SDN BASED SECURITY USING COGNITIVE ALGORITHM AGAINST DDOS

Shah Ali Faizan

This thesis is submitted in partial fulfillment of the academic requirements

for the Degree of

Master Engineering in Telecommunications

in the Faculty of Engineering and The Built Environment

University of Cape Town

December 2018

Declaration

I declare that the work presented in this dissertation is originally my own. Ideas and material generated from other researchers is explicitly stated with appropriate references.

This work is submitted for the Master of Engineering specializing in Telecommunication at the University of Cape Town. It has not been submitted to any other university for any other degree or examination.

Name: Shah Ali Faizan

Signed by candidate

Date: 09th April 2018.

Abstract

The internet and communication industry continue to develop new technologies rapidly, which has caused a boom in smart and networking device manufacturing. With new trends, operators are constantly battling towards deploying multiple systems to cater for the need of all users. The higher bandwidth utilization and flexibility demanded new networking solutions which paved way for Software Defined Network (SDN). SDN is centralized platform which works with other technologies such as Network Function Virtualization (NFV) to offer reliable, flexible and centrally controllable network solutions. It offers remote access control with logical design of the system, security and resource management.

Traditional and new developing networks despite their advantages present numerous security challenges. With growing users worldwide, bandwidth related security risks such as Distributed Denial of Service (DDOS) are of grave concern. This encourages towards reliable and rapid response solutions such as Cognitive Algorithms (CA) which can adapt to a threat in real time environment. This dissertation proposes the use of CA to deploy security and mitigation measures against potential DDOS flooding attack to avoid network failure and memory depletion in SDN.

The experiment done in proof of concept (PoC) provided proof of greater network resource utilization by limiting the attack while mitigation policies are implemented. It also shows that CA can adapt to growing and evolving network attack strength to counter as much as possible without the intervention of the operator. The work for future solutions based on CA and Artificial Intelligence (AI) for security have been established.

Acknowledgement

Firstly, I would like to thank God, for granting me the strength, courage and wisdom throughout this research and in my life, which made it possible for me to come this far.

I would also like to take this opportunity to thank my supervisor Dr. Joyce Mwangama, whose constant supervision and guidance has been critical to my research. I was better able to manage my work only due to her constructive criticism and productive input. It was an absolute joy and privilege to have an opportunity of working under her tutelage.

Finally, I would like to thank my family, especially my mother and father, who have supported me every step of the way. Their constant care and cheer has helped me cope through difficult times and helped me persevere against all the odds – Thank you.

Table of Contents

List of Figures

List of Acronyms

AI	Artificial Intelligence
API	Application Program Interface
BGP	Border Gateway Protocol
CA	Cognitive Algorithm
CAGMA	Critical Amount Guaranteed Memory Allocation
CPU	Central Processing Unit
DDOS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DMA	Dynamic Memory Allocation
DNS	Domain Name Service
DOS	Denial of Service
DPI	Deep Packet Inspection
EMC	Exact Match Cache
ER	Every Request
HLPO	High level Programmable Output
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
LOD	Line of Defense
LOIC	Low Orbital Ion Canon
MiM	Man in Middle
MRA	Memory Release Algorithm
MTD	Moving Target Defense
NAC	Network Access Control
NADA	Network Anomaly Detection Algorithm
NAPA	Network Anomaly Prediction Algorithm
NBT	North Bound Traffic
NETCONF	The “Network Configuration” Protocol
NFV	Network Function Virtualization
NIC	Network Identity Card
ODL	OpenDayLight
OS	Operating System
OSI	Open System Interconnection
OVS	Open Virtual Switch
OVSDB	Open vSwitch Data Base Management
PaaS	Platform as a Service
PDSD	Packet Data Scan Detection
PH	Priority Heuristic
pIF	Physical Interface
PING	Packet Internet Groper
PoC	Proof of Concept
REST	Representation State Transfer
RnD	Research and Development
RT	Routing Table
RTT	Round Trip Time
SaaS	Software as a Service
SAL	Service Abstraction Layer
SBT	South Bound Traffic

SDC	Software Defined Cloud
SDN	Software Defined Network
SecMC	Secure Memory Controller
SPasS	Software Platform as a Service
SRT	Session Request Traffic
TAC	Traffic Analysis Calculation
TVF	Tri-verification
vIF	Virtual Interface
VM	Virtual Machine
VNIC	Virtual Network Interface Controller

