ENHANCED CLUSTER BASED (E-CluB) PROACTIVE FRAMEWORK FOR MITIGATING DISTRIBUTED DENIAL OF SERVICE (DDOS) ATTACK IN ANYCAST NETWORKS

BY

Bashira Olawumi YUSUF
(P13SCMT8004)

A DISSERTATION SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES, AHMADU BELLO UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE DEGREE IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE,

FACULTY OF PHYSICAL SCIENCES

AHMADU BELLO UNIVERSITY,

ZARIA, NIGERIA.

JANUARY, 2017

DECLARATION

I declare that the work in this dissertation titled **"ENHANCED CLUSTER BASED (E-CluB) PROACTIVE FRAMEWORK FOR MITIGATING DISTRIBUTED DENIAL OF SERVICE ATTACK IN ANYCAST NETWORKS"** has been carried out by me in the Department of Mathematics under the supervision of Prof. Sahalu Junaidu and Dr. Isma'il Baroon. The information derived from the literature has been duly acknowledged in the text and a list of references provided. No part of this dissertation was previously presented for another degree or diploma.

Bashira Olawumi YUSUF _____

(Name of Student) Signature/Date

CERTIFICATION

This dissertation report entitled titled **"ENHANCED CLUSTER BASED (E-CluB) PROACTIVE FRAMEWORK FOR MITIGATING DISTRIBUTED DENIAL OF SERVICE ATTACK IN ANYCAST NETWORKS"** by Bashira Olawumi Yusuf meets the regulation governing the award of the degree of M.Sc. Computer Science of Ahmadu Bello University, and is approved for its contribution to knowledge and literary presentation.

……	……..
Chairman, Supervisory Committee (Prof. Sahalu B. Junaidu)	Date
…….....	……..
Member, Supervisory Committee (Dr. Ahmad I. Barroon)	Date
……	……..
External Examiner	Date
……	……..
Head of Department (Prof. Sahalu B. Junaidu)	Date
……	……..
Dean, PG School (Prof. Kabir Bala)	Date

DEDICATION

I dedicate this work to Almighty Allah, who has made this work possible. I also dedicate this dissertation to my parents, Engr. and Mrs. Yusuf M.A, my siblings and all my friends for the support given to me to make this work a reality.

ACKNOWLEDGEMENT

First and foremost, I would like to thank Almighty Allah for all He has done. I will also express my profound gratitude to my supervisors, Prof. S.B. Junaidu Sahalu and Dr. Isma'il Barroon, whose constructive criticisms and invaluable contributions have enabled me complete this work successfully.

My special thanks goes to Professor A.A. Obiniyi for his contributions despite his tight schedules.

I owe a great debt of gratitude to my parents for educating me to this level and all their support. I also give special thanks to my siblings for their kind support and contribution to the success of my studies.

Finally, I wish to acknowledge all my lecturers for their supports in the course of my studies. Also, Kawu, Yaya, Mariam Ocheja, Ramlat Muhammed, Bilikisu Abubakar, Hauwa Atiku, Mal Sani Minjibir and Abdukadeer and all my colleagues, I say thanks for your contributions to the success of this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDIX

ABBREVIATIONS and DEFINITION

ABBREVIATION	DEFINITION
DOS	Denial of Service
DDoS	Distributed Denial of Service
CluB	Cluster based
E – CluB	Enhanced Cluster based
IP	Internet Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
BGP	Border Gateway Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
UDP	User Datagram Protocol
SYN/ACK	Synchronization/Acknowledgement
DNS	Domain Server Name
SHA	Secured Hash Algorithm
GNS3	Graphical Network Simulator 3
PHP	Hypertext Preprocessor
CH	Cluster Head
ECR	Egress Checking Router
ICR	Ingress Checking Router
TEA	Tiny Encryption Algorithm
IAD	IP Address Database
SAVE	Source Address Validity Enforcement
OSPF	Open Shortest Path First
IOS	Internetwork Operating System

ABSTRACT

Distributed Denial of Service (DDoS) attacks are threats not only for the direct targets but also for the core of a network. These attacks can be so powerful that they can easily deplete the computing resources or bandwidth of the potential targets, by flooding packets on the intended server. They are also hard to detect in advance, hence methods to deal with them need to be proactive, but several researchers used reactive methods to deal with DDoS attacks. Building on an existing work which used SHA1 hashing method to encapsulate token, a proactive method was enhanced known as Enhanced Cluster Based (E-CluB) proactive framework used to improve on the distribution control aspects. The E-CluB aims at mitigating DDoS attacks by keeping the network performance degradation as little as possible using virtual machines to launch the DDoS attack using the Kali Linux Operating System. E-CluB also used Open Shortest Path First (OSPF) as its routing policy in the anycast network, including contemporary datagram options. The granularity of the existing system used 100% of its routers as the checking routers and the proposed system used 1% and achieved a better granularity control with respect to the efficiency filtering in the E-CluB. The proposed and existing systems were compared based on latency of transmission, granularity control and filtering efficiency of DDoS attacks using the wireshark analyzer and an improvement of 41.2% was achieved.

CHAPTER ONE: INTRODUCTION

In this chapter, we introduce the background to the study, problem statement, motivation, aim and objectives, research methodology, expected contribution to knowledge and the organization of the dissertation.

1.1 Background to the Study

Computer Networking is considered a branch in Computer Science, Electrical Engineering, Information Technology or Computer Engineering as they all rely on both practical and theoretical aspect of Network application in our society today. Computer Networks were originally developed to connect number of devices through wires so that devices can share some information and data with each other, but with the increase in the number of entities which needs network access and are not physically attached to any wired network, then the wireless network was developed to serve. These wireless networks are computer network that utilizes wireless connection network. There are two categories of wireless network namely:

a. Infrastructure Network
b. Infrastructure-less Network

Infrastructure Network are networks that contains fixed and wired gateways and the Infrastructure-less Network are networks that contains multi-hop wireless nodes and it has no fixed infrastructure (Kaur and Kaur, 2013).

Routing is the process of transferring a packet from source to its destination. In a routing process, the wireless sensor node will search for a path or route to communicate with the other nodes in the network. Protocols are set of activities or rules through which two or more devices communicate with each other. Routing is also the process of selecting best path in a network; it could also mean forwarding of network. It is performed in many

kinds of network which includes Circuit Switching and Packet Switching. Some Routing schemes used in delivering semantics are: (Kaur and Kaur, 2013)

a. Unicast – it delivers to a single node
b. Anycast – it delivers to many of a group node
c. Multicast – delivers to a group of nodes

Anycast is a method used to advertise one IP address from multiple points in the network topology, and with the help of dynamic routing method, the traffic is delivered to the nearest point. Anycast is a technique used to deliver packet to the closest in a group (Patridge *et al.*, 1993).

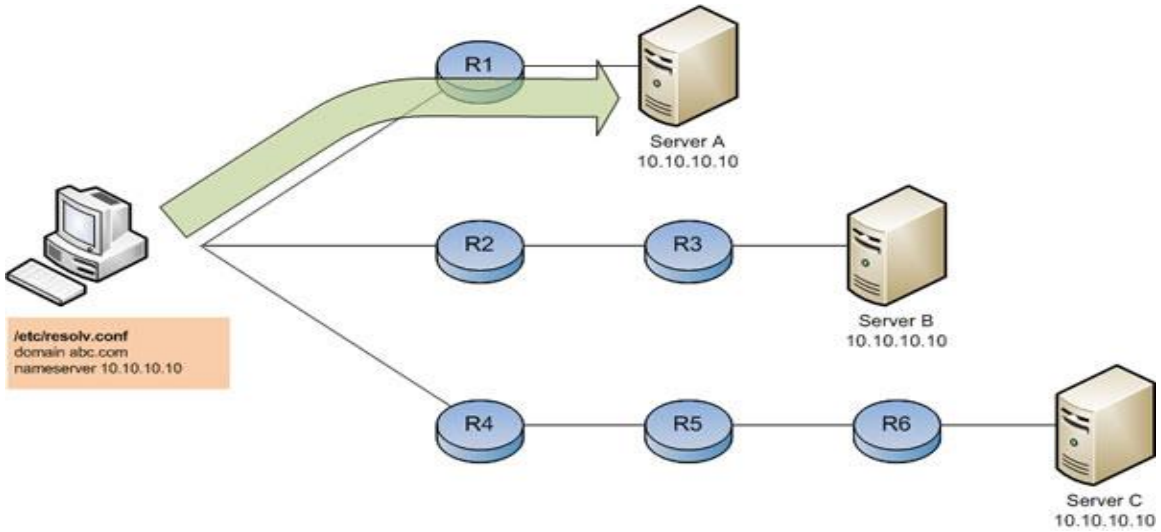

Figure 1.1: Anycast Network Topology (data:image/anycast-dns1.jpg)

A Denial of Service (DoS) attack is an attempt by the adversary to prevent the legitimate users of a service from using that service. Generally speaking, any attack that can saturate or exhaust system resources or get the system into fault status or sometimes even crashes should be identified as a DoS attack. DoS problems are not new, as they have been there for more than 20 years and keep evolving over time. The first well-known DoS is the Morris Worm which is an Internet worm developed by a graduate student (Zhang, 2012).

Nowadays, DoS attacks are usually launched in a distributed way: the attack traffic is from many attacking sources and the aggregated traffic volume is so big that it can easily deplete the victim's key computing resources, such as bandwidth and CPU time. When the adversary compromises multiple machines to launch a Denial-of-Service attack, this becomes a Distributed Denial of Service (DDoS) attack (Zhang, 2012).

Figure 1.2 below depicts the DDoS attack been launched on anycast network.

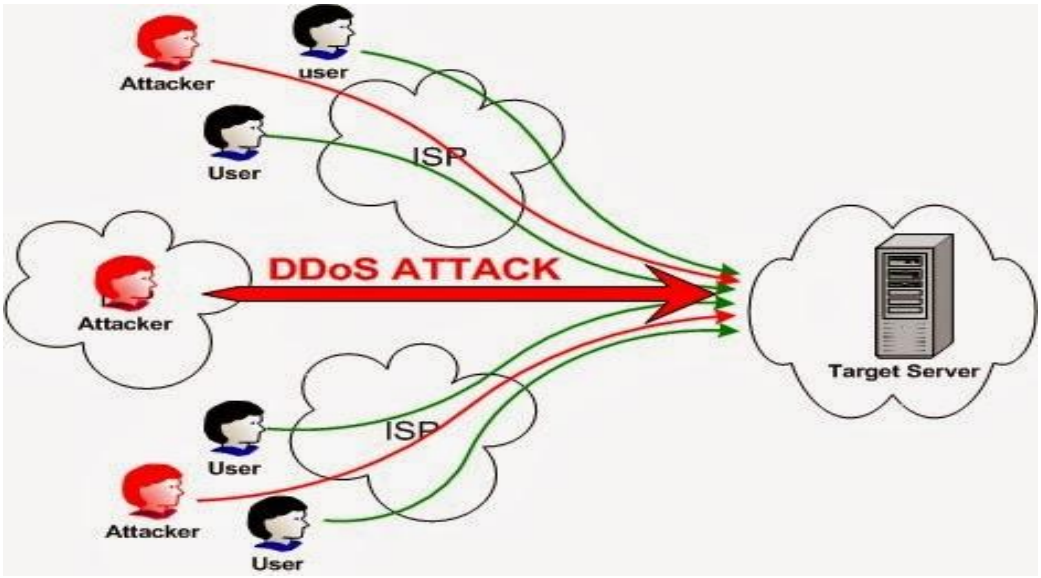

Figure 1.2: DDoS attack (http://1.bp.blogspot.com/DDoS-Attack.jpg)

Researchers found out some motives for launching DDoS attacks which are listed below: (gathered from botnet economy)

a. Some show off their skills or prove they found some system vulnerabilities like the Morris Worm
b. Economic incentive like blackmailing the victims such as companies
c. Political reasons

In 2001, researchers from CAIDA observed 12000 attacks against more than 5000 distinct targets from a 3 weeks long dataset using backscatter analysis (Moore *et al.*, 2001).

According to the report from Arbor Networks the scale of the DDoS attacks evolved a lot that it was observed a significant increase in the prevalence of attack rates in the 10Bbps range. The frequency of DDoS attacks though is not as high as year 2000 to year 2004, is still far from extinction.

