacks successfully will aid in defending the DDoS attack. Therefore, this study the focuses on DoS attacks.

### 2.1.1 DoS Attack Categories

DoS attacks involve an attacker bombarding a victim with large volumes of traffic in a bid to overwhelm the victim. There are several categorisations of DoS attacks. For this study, they are classified in two categories: application layer and network/transport layer attacks.

***Network/transport layer attacks***: Attacks at the network/transport layer are launched using the network/transport layer protocols. These attacks target the network itself, aiming to consume the bandwidth of the victim. Examples of these attacks are listed below.

*TCP flooding*: TCP flooding attacks involve the attacker sending large volumes of traffic using TCP. A variation of the TCP flooding is the syn flooding in which the attacker attempts to

exploit the nature of TCP connections. The victim is flooded syn packets without completing the three-way handshake or closing the connections without sending any data.

(Wang, Zhang, & Shin, 2002) (Acharya & Tiwari, 2016)

*UDP flooding*: UDP flooding attacks involve the attacker sending a large stream of UDP traffic to the victim to clog up the victim's network. (Criscuolo, 2000) (Acharya & Tiwari, 2016)

*ICMP flooding*: ICMP flooding attacks involve an attacker attempting to take down the victim machine by sending an unnecessarily large amount of ICMP echo requests or pings. (Acharya & Tiwari, 2016)

*Ping of Death*: In a ping of death attack, the attacker sends oversized or malformed packets with the ping command. These packets are fragmented before being sent and the victim ends up with oversized packets when reassembling. (Elleithy, Blagovic, Cheng, & Sideleau, 2006)

***Application layer attacks***: Application layer attacks aim to interrupt the operation of a specific service or application on a server to prevent other users from accessing the application or service. Some of these attacks are listed below.

*DNS Amplification*: DNS amplification attacks rely on a vulnerability inherent in domain name system (DNS) servers in which a small query is replied with a much larger payload. An attacker sends queries to public DNS servers using the victim's IP address as the source and the server responds with the large payload to the victim. (Anagnostopoulos, Kambourakis, Kopanos, Louloudakis, & Gritzalis, 2013)

*HTTP fragmentation attack:* HTTP fragmentation attacks involve the attacker establishing a connection with a victim server and sending traffic as slowly as possible and in fragments to

ensure the server maintains longer sessions with the attacker as it waits to receive traffic from the attacker. (Zargar, Joshi, & Tipper, 2013)

## 2.2 DOS ATTACK TOOLS

DoS attack tools, also known as attack scripts or DoS standalone attack tools, are software written to enable easy launch of DoS attacks (Imperva, n.d.). Though not all attack tools are written for malicious purposes, such tools end up being used for attacks. Most of these tools are used for vandalism or personal rivalries, however, they have been successfully used to cause some significant damage to some online services. These tools have made it easy for inexperienced and unskilled people to launch DoS attacks successfully.

### 2.2.1 Attack Tools Selected

**LOIC v1.0.8**: Low Orbit Ion Cannon is a DoS attack tool written in C# (KITPLOIT, 2014). Praetox Technologies developed it initially, and later released it to the public. The tool is one of the most popular DoS tools available and is hosted on many open source platforms, an example is SourceForge (SOURCEFORGE, 2014). The tool can be used for both TCP and UDP flooding attacks as well as HTTP flooding attacks. SourceForge alone indicates over 2 million downloads in total and over 5,000 downloads per week. Figure 2.1 and figure 2.2 show the configurations used in launching an attack for TCP and UDP attacks respectively. The IP address and port number are the only requirements to launch an attack, with options to adjust the speed, set the number of threads used for the attack and the message sent in the data section of the packets.

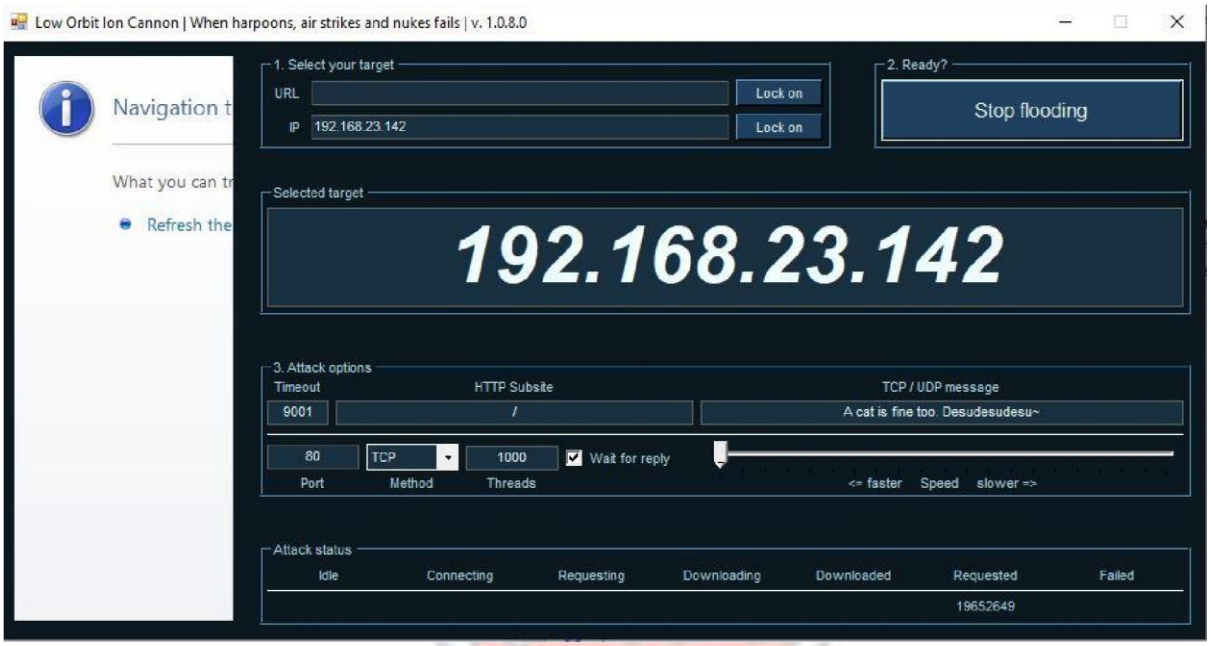


**Figure 2.1** LOIC TCP attack configuration

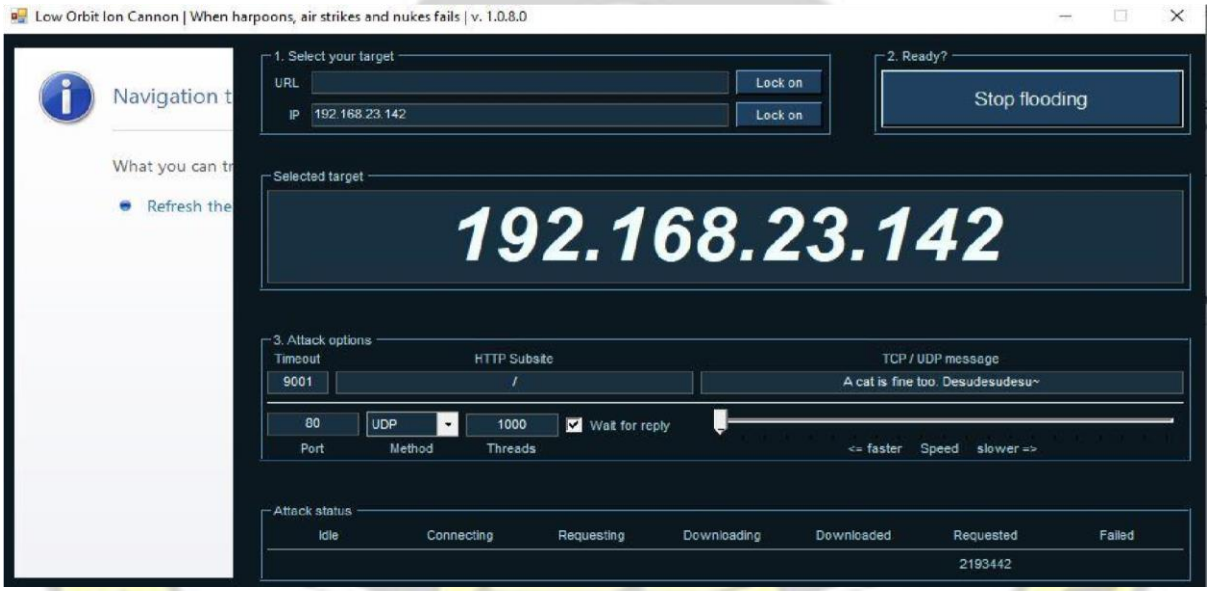


**Figure 2.2** LOIC UDP attack configuration

**HOIC v2.1**: High Orbit Ion Cannon is a DoS tool related to LOIC. It can can be used to launch TCP attacks. It has the option of including several targets. It is however unstable when a high number of threads are selected, causing the application to crash. It also has options to select the power of the attack and include a "booster" script to increase the strength of the attack. The port targeted by default is port 80. Figure 2.3 shows the interface of the tool.

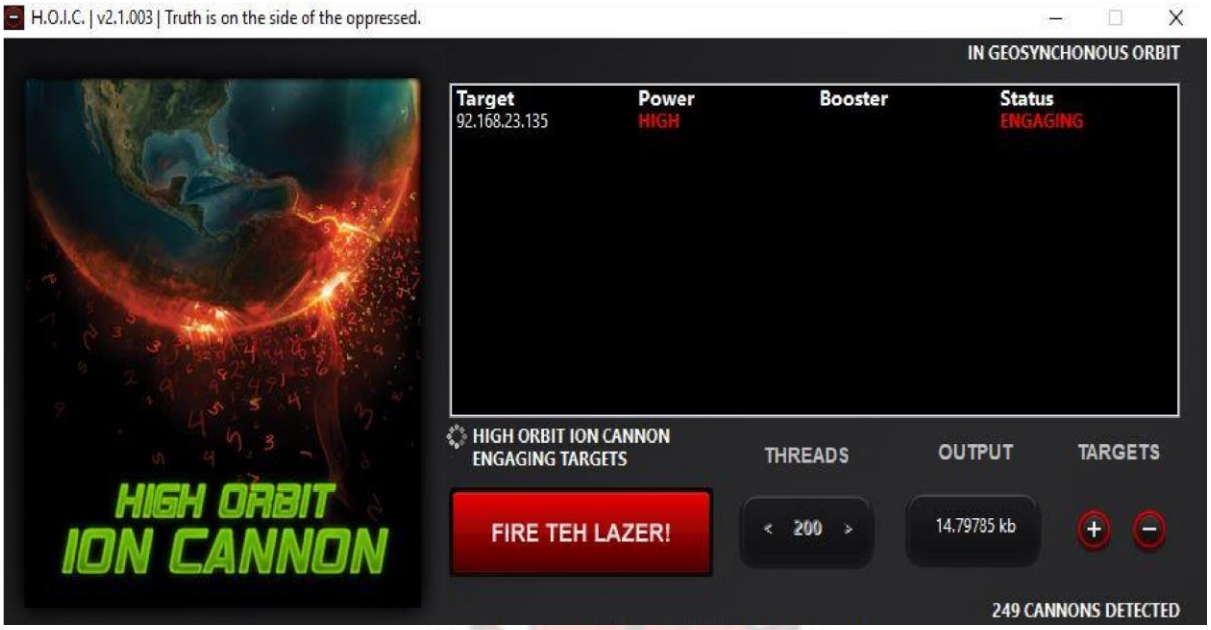


**Figure 2.3** Configuration of HOIC

**XOIC v1.2**: XOIC is an open source DoS attack tool written by DLR. It is a simple and straightforward tool which only requires the IP address, port and protocol to employ in launching the attack. It can perform TCP, UDP and ICMP attacks. Figure 2.4 shows the interface of XOIC. Figure 2.5 shows XOIC launching an attack.

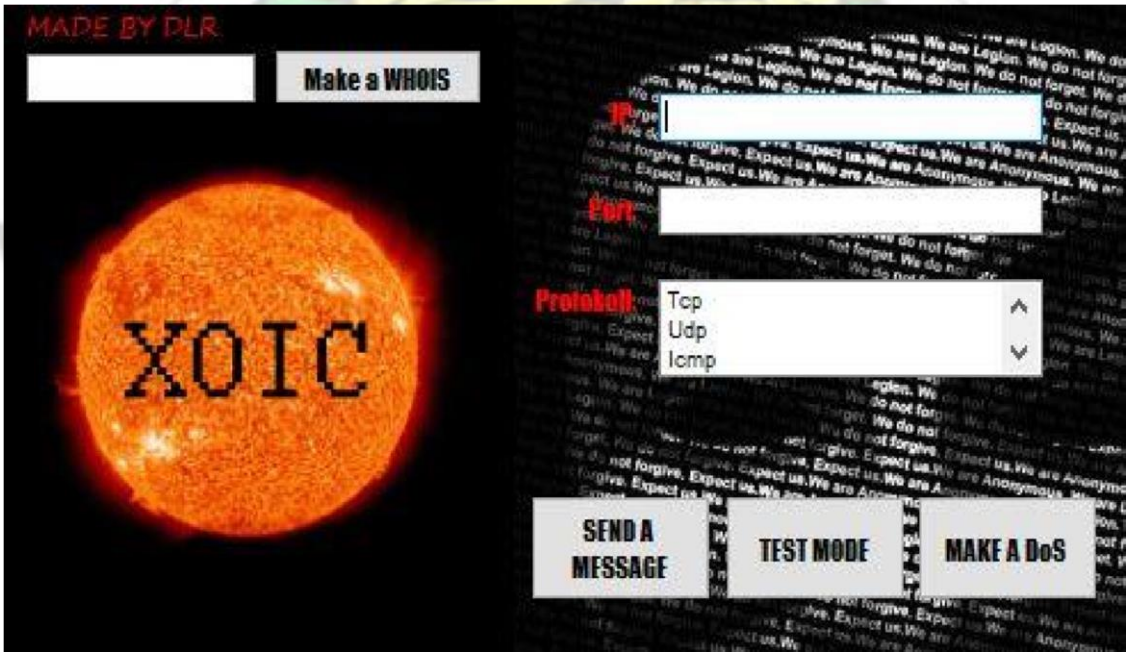


**Figure 2.4** XOIC interface


**Figure 2.5** XOIC attack

**DoS Tools By Noé**: This tool has a very simple interface and performs UDP attacks. It gives the user the option to input the IP address, port number and data to send to the target. It also has an option to determine the speed of the attack, with the highest speed being 10 and to set the message to be sent in the data section of packets. Figure 2.6 shows the interface and a sample configuration used to launch an attack.


**Figure 2.6** DoS tools by Noé interface and configuration

**FHRITP**: This simple tool only gives the user the option to input the IP address of the target. It attacks the victim at port 80 and can launch only TCP attacks. It produces a very small amount of traffic and required multiple instances to be run simultaneously to generate a significant amount of traffic. Figure 2.7 shows the interface of the tool.


**Figure 2.7** FHRITP DoS tool interface

**Moihack**: Moihack DoS tool is a simple TCP and UDP attack tool written in python. The user types the IP address, port number, protocol and choice of generating random packets on a command line interface. An example command is also given to show the syntax for launching an attack for easy reference. Figure 2.8 shows the command interface of the tool.

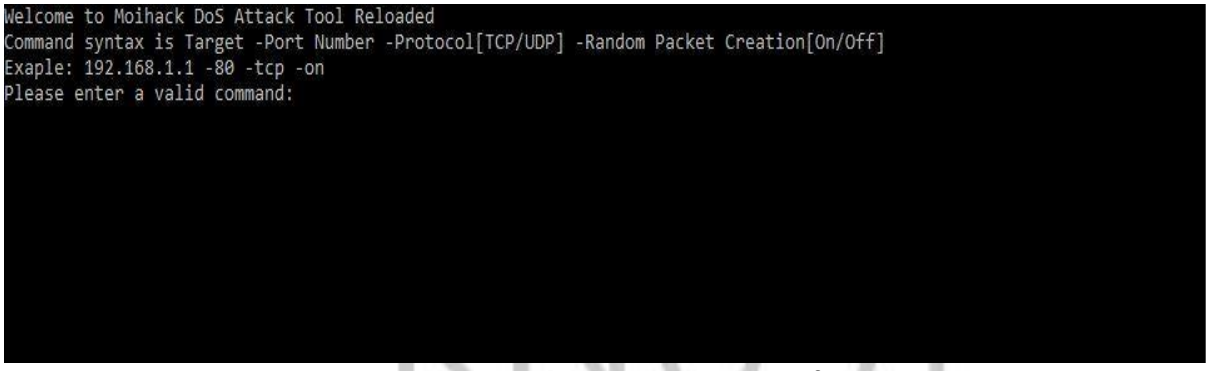


**Figure 2.8** Moihack command interface

**Anoncannon v1.06**: Anoncannon is another DoS tool based off LOIC. It only performs TCP attacks. It is unstable when high number of threads is selected. The only other attack option is the message to send in the data section of packets. Figure 2.9 shows the interface of Anoncannon with an active attack underway.

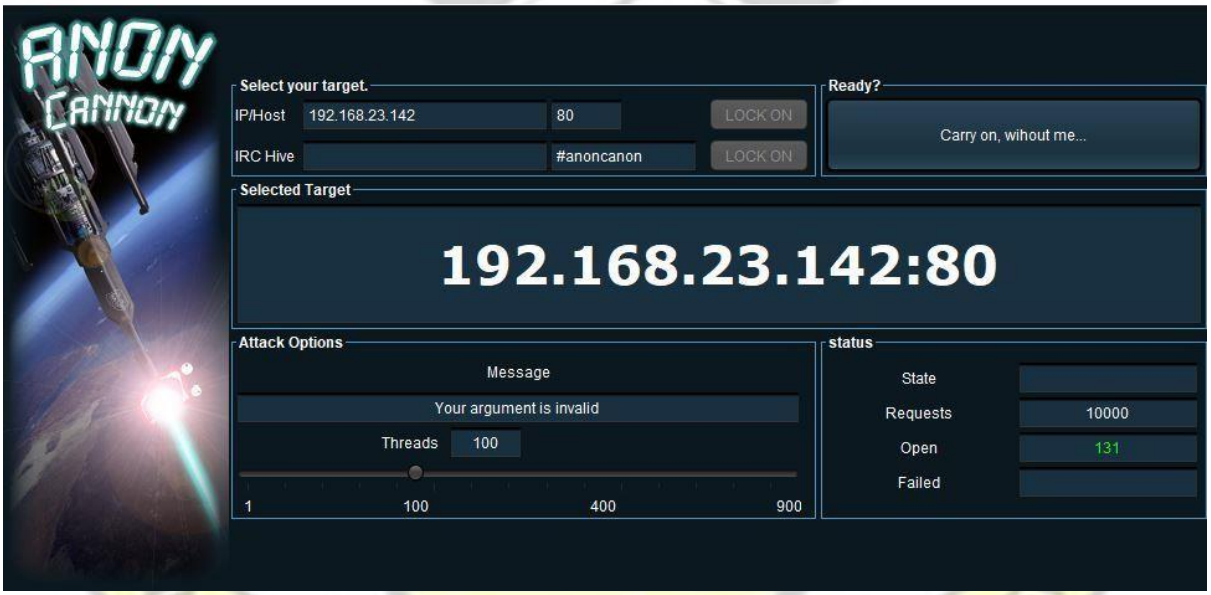


**Figure 2.9** Anoncannon DoS attack

**ByteDoS v3.2**: This simple tool has the option of launching a TCP SYN flood or ICMP flood attack, however after tests it was realised the ICMP option functions exactly as the TCP SYN option. The tool opens two windows, each giving the option of using 3 threads to launch an attack. Figure 2.10 shows the interface of the tool.

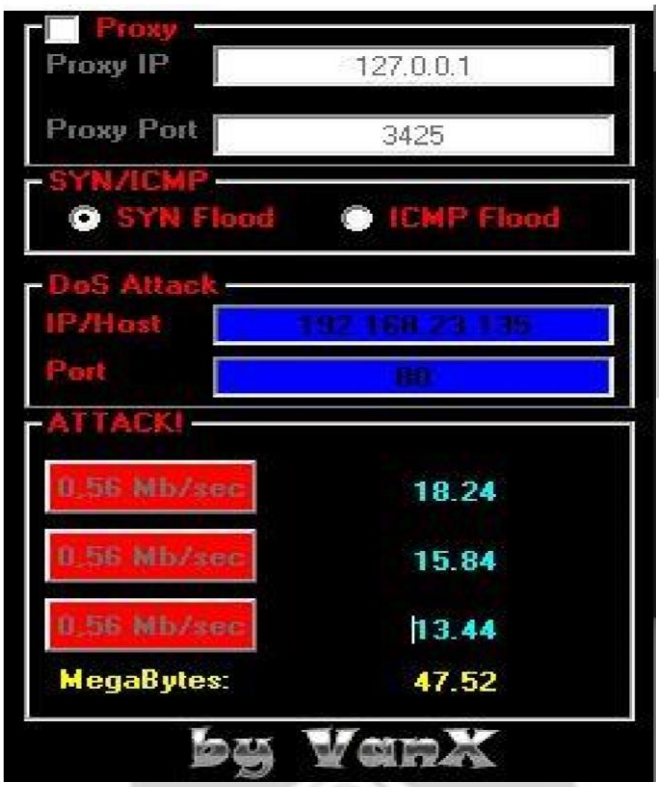


**Figure 2.10** ByteDoS tool

### 2.2.2 Studies on Attack Tools

There are several DoS attack tools available to anyone on the internet. There is however very sparse literature focused on characterising these tools, and much less focusing on defending attacks from these tools. Most of these studies also focus on obsolete and practically forgotten tools (Bhuyan, Kashyap, Bhattacharrya, & Kalita, 2014).