CHAPTER 1

INTRODUCTION

1.1. Introduction

Internet is constantly evolving to accommodate the ever-growing paradigm of technology accompanying hardware improvements such as smart phones, efficient networking devices and enhanced software. Rapid increase in web-based applications is observed across all internet forums from social media to banking and financial sectors. Applications like Facebook, Twitter, Paypal which form a fraction of these platforms, all rely on security of services. Privacy and data management is of paramount importance towards functionality of these technology giants. Constant increase in traffic generation, demands equally comprehensive and scalable security systems against adapting threats. Operators are thus faced with a constant battle for keeping up with the ever-changing security measures. One of the major problems is deploying security services in a proprietary market. Operators use vendor specific devices which have proprietary features. This presents problems in terms of finance, hardware and other related constraints. Moreover, for each new enhancement, a new node (device) must be established and extra labor to maintain it. Increasing device management and manpower becomes cumbersome for the biggest of operators. This problem has given way to the development of an effective and very versatile solution.

Research and development (RnD) ventures across multiple vendors, operators and academia have paved way for a generalized solution. Software Defined Network (SDN) is the answer to provide a centralized local solution for real time security updates, remotely. SDN enables the administrators to configure network resources very quickly and adjust network-wide traffic flow to meet changing needs dynamically [2]. Alongside SDN, Network Function Virtualization (NFV) was developed as well. Even though both are cloud based computational solution, they are two different application with different goals.

SDN implements a network control plane that has a global view of the network and decision making capabilities in a logical centralized and fully-programmable software platform, by decoupling it from the network forwarding devices [1]. SDN allows a centralized controller that oversees all the other devices working in that network. Programmability of the network devices through a remotely controlled system grants the network flexibility and dynamic solution implementation, in terms of network and security solutions. For instance, network control solutions can now be implemented between two servers oceans apart, through SDN. This also reduces the added cost of devices, professional expertise and other financial and time expenses associated with the manual deployment of the system. SDN deployment also enables scalable solutions to be implemented according to the need of the user and operator. SDN is a network control entity that provides software solutions to be implemented remotely. As the benefits of network visibility and network device programmability are discussed, the question could be asked as to who exactly will benefit. Will it be the network operator or will it, in fact, be the network intruder? [4]

Security solutions implemented in a network must counter different types of threats. One of the primary types of threats that internet has always faced is Denial of Service (DOS) attack. DOS and Distributed Denial of Service (DDOS) are some of the oldest, crude yet effective types of attacks. DDoS attacks can occur when multiple compromised systems flood the bandwidth to a targeted system with spoofed source IP addresses and its purpose is to make the network and application resources unavailable to the genuine users for a certain period of time [5]. The attack prohibits legitimate users and tenants the access to the online service and requires very little technical expertise for execution. One of many proposed solutions is a high level programable output (HLPO) to filter such attacks. Research cited in [4] suggests the use of strong programming language to prevent DDOS, through Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) by harvesting from existing systems. Traditional network compatibility with SDN solution allows for importation of logic and security policies. Following a recent trend, traditional network security appliances are migrating towards programmability concepts, and Software-Defined Networking (SDN) enables an easier network and service management, resulting an interesting means to enforce security policies [3]. HLPO integrated with SDN service platform provides Platform-as-a-service solution (PaaS) towards an SDN network. Current state of technology demands security solutions to be adaptive and flexible to counter evolving DOS & DDOS attacks. This gap was filled by incorporating Cognitive Algorithm (CA) deployment via HLPO to constantly update the filtration and security measures.

CA is a self-adapting algorithm by taking output as a source of input for constant betterment of system solution. Cognitive algorithms can be of various kinds (e.g. inductive, deductive, incremental, non-incremental), they can perform learning with or without a teacher, with attention to solve cognitive tasks of various kinds [6]. CA formulates the basis of Artificial Intelligence (AI) where multiple nodes (sources) provide the output for constant evaluation and system development towards a more refined and efficient output. CA and AI are used primarily to enhance system efficiency and to develop more economically feasible industrial, and academic methods. However, recently their relevance towards cyber defense have been highlighted many times. SDN can be integrated with CA to detect, isolate and block a DOS or DDOS attack, as proposed in this thesis. The work in [6] discusses multiple uses such as data mining and pattern evaluation which is the key for developing CA in this thesis. HLPO is implemented remotely to provide ease of execution to the operator while maintaining authority over customizable solutions through SDN. This thesis suggests security in an SDN environment through CA, implemented via HLPO to counter DOS and DDOS. Simulation, graphical output and statistical data analysis using Mininet, Wireshark and OpenDayLight (ODL) [7] respectively, provides a comprehensive insight into security measure against DOS attack.