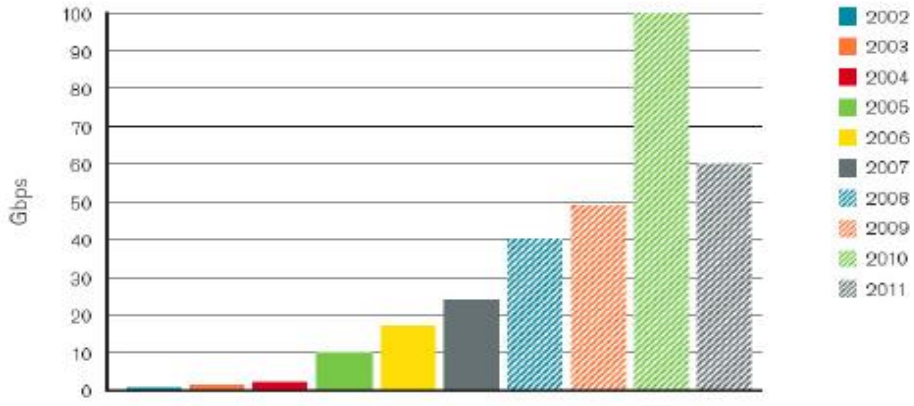

Figure 1.3: Bandwidth attacks reported from 114 service providers. (Moore *et al.*, 2001)

1.2 Categories of DDoS Attacks

Preventing or mitigating DDoS attacks is not an easy job. First we have to understand how the attacks work. Some categories of DDoS attacks are as:

1.2.1 Semantics Attacks

In semantic attacks, a single machine can complete the attack goal, since one malformed packet is enough to impede the service. Semantic attacks can be prevented by fixing the corresponding bugs in the protocols or applications. Some examples of semantics attacks are:

1.2.1.1 Teardrop

The adversary sends incorrect IP fragments to the target. The target machine may crash if it does not implement TCP/IP fragmentation re-assembly code properly (CERT, 2010).

This kind of attack can be prevented by fixing the IP implementation bugs in operating systems.

1.2.1.2 **Ping of Death**

A ping of death is an attack that the adversary sends the victim a ping packet which has more than 65535 byte. Since many systems cannot handle ping packets larger than 65535 bytes, handling packets of this size may cause a buffer overflow which may cause a system crash (Kenney, 2012).

1.2.1.3 Border Gateway Protocol (BGP) Poisoning

The BGP is used to establish routing paths between networks in Autonomous system level. The routing information is updated by exchanging the BGP advertisement between routers. Usually, the routers update their routing tables without verification of the BGP advertisements. The adversary can subvert the network communication by announcing a better route to some destinations, and then all the packets to the destinations are routed to the adversary.

Also the adversary can disturb the BGP routing by announcing fake BGP advertisements with addresses of other routers. Then the corresponding traffic will be routed to those routers which do not have optimal routes to the destination (Kent *et al.*, 2000).

1.2.2 Brute Force

Brute force attacks aim at exhausting the victim's network bandwidth or computing resources by means of flooding massive malicious packets. To deplete the victim's computation resources, the adversary usually uses the packets of Internet protocols which have request-reply scheme, such as TCP, HTTP. During the attacks, massive spurious requests are flooded to keep the target busy serving them, thus impeding the legitimate usage. To deplete the bandwidth, the adversary can basically flood any types of packets to congest the target network link. Examples can be UDP flooding and ICMP flooding.

1.2.2.1 SYN Flood

In a SYN flood attack, the adversary takes the advantage of the three-way handshake for a TCP connection. In normal execution, when a TCP server receives a SYN packet, it opens a session for this new connection and sends back a SYN/ACK packet to the initiator. When it reaches a timeout and there is no ACK packet received from the corresponding initiator, the session will be closed and the corresponding resources for the session are released. During the attack, the adversary continues sending SYN packets without sending back the final ACK packets for the TCP handshakes, the server's resource (e.g. memory) can be quickly depleted by maintaining many half open sessions, thus legitimate connection requests cannot be served (Wesley, 2007).

1.2.2.2 Hypertext Transfer Protocol (HTTP) Flood

In HTTP flood attacks, the adversary floods massive spurious HTTP requests for downloading a web file from the target server. This file is usually a large file that the server may need to load from the hard disc and spend considerable CPU time to transfer it via packets. However, continuously requesting big files can be suspicious. To avoid being detected, the adversary can instruct zombie machines to get a specific web page as the start and then follow the links on that page recursively, which can mimic the normal web browsing behaviors (Peng *et al.*, 2007).

1.2.2.3 Internet Communication Message Protocol (ICMP) Flood (Smurf Attack)

In an ICMP flood attack, the adversary floods ICMP Echo packets to some network which broadcasts these messages to all the hosts in the network. These ICMP Echo packets have the victim's IP address. All the hosts who receive the echo packet will send Echo reply packets to the victim, which exhaust the victim's bandwidth. Actually, this kind of attack is a mixture of a semantic attack with brute force. The way the attack works is based on response mechanism in ICMP. However, from the perspective of the

victim, it is brute force, as the type of the attack is just flooding packets from many machines. Similar to ICMP flooding attack, the adversary can take advantages of any reply-based protocol to launch reflected attack, by spoofing requests from the victim to a large set of Internet servers, resulting in a big volume of reply messages towards the victim network. Common protocols used in this kind of attacks include DNS queries, ICMP (Vaughn and Evron, 2006).

1.2.2.4 **User Datagram Protocol (UDP) Flood**

During a UDP flood attack, the victim's network is overwhelmed by a large volume of UDP packets (CERT, 2010). The attack packets are usually with random port numbers. When the victim receives a packet, if there is no application listening at the corresponding port, then the victim may generate an ICMP packet of "destination unreachable" to the sender. Thus massive UDP packets to the victim's inactive ports may exhaust both incoming and outgoing capacities of the victim.

1.3 Research Problem

Some limitations have been observed in the existing methods which need to be addressed to maximize the confidentiality of the authentication token of the outgoing packets, these are:

a. The SHA1 hashing method used can result to collision in the hash value
b. The checking routers are changed periodically; leading to possibility of compromising a router at the border end of the cluster.

After studying the literature of the papers reviewed, means were devised to improve on the security of the existing system using dynamically changing checking router and the

hashing method used in the proposed has more length of hash value thereby leading to less brute force attack.

1.4 Research Motivation

As recent studies have indicated, DoS and DDoS attacks remain to be a severe threat to the stability of the Internet. This area of study has received much attention in the last several years because many people believe that these attacks will be a persistent threat in the Internet and could undermine the stability and usability of the Internet (Zhang, 2012). Despite methods that are in existence today, the threat of these attacks still lingers and future attacks will be powerful (Mohammed, 2010).

So, due to the problem of less security and confidentiality of the existing system, we intend to improve on the filtering efficiency and granularity control of in the proposed system.

1.5 Research Aim and Objectives

The aim of this study is to mitigate Distributed Denial of Service (DDoS) using Enhanced Cluster based (E–CluB) proactive method in a network.

The Objectives are to:

a. design a system that will enhance security of packets in transmission thereby mitigating attacks in anycast networks;
b. implement the proposed E-CluB framework;
c. evaluate the performance of the proposed system with the existing system (Zhang, 2012) with respect to filtering efficiency, granularity control and latency of transmission of the systems.

1.6 Research Methodology

a. Conduct literature review to establish limitations of the existing solutions to DDoS attack in anycast networks.

b. Design tool such as graphical network simulator 3 (GNS3) was used to design the anycast Networks including its routers.

c. Backtrack 5 was used to launch DDoS attack on the anycast IPv6 anycast network

d. Blowfish was employed as a hashing method to secure the authentication tokens for packets in transmission.

e. The message was encrypted using the public key encryption before the message leaves the cluster.

f. The wamp server was used to implement the E-CluB framework.

g. The wireshark analyzer was used to take the traffic readings of the attack and mitigation on both the existing and proposed systems

h. The proposed system was compared with that of Zhang (2012) when the system is under attack based on filtering efficiency, granularity control and latency of transmission in both systems.

i. Discussion and presentation of results in graphical and tabular form

1.7 Organization of the Dissertation

This dissertation explains “Enhanced Cluster Based (E-CluB) Proactive framework for mitigating Distributed Denial of Service (DDoS) attacks in anycast networks”, and it consists of five chapters.

Chapter one is the dissertation Introduction: Background of study, Research Motivation, Research Aim and Objectives, Research Methodology and the Organization of the dissertation.

Chapter two focuses on Literature Review, System Study, Review of the existing System, Problems of the existing System and Proposed Solution.

Chapter three is the Design and Implementation, Analysis for the Proposed System and Implementation Organization.

Chapter Four focuses on the Result discussion and Result Analysis.

Chapter Five focuses on the Summary, Conclusions and References

CHAPTER TWO: LITERATURE REVIEW

In this chapter, we introduce the defense mechanisms in DDoS attacks, history of anycast, history of clustering, clustering based in anycast routing protocol, cryptography and related work.

2.1 Introduction

According to the activity levels, the DDoS defense mechanisms can be classified as *proactive defense, reactive defense,* and *DDoS detection* (Fu, 2012).

a. The Proactive defense aims at preventing the malicious packets from reaching the victim, thus they cannot impact the protected service. Proactive defense methods are always turned on so that the malicious packets can be dropped as soon as possible when they are identified.
b. Reactive defense aims at minimizing the loss caused by DDoS attacks. Reactive defense methods are always activated after the attacks are detected or the targeted service has already impacted.
c. DDoS detection aims at detecting DDoS attacks when they are launched.

Based on the deployment layer, the defense mechanisms can be classified as application defense and network defense, depending on whether network entities, e.g. routers, other than the end hosts are involved for mitigating the attacks. In this section, to connect with our work, we give a brief overview of the existing DDoS defense mechanism based on the classification of application defense and network defense.

The primary goal of DDoS defense is maintaining availability in the face of attack (Fadda, 2015)

2.2 History of Anycast

Anycast is a technique used to deliver a packet to one of many hosts. A group of possibly distributed hosts respond to the same address known as anycast address. A packet destined for an anycast address will be delivered to one of the hosts with that address which is close to the source. IPv6 specially defines anycast addressing as an identifier for a set of interfaces. A data packet is intended to be delivered to an anycast address and routed to the nearest node (Patridge *et al.*, 1993).

The enhancement of the anycast protocol is having multiple receivers, and only one receiver is selected from all the available ones. Hence in a way it's a point to point communication with the nearest address. Also the sender does not care which receiver from the possible list is selected (as all of the receivers will be providing the same service) and will be mirrors of each other (Liu and Shi, 2012).

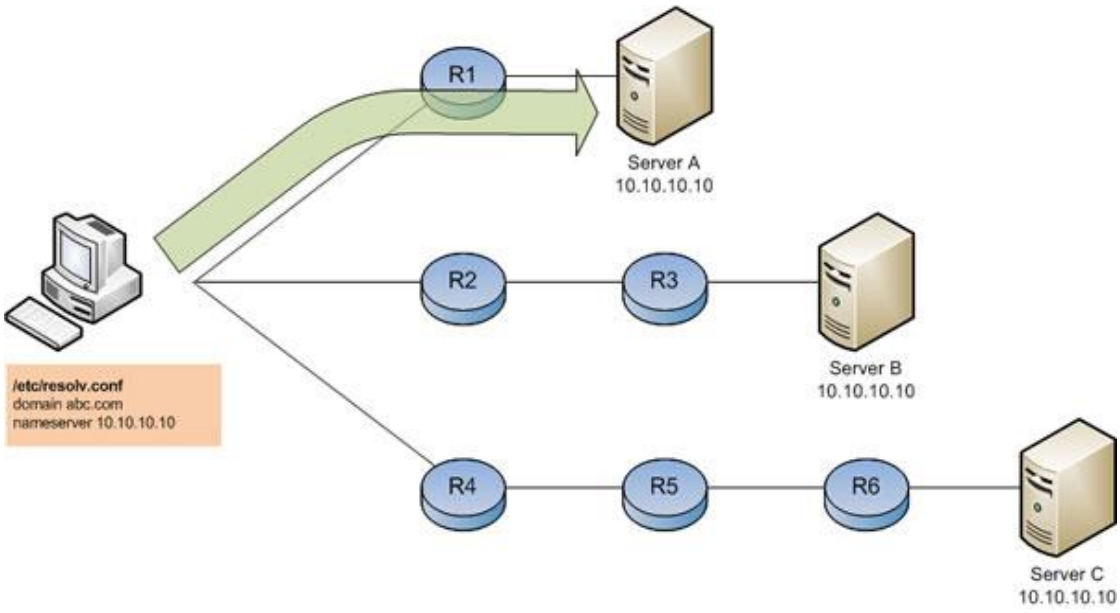

Figure 2.1: Anycast Network (data:image/anycast-dns1.jpg)

2.3 History of Clustering

Clustering nodes and organizing them hierarchically have proven to be an effective method to provide better data aggregation and scalability for the network while conserving limited energy. The main aim of clustering nodes in a network is to save or reduce energy consumption in the various nodes in a network (Liu and Shi, 2012).