(Bukac & Matyas, 2015) presented a comparative analysis of features generated by standalone DoS attacks tools. They classified traffic features of different attack tools as well as presenting the evasion techniques used by the tools. Attacks tools to implement a wide variety of DoS attacks were presented. Traffic from these tools were captured with no background data to ensure accurate analysis. They analysed traffic burst behaviour, flow count, flow parallelism, flow packet count, HTTP requests per flow and HTTP request uniform resource identifiers (URI). Future research directions were presented, however a solution to defend against the attacks was not proposed.

(Mahadev, Kumar, & Kumar, 2016) classified DDoS threats based on anomalous behaviour at the application layer. They also presented a summary of information about some DDoS tools. Threats were classified according to low rate requests, low rate session and resource depletion attacks. Some attacks tools presented Knight, Shaft, Trinity and Trinoo. They went further to describe some DoS prevention methods.



(Behal & Kumar, 2017) characterised and compared popular DDoS traffic generators and attacks tools. In their paper, they identified both legitimate and background traffic generators as well as attacks tools which have been used to launch DoS attacks. Characteristics of these tools were presented and the tools were compared with each other according to the impact to the victim (bandwidth or resource depletion), encryption of traffic, attack model, IP spoofing, interface of tool among others. They however did not present a method to defend against attacks from the attack tools.

## 2.3 REVIEW OF DOS DEFENCE MECHANISMS

Several mechanisms have been proposed to defend against DoS attacks. These mechanisms have been met with varying degrees of success. Various techniques have been used in designing these mechanisms. Some of the proposed mechanisms are presented here.

(Cui, 2016) presented a mechanism to defend against attacks in software defined networks (SDN). Software Defined Anti-DDoS is able to detect and mitigate DoS and DDoS attacks. The proposed model has a detection trigger which periodically checks for attacks and launches the detection module when there is a possible attack. The detection module uses the Back Propagation Neural Network (Rumelhart, Hinton, & Williams, 1986) to classify and differentiate between attack and legitimate traffic. The model was implemented with the RYU framework (Ryu sdn framework) and evaluated with Mininet (Lantz, Heller, &

McKeown, 2010) (Mininet: An instant virtual network on your laptop).
(Park, 2014) proposed a service-oriented detection mechanism using pseudo states. The mechanism uses a bidirectional key hashing algorithm to maintain a pseudo state machine and a hashing table. Service procedure states are represented as bit sequences to enable flow routers with packet analysis. The system checks the source and destination IP addresses and ports as well as the protocol in use for every flow. Traffic destined to a protected server is checked if it is attack traffic from the pseudo state machine. (Shon, Kim, Lee, & Moon, 2005) and (Tanachaiwiwiat & Hwang, 2003) proposed related schemes but used a principal component analysis tool to reduce the number of variables used in traffic analysis. (Lau, Rubin, Smith, & Trajkovic, 2000) also proposed a similar scheme but used a support vector machine to classify traffic.

(Yu, Zhou, Guo, & Guo, 2013) proposed a dynamic packet marking scheme based off the deterministic packet marking scheme proposed by (Belenky & Ansari, 2003). The scheme is designed to trace routers as close to the attack source as possible. Marking identities are dynamically given to routers along the path of possible attacks using a marking on demand scheme. An attack detector is set up to monitor network flow at each gateway and marks all suspicious flows with an identity which is tied to the IP address of the flow. Identities from confirmed attack flows are picked out and the IP addresses related to that attack flow are identified. (Xiang, Zhou, & Guo, 2009) proposed a related scheme using a flexible marking method to vary the length of the marking identity based on the network protocol in use by adding the type of service field for marking.

(Foroushani & Zincir-Heywood, TDFA: Traceback-based Defense against DDoS Flooding Attacks, 2013) presented a traceback based defence against flooding attacks. The proposed scheme is aimed at filtering attack packets as close to the source as possible, greatly reducing the number of packets to be received and processed by the victim. The authors proposed to use

existing detection tools residing on the edge network. The deterministic flow marking (Foroushani & Zincir-Heywood, IP traceback through (authenticated) deterministic flow marking: an empirical evaluation, 2013) method is used by the traceback component. Traffic flows are marked with outside interface IP address, the network interface identifier and a node identifier. The identifiers of detected attack traffic flows are then used to trace the attack as close to the source as possible where attack traffic can then be filtered. (Yaar, Perrig, & Song, 2006) (Chen, Shantanu, & Pulak, 2008) proposed similar methods, however they used a path fingerprint to mark each packet.

(Sahi, Lai, Li, & Diykh, 2017) presented a classifier system to detect and prevent TCP flooding attacks. The proposed scheme blacklists and drops packets from source IP addresses related to attacks. Packets from traffic flows are passed through a classifier to determine if they are malicious or legitimate. A packet is tagged to be malicious if its source IP address requests more connections to the same destination than the threshold set. The classifiers used were the K-nearest, naïve Bayes, least square support vector machine and multilayer perceptron (Hameed, Karlik, & Salman, 2016).

(Zhou, Jia, Wen, Xiang, & Zhou, 2013) presented a defence scheme to defend application layer attacks in heavy backbone web traffic. A real-time frequency vector examines entropy of traffic to differentiate between attack traffic and flash crowds. The number of HTTP ‘Get’ requests are tracked per second. The Kalman filter is used to predict the next amount. A deviation between the predicted amount and actual amount beyond the threshold set is reported as an abnormal event. Traffic from attack sources are the filtered with a bloom filter (Broder & Mitzenmacher, 2002).

(Wang, Zheng, Lou, & Hou, 2015) presented a defence architecture using software defined networks for cloud systems. DaMask, an anomaly-based detection and mitigating system was designed to secure services in cloud systems whilst incurring very little computational overhead and also having a low cost of deployment. The authors relied on existing detection algorithms in their work. Network virtualisation was used to separate network traffic destined for different locations. Upon arrival of a packet, the packet is checked to see whether it belongs to an existing flow. If it is not, the flow statistics are updated and the detection model is queried. An alert is issued and if the there is no pre-set policy defined for that alert, the packet is dropped.

(Zheng, 2018) presented a real-time attack defence system built using stock software defined switches. The mechanism uses adaptive correlation analysis to detect flooding attacks. Traffic flows are identified as containing malicious or legitimate using patterns of anomalous flows obtained from other switches. Further analysis is done on each path to determine the changes in flow statistics on each path if the statistics change rate is consistent with aggregated flows noticed on the victim's link, traffic is tagged as malicious. The system was evaluated with Project Floodlight (Project Floodlight) and Caida (Caida). (Kang, Gligor, & Sekar, 2016) presented a similar traffic-engineering model which tracks traffic source rate change during network link flooding, however the system uses modified switches.

(de Assis, Hamamoto, Abrão, & Proença, 2017) presented a game theory-based decision making model which uses detection from a Holt-Winters for Digital Signature system (de Assis, Rodrigues, & Proença, A novel anomaly detection system based on seven-dimensional flow analysis, 2013) (de Assis, Rodrigues, & Proença, A seven-dimensional flow analysis to help autonomous network management, 2014). The defence mechanism was designed to defend attacks targeting the control plane of software defined networks. Seven parallel IP flow dimensions are analysed and used to characterise traffic. These seven dimensions are the

entropy values of the source and destination addresses and ports, packet, bit and flow rates per second. For each analysed dimension, a signature is created. These signatures are then compared with existing attack signatures and if similar, traffic is flagged as an attack.

(Yusof, Ali, & M.Y., 2018) presented a detection algorithm to defend against flooding attacks. A packet threshold algorithm in combination with a support vector machine was used in their proposal. The algorithm checks the rate of traffic packets to ensure it is below a threshold, else the traffic is flagged as an attack. The support vector machine is used to train datasets to improve detection accuracy.

(Wang H. e., 2014) presented a moving target mechanism. The system uses dynamic proxies for transmission of network between authenticated clients and servants. Clients are shuffled between proxies with addresses known only to authenticated clients in the event of an attack by using a greedy algorithm.

# CHAPTER THREE

# ALGORITHM DESIGN

## 3.1 INTRODUCTION

Due to the increasing frequency and ease of launching DoS attacks, it is imperative to know the nature of traffic generated by these attacks and develop methods to detect such attacks early before a lot of harm is caused. This chapter presents the proposed method for detecting DoS attacks launched using attack tools. Figure 3.1 shows a flow chart of the steps taken from determining the features of the attack traffic to testing the proposed model.

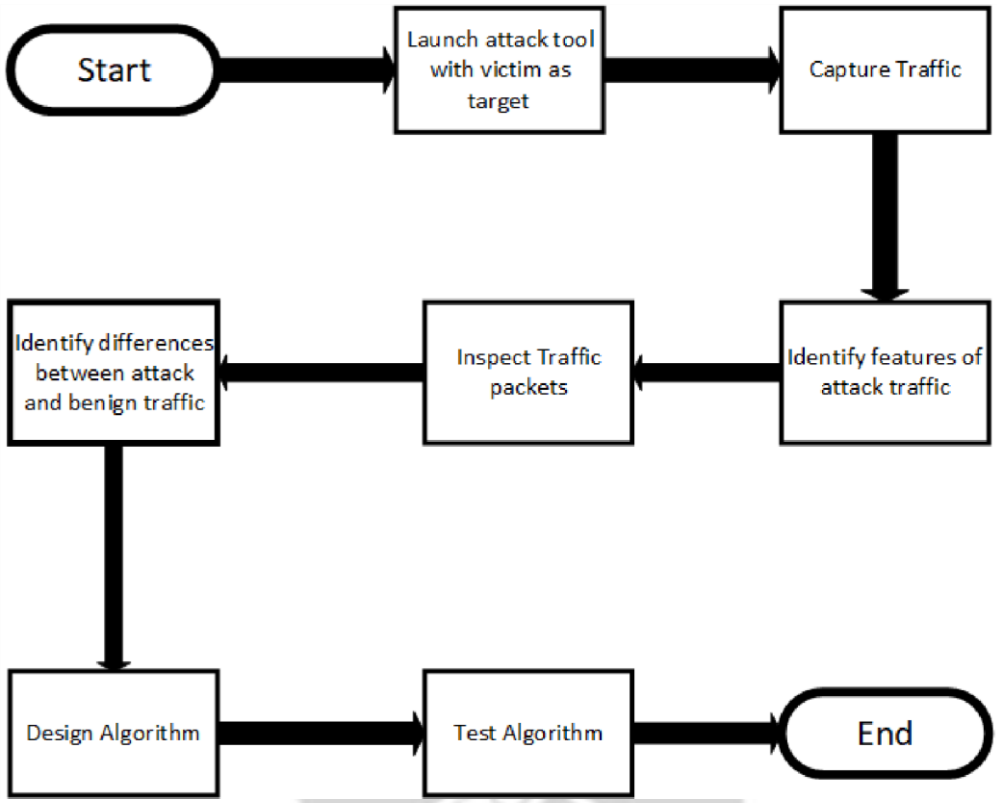


**Figure 3.1 Process Flowchart**

## 3.2 EXPERIMENTAL SETUP

Attack traffic was captured using the setup in figure 3.2. The detection algorithm was tested using the setup in figure 3.3. Wireshark Network Analyser was used to capture the attack traffic for analysis. Snort Intrusion Detection tool was used to test the proposed algorithm.

The attacking machine was running Microsoft Windows® 10 with 8GB of RAM and a 2nd Generation Intel® Core I5 processor running at 2.5GHz with 4 cores. The victim machine was a virtual machine running Linux Ubuntu 19 with 2GB of RAM and a processor running at 2.5GHz with 2 cores.

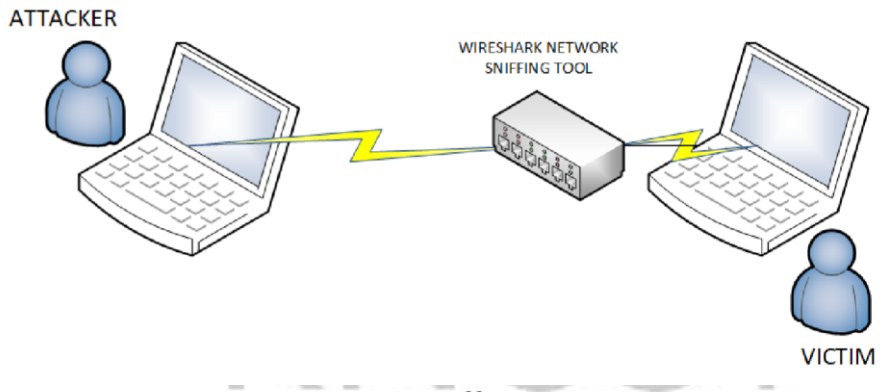


**Figure 3.2 Attack traffic capture setup**

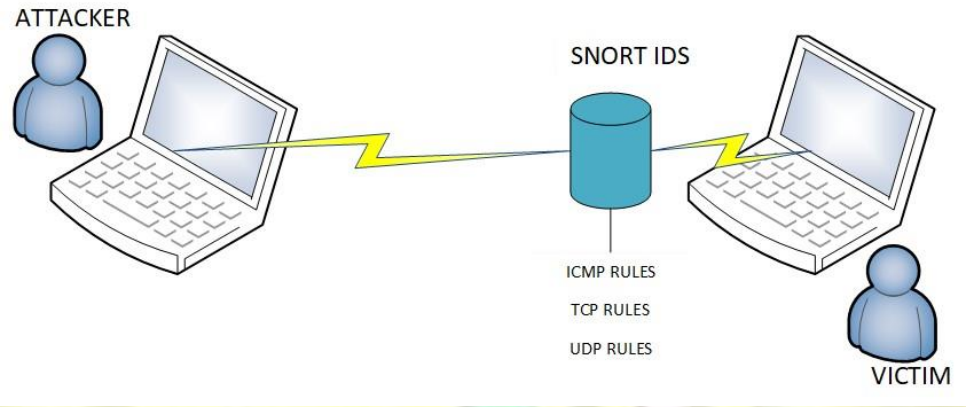


**Figure 3.3 Algorithm test setup**

## 3.3 ATTACK LAUNCH

The network between the attacker and victim was setup such that there will be no interference from any external source. Any interference from an external device sending data across the network channel would have resulted in an inaccurate representation of attack captured.

DoS attack tools used in this research are LOIC v1.0.8, HOIC v2.1, XOIC v1.2, DoS Tools By Noé, FHRITP, Moihack, Anoncannon v1.06 and ByteDoS v3.2. These tools were chosen due to their ease of access on the internet as well as their ease of use in launching a DoS attack. The tools can be easily downloaded over the internet and are relatively easy to use, assuming the target machine is known. Some of the tools could also be used to launch more than one type of attack. After the selection, it was realised that seven out of eight of the tools could be used to launch TCP flooding attacks, four could be used to launch UDP flooding attacks and only one

could be used to launch an ICMP flooding attack. For each type of DoS attack considered for this research, tools used are listed below:

1. TCP flooding attack: Anoncannon, ByteDos, FHRITP, HOIC, XOIC, Moihack and LOIC
2. UDP flooding attack: Dos Tools by Noé, LOIC, Moihack and XOIC
3. ICMP flooding attack: XOIC

Table 3.1 is a summary of the configurations used in each tool. The protocol used for the attack, the port targeted and number of threads used (for tools which gave the option) is listed. Other configurations specific to each tool are also listed.

**Table 3.1 Summary of configurations**

| DoS Tool | Protocol | Port | No. of threads (simultaneous connections) | Other |
|---|---|---|---|---|
| LOIC | TCP, UDP | 80 | 1000 | Fastest speed |
| HOIC | TCP | 80 | 200 | High Power |
| XOIC | TCP, UDP, ICMP | 80 | | |
| DoS tools by Noé | UDP | 80 | | Speed x10 |
| FHRITP | TCP | 80 | | |
| Moihack | TCP, UDP | 80 | | Random Packet Creation On |
| Anoncannon | TCP | 80 | 100 | |
| ByteDoS | TCP | 80 | 6 | |

## 3.4 TRAFFIC CAPTURE

Attack traffic was captured using Wireshark Network Analyser. Benign traffic was also captured by browsing several web pages to mimic normal use of network. Traffic was captured for approximately sixty seconds. The Dumpcap file generated by Wireshark was saved for analysis.

## 3.5 TRAFFIC FEATURES IDENTIFICATION

For each captured attack, the rate of traffic in bytes per second and packets per second as well as the average packet size were taken. Graphs of the rates over sixty seconds were also taken. Traffic rates are one of the key identifiers of DoS attacks. Most attacks generate and transmit a huge number of packets and or large volume of traffic within a short time aimed at the victim. High packet or byte rates could therefore indicate the presence of an attack. Another indicator for TCP connections is the number of three-way handshakes observed. Most TCPbased DoS attacks tend to set up numerous connections simultaneously to the victim within a short time. The higher the number of open connections, the more resources the victim uses to ensure these connections are kept open, waiting for data, before the connection is closed.

## 3.6 TRAFFIC PACKETS INSPECTION

Traffic packets captured from attack traffic were inspected to determine the type and form of packets they are. Pattern recognition was employed to aid in packet inspection to determine the patterns used in attack traffic. For TCP traffic, a three-way handshake (syn, syn-ack, ack) is generally expected for establishing a connection, followed by the payload. The number of packets with the syn flag set gives an idea of the number of connections requested by an IP address. ICMP packets have a type field and code field. The type and code field combination determine the type of control message being sent.

## 3.7 ATTACK TRAFFIC VS BENIGN TRAFFIC

Attack traffic was differentiated from benign traffic using the traffic features and patterns observed from traffic packets. The differences in traffic features and patterns between expected traffic and attack traffic is a very important guide when developing DoS defence mechanisms. Without knowledge of these differences, a defence mechanism may report a lot of false positives or negatives.

## 3.8 TRAFFIC FEATURES AND TYPES OF PACKET

For every attack tool studied, certain features were gathered. Packet rates, byte rates and packet sizes can give an indication of whether traffic is benign or not. Traffic packet rate is the number of packets sent over the network channel with time. Byte rate is the volume of data sent over channel with time. Traffic containing similar sized packets will produce a similar pattern in packet rate and byte rate. This is a key indicator for TCP attacks requesting connections and closing them after connection is given without sending any data. The difference in packet rate and byte rate patterns and values can also indicate the nature of the DoS attack. Attacks with a high byte rate to packet rate ratio indicates the transfer of larger packet sizes, meaning a lot of bogus data is being transmitted. A low byte rate to packet rate ratio means smaller packet sizes are being sent, with the focus on transmitting a large number of packets with very little or no data. Packets without data normally have a size of 60 bytes or less, therefore a byte to packet rate ratio close to 60 indicates little or no data is being sent. The types of packets sent and their contents also provide an idea of what could be or not be attack traffic. The features of traffic captured and analysed are described below.

### 3.8.1 TCP Traffic

Figure 3.4 shows a sample of benign traffic byte rate captured from browsing several pages on the internet. Figure 3.5 shows the syn packet rate to total packet rate and figure 3.6 shows the push-ack packet rate to total packet rate. For this setup, this amount and behaviour of traffic can be described as expected traffic to our network. Traffic was captured for approximately five minutes to capture a wider time window of normal behaviour.