1.1.1. Problem Statement

In the above discussed PaaS, complete network control resides with SDN Controller. All the updates to network devices are pushed through the SDN controller. The network, more than often consists of multiple subnetworks, with variation of tenants. Tenants depending upon the nature of their network utilization, generate different amounts of traffic. Large industrial firms, corporate organizations and home-based users could all reside in the same network. OpenFlow Switch is a

virtualized (software governed) switch in Cloud based and SDN platform. It utilizes openflow protocols to communicate with controller to receive and transmit data. The OpenFlow switch faces two main challenges which are: i) limited number of entrees that flow buffer supports and ii) limited number of flow tables [10]. Operators have to cater for the need of variation of tenants, with room for increase in tenants in a SDN network. Traffic analysis for such networks becomes extremely difficult for operators, and filtration, more so. With growing traffic analysis of every data packet demands higher resources which makes the overall system costly. Distinguishing large legitimate traffic in a cluster of users becomes tedious and requires more investment and professionals like network analysts.

The algorithm itself is a centralized tool for performing traffic analysis calculation (TAC) against DOS attack, across a spectrum of users. Research in [5] suggested IPS integrated with SDN & firewall by dropping packets, resting connection and logging information. So, for the entire session that IP is leased to a user, the SDN Routing Table (RT) can be constantly updated to report, prevent and isolate a flooding attack. Network programmability via SDN also ensures timely constant update of the CA for growing traffic demand of a user.

Service providers also have to facilitate precise Network Access Control (NAC) solutions. Majority of DOS attacks are used as tools for probing networks for vulnerabilities, weak points and data acquisition. In general, the DOS attacks are usually combined with other attacks like MAC/IP spoofing [8]. Furthermore, DOS or any flooding attack is usually initiated using session establishment methods. In Open System Interconnection (OSI), sessions are established at multiple layers using Internet Control Message Protocol (ICMP), Packet Internet Groper (PING), 3-Way Handshake using Transmission Control Protocol Synchronization (TCP-SYN) bit and more. Research in [5] is directed towards defense against ICMP & SYN flood attacks. Majority of scanning tools employed by hackers use these methods scanning and spoofing of a network. This presents operators with a serious challenge of securing NAC with a preemptive method against DOS, as all the session establishment tools are used for legitimate traffic transfer as well. For instance [9] points out that Domain Name Service (DNS) based DDoS amplification attacks cannot be stopped easily by traditional signature based detection mechanisms because the attack packets contain authentic data, and signature based detection systems look for specific attack-byte patterns. Separating an ICMP-DOS attack from an authentic ICMP connection request becomes more complicated.

The dynamic nature of CA allows the traffic to be monitored on the go while implementing the security solution. HLPO through CA presents a solution by analyzing the number of session requests over a fixed timeframe. CA under the umbrella of SDN, allows operators to provide scalability to SDN network and its tenants for managing larger session request traffic (SRT). Filtration of SRT, grants higher legal NAC by blocking DOS/DDOS, without compromising the authentic traffic flow. Research in [9] utilizes Lyapunov exponent to measure the complexity of a flow of packets, and classifies the traffic as either normal or anomalous, based on the magnitude of the computed exponent.

It is a well established fact that 70% and above of threats to an organization's network and network-based infrastructure originate from inside [11]. Tenants hosted in the same network may also be hackers. An in house malicious source has always been a threat to majority of the operators

and presents a unique challenge. Most security solutions are designed towards filtration of incoming packets into a network. Also, the risk of data breach from within the network is far graver as compared to a foreign source. Pre-authenticated user inside the network makes flooding, spoofing and other such attacks easier and more powerful. A number of coordinated and infected PCs can launch DOS or DDOS attacks [11]. Operator has to isolate each individual's data from one another as well.
A centralized controller grants administrator access to monitor all traffic within a network. South bound and North bound Traffic represented by SBT and NBT respectively is the data flow inwards and outwards of switch. The purpose is to distinguish internal and external data overflow, the type mimicked in a DDOS attack. NAC will be performed for north bound traffic (NBT a.k.a outgoing) to prevent and isolate a SDN network tenant, from becoming a DOS source. This ensures NAC of administrator while preventing an attack from generating from within the network.