2.4 Clustering based in Anycast Routing Protocols

Collection of nodes into a cluster in anycast is well-known as clustering. Every cluster contains a leader called the Cluster Head (CH). A CH is selected by the group of network nodes that forms the cluster. This CH collects all the information from the nodes within the cluster and sends the information to the destination node. Clustering helps to reduce/minimize the total transmission power aggregated over the nodes in the selected paths and also to balance the load between the network lifetimes (Kent *et al.*, 2000). Clustering based routing algorithms are growing to be an essential part of routing technology with several advantages such as larger scalability, lesser load and a smaller amount of energy consumption with extra robustness (Liu and Shi, 2012).

2.4.1 Characteristics of Anycast routing Protocol

Authentication Tokens

In order to go out or pass through a cluster, packets need to have permission which are called Authentication Tokens. Roughly speaking, these authentication tokens are like “passports” (permission to leave a cluster) and “visas” (permission to enter or pass through a cluster) for the traffic. To control the validity of the authentication tokens of outgoing and incoming packets, in each cluster there are designated Egress Checking Routers (ECR), Ingress Checking Routers (ICR). Transit traffic is controlled by backbone routers. Given that the attacker can observe the tokens of legitimate packets and learn which routers are the ECRs and ICRs, these tokens are binded to specific IP addresses

and checking routers, also tokens and the checking routers will be changed periodically; the latter actions are also called “hopping” (Zhang, 2012).

Coordination authorities

Every cluster needs a coordination entity, which can be implemented in a centralized manner by a single node. A single coordinator entity can be chosen per cluster. This coordinator is publicly trusted and well protected, e.g. via a secure overlay, in order to avoid turning it to an attackers target. The coordinator maintains security policies of its cluster and cooperates with coordinators in other clusters. Generally, each coordinator has the duty for the following tasks: (Zhang, 2012)

a. It decides whether to allow a host in the cluster to send outbound traffic out of this cluster, or to allow a host of another cluster to send inbound and transit traffic to/via its cluster. The coordinator will grant the authentication code of the cluster to the hosts of the approved requests. The authentication code is used for computing the authentication tokens of legitimate packets. If a host wants to apply for permission for sending inbound traffic or transit traffic to another cluster, the coordinator of the local cluster will forward the host’s request to the coordinator of the corresponding cluster.
b. It generates new authentication codes for the cluster periodically; and gives the codes to the corresponding routers for checking the validity of traffic.

Not only the tokens, but also the routers for checking them will be changed periodically. It is the coordinator that appoints the checking routers for each period of time (Fu, 2012).

2.5 Egress/Ingress Filtering Scheme

Egress filtering is the practice of monitoring and potentially restricting the flow of information outbound from one network to another. Typically it is information from a private TCP/IP computer network to the internet that is controlled.

Egress filtering helps ensure that unauthorized or malicious traffic never leaves the network. Packets that do not meet security policies are not allowed to leave they are denied passage through the egress policy. Egress filtering can reduce risk to an organization (Gezelter and Robert, 1995).

Egress filtering is defined as ensuring that your site does not emit packets from inappropriate addresses (William *et al.*, 2003).

Ingress filtering is a technique used to ensure that incoming packets are actually from the network they claim to originate. In ingress filtering, packets coming into the network are filtered if the network sending it should not send from the originating IP addresses

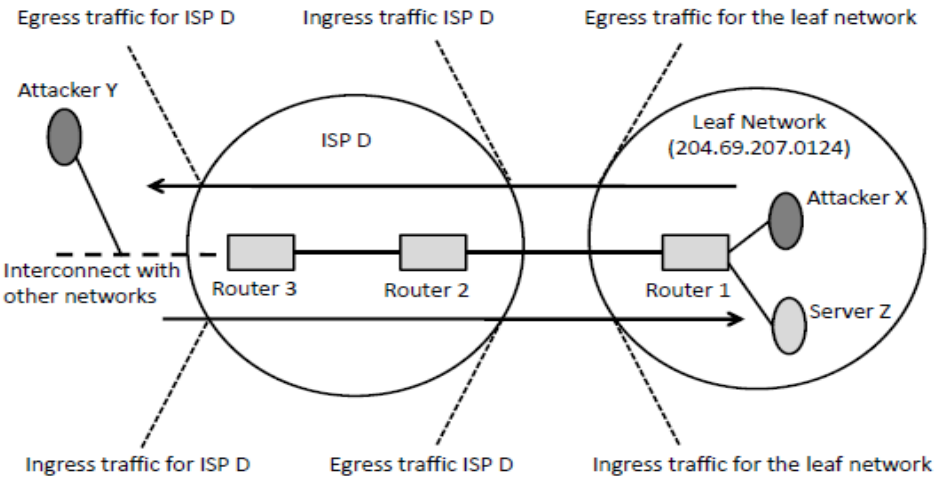

Figure 2.2 Egress/Ingress Filtering Scheme (Peng *et al.*, 2007)

2.6 DDoS Attack Tools

Table2.1: Summary of DDoS attack tools (Gupta *et al.,* 2010)

DDoS attack tool	**Commands used**	**Types of attacks Generated**	**Communication methods**
Trinoo	Not Encrypted	UDP flooding	Attacker to master- unencrypted TCP, Master to slave- unencrypted UDP, Slave to master - unencrypted UDP
TFN	Numeric code and Not Encrypted	ICMP flooding TCP flooding UDP flooding Smurf	Attacker to master - required third-party program, Master to slave- unencrypted ICMP Slave to master - none
TFN2K	Encrypted	ICMP flooding TCP flooding UDP flooding Smurf	Master to slave- can be mixture of encrypted TCP, UDP and ICMP, Slave to master - none
Stacheldraht	Encrypted	ICMP flooding TCP flooding UDP flooding Smurf	Attacker to master - encrypted TCP, Master to slave- TCP and ICMP Slave to master - none
Shaft	Not Encrypted	ICMP flooding TCP flooding UDP flooding Mixed flood	Attacker to master - unencrypted TCP Master to slave- unencrypted UDP
Mstream	Not Encrypted	TCP flooding	Attacker to master -un TCP Master to slave- unencryp Slave to master - unencryp Knight
Knight	Not Encrypted	TCP flooding UDP flooding and Urgent Pointer	Uses IRC as it's communication method
Trinity	Not Encrypted	TCP flooding UDP flooding	Uses IRC as it's communication method

Gupta *et al.* (2010) discussed in the table above an overview of the DDoS problem, available DDoS attack tools, defense challenges and principles and a classification of available DDoS preventive mechanisms.

A DDoS attacker uses many machines to launch a coordinated DOS attack against one or more targets (Douligeris and Mitrokotsa, 2003). It is launched indirectly through many compromised computing systems by sending a stream of useless aggregate traffic meant to explode victim resources. His work didn't cover the frequently network congestion on the way from a source to the target, which disrupts normal Internet operation which has lead to the increase in of DDoS attack in the last few years (Moore *et al.*, 2001).

2.7 The Basic Protocol in CluB Framework

The basic protocol in CluB consists of five sections namely: permission requesting, packet encapsulation, packet forwarding, filtering replayed packets and token refreshing.

2.7.1 Permission requesting

Before a host sends packets to other clusters, it first sends a request to the coordinator in the local cluster. If it is allowed to send packets outbound, the coordinator will give the IDs of one ECR and one ICR of the local cluster and the destination cluster respectively. The authentication codes of the clusters that the packets will go through will also be given.

Upon receiving a request from a host, the coordinator first checks whether all protocols are OK, if so then the coordinator forwards this request to the coordinator of the destination cluster, while each coordinator of the intermediate clusters will also get this request. They too, will decide whether to grant the permission to the host according to their local policies. If everything is OK, the destination coordinator provides the address of one of its ICRs and its inbound-authentication codes; similarly, all the intermediate

coordinators provide their transit authentication codes. After the local coordinator gets the reply messages, it will hash each of the authentication codes as well as the outbound-authentication code using a hashing method. Then the local coordinator will put these hash values and the addresses of ECR and ICR into one message and return it to the requesting host, encrypted using the host's public key (Zhang, 2012).

2.7.2 Packet encapsulation

If the host gets all the information it needs for sending packets, it will form the packets as required by the protocol, essentially adding a new header. After successfully getting the reply, the requesting host can compute the authentication tokens for its packets and send them to the desired destination. Regular packets have some more header fields than the normal (e.g. IP) packets. These extra fields form a data structure, called routing vector, containing everything that a packet needs for going through the checking points in the process. Furthermore, the first entry of the vector contains the address of an ECR of the source cluster and the cluster's outbound authentication token. The last entry of the vector contains one of the ICRs of the destination cluster and corresponding inbound-authentication token. The intermediate entries contain the corresponding transit-authentication tokens. Each entry also contains one bit, called checking bit, indicating whether the packet has been checked by the corresponding router (Fu, 2012).

An example can be used o illustrate the packet forwarding procedure. As shown in Figure 3.2 below, host h_1^1 in cluster C_1 wants to send packets to h_1^3 in cluster C_3. The routing vector in the packets will be like this:

$$\{C_1; r_1^1; 0; tok_1\}, \{C_2; border; 0; tok_2\}, \{C_3; r_3^1; 0; tok_3\}$$

where tok_1, tok_2, tok_3 are the outbound, transit and inbound authentication tokens, and r_1^1 is an ECR in C^1, r_3^1 is an ICR in C_3. Initially, all the checking bits are set to 0 (Zhang, 2012).

2.7.3 Packet forwarding

An outbound packet will be checked by current ECR in the source cluster, and will be routed out of the source cluster through the border router. Then it will pass through intermediate clusters, and will be forwarded into the destination cluster through the border router of that cluster, while in the destination cluster, the packet will be checked by the current of ICR and finally be forwarded to the destination host. Each authentication token will be checked by the corresponding checking routers in each cluster. If everything is valid, the packet will reach its destination (Zhang, 2012).