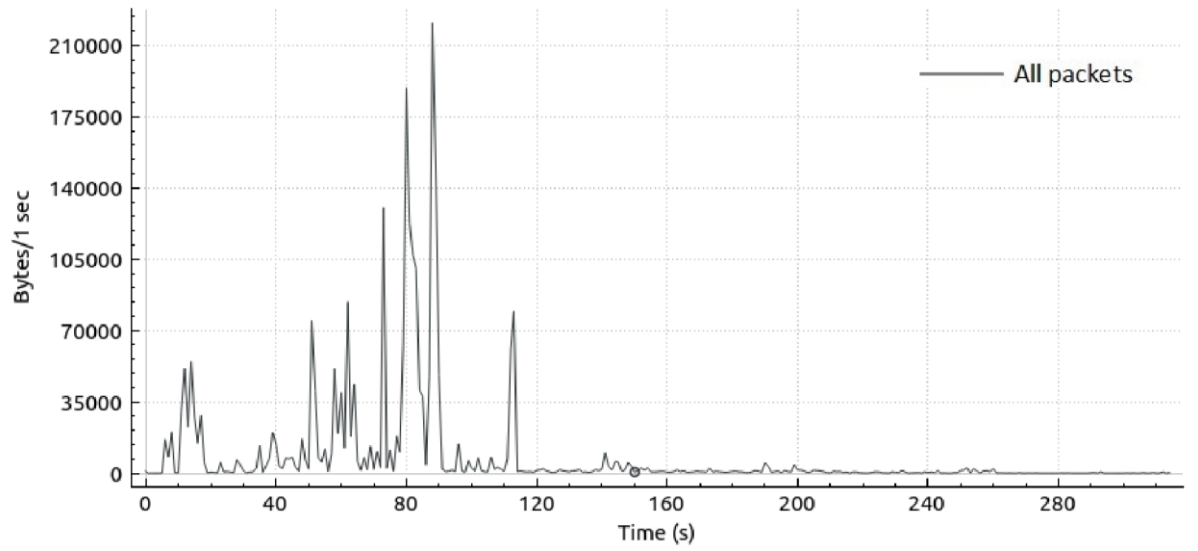


**Figure 3.4** Benign TCP Byte rate

The byte rate indicates the amount of data in bytes sent over the channel. This translates into the bandwidth of the network channel being used. The highest recorded data rate was about 220kB/s. Traffic rate was below 10kB/s for majority of the time. The recorded average byte rate was 8,116B/s (8kB/s).

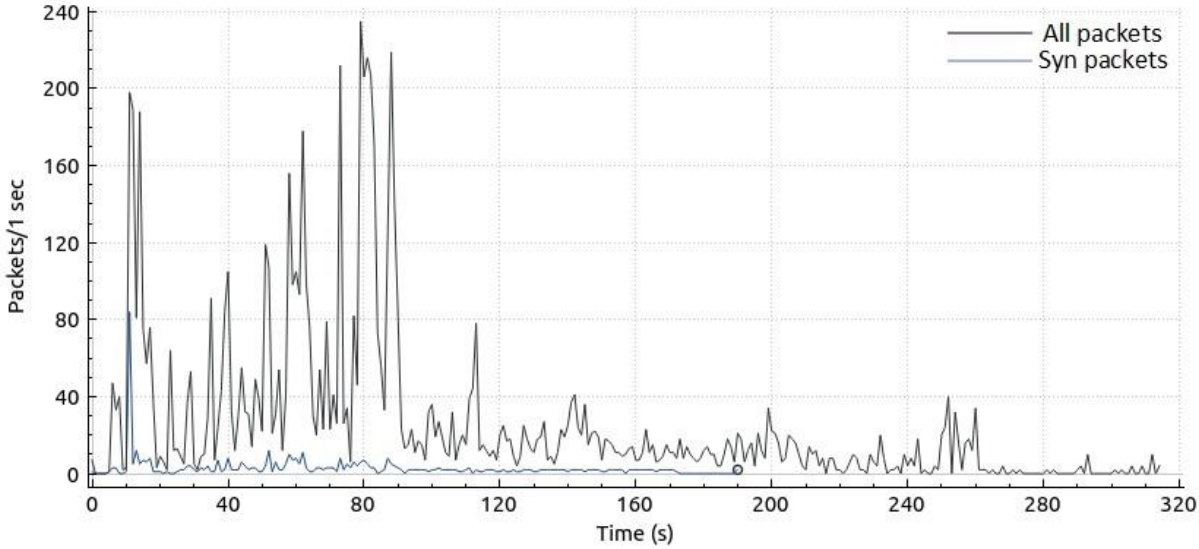


**Figure 3.5** Benign TCP packet and Syn rate

The packet rate indicates the rate at which traffic packets are sent across the network channel. The higher the number of packets, the more time it takes to process packets. For syn packets, a

high number per second indicates a high number of connection requests from the source. Benign traffic usually contains few syn packets per second, however, attack traffic usually contains numerous syn packets per second. This is to exploit the nature of the TCP protocol in which a connection established is left opened for a while before being closed either by the source or the destination when there is no activity over that connection. A high number of connection requests can also be observed by the number of ports used by a single source to request connections within a short period.

The average packet rate for the benign traffic captured as seen in figure 3.5 was 25 packets per second with the peak at 232 packets per second. The number of syn packets recorded over the entire five-minute period was also less than 300, approximately 1 per second.

Figure 3.6 was included due to the nature of TCP traffic produced by LOIC tool. The figure shows the rate of Push-Ack packets compared with total packets sent over the network. PushAck packets indicate payload data is being sent over the channel. TCP flooding DoS attacks are generally expected to request more connections than send actual payload data to exploit the three-way handshake. Push-Ack packets are expected to be the majority of packets in a TCP connection, indicating actual payload data is being sent as seen in figure 3.6.

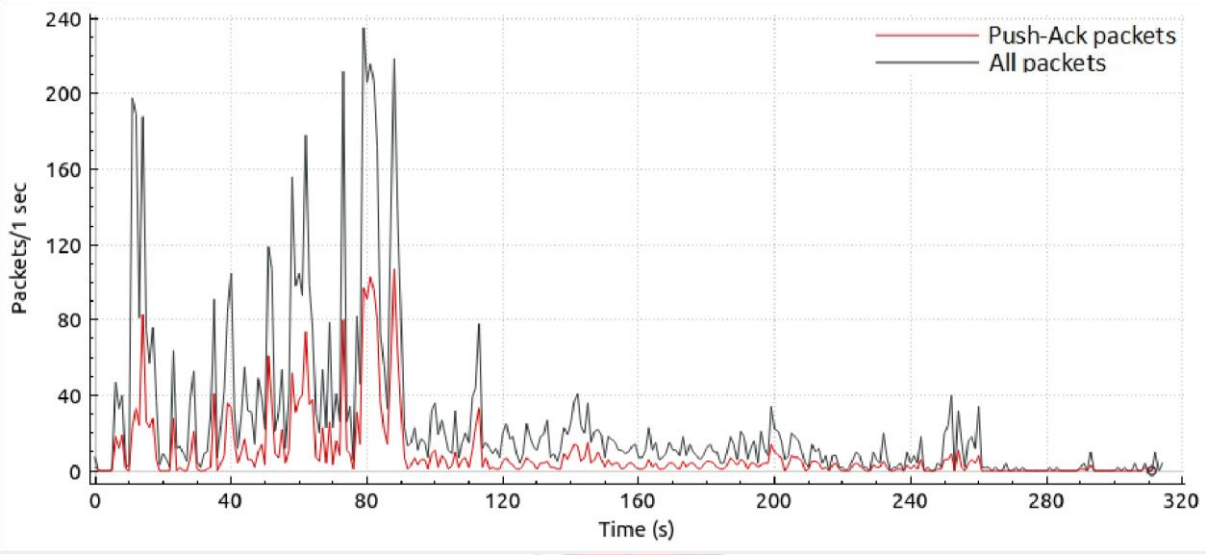


**Figure 3.6** Benign TCP packet and push-ack rate

*3.8.1.1 Anoncannon*

Figure 3.7 shows the traffic byte rate of Anoncannon tool. The distribution of bandwidth usage is uneven with no bandwidth used at certain times. The average byte rate recorded however was 188kB/s, over 20 times that of benign traffic captured.

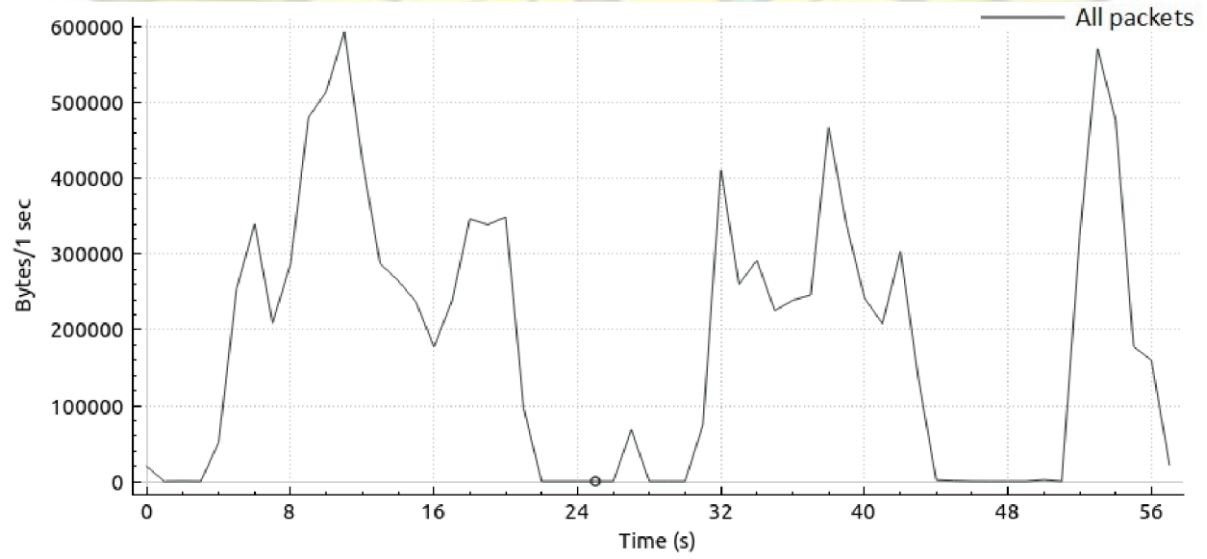


**Figure 3.7** Anoncannon Byte rate

Packet rate of Anoncannon tool as seen in figure 3.8 was irregular with random spikes and periods of inactivity. Syn packet rate was relatively high, with 106 syn requests recorded in the first second. Syn packet rate increased to as high as close to 2000 syn packets per second.

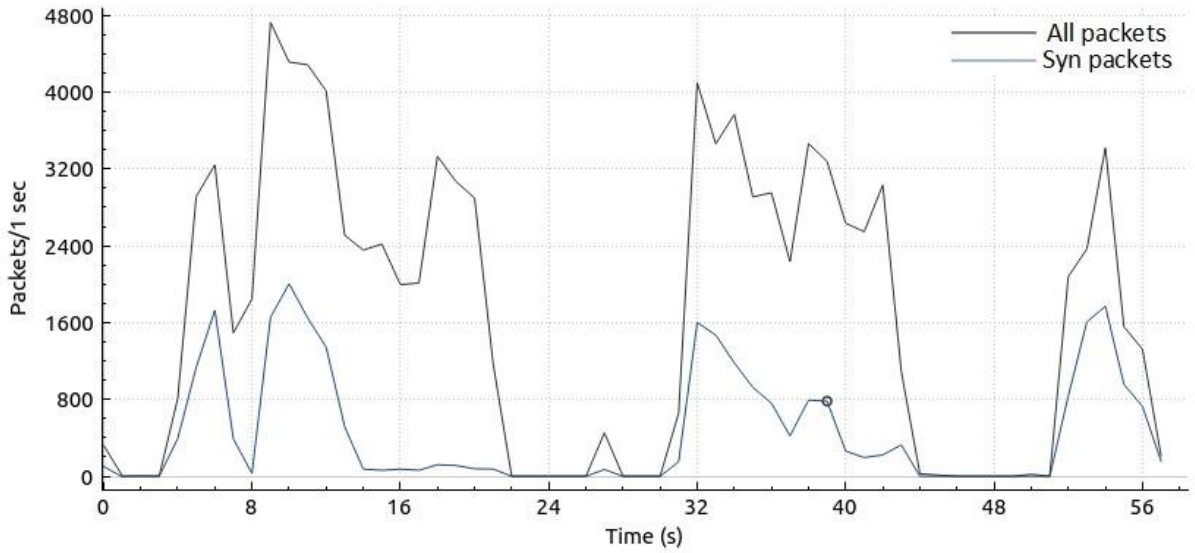


**Figure 3.8** Anoncannon packet and Syn rate

*3.8.1.2 ByteDoS*

Figure 3.9 shows the traffic byte rate of ByteDoS tool. Bandwidth use rose steadily to about 20kB/s and hovered around the same rate with slight increases and decreases. The average byte rate recorded was 19kB/s, which was relatively low compared to other attack tools.

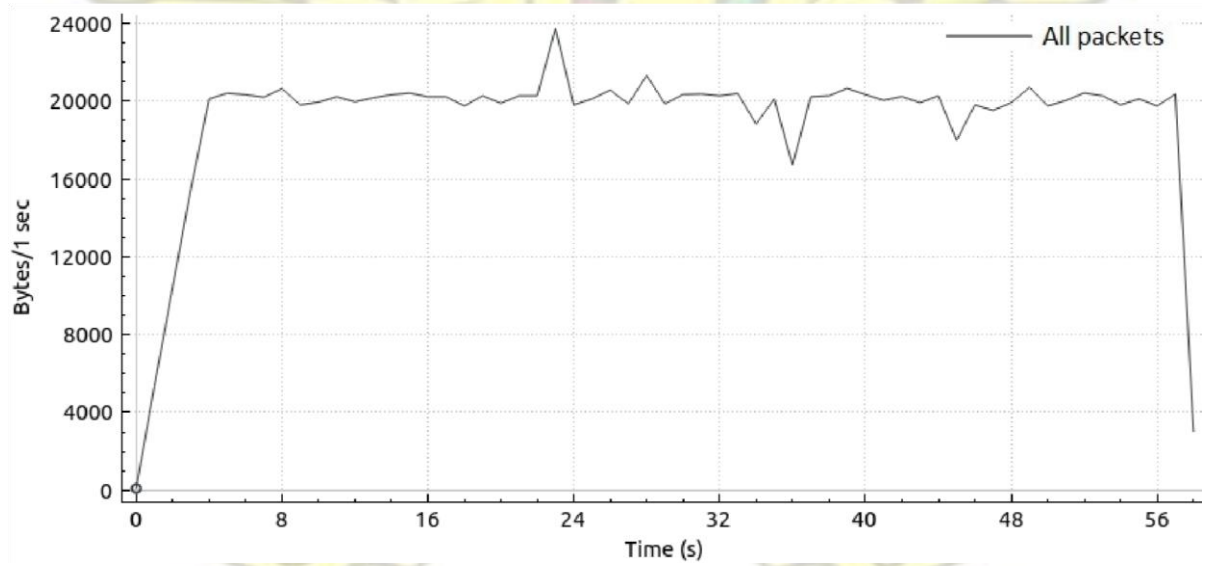


**Figure 3.9** ByteDoS Byte rate

Packet rate of ByteDoS tool as seen in figure 3.10 was relatively stable with very few deviations. Syn packet rate was also relatively stable, with an average of close to 60 syn per minute.

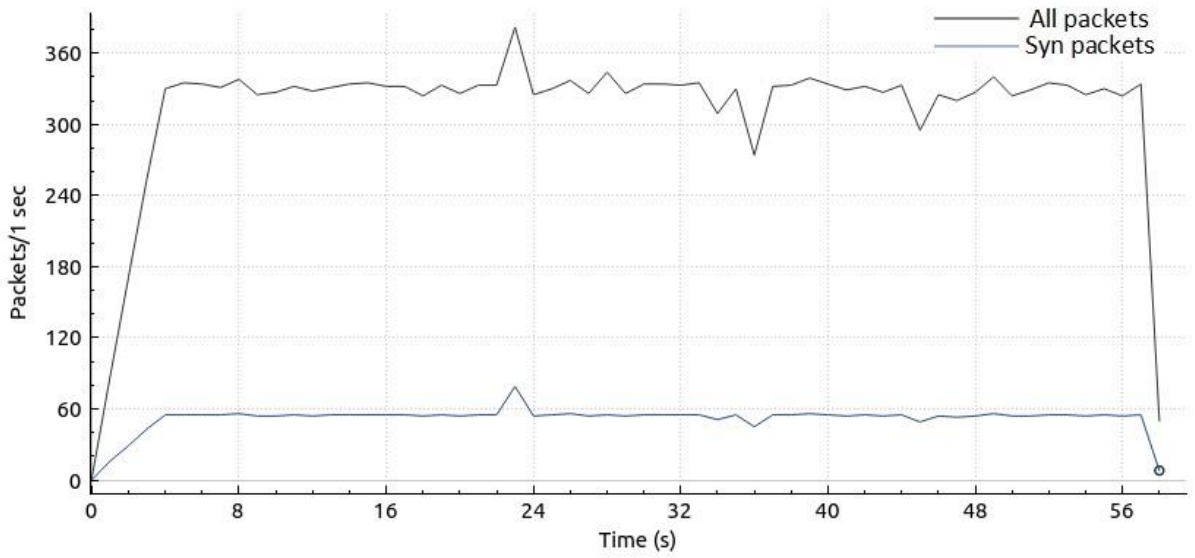


**Figure 3.10** ByteDoS packet and Syn rate

*3.8.1.3 FHRITP*

Of the TCP attacks tools studied, FHRITP produced the least traffic even after multiple instances of it were run. Traffic was generated at an average of 8kB/s, similar to benign traffic. Regular peaks were produced. It was however noticed that the tool did not always produce traffic when run. Figure 3.11 shows the byte rate of the tool.

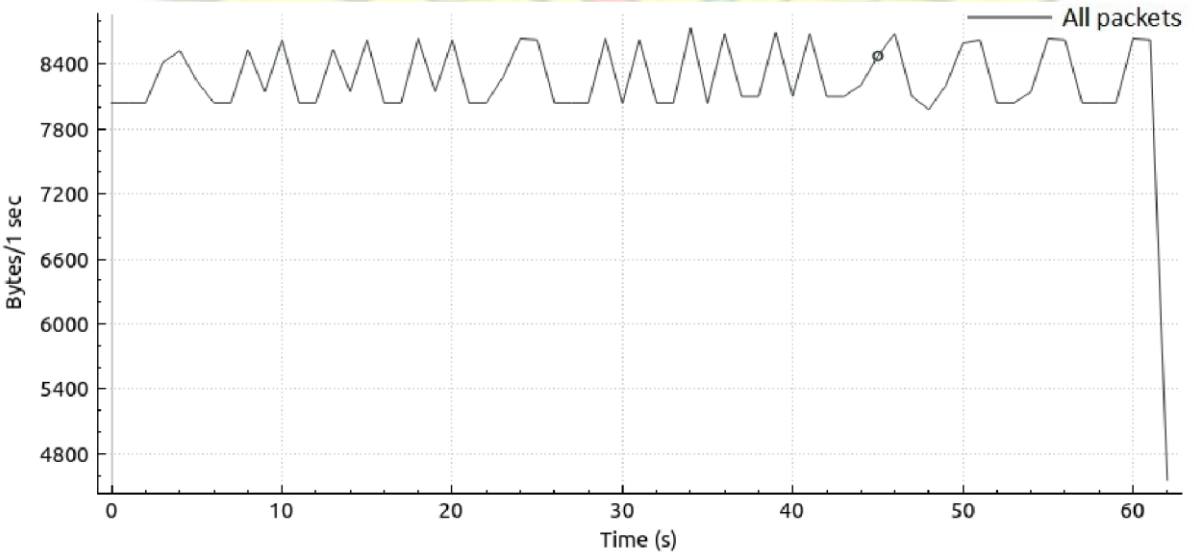


**Figure 3.11** FHRITP Byte rate

Packet rate of FHRITP tool as seen in figure 3.12 was fairly constant. An average of 137 packets per second was recorded. Syn packet rate was also constant at 67 packets per second.

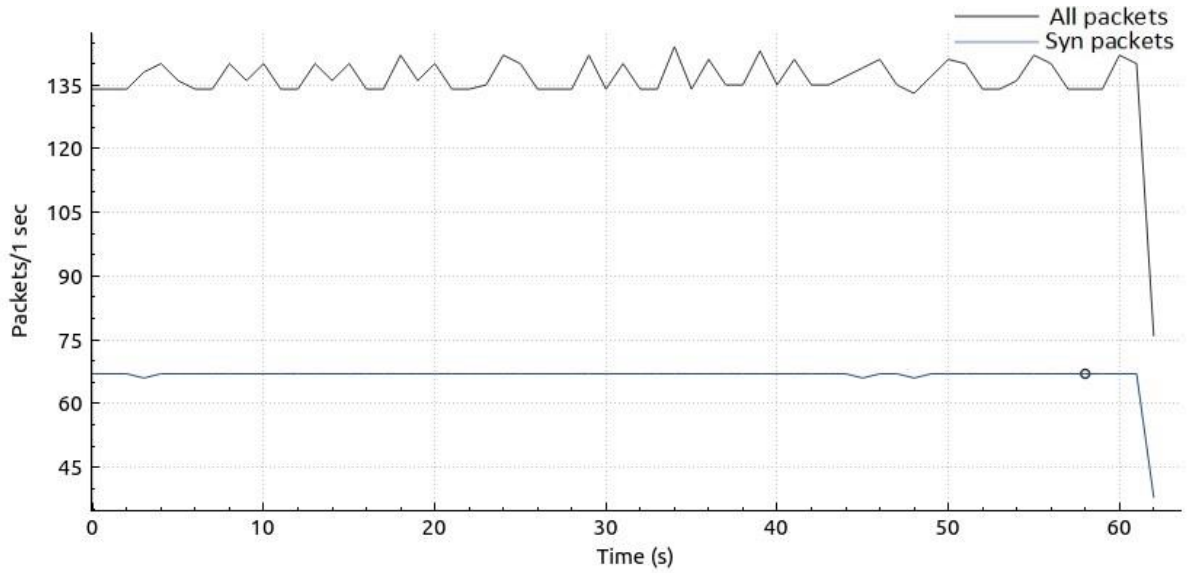


**Figure 3.12** FHRITP packet and Syn rate

*3.8.1.4 HOIC*

HOIC DoS attack tool produced a constant rate of traffic for about 46 seconds before decreasing and increasing sharply. This was as a result of the tool crashing around the forty sixth second. Figure 3.13 shows the traffic byte rate of HOIC tool. The average byte rate recorded was 4.8MB/s.