1.2. Aims and Objectives

The aim of this thesis is to develop a security measure against DDOS attack in a SDN controller using CA deployment. This method introduces CA through HLPO to ensure operator-user specific solution to cater for both party needs. It does so by identifying illegal growing traffic as a flooding attack, securing the network by isolating the IP address and then blocking the malicious source from network tenants. The solution also suggests adaptability in security measures for flexibility towards ever changing traffic demand of the tenants. It not only acts as filter for south bound traffic (SBT a.k.a incoming) traffic but for NBT to prevent a network from becoming a DOS source. This could be made possible by following objectives:

- To evaluate various challenges associated with DOS and DDOS attack in SDN network and existing related solutions.
- To propose and design a CA using HLPO to regulate traffic in the SDN data plane and flow of legitimate traffic in and out of the network.
- To design and calculate the SRT ranges for maximum threshold, malicious source isolation and maximum efficiency.
- By making algorithm unbiased and flexible towards multiple tenant traffic capacity while keeping the traffic transparent to the administrator.
- To implement, evaluate and compare the proposed security solution in SDN network.

1.3. Scope

The proposed architecture is for SDN based networks that are prone to high level DOS or DDOS attack. This configuration allows operators to closely monitor traffic while providing real time updates without the involvement of a middle party. The SDN platform overcomes proprietary only solution implementation, making the network more agile, robust, and remotely programmable. This is particularly well for operators with IOT (Internet of Things) devices, which generate large amount of traffic and requires round the clock NAC and DOS security. Individual CA flexibility

for each tenant also proposes localized security zones within the network to maximize network efficiency for each host. It gives room for NBT-IP monitoring for tracking and isolation purposes, which can provide supporting information for countermeasures towards an attacking source. All of this attributes to an improved memory capacity, which is a primary target for any flooding attack (memory depletion). Cache memory containing routing, IP and network information in particular, is susceptible to this attack. This could translate into better and faster system management with increased output. The security solution highlights the following in particular:

- To confine access to legitimate traffic only and block the rest of the data packets (in SBT).
- To prevent flooding attack by identification, isolation and blockade of malicious IP source.
- To improve NAC by filtration of SRT.
- To prevent in-house deployment of a DOS source in the network.
- And to improve the scalability of the algorithm to adapt towards security changes.

1.4. Dissertation Outline

A brief yet precise insight into SDN controller and network function scalability has been made in this chapter. It further elaborated security problems associated with the DOS and DDOS attacks. Short introduction towards the existing scope of solution was made, along with aims & objectives.

Chapter 2 discusses in depth the details of SDN technology architecture and CA. Use of CA in SDN for security mitigation as well resource management of memory through Dynamic Memory Allocation (DMA) is reviewed. The proposed security solution of CA and SDN based security is reviewed for existing literature and proposed methodology. This chapter will also define the DDOS in general and its classifications. A statistical analysis of recorded flooding attacks and prevention measures proposed in RnD will be discussed. IOT devices and their association with flooding attacks would be briefly elaborated for better understanding. And it would be concluded on the proposed method of solution, as a results of literature review and its comparison with existing methods protocols.

Chapter 3 hosts the framework that is proposed and strengthened by literature review, for SDN Controller implementation of CA deployment. Focus of this chapter will be towards SDN controller environment while comparing CA and existing security protocols. Markov's Chain rule, which constitutes the basics for development of CA will be discussed thoroughly and proposed CA will be brought forward. Finally, the chapter will be concluded leaving room for future possibilities of SDN and CA integration platforms.

Chapter 4 is the proof of concept (PoC) and its implementation for CA in SDN environment. This chapter entails software and tools necessary for the design, simulation and output analysis of the proposed solution. Major components linked with PoC are Mininet, POX SDN Controller, Wireshark and OpenDayLight (ODL) [7]. Few other tools for result enhancement will also be mentioned here. Architectural design of the network in ODL will also be demonstrated. The software associated with HLPO and its constructed code tests are performed to demonstrate its purpose.

Chapter 5 discusses the results obtained by implementing PoC in previous chapter to analyze security output efficiency. The results will be verified through a method of tri-verification (TVF), where SDN CA will be supported by comparing and analyzing with two independent sources, executed on different software platforms. More so, calculations are presented as graphical output for DOS strength across the network, efficiency spectrum of the algorithm and percentage evaluation gain and loss of the system.