Considering the example illustrated in the Figure below: A regular packet from h_1^1 to h_3^1 will be first routed to router r_1^1 for checking the outbound authentication token, after which it will be routed to b_1^1 and finally out of cluster C_1, with next hop C_2. In cluster C_2 the packet will be forwarded by the backbone router(s) where the transit authentication token is checked. After passing through C_2, the packet will enter into cluster C_3 via b_3^1 and be routed to router r_3^1 (ICR). After r_3^1 checks the inbound authentication token, the packet can be finally routed to its destination h_3^1. Below he detailed some important steps of the forwarding process using the running example in the figure below:

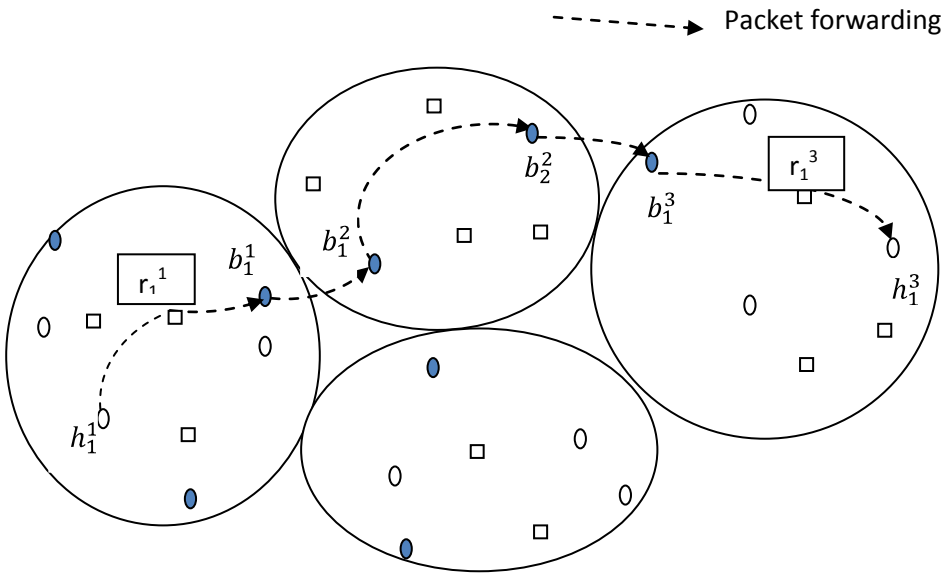

Figure 2.3: Packet forwarding (Zhang, 2012)

2.7.4 Token-Refreshing

Since the ECRs, ICRs and all authentication codes change periodically, a host sending outbound packets may have to request them when a new period begins, which repeats the initial steps. To care for enhanced efficiency considering the cost in the requesting process, each coordinator of every cluster give the information about its ICRs, inbound and transit authentication codes for the new period to the coordinators of other clusters. So upon receiving a refresh request from a host who got the sending permission before, the coordinator does not have to forward the request to the other coordinator(s) that may be involved. If the local cluster did not receive any report of possible misbehaviors of this host, new *"hopping"* information will be granted to the host (Zhang, 2012).

2.8 Cryptography

Cryptography has often been used to protect the wrong things, or used to protect them in the wrong way. Cryptography refers to the science and art of designing ciphers; Cryptanalysis is the science and art of breaking them; while Cryptology, often shortened

to just crypto, is the study of both Cryptography and Cryptanalysis. The input to an encryption process is commonly called the *plaintext*, and the output is called the *cipher text.* There are a number of cryptographic primitive's building blocks, such as block ciphers, stream ciphers, and hash functions. (Caesar and Kennedy, 2010)

There are basically two ways of encryption: Symmetric and Asymmetric cipher encryption.

2.8.1 Symmetric Encryption

Symmetric encryption methods use mathematical operations that can be programmed into extremely fast computing algorithms so that the encryption and decryption processes are executed quickly by even small computers (Whitman *et al.*, 2002).

A symmetric encryption requires the following (Stallings, 2005):

a. Plaintext: This is the original intelligible message or data that is fed into the algorithm as input.
b. Encryption algorithm: The encryption algorithm performs various substitutions and transformations on the plaintext.
c. Secret key: The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.
d. Ciphertext: This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts. The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.

e. Decryption algorithm: This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

2.8.2 Asymmetric Encryption

Asymmetric encryption uses two different but related keys, and either key can be used to encrypt or decrypt the message. If, however, key A is used to encrypt the message, only key B can decrypt it, and if key B is used to encrypt a message, only key A can decrypt it. Asymmetric encryption can be used to provide elegant solutions to problems of secrecy and verification. This technique has its highest value when one key is used as a private key, which means that it is kept secret (much like the key in symmetric encryption), known only to the owner of the key pair, and the other key serves as a public key, which means that it is stored in a public location where anyone can use it. This is why the more common name for asymmetric encryption is *public-key encryption* (Whitman *et al.*, 2002).

The RSA algorithm is based on the computational difficulty of factoring large composite numbers and computing the e^{th} roots modulo, a composite number for a specified odd integer *e*.

In RSA's asymmetric algorithm, which is the basis of most modern public-key infrastructure (PKI) systems the following steps are considered:

a. Key generation: Prime factors p and q are selected by a statistical technique known as probabilistic optimality testing and then multiplied together to form n. The encryption exponent e is selected, and the decryption exponent d is calculated. Encryption in RSA is accomplished by raising the message *M* to a nonnegative integer power *e*. The product is then divided by the nonnegative modulus *n* (n should have a bit length of at least 1024 bits), and the remainder

is the ciphertext *C*. This process results in a one-way operation (shown below) when n is a very large number.

$$c = \frac{Me}{mod\ n}$$

In the decryption process, the ciphertext C is raised to the power *d*, a nonnegative integer, as follows:

$$D = e - 1\ mod((p-1)(q-1))$$

C is then reduced by modulo n. In order for the recipient to calculate the decryption key, the *p* and *q* factors must be known. The modulus n, which is a composite number, is determined by multiplying two large nonnegative prime numbers, p and q

a. Encryption: M is raised to the power of e, reduced by modulo n, and remainder C is the ciphertext.

b. Decryption: C is raised to the power of d and reduced by modulo n.

Cryptographic hash functions are hash functions needed for securing applications. A cryptographic hash function is an algorithm for which it is computationally infeasible because no attack is significantly more efficient than brute force. Some hash functions include: SHA1, Blowfish (Stallings, 2005).

2.8.3 Secured Hash Algorithm 1 (SHA1)

Secured Hash Algorithm 1 (SHA1) is an algorithm that hashes message of 2^{64} bit and produces hash result of 160bits as its hash value (Manuel, 2008). The SHA1 is used to compute a message digest for a message or data file that is provided as input. The message or data file should be considered to be a bit string. The length of the message is the number of bits in the message (James, 2003). SHA1 using the function below for hashing:

SHA1($input)

where SHA1( ) is the method and $input is the parameter for hashing

2.8.4 Blowfish

Blowfish is a symmetric fast block cipher, except when changing keys. Each new key requires pre-processing equivalent to encrypting about 4kilobytes of each texts, which is very slow compared to other block ciphers. The slow key changing is a benefit in blowfish; it has a hash format which is passed in the blowfish function (Gonzalez, 2007).

Some advantages of Blowfish are (Rajender *et al.*, 2014):

a. Blowfish uses a key (salt) for hashing not less than 22 characters.
b. Blowfish encrypts with a hash format ($2y$).
c. Blowfish has 32 – 448bits hash value.
d. Blowfish cipher is slower to process request thereby leading to less brute forcing attack.

The three security characteristics are Confidentiality, Availability and Integrity (Sattarova and Tao-hoon, 2007).

2.9 Related Works

A number of reactive and proactive methods have been developed to mitigate DDoS attack on the network, some of which includes:

Timothy (2005) proposed method of mitigating Network-based Denial of Service attacks with Client Puzzles. He first made some assumptions clear in the protocol Transmission Control Protocol (pTCP) before he introduced the pTCP which are shown in the table below:

Table 2.2: Assumptions of pTCP

Assumption 1:	Attackers are generally more aggressive and send more requests than the average legitimate client.
Assumption 2:	The server under attack can process every incoming packet and send responses to each client.
Assumption 3:	If a stateful firewall is protecting the victim server, the puzzle mechanism can be embedded into the firewall, or rules can be added to the firewall (to allow ACK packets without state information to pass through) if the puzzle mechanism is chosen to be implemented at the server.

The limitation of his work is that the three-way handshake implemented in TCP has lead to security problems, mainly because the server can allocate resources before clients are authenticated.

Feng (2003) documented the need to place client puzzles in the TCP or IP-layer. They show how client puzzles can be integrated within the TCP stack to prevent resource-exhaustion DoS attacks. Chained Puzzles is an extension to the Internet Protocol (IP) that utilizes client puzzles to mitigate the crippling effects of a large-scale DDoS flooding attack by forcing each client to solve a cryptographic problem before allowing them to send packets into the network. The limitation of his work is that the pTCP does not provide a solid defense against large-scale flooding attacks such as the DDoS and the server may not be able to process all incoming packets in large flooding attacks. In our proposed work, the server will be able to process all incoming packets no matter how large the packet is on the network.

Wheeler and Needham (1994) proposed the Tiny Encryption Algorithm (TEA) algorithm which is a block cipher encryption algorithm. Both the encryption and decryption algorithms are Feistel routines that encrypt or decrypt data by several rounds of addition, subtraction, bit-shifting, and exclusive-OR operations. The goal of the encryption algorithm is to create as much diffusion as possible by incorporating many rounds or

iterations of these operations. Due to some weaknesses observed in TEA after it was released as an encryption algorithm Wheeler and Needham developed an extension to TEA, called XTEA.

Peng *et. al.* (2003) relied on the idea and use IP address database (IAD) to keep frequent source IP addresses. During an attack, if the source address of a packet is not in IAD, the packet is dropped. Hash based/Bloom filter techniques are used for fast searching of IP in IAD. This scheme is robust, and does not need the cooperation of the whole Internet community.

The main strength of their work is that the destination can now control the traffic according to its own policy, thereby reducing the chances of DDoS attack, its limitation is that packets without capabilities are treated as legacy and might get dropped at the router when congestion happens.

Li *et al. (*2002) proposed a new protocol called the Source Address Validity Enforcement (SAVE) protocol, which enables routers to update the information of expected source IP addresses on each link and block any IP packet with an unexpected source IP address. The aim of the SAVE protocol is to provide routers with information about the range of source IP addresses that should be expected at each interface. Its limitation is that any IP address not within the range will not receive and send any packet.

Mohammed (2010) proposed a Novel Source – End Packet Filtering Scheme for IP Spoofing Prevention. He partitioned prevention filtering schemes into two namely: Source - End filtering Schemes and the Victim – End filtering Schemes. They both prevent attacks by dropping the attacked packets before reaching the victim.

The overall architecture of the proposed filtering scheme is installed at an edge router connecting a stub network to the Internet. The main idea behind the proposed filtering

scheme is that the edge router first securely generates a key for each IP address in its stub network and each client in the stub network receives its valid key from the router. Limitation of the work is that it only verifies the first packet and subsequently allows other incoming packets from that IP address flow without verification. In our proposed work all packets coming in and out for transmission purpose will all be checked at all time.

Zhang (2012) proposed a proactive method of mitigating DDoS attacks using CluB (Cluster Based) proactive framework to mitigate DDoS attacks. Each cluster C_i will mainly control three types of inter-cluster traffic which are Outbound traffic i.e. from C_i to other clusters, Inbound traffic i.e. from other clusters to C_i and Transit traffic i.e. traffic that passes through C_i and goes to another cluster.

The limitations observed in his work are as follows:

a. If an attacker registers successfully with a router, it will be allocated a valid key on the network.
b. The work didn't consider compromised routers, if an attacker pass the gateway router his/her packet will be sent successfully
c. The work is too expensive to implement with respect to management of the network as it has to implement egress/ingress checking on all routers in the network.

In our proposed work, the gateway will also be a cordinator entity therby dealing with the issue of compromising a router and all implementation takes place on the gateway as well as the Stronger hashing method was used to avoid brute-force attack on the gateway router.

CHAPTER THREE: PROPOSED E-CluB FRAMEWORK

To achieve the design of the E–CluB framework, it is necessary to have a clear and easy understanding of the system architecture which requires a good and user friendly environment for configuration. Fortunately, Graphical Network Simulator 3 (GN3), Hypertext Preprocessor (PHP) programming language were used in designing the E-CluB framework making the framework user friendly and the Kali Linux Operating System (Backtrack 5) was used to launch the DDoS attack along-side the wireshark analyzer was used to get the simulation results of the existing and proposed system.

3.1: E-CluB Network Architecture

System architecture describes how the major components of the system are organized and how they interact with one other. This will allow developers to visualize what the proposed system will be like. A well thought out system architecture will ensure that the correct components are selected to enable them interact among themselves. Without a well thought out architecture unforeseen problems might be encountered that will affect the success of the framework.

The architecture of the framework developed is made up of five major phases. An overview of the architecture can be seen in figure 3.1 below.