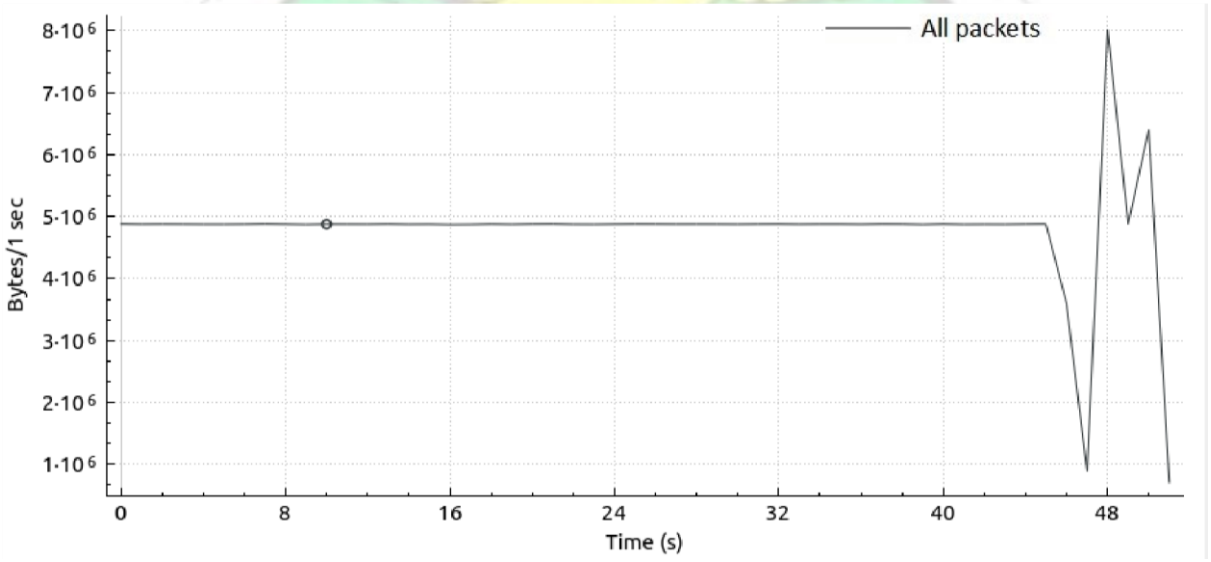


**Figure 3.13** HOIC Byte rate

Total packet rate of HOIC tool was constant with the sharp decrease and increase at the end. Syn packet rate was also constant. An average of 137 packets per second was recorded. Syn

packet rate was also constant at 67 packets per second. Figure 3.14 shows the total packet and syn packet rate.

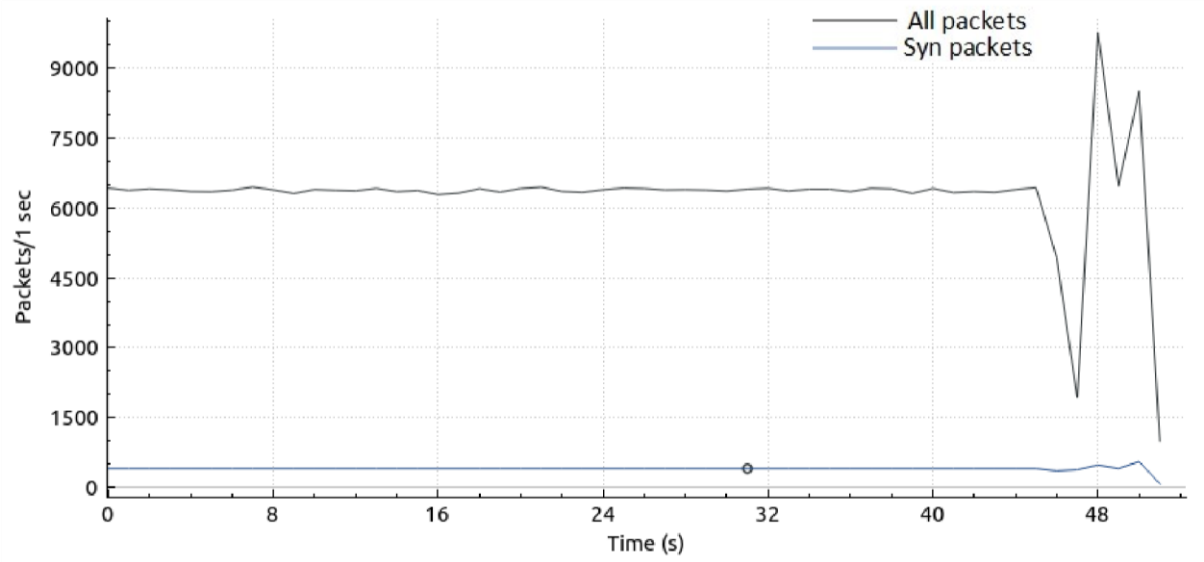


**Figure 3.14** HOIC packet and Syn rate

*3.8.1.5 XOIC TCP*

XOIC DoS attack tool produced traffic largely at a rate between 300kB/s and 380kB/s and 5000-6000 packets per second. The pattern of traffic generation however was random. Syn packets were also generated at a high and fairly constant rate, mostly hovering around 1000 packets per second. Figure 3.15 shows the traffic byte rate of XOIC tool. The average byte rate recorded was 333kB/s. Total packet rate and syn packet rate is shown in figure 3.16. The average packet rate recorded in the study was 5469 packets per second.

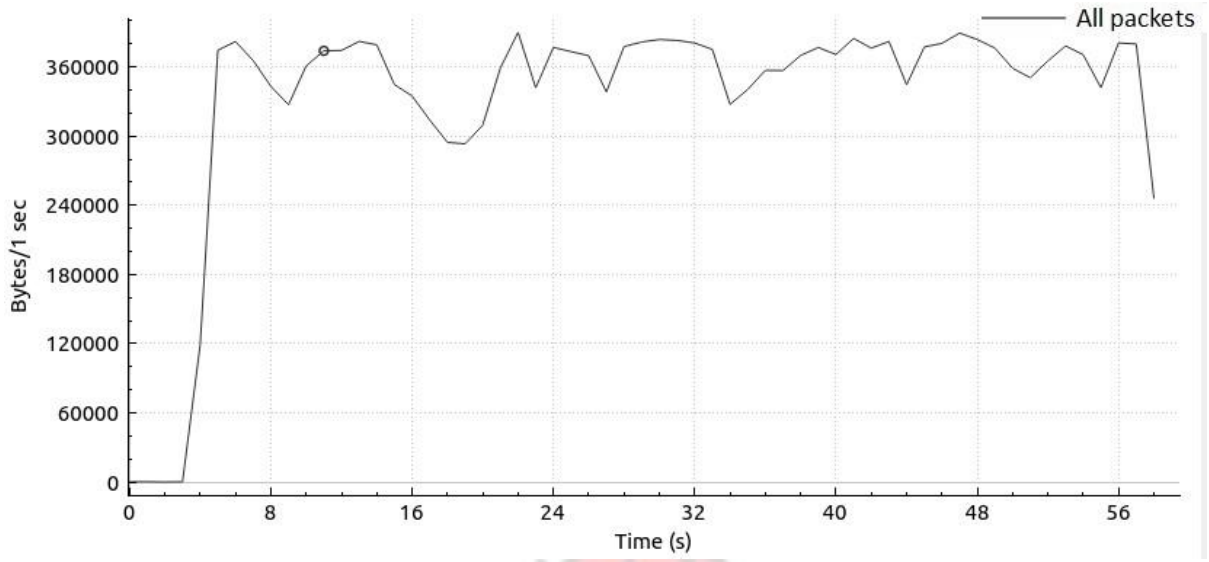


**Figure 3.15** XOIC TCP Byte rate

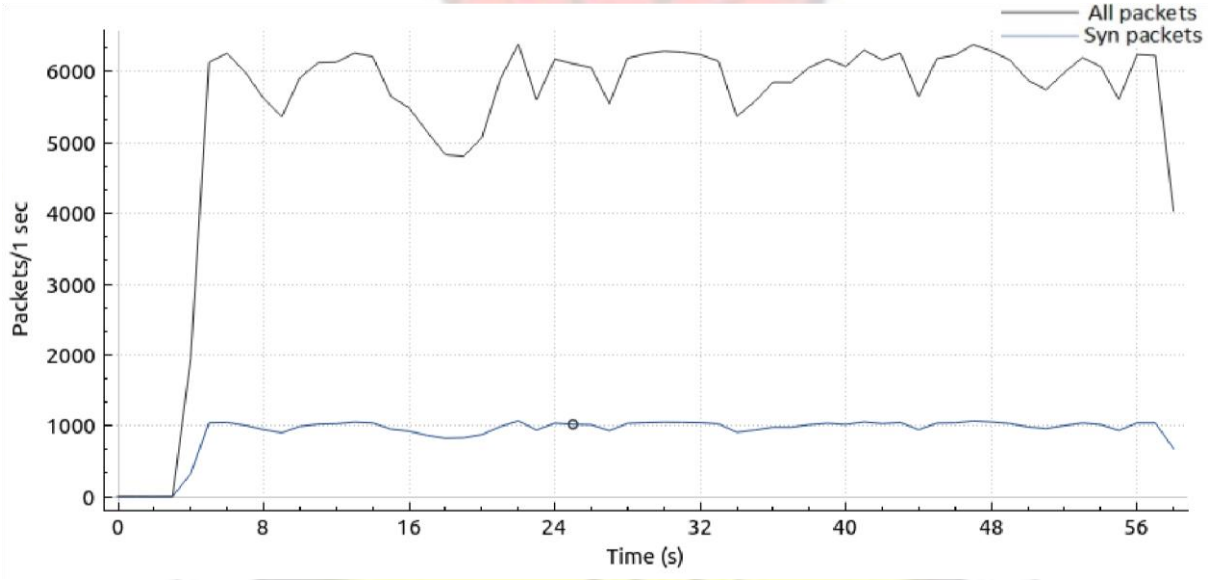


**Figure 3.16** XOIC TCP packet and Syn rate

*3.8.1.6 Moihack TCP*

The Moihack tool produced traffic with no regular pattern, with very significant traffic rate changes at three points. The average byte rate recorded was 2.8MB/s and the average packet rate was 7931 packets per second. Syn packet rate was also high, averaging about 1000 syn packets per second. Figure 3.17 shows the traffic byte rate of Moihack tool and total packet rate and syn packet rate is shown in figure 3.18.

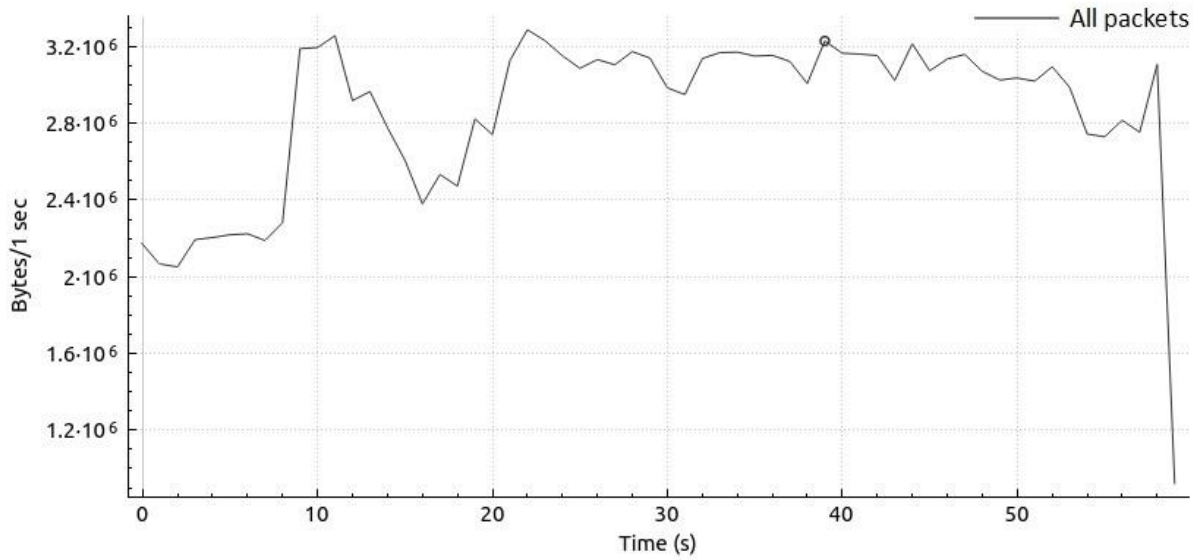


**Figure 3.17** Moihack TCP Byte rate

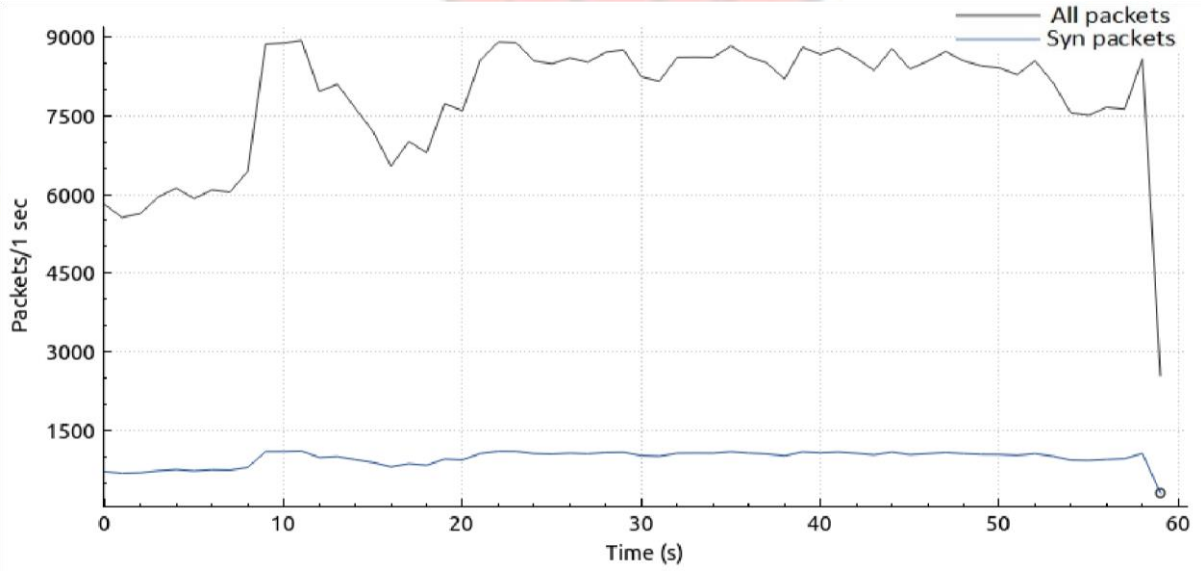


**Figure 3.18** Moihack TCP packet and Syn rate

*3.8.1.7 LOIC TCP*

The LOIC DoS attack tool produced traffic with several spikes. A very large amount of data was transmitted over a short period of time as well. Due to this, traffic was captured for 30 seconds. It was noted that even though the packet rate was very high, averaging 11,829 packets per second over 30 seconds, the syn packet rate was extremely low, with only 32 syn packets sent to the victim. This indicated the tool was focused on sending out large volumes of data over few connections rather than opening connections and closing them without sending data

or sending very little data. The Push-Ack packet rate was therefore included to compare the total packets sent with the number of data packets sent. Figure 3.21 shows the total packet rate and push-ack packet rate. The traffic byte rate is shown in figure 3.19. The average byte rate recorded was 10MB/s. Total packet rate and syn packet rate is shown in figure 3.20.

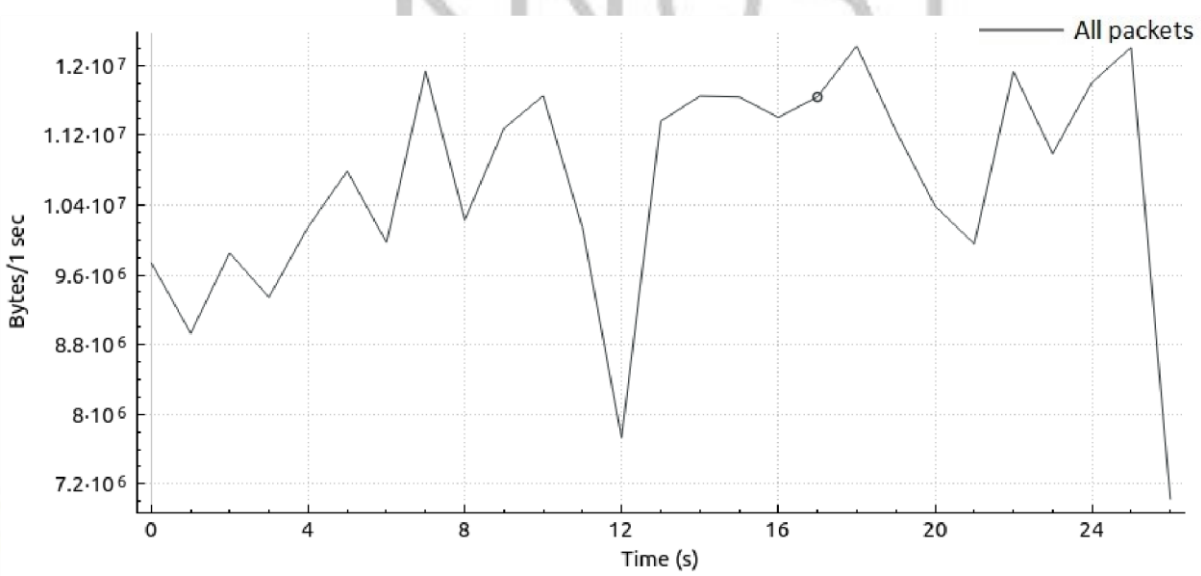


**Figure 3.19** LOIC TCP Byte rate

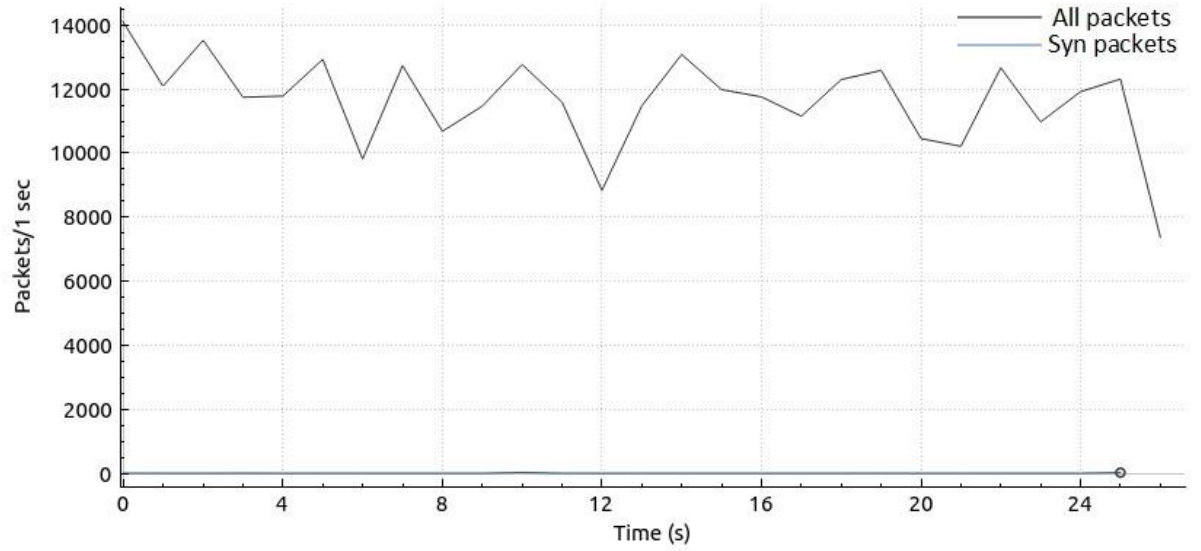


**Figure 3.20** LOIC TCP packet and Syn rate

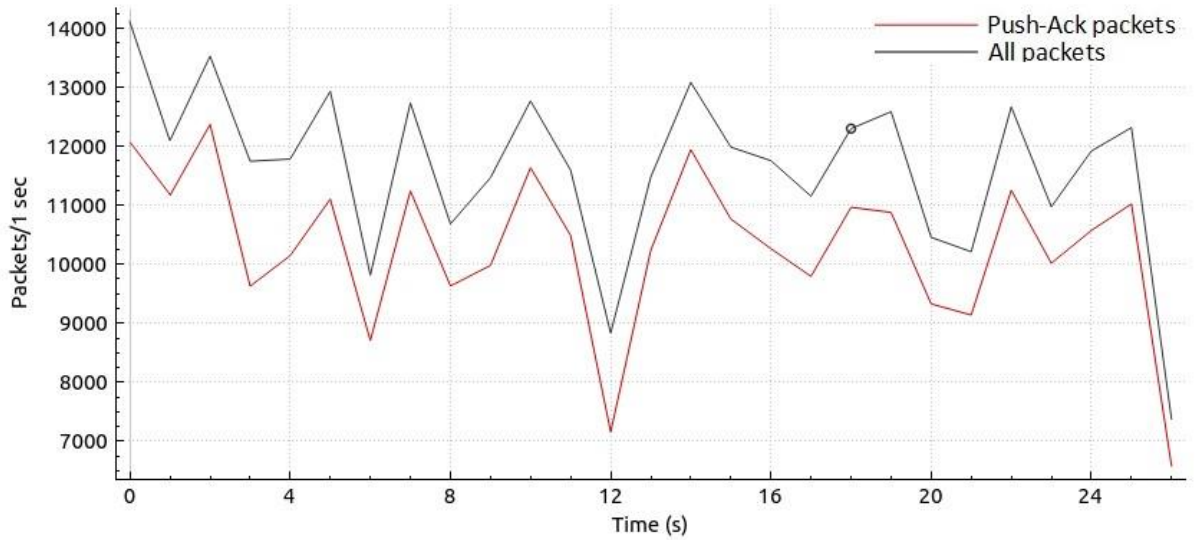


**Figure 3.21** LOIC TCP packet and push-ack rate

### 3.8.2 UDP Traffic

Figure 3.22 shows a sample of benign UDP traffic byte rate captured from browsing several pages on the internet. Figure 3.23 shows the packet rate. Traffic was captured for approximately five minutes. The benign packet rate was recorded at an average of 42 packets per second and byte rate had an average of 516B/s.