Chapter 6 is the conclusion of the thesis by defining the findings and their possible advantages in research and industrial community. The future prospects of CA as security tool is discussed in detail, with integration towards remote network deployment via SDN.

1.5. Chapter Summary

Chapter 1 provides introduction for the technologies and methodologies used for the thesis. It discusses developing challenges in a SDN network environment directly relating to DOS and DDOS attacks. Aims and objectives specifically define methodologies involved in providing solutions for those issues. Scope covers the security measures in step by step explanation and tasks associated; and section 1.4 sheds a brief light on the various topics associated with upcoming chapters.

CHAPTER 2

LITERATURE REVIEW

2.1. Introduction

This chapter presents in detail the work done on Software Defined Network (SDN) and Cognitive Algorithm, and studies how the two can be integrated to provide protection against Denial of Service (DOS) attacks. Previous security solutions in SDN architecture are examined for strengths and weaknesses to provide Platform as a Service (PaaS) solutions. Existing research opting wavelet analysis and other non-algorithm-based solutions for DOS protection are brought forward. All of the reviews, including general SDN security as well as the algorithm mitigation are analyzed, and are used to formulate self-adapting, Cognitive Algorithm (CA). From this, the algorithm is extended to a SDN deployment to establish an Anti-DDOS platform by enabling remote updates and security changes. Brief introduction into DDOS and its targeted application platforms is supported with statistical attack history. This is to signify the importance of security solution implementation and existing research. Therefore, security solution is cognitive algorithm implemented in SDN application platform.

2.2. Software Defined Network (SDN)

2.2.1. Software Defined Network Architecture

Software-Defined Networking (SDN) allows for fast reactions to security threats by dynamically enforcing simple forwarding rules as counter-measures [3]. SDN decouples the control and data planes, wherein a logically centralized controller performs forwarding decisions on behalf of switches [12]. Centralized controller allows for network alterations through software or platforms to provide Software as a Service (SaaS) or Platform as a Service (PaaS) respectively. Services through SDN are offered through three different planes which are: application, control and data (Infrastructure) planes as shown in figure 2.1.

The application layer comprises of software-based network applications that manage data and provide logic to the controller. Updates, logical control and security solutions are enforced through software or network program at the application layer to relay decision to data planes. The data plane consists of network elements, each element contains a series of network resources and business processes, SDN resources are based on the underlying physical entity and capabilities supported and given [13]. Application Program Interfaces (APIs) use protocols enforced by users to deliver best flow routes and constantly update Flow Tables (FT). The control layer is the core of an SDN, which consists of controllers that take responsibility of managing traffic flow in a network, flow forwarding and taking decision on routing using programming [10]. The programming usually is a HLPO which is formulated and maintained in a software at application layer, to be executed at control or data plane layer. This is discussed in detail in chapter 3 and 4. Researchers can write their protocols freely on the control layer, so that the network can be creative

and implement the protocol of API OpenFlow, which is a powerful promoter of this idea [13]. SDN protocols provide more flexibility to the administrator as compared to a conventional routing network, as discussed in [14].