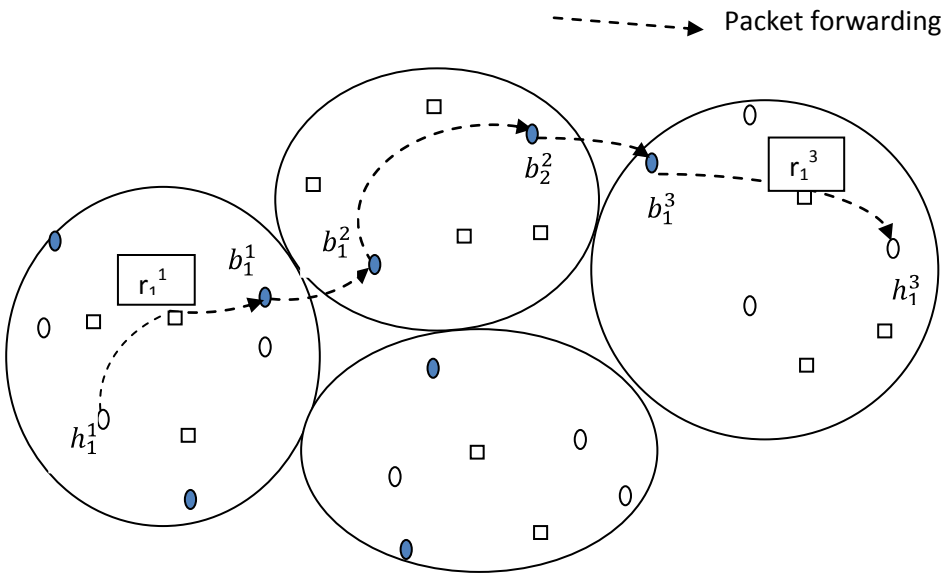

Figure 3.1: Proposed System architecture

The architecture of the E-CLUB framework in figure 3.1 above consists of permission requesting, packet encapsulation, packet forwarding, filtering replayed packets, and token refreshing which are all shown in the architecture above.

3.1.1 Permission Requesting: the coordinating entity i.e. the coordinator can also be the gateway router at certain periods, the notification message reply's the sending host and also the same notification message goes to the border router so as to tackle the issue of compromising a checking router at the border end. The notification message contains the source IP, destination IP, checking router ID, current sequence which is hashed with the Blowfish function, outbound, inbound and transit authentication code for the packet to be transmitted. This notification message will be sent to the sending host and also the border router of the cluster for hashing to take place in order to get the authentication token for transmission. The intermediate clusters are chosen based on the Open Shortest path first (OSPF) which is a dynamic routing protocol.

3.1.2 Packet Encapsulation: the sending host computes its token as the border router also computes it's token so as to compare with the token of the packet in transmission.

The token computed consist of the source IP, destination IP, the hashed sequence number and the checking router ID received from the coordinator. The packet intended to be sent will be encrypted using the public key encryption so that it will be decrypted at the receiving end using the receiver's private key. It then sends the token that has been generated alongside a sequence number with the already encrypted packet to the destination IP address through the border routers. The hashing is done using the blowfish function below while the encryption is done using the public key encryption.

```
function blowFEncrypt($input){
        $hash_format="$2y$10$";
        $salt="HappyhappYHApPyHappyhAPPY";
        $format_and_salt = $hash_format.$salt;
        $hash = crypt($input,$format_and_salt);
        return $hash;
}
```

The packet was encrypted using public key encryption method; which used key A and B for encryption and decryption respectively, where key A is the public key known publicly and Key B is the Secret key known only to the user.

3.1.3 Packet Forwarding: from figure 3.1 above, a packet was routed from host1 (h1) to host3 (h_3) through the gateway router b_1^1 where all the properties were checked by comparing the hashed authentication token and sequence number if they correspond with the token the gateway router hashed before forwarding the packet to its intended destination. The token will be stored in the Bloom Filter to avoid an attacker replaying token for the current packet in transmission else if the token does not correspond then the packet will be labeled from an attacker.

3.1.3.1 Forwarding within the source Cluster: when a packet is ready to be sent out from one host to another within the same cluster. Then the coordinator for that cluster gives the outbound/inbound authentication code for the packet. The sending host sends to

the coordinator for permission and gets an outbound authentication token and a sequence number for its sending packet and then finally sends the packet out through the gateway and then the gateway also receives and compares all properties of the ICR then forwards it to the destination host.

3.1.3.2 Forwarding in the intermediate Cluster and the destination cluster: these intermediate clusters are chosen based on the open shortest path first routing protocol. When packets pass through an intermediate cluster, they will be forwarded by the backbone routers. When a backbone router receives a transit packet, it will check the value for the transit-authentication token contained in the packet by recomputing the token and comparing the two values. If the two values match, then the packet will be routed to the next cluster. After being forwarded through the intermediate cluster, the packets will reach the destination cluster.

3.1.4 Filtering replayed packets: in E-CluB, the authentication tokens were implemented by directly hashing the values of the sequence number, source IP, destination IP and the cluster IDs using blowfish hash function. Since the authentication codes are hashed with source IP, and destination IP the attacker can only "replay" packets during the current period. So, to prevent replaying packets in current period, the authentication token were packet specific by hashing the sequence number, cluster ID, source IP and destination IP.

3.1.5 Token Refreshing: since the ECRs/ICRs changes periodically, a host sending outbound packets will have to always request them when a new period begins. To enhance the efficiency in respect to cost in the requesting process, the coordinator of each cluster gives information about the ECRs/ICRs, inbound and transit authentication codes for each new periods.

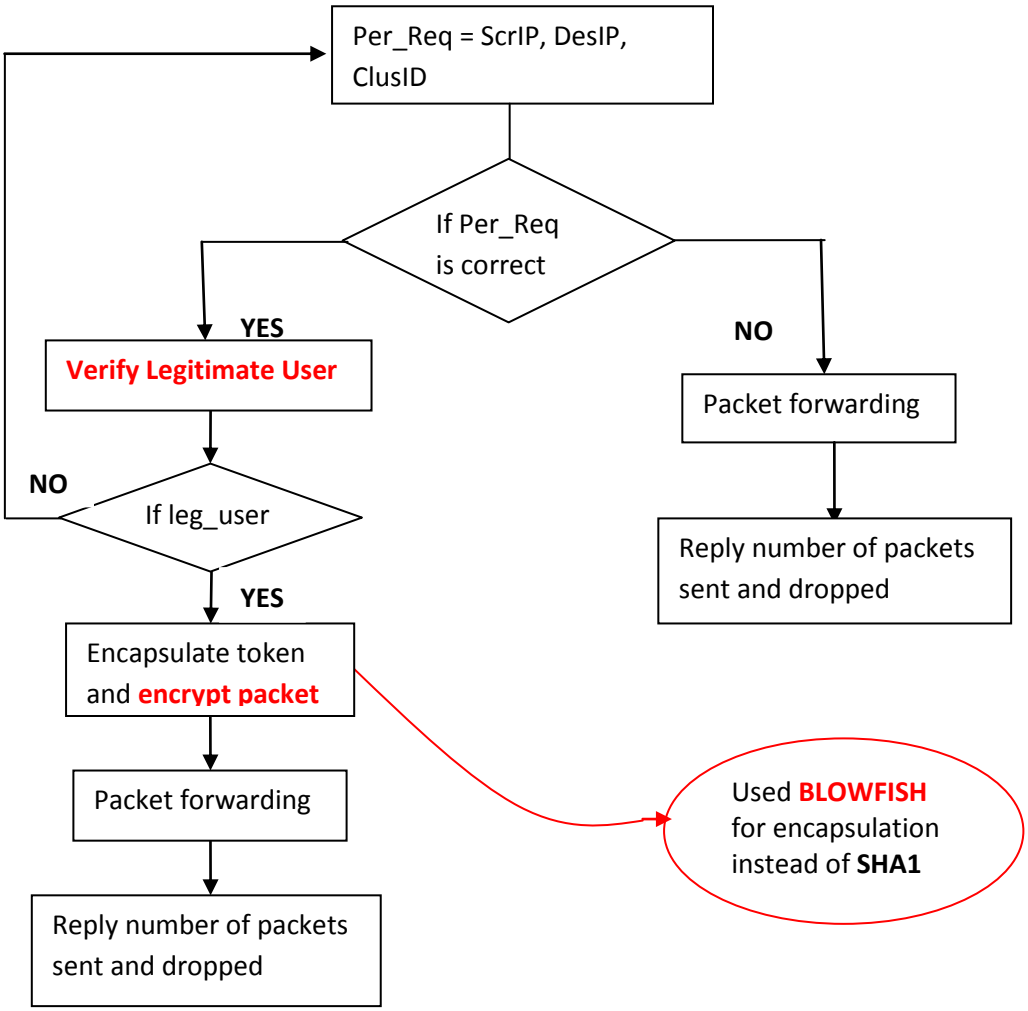

Figure 3.2: Flow Diagram of proposed System

3.1.6 Proposed E-CluB Algorithm

//srcIP is the sourceIP

// dstIP is the destination IP

//AuthToken is the authentication token generated

// Authcode is a Boolean value for notification reply

//salt is the key for hashing

initiate isECR; /* This denotes whether this router is an ECR*/

clear the Bloom filter; /* Used for checking replay packets */

AuthToken ← Outbound authentication token for current period;

PID ← the number of current period;

routerECR ← PKout.ECR; //when receive an outbound packet PKout

/*check if isECR and then should check the incoming packet*/

if isECR = true && routerID = = routerECR then

SeqNum← PKout.SeqNumber;

hash ← HASH(srcIP||desIP||AuthCode||PID||routerID);

if HASH(salt ⊕SeqNum||routerID) = = PKout.OUTAuthToken then

insert(SeqNum||srcIP||desIP)

/* into Bloom filter, if it already exist return False, otherwise return True.*/

PKout.outBit = 1;

if PKout.OUTAuthToken = = Cluster.OUTAuthToken;

route the packet out of the cluster;

else

drop PKout;

end if

else

drop PKout;

end if

end if

if isECR != true and routerID = = routerECR then

drop PKout;

end if

if !PKout.outBit and routerID ≠ routerECR then

route PKout to routerECR;

end if

3.2 Graphical Network Simulator 3 (GNS3) is a network simulator that allows simulation of complex networks. To provide complete and accurate simulations, GNS3 is strongly linked with:

a. Dynamips, a Cisco Internetwork Operating System (IOS) emulator.

b. Dynagen, a text-based front end for Dynamips.

c. Qemu, a generic and open source machine emulator and virtualizer.

d. VirtualBox, free and powerful virtualization software.

GNS3 is an excellent complementary tool to real laboratories for network engineers and administrators. Dynamips is the core program that allows IOS emulation. GNS3 runs on top of Dynamips to create more graphical environment and user friendly environment for its users.

Dynamips software emulates Cisco IOS on a traditional PC and GNS3 control the network logics. The GNS3 was used to design the IPv6 anycast network.

3.3 Backtrack 5

Backtrack 5 is a notorious Digital Forensic and Intrusion Detection software bundle with a whole lot of tools for Penetration Testing. It is based on Linux and includes 300 plus tools. The Backtrack was used on a Kali Linux Operating System to launch the DDoS attack on the anycast network and its subordinate tool such as the wireshark was used to filter the probe requests on the attack.

3.3.1 Wireshark

The wireshark is a network analyzer tool used to analyze and capture data on a network. Wireshark is software that understands the structure of different networking protocols. Thus, it is able to display the encapsulation and the fields along with their meanings of different packets specified by different networking protocols. The wireshark was used to

get simulation of DDoS attack on IPv6 anycast network and so also the mitigation results on the existing and proposed system were captured using the wireshark tool.

3.4 E-CLUB Framework Topology Design

The topology will take one address scheme and interface design which is the IPv6 address since the network is considered an anycast network.

3.4.1 IPv6 Address Schemes and Interfaces Design

There are ten router and eight end-nodes that are participating in the anycast session designed using the GNS3. IPv6 unicast-routing, IPv6 anycast-routing and IPv6 OSPFv3 were enabled in the network for the routing to take place. Each of the interfaces on the router was configured with IPv6 global unicast/anycast address and PIM-SM enabled. The network was configured to work as an anycast network where by all the routers will be up and functioning as an anycast router.