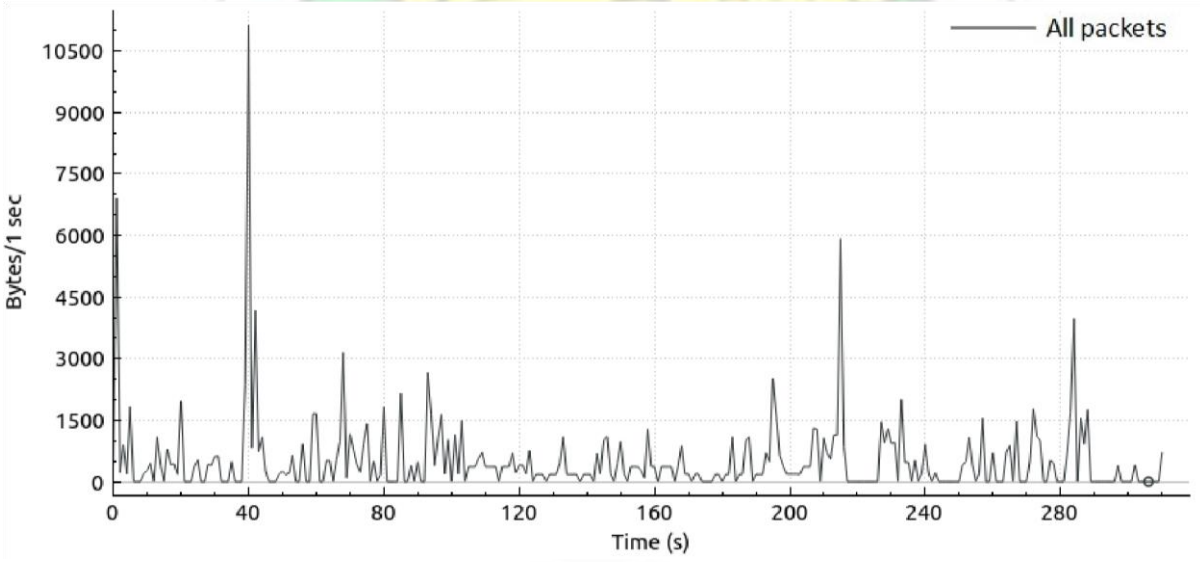


**Figure 3.22** Benign UDP Byte rate

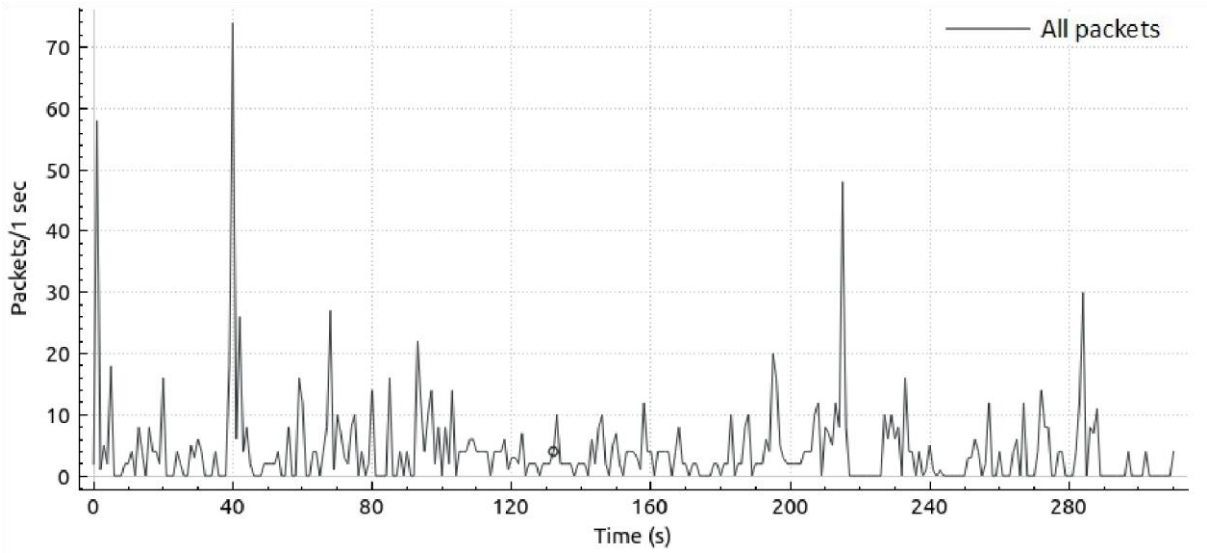


**Figure 3.23** Benign UDP packet rate

*3.8.2.1 LOIC UDP*

The LOIC tool sent packets at a very high rate, averaging 33,221 packets per second. Byte was also high at an average of 2.4MB/s. Traffic byte rate remained above 1MB/s for most of the period, dropping below that only once throughout the capture period. Figure 3.24 shows the byte rate and figure 3.25 shows the packet rate.

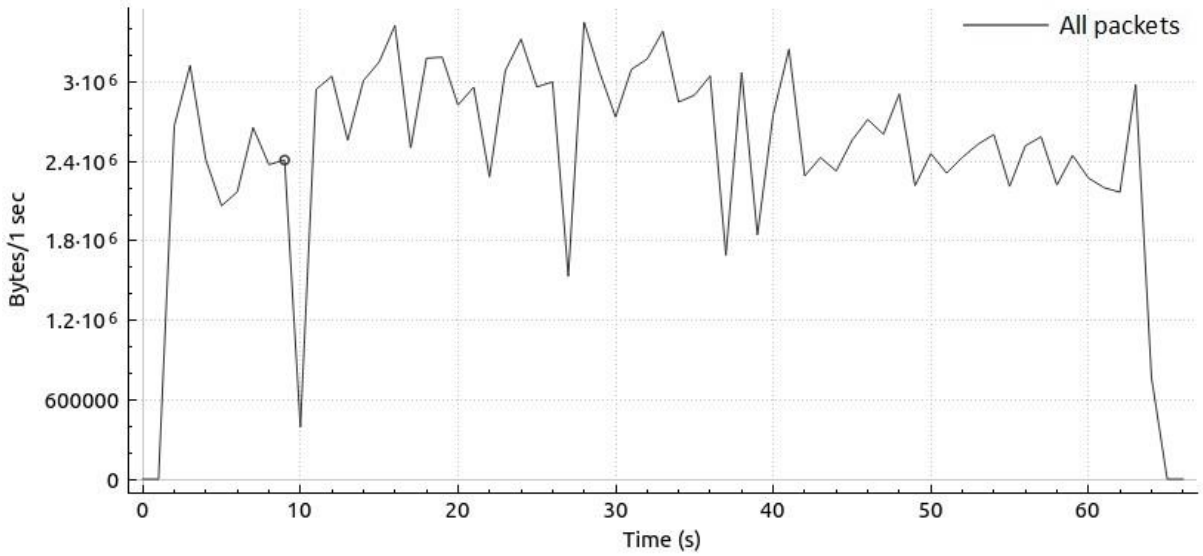


**Figure 3.24** LOIC UDP Byte rate

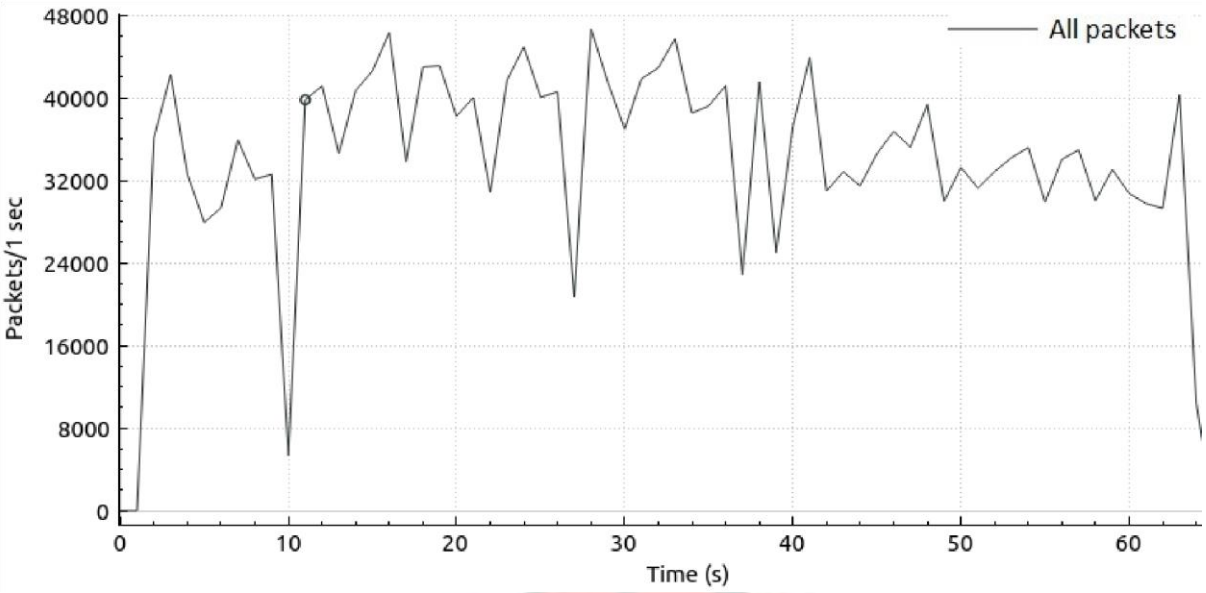


**Figure 3.25** LOIC UDP packet rate

*3.8.2.2 DoS tools by Noe*

This tool produced traffic packets at a very low rate, lower than benign traffic. 10 packets per second at 600B/s was generally the baseline, rising and falling back almost immediately throughout the entire capture period. Average packet rate and byte rate recorded was 13 packets per second and 876B/s respectively. Figure 3.26 shows the byte rate and figure 3.27 shows the packet rate.

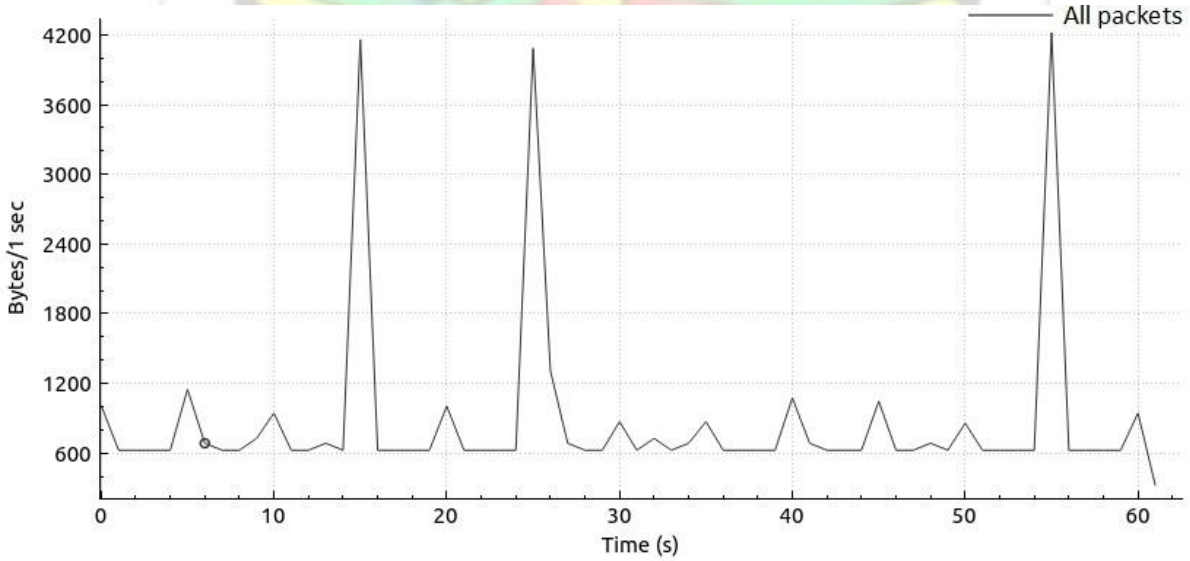


**Figure 3.26** DoS Tools by Noe UDP Byte rate

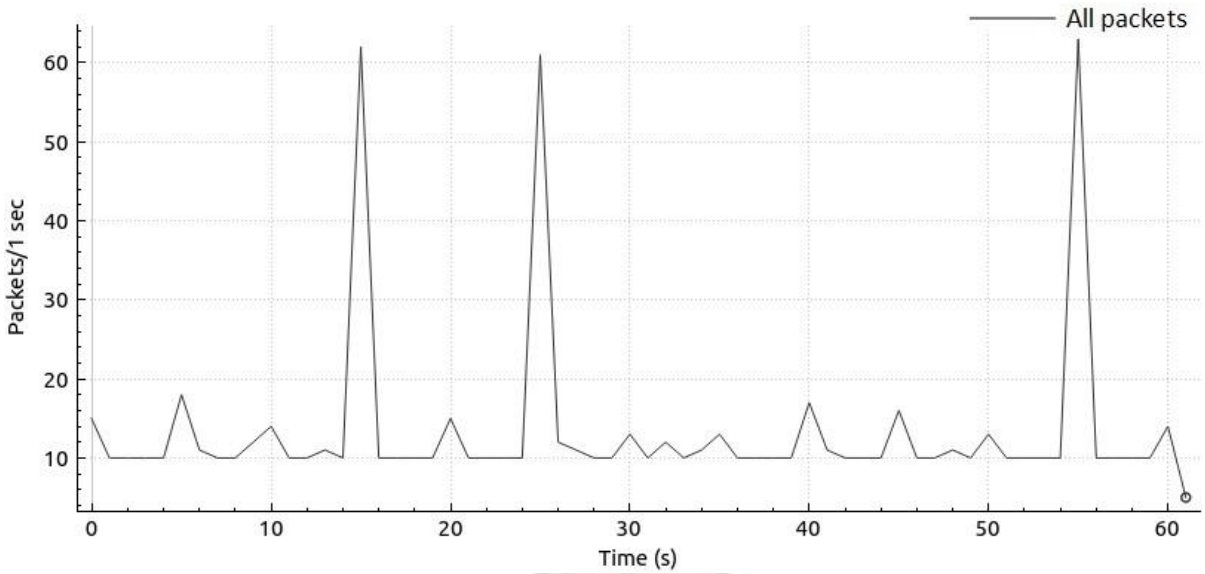


**Figure 3.27** DoS tools by Noe UDP packet rate

*3.8.2.3 Moihack UDP*

The Moihack tool generated some huge sized packets which had to be fragmented. Packet fragmentation is used by some DoS attacks to force the victim to use resources when reassembling packets. The Moihack tool fragmented its packets into two or three during the capture period. Average traffic rates were recorded at 3.6MB/s and 3,093 packets per second. Figure 3.28 shows the byte rate and figure 3.29 shows the packet rate.

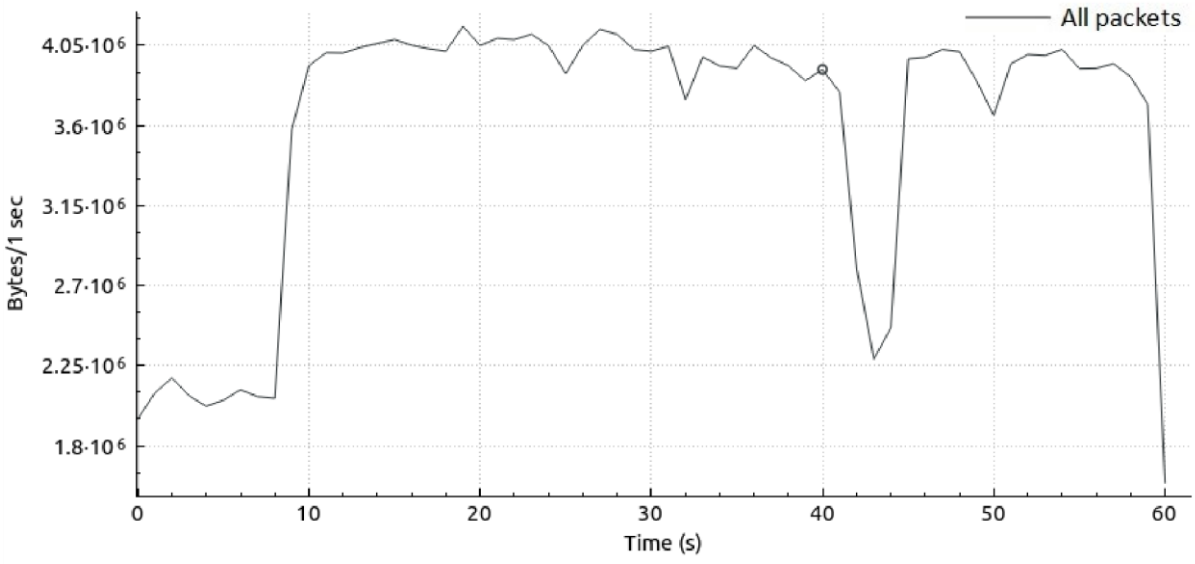


**Figure 3.28** Moihack UDP Byte rate

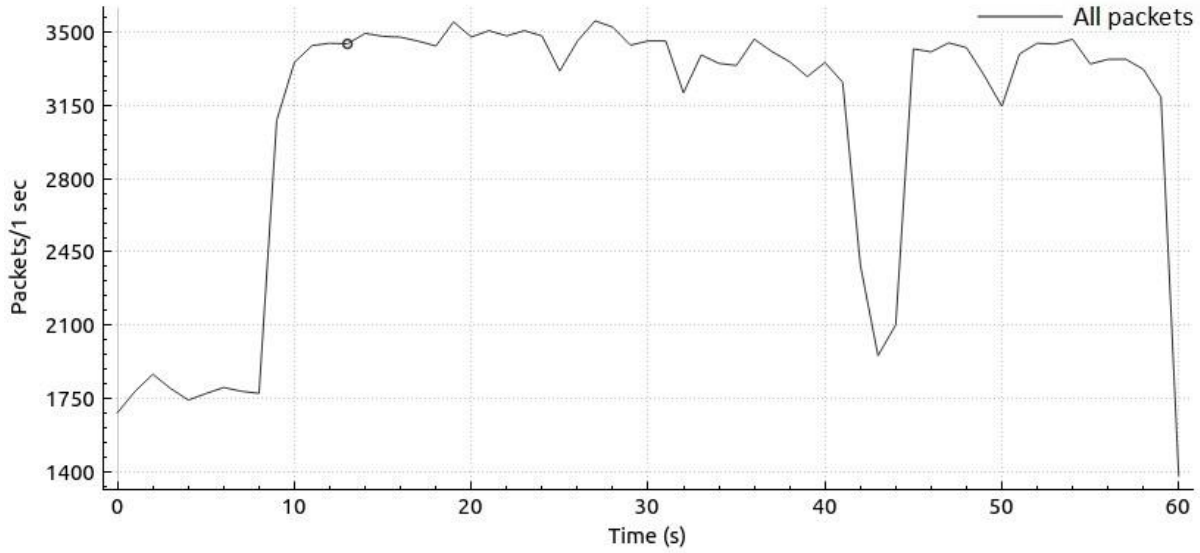


**Figure 3.29** Moihack UDP packet rate

*3.8.2.4 XOIC UDP*

The XOIC tool rose above 4000 packets per second and 240kB/s and stayed there for the entire capture period, dropping below that and rising immediately just once. Average byte rate was recorded at 267kB/s and packet rate at 4451 packets per second. Figure 3.30 shows the byte rate and figure 3.31 shows the packet rate.