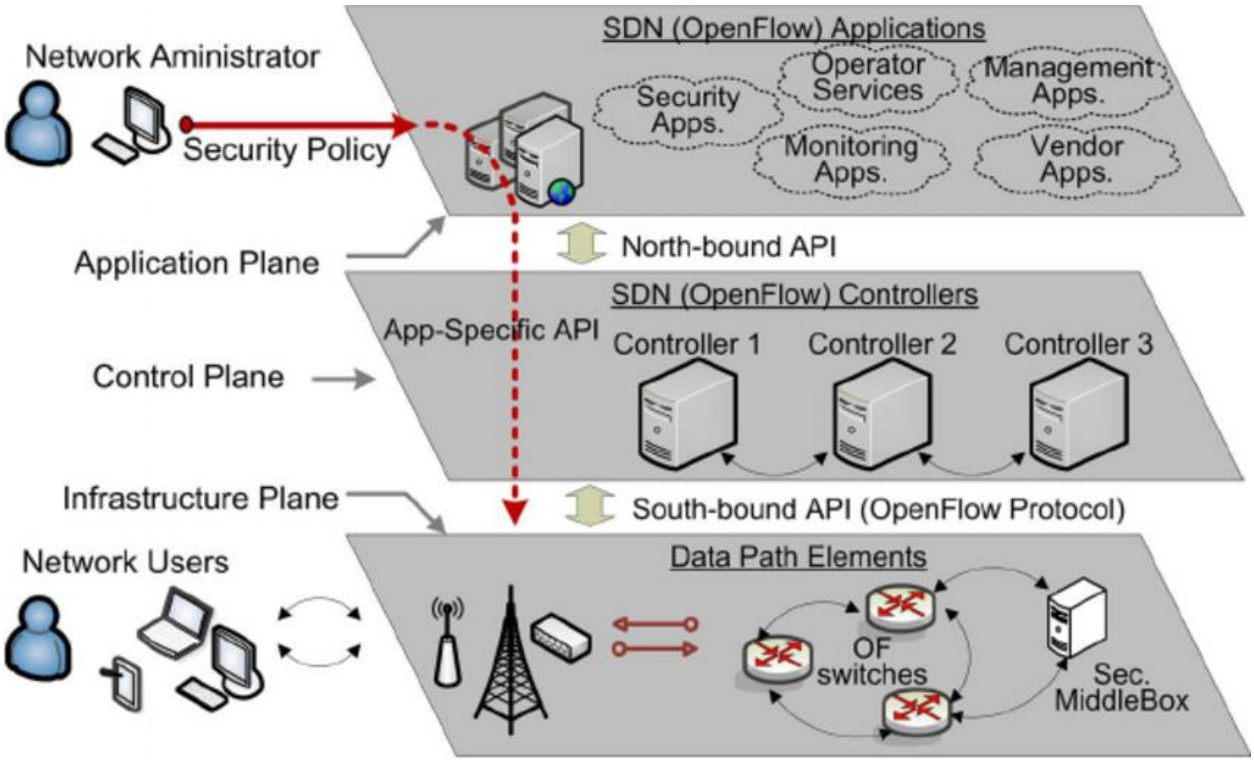

Figure 2.1 [28] SDN Architecture

Some of the common examples of such APIs are:

- Border Gateway Protocol (BGP) is a protocol used for exchanging routing information between gateway hosts in a network of autonomous systems [15].
- The Network Configuration Protocol (NETCONF) is an Internet Engineering Task Force (IETF) network management protocol. It provides an administrator or network engineer with a secure way to configure a firewall, switch, router, or other network device [15].
- Open vSwitch DataBase Management Protocol (OVSDB) to configure Open vSwitch.

This integration has proven more efficient in optimizing network solutions, traffic monitoring and customized security algorithms for Intrusion Preventions Systems (IPS). Software security platforms integrated with network devices such as switches, have provided more robust security options. Work in paper [17] presents one example of building a firewall over SDN, using the POX controller and Python Programming Language. The collaboration of a SDN Controller with programming languages for implementing IDS and IPS, opens room for advancements in security enforcement. Different software and programming security solutions can be modified for SDN requirements, removing compatibility and proprietary complications, in standard routing network [14]. Advanced and more complex security algorithms are being employed using software and networking tools at SDN application layer.

2.2.2. SDN Security Solutions

Security solutions using SDN have been used to counter multiple arrays of attacks from DDOS, Man in the Middle (MiM), IP Spoofing and more. This research is focused towards mitigation and security methodologies deployed towards flooding, and DDOS attacks. Work done by researchers in [3] proposes a novel approach of stateful monitoring by StateSec. It works by matching configurable traffic features (e.g., IP src/dst, port src/dst) without resorting to the controller and the solution is implemented at Open vSwitch (OVS). OVS is a SDN solution for on the go network deployment that is discussed in detail in 4.2.2 and chapter 3. It employs an entropy-based algorithm to detect variety of attacks such as DDOS and Port Scan. The solution suggested in this work uses entropy to detect abnormalities in data patterns and executes pre-installed reactive countermeasures. Computational delegation from controller to network devices such as switches is critical. This is suggested to relieve computational power at controller to carry higher operations.

Intrusion Preventions System (IPS) and Intrusion Detection System (IDS) are presented in [5] which perform filtration of the network packets. It uses Software Defined Cloud (SDC) to provide network traffic filtration by using IDS and IPS at a SDC firewall. Dynamically distributed firewalls combined with IPS and IDS, provide a secure NAC as demonstrated by conducting an ICMP DDOS attack of 10,000 packets per second. Research in [3,5] suggests functionality outsourcing from controller to switch which provides stronger SDN.