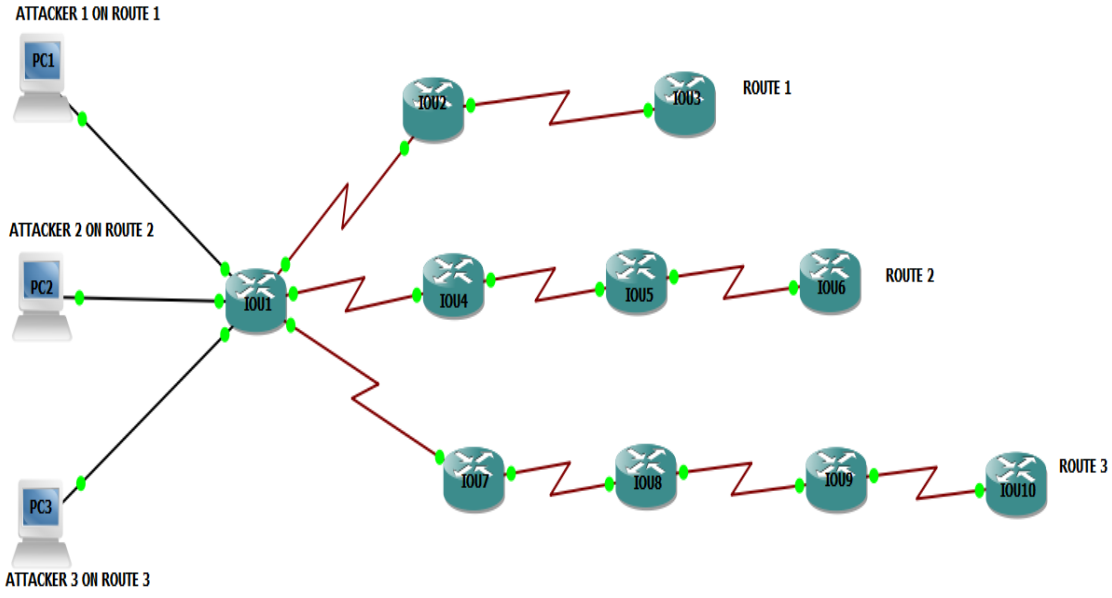

Figure 3.3: Anycast network showing DDoS attack in a cluster

3.5 Chapter Summary

This chapter designed the various architectural components and the topology of the E-CluB Framework which is shown using the flow chart. The next chapter discusses the result of the proposed system in accordance to the granularity control, filtering efficiency and latency of transmission.

CHAPTER FOUR: IMPLEMENTATION AND RESULT

This chapter presents an overview of the various tools and technology used in this dissertation. The user interfaces of the developed system are presented so also the testing and discussion of results. Finally, the system was compared with the closest work in terms of filtering efficiency, granularity control and latency in transmission of packets.

4.1 System Requirements

This section describes the tools and technologies used in this work: Apache Web Server, Dreamweaver, Hypertext Preprocessor (PHP) were used and discussed below.

4.1.0: Apache Web Server

Apache web Server is an open source web server creation, deployment and management software. It was designed to create web servers that have ability to host one or more HTTP-based websites. It can also support multiple programming languages, server side scripting, an authentication mechanism and also database. It can also be enhanced by manipulating the codes base or adding multiple extensions.

It is widely used by web hosting companies for the purpose of providing shared/virtual hosting as default Apache web server also supports and distinguishes between different hosts that reside on the same machine.

4.1.1: Dreamweaver

Dreamweaver is an adobe development, created by Macromedia and later acquired by Adobe. It is available for windows to write codes (program). Dreamweaver let users design code and manage websites as well as mobile content.

4.1.2: Hypertext Preprocessor (PHP)

PHP is an open-source HTML embedded server side scripting language which is used to develop dynamic and interactive web applications and also used as a general purpose

programming language. Lot of syntax is borrowed from other programming languages such as C, Pearl. To describe a PHP file we say it is a file with .php extension that contains the html and other scripts that runs on a web server. PHP supports many databases (servers using MySQL and PHP combination widely) and it is also a server side scripting language which means that all code in PHP can be processed on the web servers rather than on the machine. PHP can be used to create web applications such as facebook, yahoo, wikipedia. It works on many operating Systems like Linux, Windows, Mac OS.

4.2 Implementation Details

The E–CluB framework is a system that is made up of several user friendly interfaces which allow users to conveniently send packets and get result from the underlying data store. The E-CluB system is made up of the following user interfaces:

4.2.1 DDoS Attack on an IPv6 anycast network

Figure 4.1 describes the DDoS attack on an IPv6 anycast network, which was launched with the Backtrack 5 on the Kali Linux OS. The attack was launched using 9 virtual machines and 1 machine was the targeted server without any mitigation technique to show how the attacker intends to bring down the targeted server with its flooded packets varying and the other was the analyzer which analyzed the simulation result of the attack (wireshark) and the remaining 7 virtual machines were used in the process.

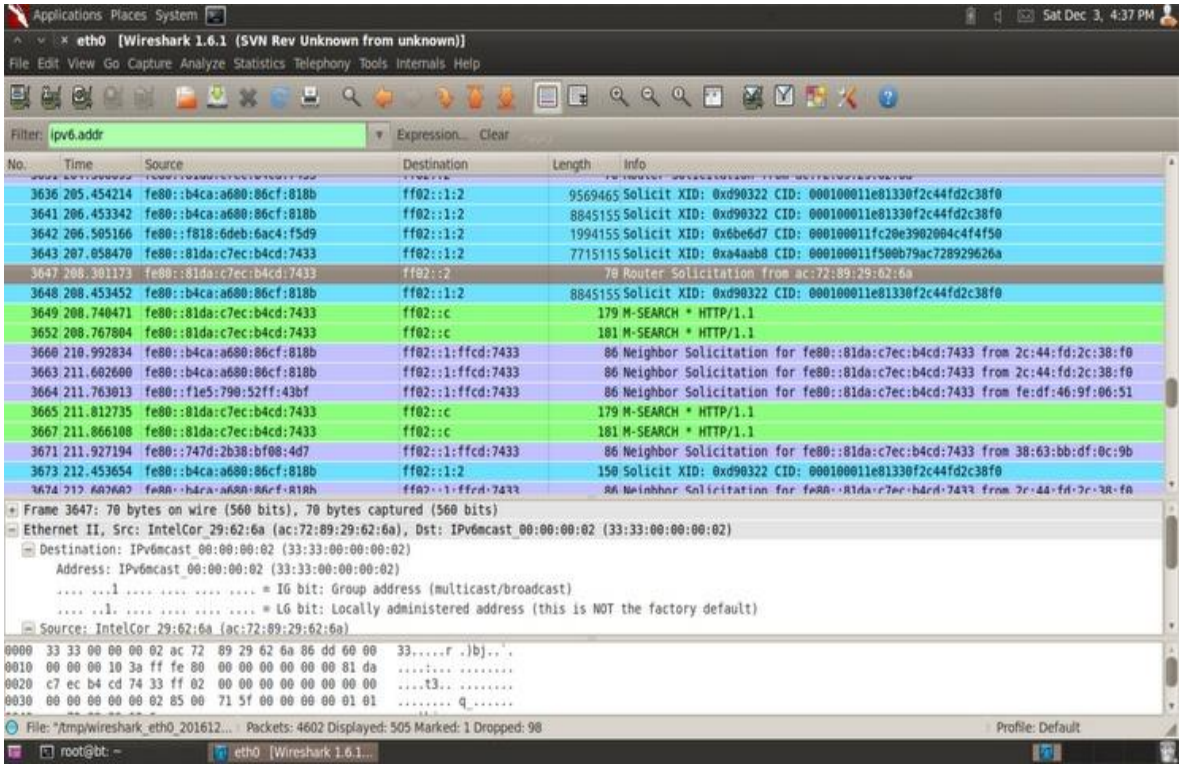

Figure 4.1: DDoS attack on IPv6 anycast network Interface

4.3.2 DDoS attack Mitigation on an IPv6 anycast network Interface

The figure 4.2 (a and b) below shows the mitigation effect of the DDoS attack on the anycast network.

Figure 4.2a below shows the mitigation technique interface of the existing method (Zhang, 2012), the attack in figure 4.1 above was mitigated with the SHA1 hashing as proposed by Zhang 2012 and the simulation result was shown using the wireshark analyzing tool to show the effect of the mitigation technique on the attack.

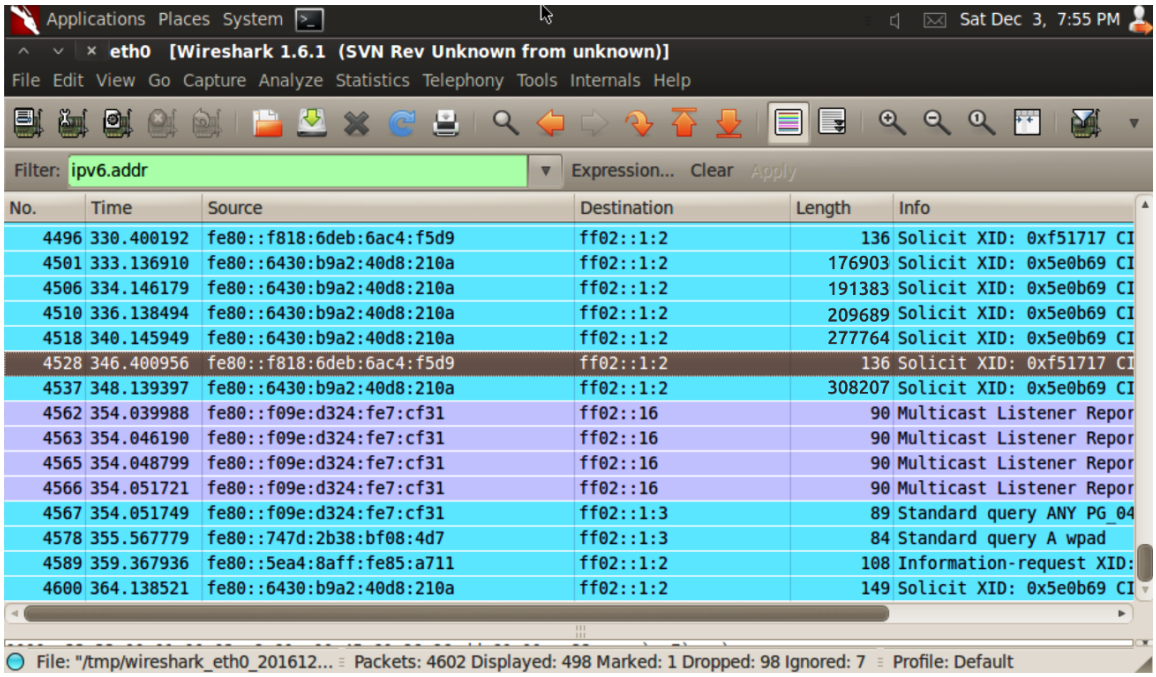

Figure 4.2a: Existing method mitigation techniques interface

Figure 4.2b below shows the mitigation technique of the proposed method, the attack in figure 4.1 above was mitigated with the proposed Blowfish hashing method.