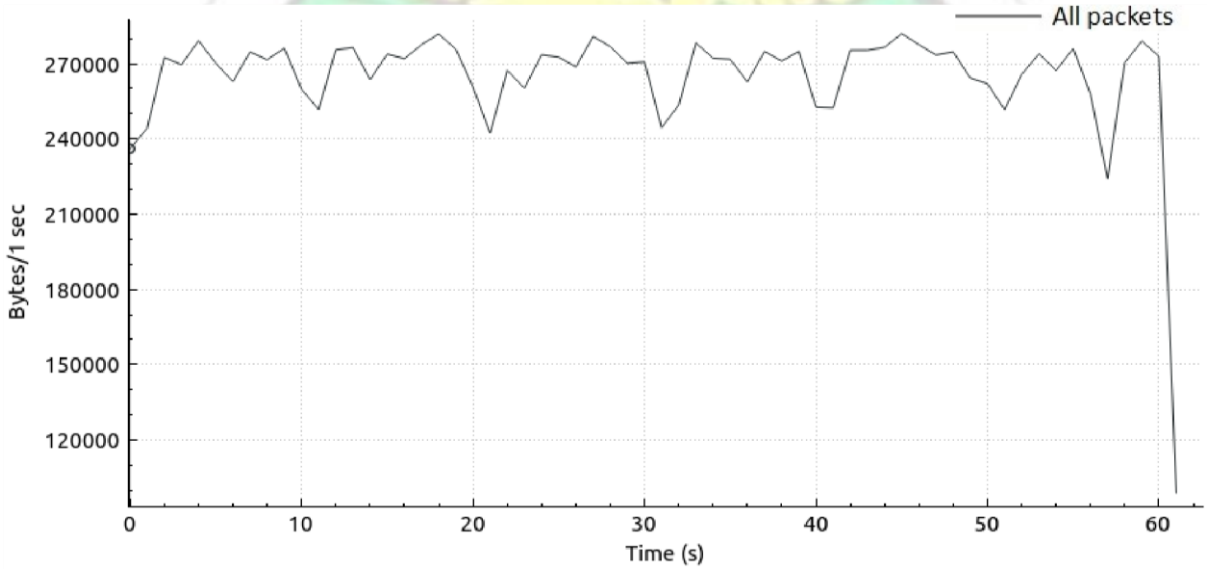


**Figure 3.30** XOIC UDP Byte rate

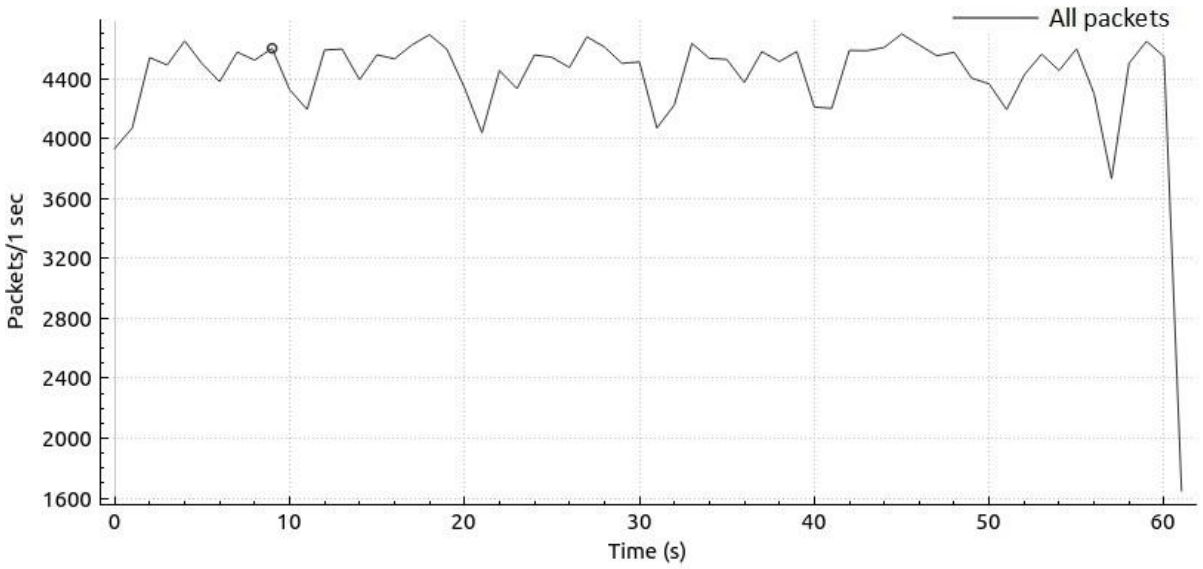


**Figure 3.31** XOIC UDP packet rate

### 3.8.3 ICMP Traffic

Figure 3.32 shows a sample of benign ICMP traffic byte rate captured during a ping session. Figure 3.33 shows the packet rate. Traffic was captured for approximately four and a half minutes. The benign packet rate was recorded at an average of 1563 packets per second and byte rate recorded an average of 79kB/s.

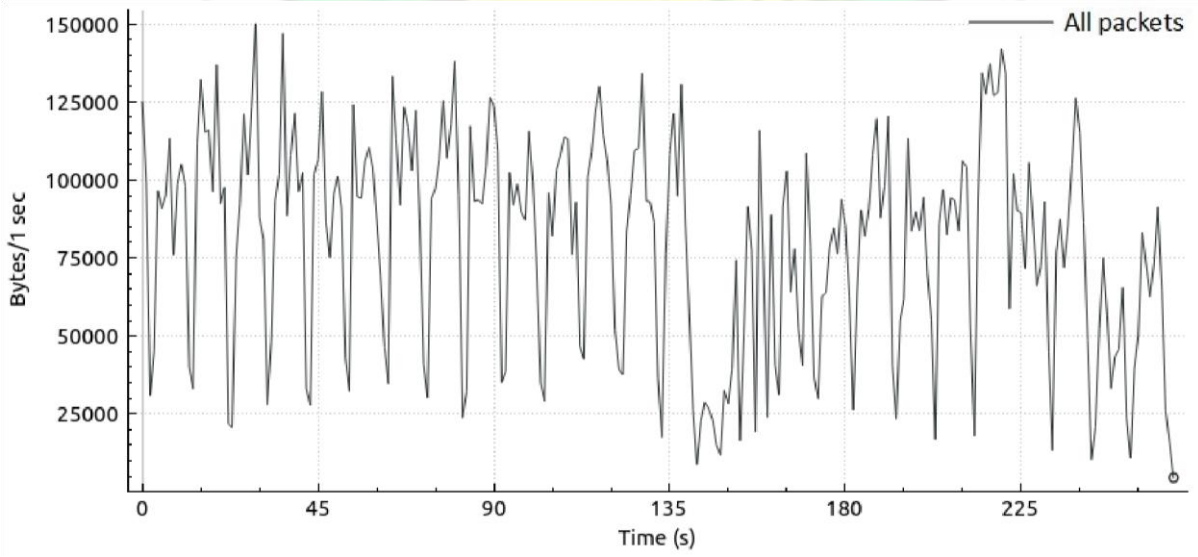


**Figure 3.32** Benign ICMP Byte rate

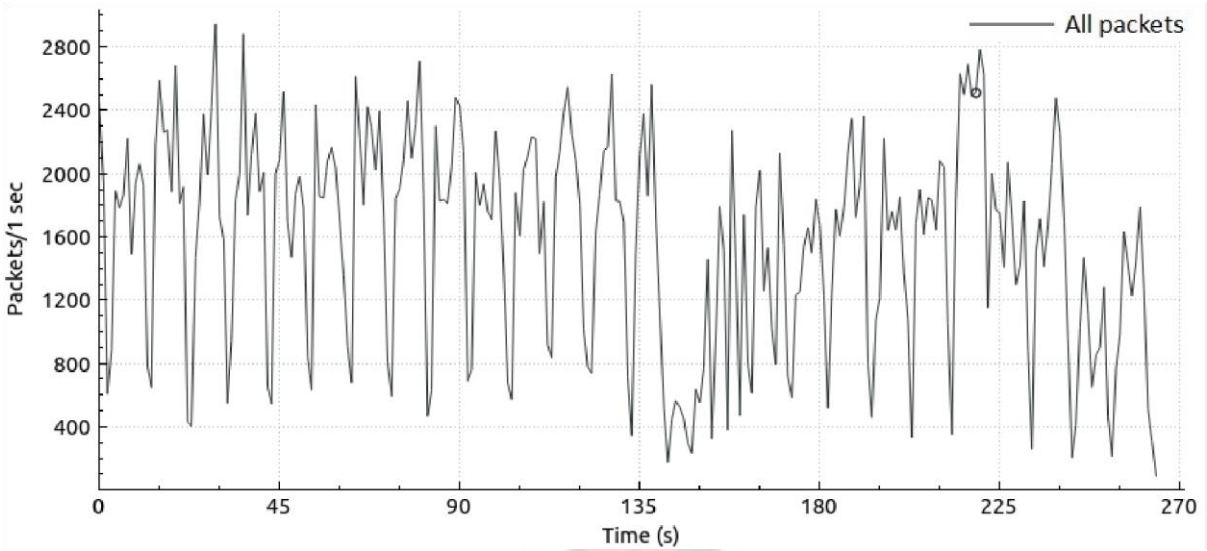


**Figure 3.33** Benign ICMP packet rate

*3.8.3.1 XOIC ICMP*

The XOIC tool generated packets at an average rate of 4,655 packets per second and an average byte rate of 237kB/s. For most part of the capture period, traffic was above 210kB/s and 4200 packets per second with only one noticeably sharp drop, which normalised almost immediately. Figure 3.34 shows the byte rate and figure 3.35 shows the packet rate of the XOIC DoS tool.

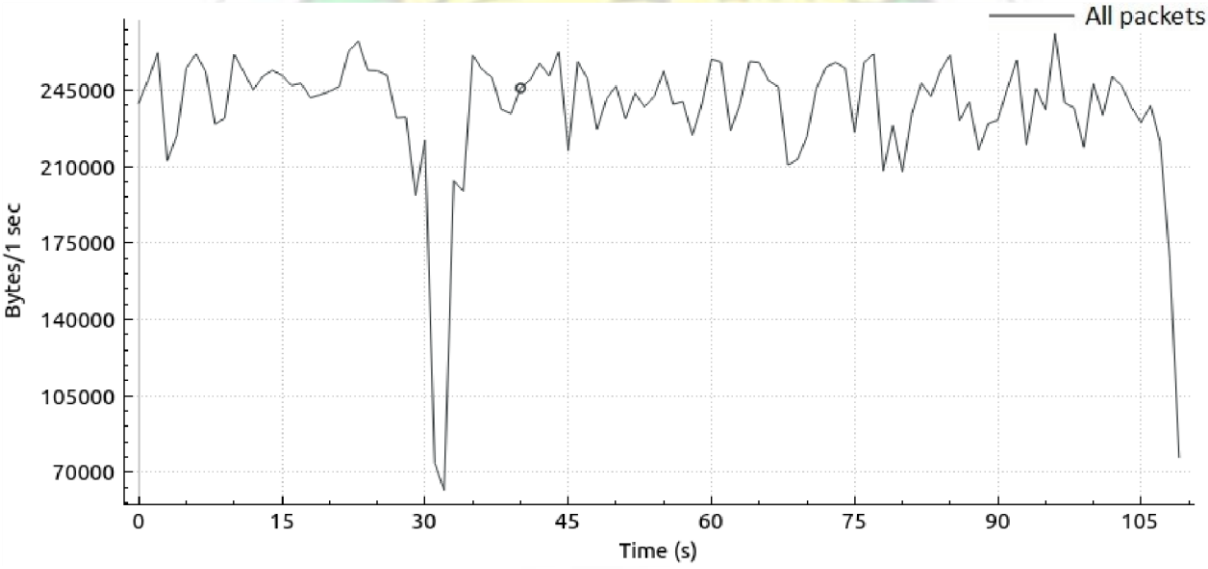


**Figure 3.34** XOIC ICMP Byte rate

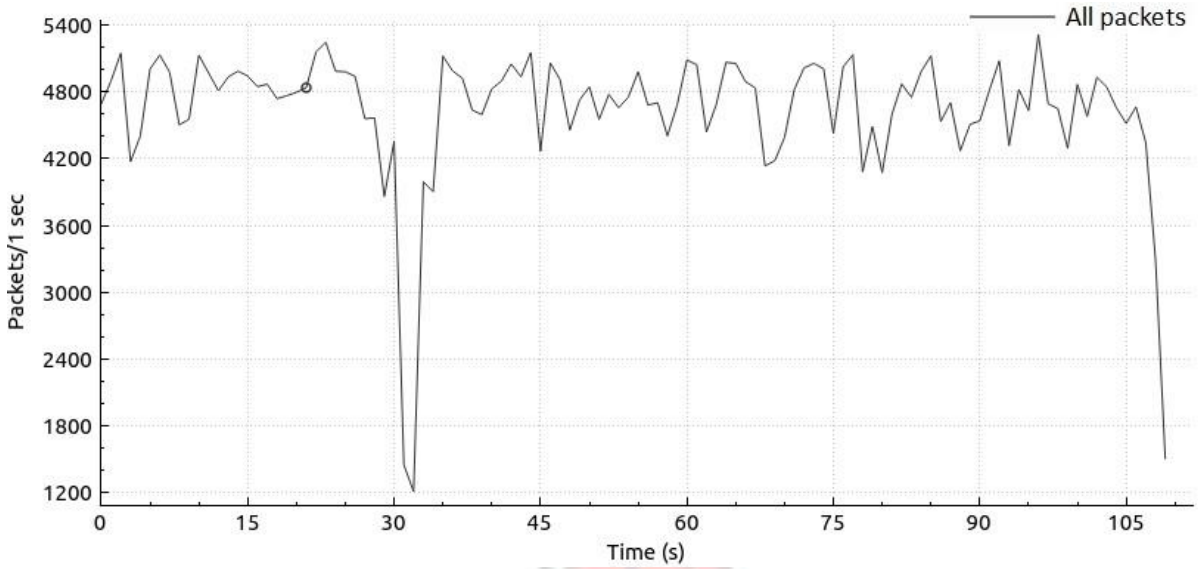


**Figure 3.35** XOIC ICMP Packet rate

Table 3.2 is a summary of the attack characteristics realised.

**Table 3.2** Summary of attack characteristics

| **Attack tool** | **Avg. packet rate** | **Avg. Byte rate** | **Avg. Packet size (Byte)** | **Comments** |
|---|---|---|---|---|
| Anoncannon TCP | 1700.2 | 188k | 111 | Connection request rate unpredictable.<br>Sends one instance of message string several times per connection granted |
| ByteDoS TCP | 315.9 | 19k | 61 | Requests for connections and closes without sending any data |
| FHRITP TCP | 136.9 | 8,282 | 60 | Requests for connections and closes without sending any data |
| HOIC TCP | 1703.1 | 307k | 180 | Open connections and requests default Apache web page<br>Sends no data |

| | | | | |
|---|---|---|---|---|
| XOIC TCP | 5469.2 | 333k | 61 | Requests for connections and closes without sending any data |
| LOIC TCP | 11,829.9 | 10M | 914 | Requests very few connections<br>Sends large number of packets with message repeated in packets a random number of times per connection granted |
| Moihack TCP | 7931 | 2,889k | 364 | Sends multiple packets with symbols, characters and numbers. |
| DoS tools by Noe UDP | 13.5 | 876 | 65 | Sends one instance of the message input by the user per packet<br>Uses several transmission channels to flood victim |
| LOIC UDP | 33,221.4 | 2,458k | 74 | Uses very few transmission channels to flood victim Sends one instance of the message in payload data |
| XOIC UDP | 4450.9 | 267k | 60 | Uses several transmission channels to flood victim Sends one instance of the message in payload data |
| Moihack UDP | 3,093.4 | 3,612k | 1168 | Sends sized randomly sized large packets which are fragmented<br>Uses several transmission channels to flood victim |
| XOIC ICMP | 4654.9 | 237k | 51 | Sends ICMP echo requests (pings) |

## 3.9 PROPOSED ALGORITHM

After analysing the attack traffic and determining the differences between attack traffic and benign traffic, a signature-based detection algorithm based on support-vector machine (SVM) classifier to detect attack traffic from the tools used in the research was proposed. SVM is a supervised data mining technique which uses traffic features to classify traffic as attack or benign. SVM is known to have high accuracy and low false positive rates, which are desired of a good detection scheme. This is possible because previously studied traffic features are used to develop a pattern for matching new traffic. The algorithm encompassed detection of TCP, UDP and ICMP attacks launched by the studied DoS tools and is presented in the next chapter.

### 3.9.1 Algorithm

Based on the differences noticed between the attack traffic and benign traffic, an algorithm was proposed. The algorithm incorporates detection for TCP, UDP and ICMP DoS attacks. The algorithm begins by monitoring the network channel for activity. If there is activity, the protocol of traffic is checked. Depending on the protocol of traffic detected, the portion of the algorithm for that protocol is run. Figure 3.36 shows the detection algorithm for ICMP traffic.

```
1 Monitor network channel for traffic activity
2 For (channel activity)
3    Determine traffic protocol
4         If (Traffic protocol == ICMP)
5              Check ICMP Type field
6              Check ICMP Code value
7              Check fragment bit in IP header
8                   If (ICMP type==8 && ICMP code==0 && Fragment bit==D)
9                   Check source IP address
10                  Check number of packets sent in time T seconds
11                       If (number of packets sent >= threshold)
12                            Alert ICMP flood attack
13                       Else go to 1
14                  Else go to 1
```

**Figure 3.36** ICMP attack detection algorithm

For ICMP traffic, the ICMP type field and code values are checked as well as the fragment bit in the IP header. The type field value of 8 and code value of 0 indicate an ICMP echo request. The fragment bit is checked to ensure it is set to Don't fragment, a characteristic of the XOIC ICMP tool. This is to ensure traffic matches exactly what is intended to be detected, to prevent processing of packets which may have some of these characteristics. If the conditions of the checks are met, the source IP address is checked and monitored for the number of packets it will send within the period T seconds. If the number of packets exceed the threshold number of packets expected within the time frame, an alert is given of an attack.

The flow chart of the process is shown in figure 3.37.