Rajat and Markku in [12] present a DDOS mitigation technique using SDN by configuring an optimal idle timeout value and flow aggregation. This work performs an immediate flow memory dump in event of a DDOS attack and the intelligent analysis of how state rules prevent overflow of a Flow table memory. Similar work in [17] suggests a simple and effective enforcement of a "Rate Limiter" to control packets being sent to controller. The rate limiter is implemented using sFlow which notifies mitigation software at SDN application layer for solution. It suggests how the use of SDN in an OVS offers flexibility, reliability and operational strength.

Recently, the SGuard solution has been popular amongst SDN security schemes. As described in [8], a security application on top of the NOX controller that mainly contains two modules: The access control module and the classification module. In support of the work in [3, 5], controller saturation alleviation is a key factor mentioned in this research. Lightweight controllers are better utilized for optimizing NAC, DDOS mitigation and security procedures in an affordable system. Source tracing is achieved through access control by logging information, in order to remap abnormal packets to their source. The solution provides a light and cheap answer to DDOS mitigating and source tracking; perfect for commercial security.

2.3. Dynamic Memory Limiter

2.3.1. Dynamic Memory Allocation

Flooding attacks are particularly dependent on the nature of intent of an attacker. The goal is to deny the user of access or to conceal the malicious packets for a graver purpose such as MAC or IP Spoofing. Regardless of the approach, cache memory (routing tables/flow tables) used for storing forwarding protocols is the main target.
Switches and networking devices have limited memory. For example, the HP 560zl OVS can store only 1500 flow rules before flooding completely [12]. The concept of a dynamic memory limiter is to impose a changing memory state, to accommodate data within a defined threshold. The dynamic nature is due to variations in DDOS attacks as most of them are hard to detect because either they appear very suddenly or slowly and consistently making greater complications [3]. Virtualization and remote access to the network has allowed for better resource management of the network. An efficient memory system has been discussed in [18] operating on a global allocation technique along with memory releasing and memory growing for satisfying the memory demand during run time. This proposed model is for a cloud virtualization platform and delivers result by information collection, monitoring, statistical analysis and optimal memory allocation. Cloud environments are supported by Openstack and Devstack support platforms that deploy the system in a Virtual Tenant Network (VTN). Calculations in [18] are promising and delivered through designing a Memory Release Algorithm (MRA). Resource management and memory allocation are vital in a SDN prone to DDOS. Flow table memory control decreases the risk of sudden resource depletion and acts as an early warning for mitigation actions to comply accordingly. Software memory allocation in real life applications such as VLC-Player, My SQL and few more is demonstrated in [19]. Time limitation is a factor towards memory allocation which is briefly discussed in this paper. Many virtualization systems have a scaling function with which memory allocation size for a virtual machine can be dynamically changed without restarting the virtual machine [20]. Cache Hit Ratio is buffer size management in virtual machine which is used in [20] to demonstrate strong results towards memory management in network virtualization as shown in figure 2.2