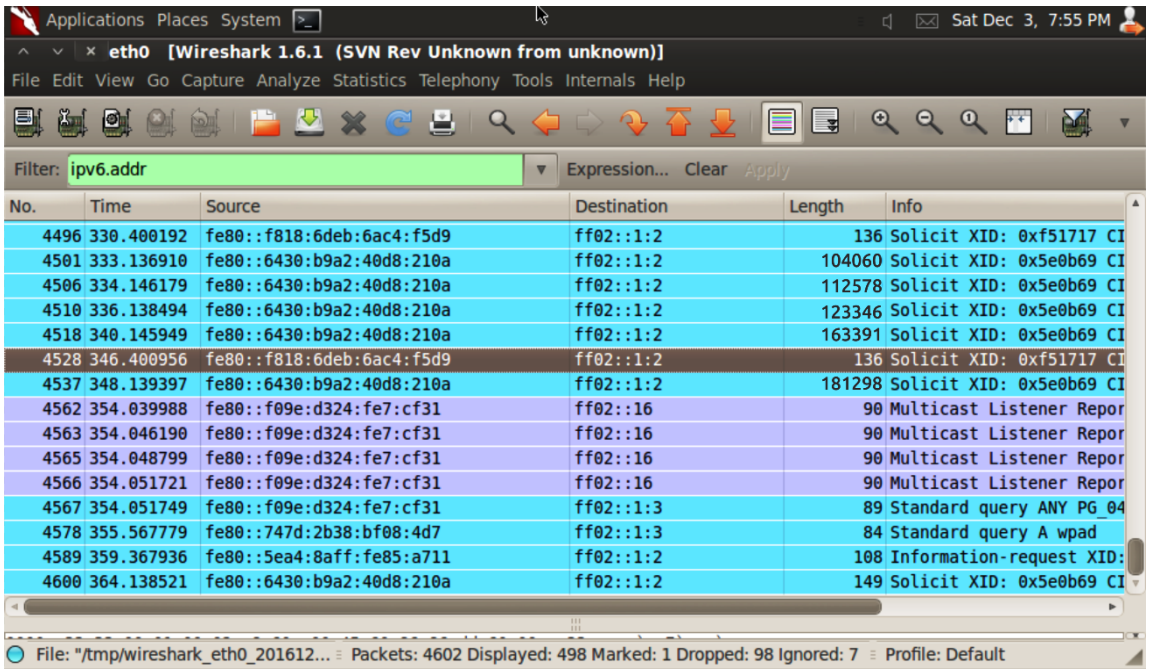

Figure 4.2b: Proposed method mitigation techniques interface

4.4.4 Cluster Database

The figure 4.4 below is a database of the 4 clusters namely c1, c2, c3, c4, which has 5 hosts respectively in each cluster. These hosts are routers and end-devices which respond to activities (such as sending and receiving) within and out of a cluster. Packets can be sent within a cluster i.e. within c1 and packets can also be sent from one cluster to another depending on the sending and receiving host cluster ID. The IP address of the network is an IPv6 anycast address.

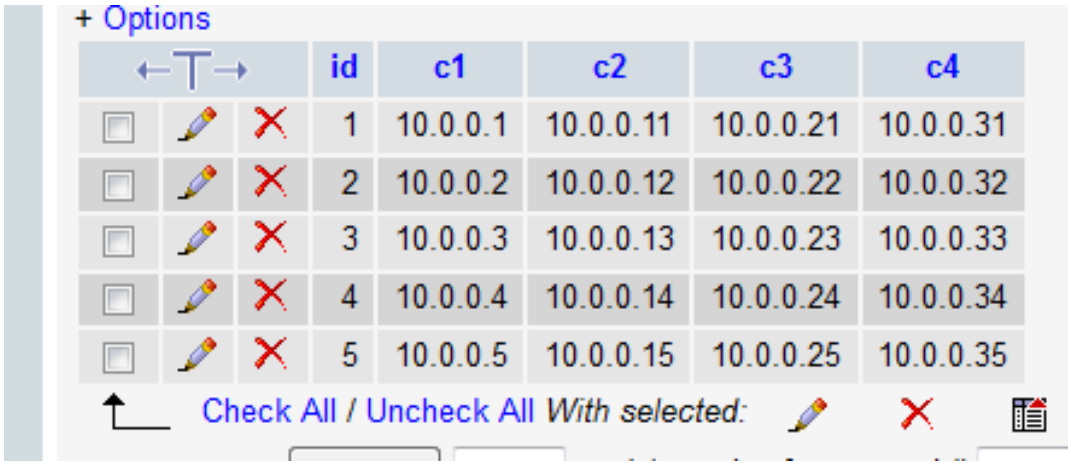

Figure 4.3: Cluster of Nodes

4.4 Result Analysis

The performance of the proposed system was tested with various lengths of packets. The proposed system was compared with the work of Zhang (2012) based on the following parameters:

a. Granularity control

b. Efficiency control

c. Latency of transmission

4.4.1 Result based on Granularity control of clusters

Each cluster in the E-CluB framework is mapped an autonomous system on the internet. Optimizing the granularity of clusters will involve security and processing load factors where CluB used 1000 routers and proposed E-CluB used 1% of the routers in CluB as its checking routers to reduce the probability of compromising routers.

If a router is compromised the inbound/outbound authentication code will be revealed. So to reduce the probability of revelation of the authentication codes the checking routers was reduced to 1% of the existing system which enhanced the performance of the system.

4.4.2 Result based on efficiency filtering of CluB and E-CluB

The graph and table below shows the analysis of packets sent under attack in CluB and E-CluB. The existing system using SHA1 hashing method has more packets leaving a cluster compared to the proposed system using BLOWFISH hashing method which has fewer amounts of packets leaving a cluster when packets are sent by attackers or assumed illegitimate sender. Some amount of packets still leaves the cluster as no 100% security is assured over a Network but the mitigation techniques differs to reduce non-legitimate packets transmitting over a Network.

Table 4.1: Filtering Efficiency

Attacking packet sent (in byte)	CluB (mitigation in byte)	% evaluation	E-CluB (mitigation in byte)	% evaluation
8845155	Sent:176903 Dropped:86682512	1.99	Sent: 104060 Dropped: 8741095	1.17
9569465	Sent: 191383 Dropped:9377772	1.99	Sent: 112578 Dropped: 9456577	1.17
10484455	Sent: 209689 Dropped: 10274766	1.99	Sent: 123346 Dropped: 10361109	1.17
13888243	Sent:277764 Dropped:13610479	1.99	Sent: 163391 Dropped: 13724852	1.17
15410352	Sent: 308207 Dropped: 15102145	1.99	Sent: 181298 Dropped: 15229054	1.17

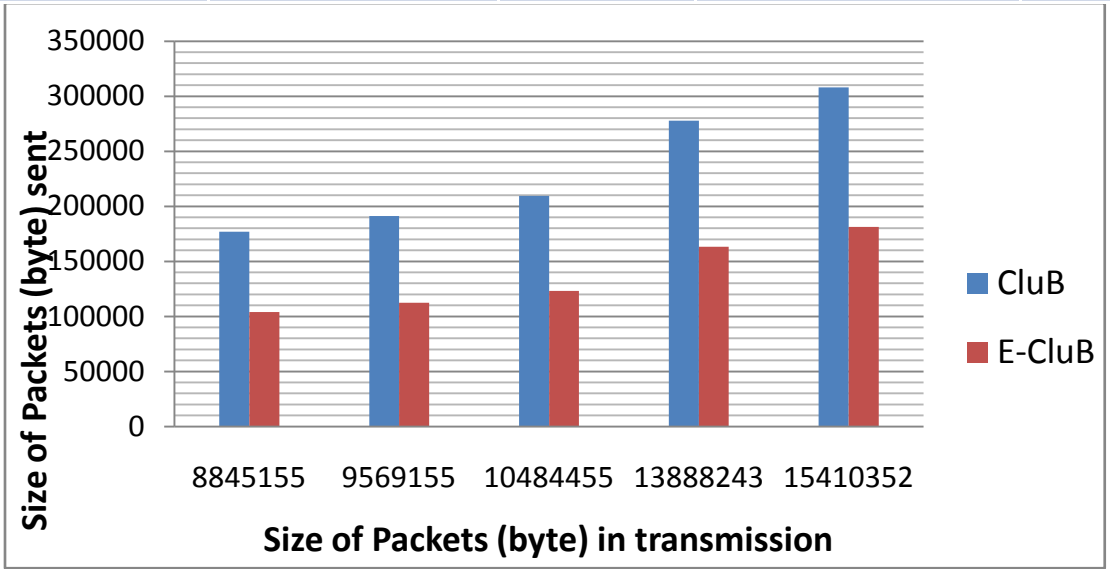

Figure 4.4: Graphical Analysis of Efficiency Filtering

The table 4.1 above gives the effectiveness of the proposed E-CluB over the existing CluB in a tabular form. The figure 4.4 above plots the size of packets against the size of packets sent when efficiency has been filtered in E-CluB and CluB.

4.4.3 Result based on latency of transmission

The time needed in the proposed E-CluB is higher compared to the CluB which makes it difficult for the attacker. The hashing method used in CluB has faster transmission time as compared to the hashing method used in E-CluB as proved by Schneier (1993)

The existing work has a slower time of transmission of packets because the enhanced algorithm in E-CluB has been proven slow over other hashing methods.

4.5: Chapter Summary

This chapter has provided an overview of the various tools and technology used. The E–CluB framework accepts any form of data in packets and covert them to bytes for record purpose. The system comparison with the closest work shows that, E–CluB framework has improvement over the CluB with about 41.2% based on filtering efficiency of packets.

CHAPTER FIVE: Summary, Conclusion, and Recommendation

5.1 Summary

The aim of this dissertation was to find out the impact of mitigating DDoS attacks on anycast Networks. It was achieved by implementing and simulating the existing and proposed systems with equal datagram's. First, was on the existing system and the proposed system. Then, finally both were compared and results were shown in tabular and graphical form.

The performance of the systems were measured based on the following parameters; granularity control, filtering efficiency and latency of time.

5.2 Conclusion

E-CluB is a proactive method proposed for mitigating DDoS attacks. It is a distributed method that requires clustering in the network and can combine with the OSPF routing protocol. It was deduced from the experiments that, E–CluB framework has a better performance compared to the CluB framework with 41.2% improvement over the existing system. Therefore, we recommend that after more investigations, E–CluB should also be used in securing sequence numbers and authentication tokens in an anycast network so as to mitigate DDoS attack.

5.3 Recommendation

To improve on this research, a live implementation of the E-CluB framework should be carried out instead of the virtualization done in this dissertation.

5.4 Contribution to Knowledge

Experiments were carried out to show that preventing DDoS attacks in anycast Networks using the BLOWFISH hashing method has more improvement of 41.2% in terms of filtering efficiency compared to the existing hashing method.

References

Abadi, Burrows, M., Manesse, M. and Wobber, T. (2003). Moderately Hard, Memory-bound Functions. *In Proceedings of the Network and Distributed System Security Symposuim, 2*(2), 25 - 39.

Anderson, T., Roscoe, D. and Wetherall. (2004). Preventing Internet Denial of service with capabilities. *In Proceedings of ACM SIGCOMM Computer Communication Review, 34*(1), 39 - 44.

Baker, F. (1998). Requirements for IP version 4 routers. *International Engineering Task Force (IETF), 3*(4), 8 - 12.

Barlow, J. and W, T. (2000). TFN2K - An analysis. *Axent Security Team Springer* , 5 - 12.

Bremler- Barr, A. and Leavy, H. (2005). Spoofing Prevention Method. (F. I. Publication, Ed.) *in proceedings of IEEE INFOCOM* , 2- 180.

Caesar, J. and Kennedy, J. (2010). *Security Engineering: A Guide to Building Dependable Distributed Systems.* United States: Pearson Practice Hall.

Casado, M. Freedman, Petit, J., and Mckeown, N. (2007). Ethane Taking control of the enterprise. *In proceedings of ACM SIGCOMM Computer Communication Review , 37*(4), 1-`12.

Dietrich, S., Dittrich, S. and Long, N. (2000). Analyzing Distributed Denial of Service tools: The Shaft case. *In proceedings of the 14th Systems Administration Conference, 5(4),* 329-339.

Dittrich. (1999). *The Tribe Flood Network Distributed Denial of Service attack tool.* University of Washington. Washington: htttp://staff.washington.edu/dittrich/misc/tfn.analysis.txt.

Dittrich, D. (1999). *The Stecheldraht Distributed Denial of Service attack tool* . Washington: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt.

Douligeris and Mitrokotsa, A. (2003). DDoS attacks and defense mechanisms: classification. *3rd IEEE International Symposium on Signal Processing and Information Technology* , *9*(3) 190 - 193.

Douligeris, C. and Mitrokotsa, A. (2004). DDoS attacks and defense mechanisns: Classifications. *Proceedings of the 3rd IEEE International Symposuim on Signal Processing and information Technology , 44* (5), 643 - 666.

Felix, L., Rubin, S. and Smith, M. (2000). Distributed Denial of Service Attacks. *In Proceedings of IEEE International Conference on Systems, Man and Cybernetics , 3*, 2275 - 2280.

Feng, W. (2003). The case for TCP/IP Puzzles. *In proceedings of the ACM SIGCOMM Workshop on future Directions in network architecture , 3*(5), 322 - 327.

Ferguson, P. and Senie, D. (1998). Network ingress filtering: Defeating Denial of service attacks which employ IP source address spoofing. *Internet Engineering Task Force (IETF) ,* 5(6), 2246-2267.

Mitigation. *In Proceedings of SAC, ACM , 38*(3), 147-180.