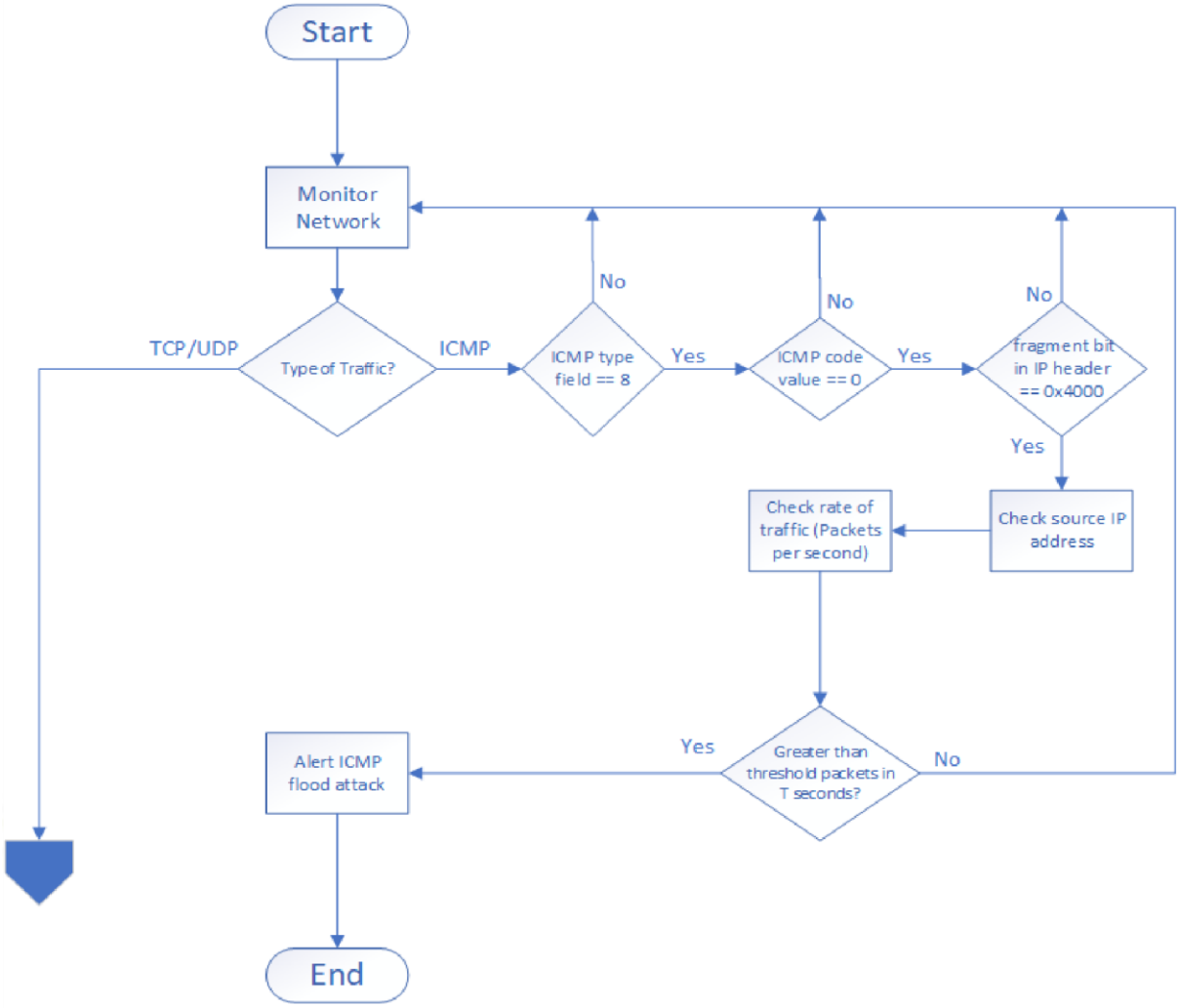


**Figure 3.37** Flow chart of ICMP attack detection algorithm

For UDP traffic, the port number is checked to ensure it is an ephemeral port. The fragment bit in the IP header is then checked. If the More fragments bit is set, it is further checked to see if it is the first fragment, which has a fragment offset set to 0. This is to ensure only the first packet in a series of fragmented packets is counted. If the More fragments and fragment offset is passed, the process moves to check the IP address of the source and monitors for the number of fragmented packets with offset set to 0 it sends within the period T seconds. If the More fragments condition fails, the process moves straight to check the IP address of the source and monitors for the number of packets it sends within the period T seconds. If the number of packets exceed the threshold number of packets expected within the time frame, an alert is

given of an attack. The algorithm is shown in figure 3.38 and the flow chart of the process is shown in figure 3.39.

```
15          Else If (Traffic protocol == UDP)
16                Check port number
17                If (Port > 1025)
18                      Check fragment bit in IP header
19                      If (fragment bit == M)
20                            Check fragment offset
21                            If (fragment offset == 0)
22                                  go to 26
23                            else
24                                  go to 1
25                      else
26                            Check source IP address
27                            Check number of packets sent in time T seconds
28                            If (number of packets sent >= threshold)
29                                  Alert UDP flood attack
30                            Else go to 1
31                Else go to 1
```

**Figure 3.38** UDP attack detection algorithm

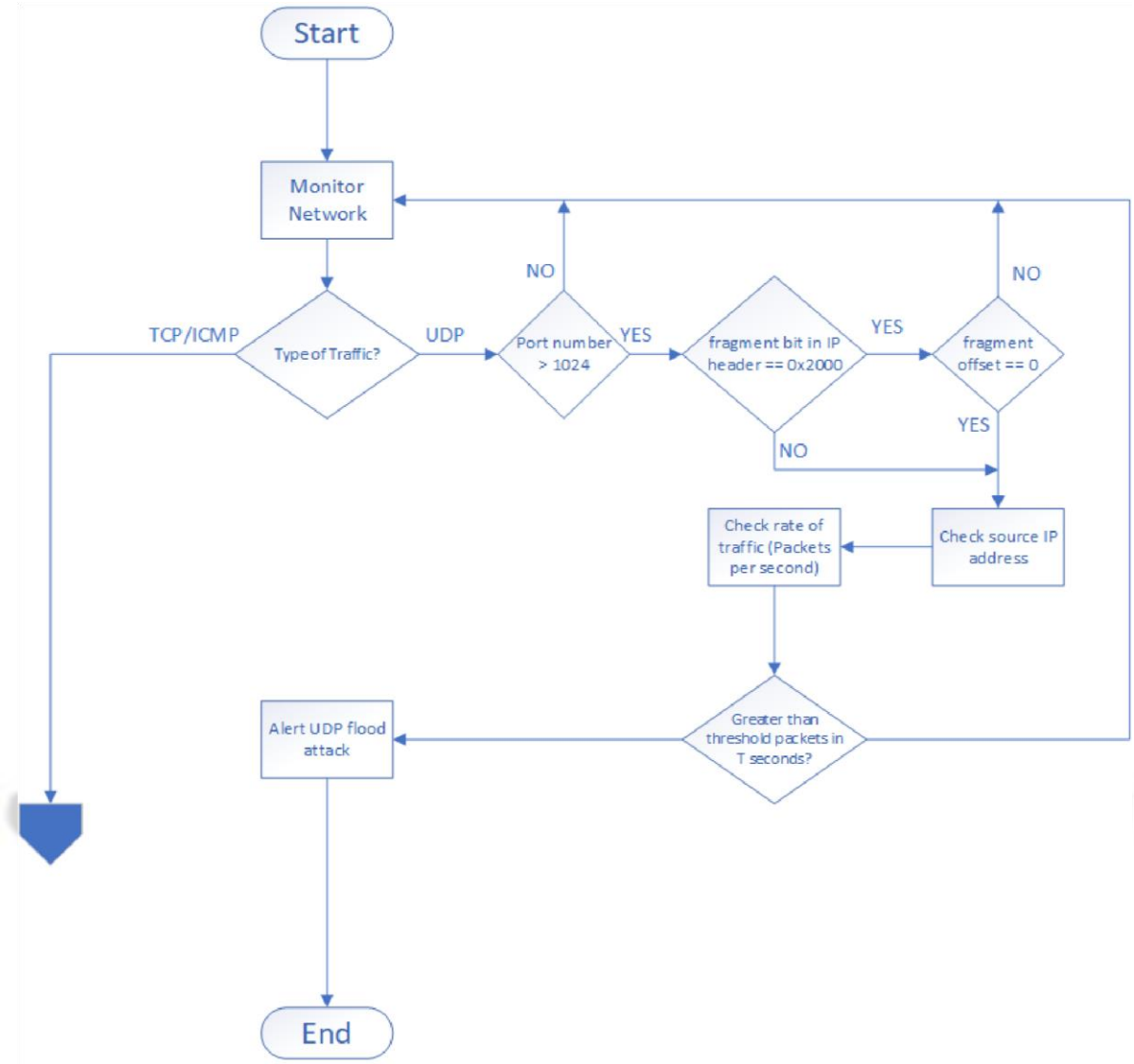


**Figure 3.39** Flow chart of UDP attack detection algorithm

For TCP traffic, the port number is checked to ensure it is an ephemeral port. The fragment bit in the IP header is then checked and the TCP flags are checked. If the fragment bit is set to

Don't fragment and the TCP flags is set to Syn or Push-Ack, the IP address of the source is checked and monitored for the number of packets it sends within the period T seconds. If the number of packets exceed the threshold number of packets expected within the time frame, an alert is given of an attack. The algorithm is shown in figure 3.40 and the flow chart of the process is shown in figure 3.41.

```
24      Else If (Traffic protocol == TCP)
25          Check Port number
26          Check Fragment bit in IP header
27          Check TCP flags
28              If (Port > 1025 && Fragment bit == D && TCP flag == S || PA )
29                  Check source IP address
30                  Check number of packets sent in time T seconds
31                      If (number of packets sent >= threshold)
32                          Alert TCP flood attack
33                      Else go to 1
34              Else go to 1
35  Else go to 1
```

**Figure 3.40** TCP attack detection algorithm

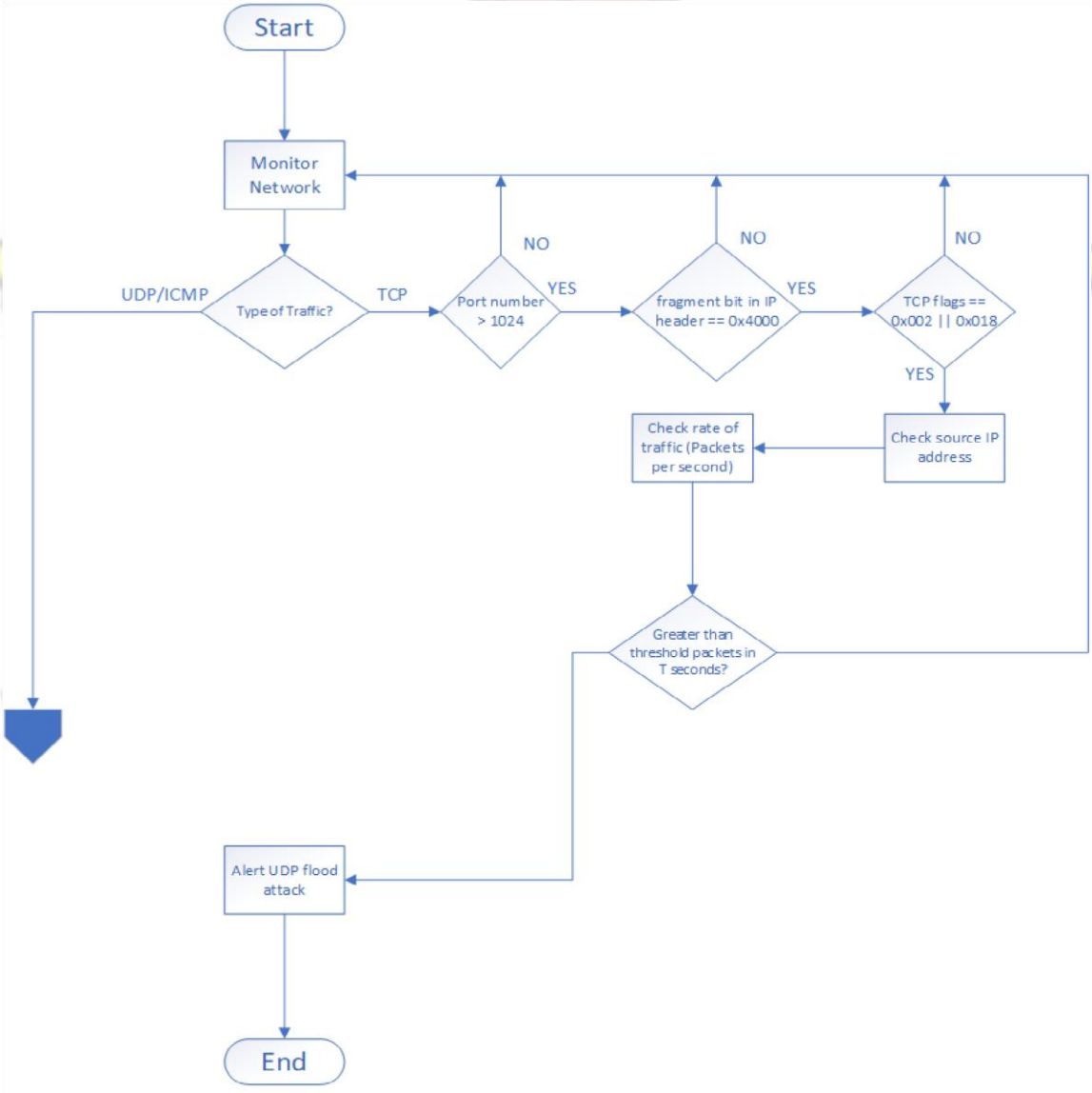


**Figure 3.41** Flow chart of TCP attack detection algorithm

### 3.10 TESTING

The algorithm was tested using Snort® IDS. Snort is an open source network intrusion detection system that performs real-time traffic analysis. The software uses rules created by the user to define anomalous traffic. The results of the test are presented in the next chapter.

# CHAPTER FOUR

# RESULTS AND DISCUSSION

## 4.1 INTRODUCTION

In this chapter, the results of testing and evaluating the algorithm are presented and discussed. The results of the detection scheme are also compared with existing DoS detection schemes.

## 4.2 DISCUSSION OF ATTACK FEATURES

After analysing traffic generated by the attack tools, it was realised that DoS attack traffic are increasingly becoming similar to benign traffic in terms of packet structure. The differences between TCP attack and benign traffic mostly was the number of connection requests. Attack traffic generally used several ports to request connections simultaneously. Normal TCP operations requests just one or very few connections for data transfer. Benign traffic captured had several but few connection requests, however, even those connection requests involved different IP addresses. LOIC TCP tool on the other hand was the only tool to request few connections. It focused on sending a lot of data over the few connections established to mimic benign traffic, making it nearly impossible to detect with the number of requests. The difference in volume of data transmitted was therefore used in detection. All the tools had a quite steady connection request rate except for the Anoncannon tool. This tool had a random connection request rate, requesting packets without any pattern. LOIC TCP had the highest bandwidth consumption at 10MB/s. All tools had the “Don’t fragment” bit set in their packets; however, some benign packets were also noticed to have this bit set as well. All ports used by the attack tools were ephemeral ports.

UDP attack traffic had no significant packet difference compared with the benign packets. The difference was in the rate of packets being sent and number of ports used in sending the packets, which was a lot compared to benign traffic, indicating the number of transmission channels

used in flooding the victim. The LOIC UDP tool however, used very few ports, just like with its TCP attack. It was also noticed that the UDP tools focused on sending smaller sized packets.

The XOIC ICMP tool sent many ICMP echo requests, more than what is sent in a normal ping session. The Don't fragment bit was also set, same as that of benign traffic.



## 4.3 EVALUATION

Snort IDS was used to test the algorithm. The detection rules written based on the algorithm were implemented and tested five times for each DoS attack to have a good idea of how fast detection of an attack was. The detection times are shown in figure 4.39. Benign traffic was also tested to determine false positive occurrences. The time (T) was set at 60 seconds. For ICMP traffic, the threshold number of packets was set at 6,000 packets. For UDP traffic, the threshold was set at 60, 000 packets for unfragmented traffic and 20,000 packets for fragmented traffic. For TCP traffic, the threshold for Syn packets was set at 600 packets and 30,000 for Push-ack packets. The thresholds were chosen to incorporate worst-case scenarios whilst using the victim machine for network access. After the first three tests, the final two tests were done using thresholds reduced to half the initial values to reduce the detection times and test for false positives. The detection method was compared with other detection methods proposed in literature.

## 4.4 DISCUSSION OF TESTS

The detection times during the runtimes of the algorithm and attack tools is shown in figure 4.1. Early detection of DoS attacks ensures they are stopped before causing a lot of damage. Of the seven TCP attacks tested, five were consistently detected under five seconds.

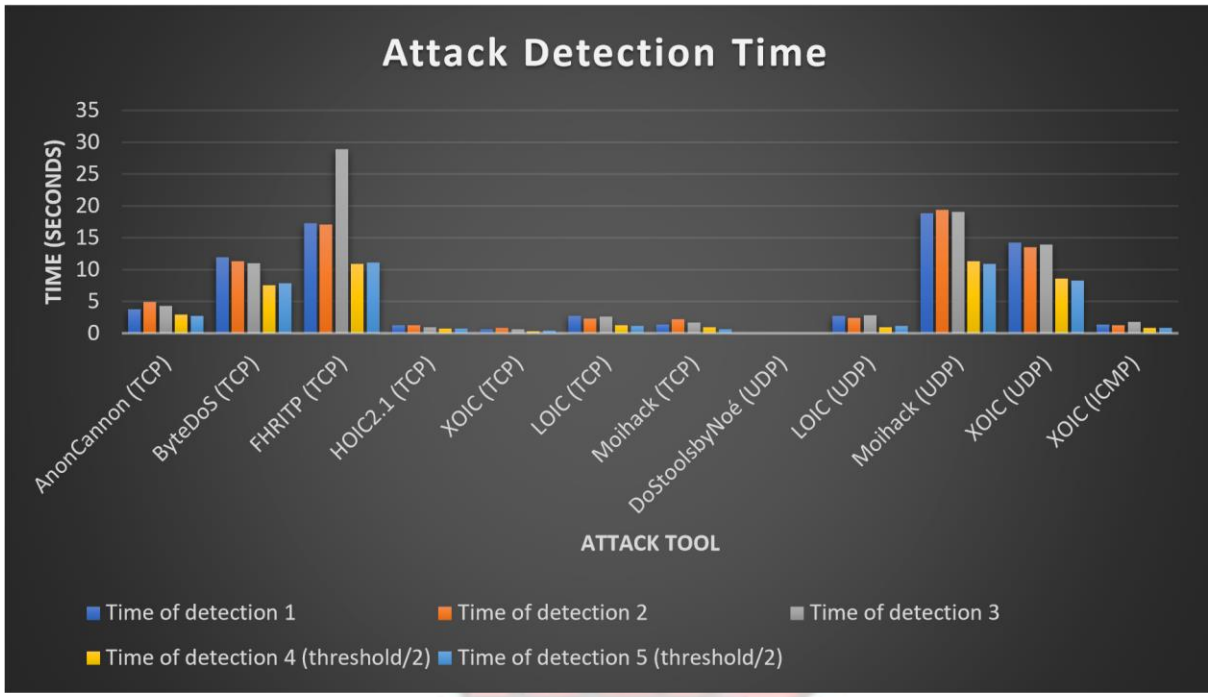


**Figure 4.1** Attack detection times

ByteDoS and FHRITP tools had high detection times due to the low rate of traffic produced by these tools. The highest recorded TCP attack detection time was 28.9 seconds with the FHRITP tool. This was most likely as a result of the tool not producing traffic for some periods of time after it was launched. This bug was encountered several times when testing

this tool.

The LOIC UDP was detected consistently under five seconds. However, the Moihack and XOIC attacks were detected between ten and twenty seconds for most of the period. This was due to the huge difference in attack packets sent by the LOIC tool and the other tools for UDP attacks. DoS tools by Noe was never detected during the attack due to the very low rate at which it sends packets and therefore will not be capable of launching a disruptive DoS attack alone.

The ICMP attack was also detected under five seconds. The benign ICMP traffic was also detected around the around seven seconds. Pinging sessions are used to determine if a server

or machine has connection to the network. Since it carries no important message, dropping even benign ping packets after a period helps prevent unnecessary use of network bandwidth. Due to this, ICMP traffic was not included in finding the false positive rates.

False positive rates were compared with the model proposed by (Yusof, Ali, & M.Y., 2018), Efficient Data Adapted Decision tree (EDADT) proposed by (Nadiammai & Hemalatha,

2013), Lightweight detection algorithm proposed by (Li, Yang, Li, & Yang, 2015), Modified K-means proposed by (Pramana, Purwanto, & Suratman, 2015) and a logistic regression model proposed by (Yadav & Selvakumar, 2015). The false positive rate for the proposed model was 0, outperforming the algorithms it was compared with, as shown in figure 4.2.

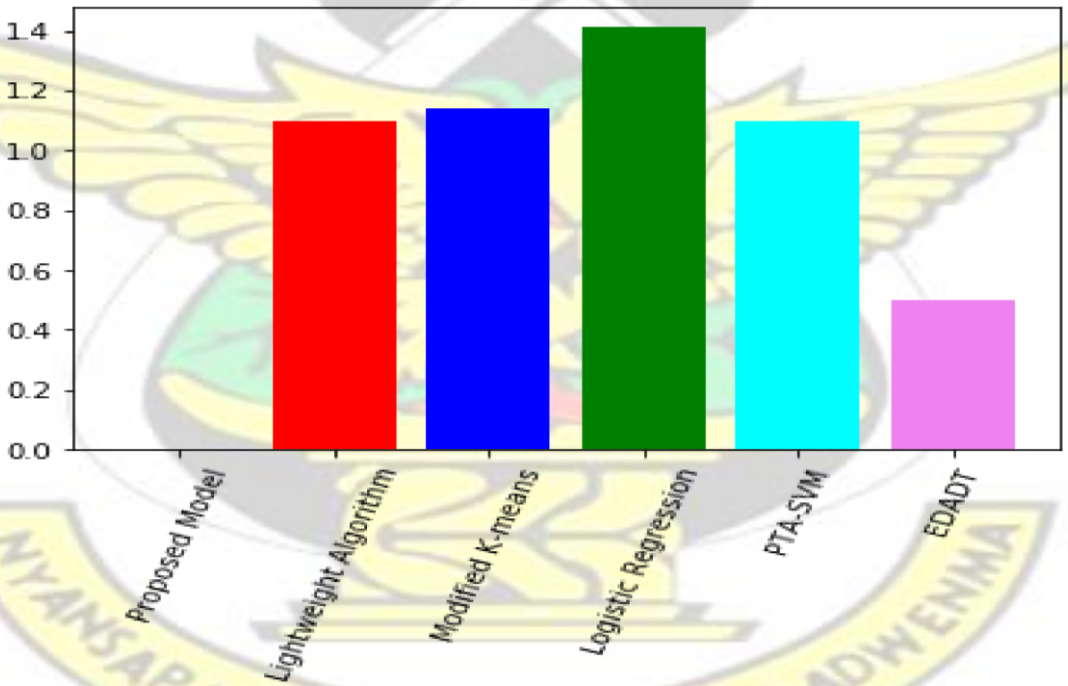


**Figure 4.2** False positive rates

New snort rules proposed by (Mehdi, 2017) was included in the detection accuracy comparison. The detection accuracy however was realised to be 92%, outperforming the model proposed by (Mehdi, 2017), and a little lower than most of the other algorithms as seen in figure 4.3.

This was due to the inability of the algorithm to detect the very low rate attack by the Noe DoS tool. The proposed model performed well in detecting attacks from the other tools.

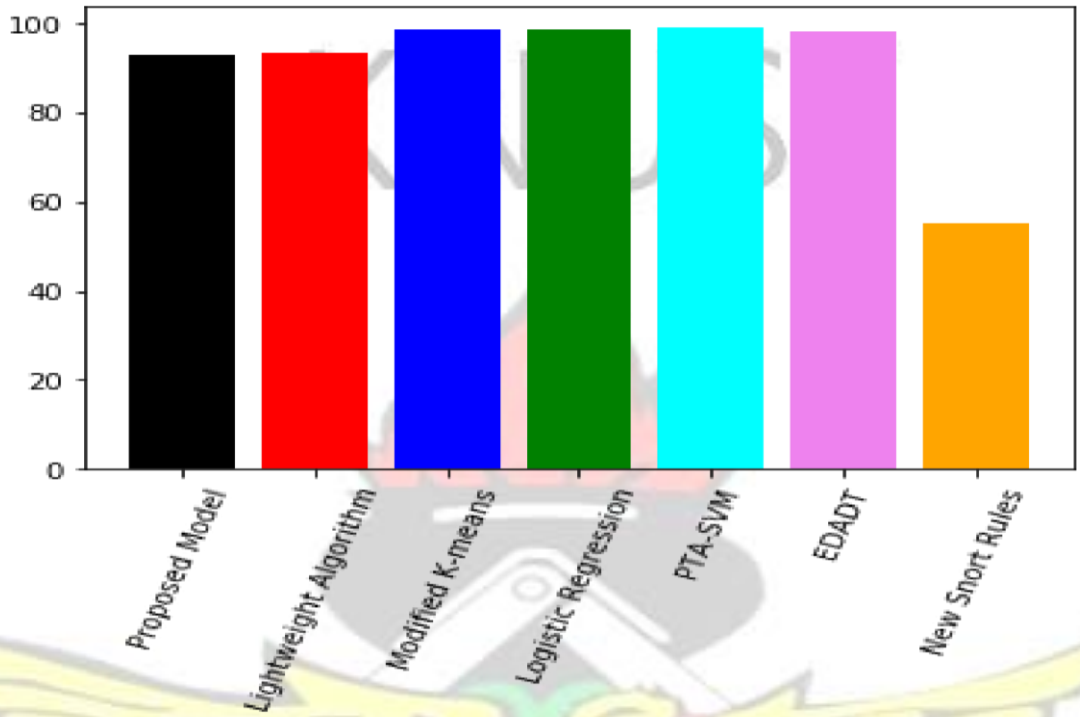


**Figure 4.3 Detection Accuracy**

## CHAPTER FIVE

## CONCLUSION

Denial of Service (DoS) attacks are on the rise in terms of number and size. These attacks can last for hours or even days, rendering the network unusable and causing financial losses. For an organisation which renders network-based services, this could be a bane to their operations. Aside the loss of revenue, there could be a loss of reputation and consumers, in the case of e-commerce organisations to other competing groups.

The availability of easily accessible DoS tools has made even the most unskilled attacker capable of launching a DoS attack. This has caused an increase in the frequency of attacks. Several DoS defence schemes have been proposed in literature; however, no proposed defence

scheme alone can defend against all DoS attacks. There has also been no defence scheme targeted at easily accessible DoS tools. There was therefore the need to propose a defence scheme to combat attacks launched using these tools.

This study therefore sought to analyse traffic generated from these DoS attack tools and propose a detection scheme to detect attacks launched by these tools. For this study, tools which could launch TCP, UDP or ICMP flooding attacks were considered. In total, eight tools were used as part of the study. Seven of the tools could be used to launch TCP attacks, four could be used for UDP attacks and only one for ICMP attacks. The experimental setup included a virtual machine as the target machine for the attacks, the host machine as the attacker, Wireshark Network Analyser for traffic analysis and Snort IDS to test the proposed algorithm. The algorithm for attack detection was designed based on the disparities between benign traffic and traffic generated by the attack tools. The proposed scheme was tested and had a high detection accuracy, low false positive rate and fast detection time. The results of

the tests were compared with existing DoS defence mechanisms.

## 5.1 RECOMMENDATIONS

One of the key challenges with DoS defence mechanisms is their generic nature. Most DoS defences are designed for generic use against DoS attacks and therefore do not target specific attacks, especially modern attacks, thereby reducing their accuracy. DoS defences should therefore be aimed at specific attacks to increase their accuracy in defending against those attacks.

Another challenge is the datasets used in testing the defence mechanisms. Most defences are tested with old datasets like the KDD Cup 99 dataset and DARPA datasets (Bukac & Matyas, 2015). These datasets are old and may not reflect the current situation with DoS attacks. New datasets must therefore be generated to ensure DoS defences are tested with current attack data.

## 5.2 FUTURE WORK

In the future, more DoS attack types will be analysed with defence mechanisms designed to combat them as well.

DDoS attacks will also be considered for future studies. Being able to defend against DoS attacks helps with guidance when designing defence mechanisms against DDoS attacks.

Anomaly-based detection methods will be combined with this signature-based method to provide a way of detecting attacks from tools not in the open yet, especially tools which will be based off some of the tools included in this study.

## REFERENCES


Acharya, S., & Tiwari, N. (2016). Survey Of DDoS Attacks Based On TCP/IP Protocol Vulnerabilities. *IOSR Journal of Computer Engineering (IOSR-JCE), 18*(3), 68-76. doi:10.9790/0661-1803046876

Amazon Web Services. (2016). *AWS Best Practices for DDoS Resiliency.*

Anagnostopoulos, M., Kambourakis, G., Kopanos, P., Louloudakis, G., & Gritzalis, S. (2013). DNS amplification attack revisited. *Computers and Security, 39*, 475-485. doi:10.1016/j.cose.2013.10.001

Behal, S., & Kumar, K. (2017). Characterization and Comparison of DDoS Attack Tools and Traffic Generators - A Review. *International Journal of Network Security, 19*(3), 383-393. doi:10.6633/IJNS.201703.19(3).07

Belenky, A., & Ansari, N. (2003). Ip traceback with deterministic packet marking. *IEEE Communications Letters, 7*(4), 162–164. doi:10.1109/LCOMM.2003.811200

Bhuyan, M. H., Kashyap, H. J., Bhattacharrya, D. K., & Kalita, J. K. (2014). Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions. *The Computer Journal, 57*(4), 537-556. doi:10.1093/comjnl/bxt031

Bijalwan, A., Wazid, M., Pilli, E. S., & Joshi, R. C. (2015). Forensics of Random-UDP Flooding Attacks. *JOURNAL OF NETWORKS, 10*(5), 287-293. doi:10.4304/jnw.10.5.287-293

Broder, A., & Mitzenmacher, M. (2002). Network applications of bloom filters: A survey. *Internet Mathematics, 1,* 636–646. doi:10.1080/15427951.2004.10129096

Bukac, V., & Matyas, V. (2015). Analyzing Traffic Features of Common Standalone DoS Attack Tools. *Security, Privacy, and Applied Cryptography Engineering*, (pp. 21-40). doi:10.1007/978-3-319-24126-5

*Caida*. (n.d.). Retrieved 08 20, 2019, from https://www.caida.org/data/passive/trace_stats/chicago-B/2015/?monitor=20150219-130000.UTC

Chen, Y., Shantanu, D., & Pulak, D. (2008). Detecting and preventing IP-spoofed distributed DoS attacks. *International Journal of Network Security, 7*(1), 69–80.

Criscuolo, P. J. (2000). *Distributed Denial of Service Trin00, Tribe Flood Network, Tribe Flood Network 2000 and Stacheldraht CIAC-2319.* Lawrence Livermore NAtional Laboratory, Department of Energy COmputer Incident Advisort Capability (CIAC).

Cui, Y. e. (2016). SD-Anti-DDoS: Fast and Efficient DDoS Defense in Software-Defined Networks. *Journal of Network and Computer Applications, 68*, 65-79. doi:10.1016/j.jnca.2016.04.005

Das, A., & Chaudhuri, A. (2017). Derivation and Simulation of an Efficient QoS Scheme in MANET through Optimised Messaging Based on ABCO Using QualNet. In *Nature-Inspired Computing: Concepts, Methodologies, Tools, and Applications* (pp. 396-425). doi:10.4018/978-1-5225-0788-8.ch016

de Assis, M., Hamamoto, A., Abrão, T., & Proença, L. (2017). A Game Theoretical Based System Using Holt-Winters and Genetic Algorithm with Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks. *IEEE Access*. doi:10.1109/ACCESS.2017.2702341

de Assis, M., Rodrigues, J., & Proença, M. (2013). A novel anomaly detection system based on seven-dimensional flow analysis. *IEEE Global Telecommunications Conference*, (pp. 735-740). doi:10.1109/GLOCOM.2013.6831160

de Assis, M., Rodrigues, J., & Proença, M. (2014). A seven-dimensional flow analysis to help autonomous network management. *Information Sciences, 278*, 900-913. doi:10.1016/j.ins.2014.03.102

Dua, S., & Du, X. (2016). *Data Mining and Machine Learning in Cybersecurity* (1st ed.). New York: Auerbach Publications. doi:10.1201/b10867

Elleithy, K. M., Blagovic, D., Cheng, W., & Sideleau, P. (2006). Denial of Service Attack Techniques: Analysis, Implementation and Comparison. *Journal of Systemics, Cybernetics and Informatics, 3*, 66-71.

Foroushani, V., & Zincir-Heywood, A. (2013). IP traceback through (authenticated) deterministic flow marking: an empirical evaluation. *EURASIP Journal on Information Security*. doi:10.1186/1687-417X-2013-5

Foroushani, V., & Zincir-Heywood, A. (2013). TDFA: Traceback-based Defense against DDoS Flooding Attacks. *IEEE International Conference on Advanced Information Networking and Applications*, (pp. 597-604). doi:10.1109/AINA.2014.73

Grance, T., Kent, K., & Kim, B. (2004). *Computer Security Incident Handling Guide.* National Institute of Standards and Technology, Computer Security Division.

Hameed, A., Karlik, B., & Salman, M. (2016). Back-propagation algorithm with variable adaptive momentum. *Knowledge-Based Systems, 114*, 79-87. doi:10.1016/j.knosys.2016.10.001

Hannula, M. (2011). How The Internet Affects Productivity. *International Business & Economics Research Journal (IBER), 1*(2), 83-91. doi:10.19030/iber.v1i2.3896

Holl, P. (2015). Exploring DDoS Defense Mechanisms. *Network Architectures and Services,* (pp. 25-32). doi:10.2313/NET-2015-03-1_04

Imperva. (n.d.). *DDoS Attack Scripts*. Retrieved 08 20, 2019, from Imperva.com: https://www.imperva.com/learn/application-security/ddos-attack-scripts/

Imperva Inc. (2019). *DDoS Attack Scripts*. Retrieved 08 20, 2019, from Imperva: https://www.imperva.com/learn/application-security/ddos-attack-scripts/

Kang, S., Gligor, V., & Sekar, V. (2016). SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks. *Network and Distributed System Security Symposium*. doi:10.14722/ndss.2016.23147

KITPLOIT. (2014). *LOIC 1.0.8 (Low Orbit Ion Cannon) - A network stress testing application.* Retrieved 08 20, 2019, from Kitploit: https://www.kitploit.com/2014/12/loic-108-low-orbit-ion-cannon-network.html

Kolahi, S., Alghalbi, A. A., Alotaibi, A. F., Ahmed, S. S., & Lad, D. (2014). Performance comparison of defense mechanisms against TCP SYN flood DDoS attack. *2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT),* (pp. 143-147). St. Petersburg.

doi:10.1109/ICUMT.2014.7002093

Kolahi, S., Treseangrat, K., & Sarrafpour, B. (2015). Analysis of UDP DDoS flood cyber attack and defense mechanisms on Web Server with Linux Ubuntu 13. *2015 International Conference on Communications, Signal Processing, and their Applications (ICCSPA'15)*, (pp. 1-5). Sharjah. doi:10.1109/ICCSPA.2015.7081286

Lantz, B., Heller, B., & McKeown, N. (2010). A network in a laptop: rapid prototyping for software-defined networks. *ACM SIGCOMM Workshop on Hot Topics in Networks.*

doi:10.1145/1868447.1868466

Lau, F., Rubin, S., Smith, M., & Trajkovic, L. (2000). Distributed Denial of Service Attacks. *Systems, Man and Cybernetics IEEE International Conference*, (pp. 2275-2280). doi:10.1109/ICSMC.2000.886455

Li, C., Yang, Z., Li, F., & Yang, Y. (2015). A Lightweight DDoS Flooding Attack Detection Algorithm Based on Synchronous Long Flows. *IEEE Global Communications Conference (GLOBECOM),* (pp. 1-6). San Diego, CA. doi:10.1109/GLOCOM.2015.7417159

Mahadev, Kumar, V., & Kumar, K. (2016). Classification of DDoS attack tools and its handling techniques and strategy at application layer. *2016 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Fall)*, (pp. 1-6). doi:10.1109/ICACCAF.2016.7749002

Matthews, Tim. (2014). *Incapsula Survey: What DDoS Attacks Really Cost Businesses.* Retrieved from http://lp.incapsula.com/ddos-impact-

report.html?_ga=2.237874052.526230773.1567034356-1341553771.1564151208

Mehdi, M. (2017). An Approach for Detecting and Preventing DDoS Attacks in Campus. *Automatic Control and Computer Sciences*, 13-23.

Merouane, M. (2017). An Approach for Detecting and Preventing DDoS Attacks in Campus. *Automatic Control and Computer Sciences, 51*(1), 13-23. doi:10.3103/S0146411616060043

*Mininet: An instant virtual network on your laptop.* (n.d.). Retrieved 08 20, 2019, from http://mininet.org/

Miniwatts Marketing Group. (2019, 7 25). *INTERNET USAGE STATISTICS*. Retrieved 08 20, 2019, from Internet World Stats: https://internetworldstats.com/stats.htm

Morales, Carlos. (2018, 03 05). *NETSCOUT Arbor Confirms 1.7 Tbps DDoS Attack; The Terabit Attack Era Is Upon Us*. Retrieved 08 20, 2019, from NETSCOUT: https://www.netscout.com/blog/asert/netscout-arbor-confirms-17-tbps-ddos-attackterabit-attack-era

Nadiammai, G. V., & Hemalatha, M. (2013). Effective approach toward Intrusion Detection System using data mining techniques. *Egyptian Informatics Journal*. doi:10.1016/j.eij.2013.10.003

Newman, Lily Hay. (2018, 03 01). *GitHub Survived the Biggest DDoS Attack Ever Recorded*. Retrieved 08 20, 2019, from WIRED: https://www.wired.com/story/github-ddosmemcached

Park, P. e. (2014). A Service-oriented DDoS detection mechanism using pseudo state in a flow router. *Springer Science + Business Media*. doi:10.1007/s11042-014-2100-5

Pramana, M., Purwanto, Y., & Suratman, F. (2015). DDoS detection using modified K-means clustering with chain initialization over landmark window. *International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, (pp. 7-11). doi:10.1109/ICCEREC.2015.7337056

*Project Floodlight.* (n.d.). Retrieved 08 20, 2019, from https://www.projectfloodlight.org/floodlight

Purwanto, Y., Kuspriyanto, Hendrawan, T., & Rahardjo, B. (2015). Traffic anomaly detection in DDos flooding attack. *8th International Conference on Telecommunication Systems Services and Applications (TSSA)*, (pp. 1-6). Kuta. doi:10.1109/TSSA.2014.7065953

Rajkumar, M., & Jitendra, N. (2013). A Survey on Latest DoS Attacks:Classification and Defense Mechanisms. *International Journal of Innovative Research in Computer and Communication Engineering, 1*(8), 1847-1860.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning representations by backpropagating errors. *Nature, 323*, 533-536. doi:10.1038/323533a0

*Ryu sdn framework.* (n.d.). Retrieved 08 20, 2019, from http://osrg.github.io/ryu/

Sahi, A., Lai, D., Li, Y., & Diykh, M. (2017). An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment. *IEEE Access PP (99)*. doi:10.1109/ACCESS.2017.2688460

Shon, T., Kim, Y., Lee, C., & Moon, J. (2005). A machine learning framework for network anomaly detection using SVM and GA. *6th Annual IEEE System, Man and Cybernetics Information Assurance Workshop, SMC*, (pp. 176-183).

doi:10.1109/IAW.2005.1495950

SOURCEFORGE. (2014, 12 13). *LOIC*. Retrieved 08 20, 2019, from Sourceforge: https://sourceforge.net/projects/loic/files/loic/loic-1.0.8/

Sunshine, C. A. (Ed.). (1989). *Computer network architectures and protocols* (2nd ed.). New York: Plenum Press. doi:10.1007/978-1-4613-0809-6

Tanachaiwiwiat, S., & Hwang, K. (2003). Differential packet filtering against DDoS flood attacks. *ACM Conference on Computer and Communications Security (CCS).*

Wang, B., Zheng, Y., Lou, W., & Hou, T. (2015). DDoS attack protection in the era of cloud computing and Software-Defined Networking. *Computer Networks*. doi:10.1016/j.comnet.2015.02.026

Wang, H. e. (2014). A moving target DDoS defense mechanism. *Computer Communications, 46*. doi:10.1016/j.comcom.2014.03.009

Wang, H., Zhang, D., & Shin, K. G. (2002). Detecting SYN flooding attacks. *Twenty-First Annual Joint Conference of IEEE Computer and Communications Societies . 3.* IEEE.

Xiang, Y., Zhou, W., & Guo, M. (2009). Flexible deterministic packet marking: An ip traceback system to find the real source of attacks. *IEEE Transactions on Parallel and Distributed Systems, 20*(4), 567-580. doi:10.1109/TPDS.2008.132

Yaar, A., Perrig, A., & Song, D. (2006). StackPi: new packet marking and filtering mechanisms for DDoS and IP Spoofing Defense. *IEEE Journal on Selected Areas in Communications, 24*(10), 1853–1863. doi:10.1109/JSAC.2006.877138

Yadav, S., & Selvakumar, S. (2015). Detection of application layer DDoS attack by modeling user behavior using logistic regression. *4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*, (pp. 1-6). doi:10.1109/ICRITO.2015.7359289

Yu, S., Zhou, W., Guo, S., & Guo, M. (2013). A Dynamical Deterministic Packet Marking Scheme for DDoS Traceback. *IEEE Global Communications Conference (GLOBECOM)*, (pp. 729-734). doi:10.1109/GLOCOM.2013.6831159

Yusof, M., Ali, F., & M.Y., D. (2018). Detection and Defense Algorithms of Different Types of DDoS Attacks Using Machine Learning. (R. Alfred , H. Lida, A. Ibrahim, & Y. Lim, Eds.) *Computational Science and Technology*, 370-379. doi:10.1007/978-981-10-8276-4_35

Zargar, S. T., Joshi, J., & Tipper, D. (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Communications Surveys & Tutorials, 15*(4), 2046-2069. doi:10.1109/SURV.2013.031413.00127

Zheng, J. e. (2018). Realtime DDoS Defense Using COTS SDN Switches via Adaptive Correlation Analysis. *IEEE Transactions on Information Forensics and Security, 13*(7), 1838-1853. doi:10.1109/TIFS.2018.2805600

Zhou, W., Jia, W., Wen, S., Xiang, Y., & Zhou, W. (2013). Detection and defense of application-layer DDoS attacks in backbone web traffic. *Future Generation Computer Systems, 38,* 36-46. doi:10.1016/j.future.2013.08.002