Garfinkel, S. and Spafford, G. (2001). *Web Security, Privacy and Commerce* (2nd Edition ed.). London: O'Reily Medai Inc.

Geng, X. and Whitson, A. B. (2000). Defending Distributed Denial of Service attacks. *International Journal of Electriacal and Elaectronic Engineering , 2*(4), 36 - 42.

Gibson, S. (2002). Distributed reflection Denial of Service. *A Technical report ,* 29.

Gonzalez, T. (2007). A Reflection Attack on Blowfish. *Journal of Latex Class Files , 6*(1), 1-6.

Gupta, B. B., Joshi, C. and Misra, M. (2010). Distributed Denial of service Techniques. 455-627.

Hammouda, S. and Trabesl, S. (2007). An Enhamced Secure ARP Protocol and LAN switch for preventing ARP based attacks. *ACM SIGCOMM , 4*(2), 8-22.

Hazehurst, S. (2000). Algorithms for Analyzing Firewall and Router Access Lists. *In proceedings of workshop on dependable IP systems and platforms ICDSN, 8*(4), 5-18.

James, J. (2003). *Vocal Technologies: SHA1 Encryption algorithm.* New York: Vocal Technologies. *5*(3), 2-14.

Kaur, S. and Kaur, S. (2013). Analysis of Zone Routing Protocol in MANET. *International Journal of Research in Engineering and Technology , 2*(9), 9-12.

Kent, Lynn, C., Mikkelson, J. and Seo, K. (2000). Secure border gateway protocol (s-bgp) - real world performance and deployment issues. *NDSS, 5*(3), 14-23.

Keromytis, A. D., Misra, V. and Rubenstein, D. (2002). SOS: Secure Overlay Services. *In the Proceedings ACM SIGCOMM , 8*(8), 61-72.

Kumar, K., Joshi, R. C. and Singh, K. (2006). An Integrated Approach for Defending against Distributed Denial of Service (DDoS) attacks. *IT Madras, 7*(4), 1-17.

Li, J., Mirkovic, M., Wang, and Reiher, P. (2002). SAVE: Source address Validity Enforcement Prootcol. *Proceedings of IEEE INFOCOM ,* 3(3), 1557-1566.

Liu, and Shi. (2012). Clustering Routing Algorithms In Wireless Sensor Networks: An Overview. *Internet and Information Systems, 6* (7), 1735-1755.

Manuel, S. (2008). Classification and Generation of Disturbance Vectors for Collision attacks against SHA1. 320-469.

McNevin, T. J. (2005). Mitigating Network-based Denail of service attacks with Client puzzles. *Internet and Information Systems, 2*(2), 1-19.

Mirkovic, J. and Reiher, P. (2004). A Taxanomy of DDoS and defense Mechanisns. *Computer Communication Review , 34*(2), 39-53.

Mitzenmacher, A. B. (2006). Network applications of bloom filters. *A Survey on internet Mathematics* , 636-646.

Mohammed, N. D. (2010). Defense against Distributed Denial of Service attacks in Computer Networks. In N. D. Mohammed, *thesis*.Egypt: Assuit University. 1-99.

Molsa, J. (2005). Mitigating denial of service attacks: A tutorial. *Journal of Computer Security , 13*, 807-837.

Moore, D., Shannon, C., Brown, D. J., Voelker, G. and Salvage, S. (2006). inferring internet Denial of Service activity. *ACM Transactions on Computer Systems , 24* (2), 24.

Moore, G., Voelker, G. and Savage, S. (2001). Inferring denial of service activity. *In proceedings of the 10th usenis security symposium.* , 9 -22.

Network, J. (2006). Combating Bots and Mitigating DDoS attacks. *Juniper Networks* , 14.

Paduch, J., Levy, J. and Khan, B. (2009). Using Secure permutational convert channel to detect local and wide area inteposition attacks. *International Conference on Wireless Communication and Mobile Computing* , *5*(6), 79-83.

Park, K. and Lee, H. (2001). On effectiveness of router-based packet filtering for distributed DoS attack prevention in power-law internets. *In Proceedings of ACM SIGCOMM , 8*(2), 15-18.

Patridge, Mendez and Milliken. (1993, November). Host anycasting service. *Internet Research Task Force (IRTF)* , 3 - 9.

Peck, T., Leckie and Ramamohararao. (2003). Protection from Distributed Denial of service attack using history-based IP filtering. *In Proceedings of IEEE Internartiona Conference on Communications , 1*(3), 482 - 486.

Peng, Ramamohanarao, K. and Leckie, C. (2007). "Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems,. *ACM Computing Surveys, , 39*(1), 1-42.

PHP. (n.d.). Retrieved July 15, 2016, from W3schools: http://www.w3schools.com

Pretty Good. (1999). An Introduction to Crytography. *Preety Good Privacy* , 6-99.

Rajender, K., Satish, K. and Balwinder, S. (2014). A Novel Approach to Blowfish Encryption Algorithm. *International Journal of Advance Foundation and Research in Science and Engineering , 1*(2), 1-9.

Robinson, M., Mirkovic, J., Schnaider, M. and Michel, S. (2003). Challenges and principles of DDOS defense. *SIGCOMM , 4*(2), 12-45.

Sattarova, F. and Tao-hoon, K. (2007). IT Security Review: Privacy, Protection, Access Control, Assurance and Security System. *International Journal of Multimedia and Ubiquitous Engineering , 4*(3), 306-791.

Stallings, W. (2005). *Cryrography and Network Security Principles and Practices* (5th Edition ed.). United States of America: Pearson Prentice Hall.

Tapakula, U. K. and Varadharajan, V. (2003). A practical Method to conteract denial of service attacks. (S. Verlag, Ed.) *Proceedings of the 26th Australian Conference on Computer Science ACSC* , 46-78.

Vaughn and Evron, G. (2006). DNS Amplification Attacks. *CERT. , 2*(2), 1-16.

Wang, X. and Reiter, M. (2003). Defending against denial of service of attacks with puzzle auctions. *In proceedings of the IEEE Symposuim on security and privacy , 3*(4), 78-92.

Wesley, E. (2007). TCP SYN Flooding Attacks and Common Mitigations. *IETF Trust , 5*(3), 1-19.

Wheeler and Needham, R. (1994). TEA, a Tiny Encryption Algorithm. *Springer , 5(*4), 9-16.

Whitman, Micheal, Mattord and Herbert. (2002). *Principles of Information Security* (4th Edition ed.). Georgia, United State of America: Kennesaw State University.

William, C., Steve, B. and Aviel, R. (2003). *Wily Hacker* (2nd edition ed.). Boston: Addiley - Wesley.

Yadav, A. K. and Rana, P. (2015). Cluster based Routing Wireless Sensor Networks. *Comparative Study , 125*(13), 31-36.

Zhang, F. (2012). CluB: A Cluster based Proactive method for mitigating DDOS attacks. *In Proceedings of IEEE Computer Society, 6*(3), 1-12.

APPENDIX I: Sample Program Coding

Index Page

```
 include("scripts/connection/connection.php");

<!DOCTYPE html />

<html lang="en">

<head>

<!-- <link href="style/styles.css" rel="stylesheet" type="text/css" /> -->

<style> .container{width:400px; margin-left:auto; margin-right:auto;

 } </style>

functioninputDstSrcCltr(){

?>

<div style='margin-top:200px;'class="container">

<form action="code.php" method="post">

<p>

Destination IP: <input type="text" name="dst" id="dst" /> </p>

<p>

Source IP andnbsp; andnbsp; andnbsp;: <input type="text" name="src" id="src" /> </p>

<p>

Cluster IDandnbsp; andnbsp; andnbsp;: <input type="text" name="clstr" id="clstr" />
</p>

<p> andnbsp; andnbsp; andnbsp; andnbsp; andnbsp; andnbsp;  andnbsp; andnbsp;
andnbsp; andnbsp; andnbsp; andnbsp;<input type="submit" name="sbmit_btn1"
id="sbmit_btn1" value="<< Go >>" /> </p>

</form> </div><!-- End of container1 -->

 }

</head>

<body>

inputDstSrcCltr();

</body>

</html>
```

Token Page

```
session_start();
                $_SESSION['DST'];
                $_SESSION['SRC'];
                $_SESSION['CLSTR'];
                $_SESSION['RANDOM_SQN'];
                $_SESSION['SQN'];
 include("scripts/connection/connection.php");
<!DOCTYPE html />
<html lang="en">
<head>
<!-- <link href="style/styles.css" rel="stylesheet" type="text/css" /> -->
<style>
 .container{width:400px; margin-left:auto; margin-right:auto;
 } </style>
```

Message Page

```
session_start();
                $_SESSION['DST'];
                $_SESSION['SRC'];
                $_SESSION['CLSTR'];
                $_SESSION['RANDOM_SQN'];
                $_SESSION['VRF_CODE'];
function blowFEncrypt($password){
        $hash_format="$2y$10$";
        $salt="Salt22CharactersOrMore";
        $format_and_salt = $hash_format.$salt;
        $hash = crypt($password,$format_and_salt);
        return $hash;
}
```

```
                $entrd = $_POST['code'];
                $_SESSION['ENTERED_SQN']=blowFEncrypt($entrd);
 include("scripts/connection/connection.php");
<!DOCTYPE html />
<html lang="en">
<head>
<!-- <link href="style/styles.css" rel="stylesheet" type="text/css" /> -->
<style>
 .container{width:400px; margin-left:auto; margin-right:auto;
 }
</style>
function getMsg(){ //messge field function ?>
<div style='margin-top:200px;' class="container">
<form action="send.php" method="post">
<p>
<table><tr>
<td>Message  andnbsp; andnbsp; :</td><td> <textarea cols="30" rows="5" name="msg" id="msg" ></textarea></td>
</tr></table></p>
<p>
 andnbsp; andnbsp; andnbsp; andnbsp; andnbsp; andnbsp;  andnbsp; andnbsp; andnbsp; andnbsp; andnbsp; andnbsp;<input type="submit" name="sbmit_btn2" id="sbmit_btn2" value="<< Go >>" />
</p></form>
</div><!-- End of container2 -->
</head>
<body>
if(empty($_POST['code'])){
        header("location:code.php");
}
```

```
getMsg();
</body> </html>
```

Server Side Code

```
$text="";
function getByte($text){
$filename = "tmp.txt";
$file = fopen( $filename, "wr" );
fwrite($file,$text);
if( $file == false ){
echo ( "Error in opening file" );
exit( );
}
$filesize = filesize( $filename );
//$filetext = fread( $file, $filesize );
fclose( $file );
//echo ( "File size : $filesize bytes" );
return $filesize;
}
if(isset($_POST["msg"])){
        $msg = $_POST["msg"];
        getByte($msg);
        if($sqn == $gsqn){
                $tbyte = getByte($msg);
                $fragleft='0';
                }else{
                        $tbyte=(getByte($msg);
                        $fragleft=(getByte($msg)-$tbyte);
                        }
echo "<table width='40%' style='margin-top:200px;' align='center'>";
```

```
echo "<tr><td>Total Number of Packets : </td><td>".getByte($msg)." byte </td></tr>";
echo "<tr><td>Number of Packets Sent : </td><td>".$tbyte." byte </td></tr>";
echo "<tr><td>Number of Packets Dropped : </td><td>".$fragleft." byte </td></tr>";
echo "</table>";
        }
```

DDoS attack Code

```
  if ( $working[$z] == 1 ) {
      if ($cache) {
         $rand = "?" . int( rand(99999999999999) );
      }
      else {
         $rand = "";
      }
      my $primarypayload =
         "$method /$rand HTTP/1.1\r\n"
       . "Host: $sendhost\r\n"
       . "Content-Length: 42\r\n";
      my $handle = $sock[$z];
      if ($handle) {
         print $handle "$primarypayload";
         if ( $SIG{__WARN__} ) {
            $working[$z] = 0;
            close $handle;
            $failed++;
            $failedconnections++;
         }
         else {
```

```
            $packetcount++;
            $working[$z] = 1;
        }
    }
    else {
        $working[$z] = 0;
        $failed++;
        $failedconnections++;
    }
}
else {
    $working[$z] = 0;
    $failed++;
    $failedconnections++;
```