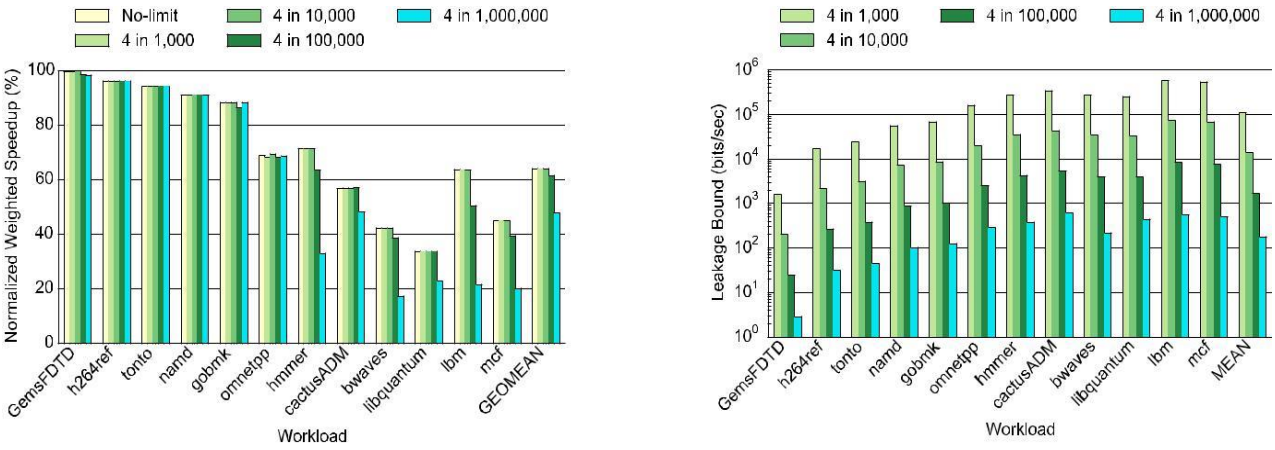

Figure 2.2 [20] Memory performance with limit on ER Violation

Memory management is constructed using an algorithm to accommodate varying memory needs. Suggested algorithms in [18] are amongst the many designed for network efficiency and are customized to changing topology and requirements. A network administrator can define the parameters for a developing SDN, improving its scalability and performance for up to date security measures for Flow table management against flooding attacks.

2.3.2. Dynamic Memory Limiter Defense

Dynamic memory limiter methods have been proposed to keep uniform memory allocation per host or device to avoid centralized flooding of a network. This is achieved by varying memory assignment per user within the confined limits of the networking system. Timing channel attacks, often used by flooding attacks to gain NAC, have been discussed in [20]. Secure dynamic memory allocation through a secure memory controller (SecMC) is suggested in this paper. It brilliantly utilizes Dynamic Random-Access Memory (DRAM) to provide a dynamic storage solution based on an error violation algorithm. The algorithm processes every request (ER) for violation of memory allocation. ER is a heavyweight solution but effective in response to growing data allocation towards specific memory set. However, this algorithm sets a tradeoff where security is compromised for performance and the system is susceptible to information leakage. Processing ER causes memory overload during high traffic and creates security gaps during packet drop for malware and network probing.

Flow tables and memory units in networking devices work parallel with IPS, IDS and such firewalls to maintain system operation even when under attack. In paper [21], critical amount guaranteed memory allocation (CAGMA) in virtual machines (VMs) is suggested for this purpose. The maximum memory assigned to a VM is calculated using formula shown in (1) below:

$$\text{CMA}\ (vmi) = max_{\tau_{i,j} \in T_i}\{MemU(\tau_{i,j})\} \quad (1)$$

Critical Amount Memory Allocation [21]

The rationale behinds our proposed CAGMA is to guarantee the available memory of each VM never less than a critical memory amount (CMA), denoted as CMA(vmi), for each vmi ∈ VM. VM is each virtual machine and "I" is the count or number of machines. The allocated memory amount and the available memory amount of a vmi are defined byALM(vmi) and AVM(vmi), respectively. The memory utilization of a vmi is defined by MU(vmi) which can be calculated by MU(vmi)= ALM(vmi)−AVM(vmi). We also consider a set ALM(vmi) of tasks Ti = {τi,1,τi,2, ⋯ ,τi,mi} which being executed in vmi. The memory usage of a task τi,j (i.e., the required memory amount of τi,j) is defined by MemU(τi,j).
Work of [21] promotes the idea of critical memory utilization and limitation for a network attack. The algorithm and CAGMA designed here is for memory efficiency and resource management. However, a security algorithm integrated towards the existing set up shown in (1) could provide a limiter for DOS prevention and securing the network. This provides a fruitful result towards an affordable, reliable and redundant system for industrial security functions.

2.4. Cognitive Algorithm

2.4.1. Design and Development

As described in [6] and the previous chapter, Cognitive Algorithm (CA) is a design and implementation, of self-assessing and modifying solution towards output efficiency. Elegant design of CA demands complex solutions to be handled in real time with improved answer for the future. This means that solution has to be advance enough to surpass average algorithm computation while draining minimum processing resources. Object classification literature shows that conventional algorithms are evolving towards cognitive informatics and cognitive computing machines [12]. Work in [25] points towards the development of cognitive decision algorithms for predictability models for lottery and numerous financial sector outputs. Priority Heuristic (PH) is utilized for analyzing future statistical prediction using past data analysis. Mathematical models for performing straightforward decisions are evolving towards pattern recognition, forecast modeling and complex solutions.

Machova and Paralic in their work in [6] conclude the use of CAs towards basic fundamentals, to either improve quality of system of knowledge or to yield knowledge of a system to gain profit or competitive edge. The quality of the learning algorithm design depends on the suitable selection of one or more basic principles [6]. Each algorithm design must be altered based on the functionality requirements and variable constraints of the system.

This thesis suggests utilization of Markov's Chain Rule for the development of a CA for ICMP-DDOS count check. The mathematical model for system design analysis of DDOS will be formulated based on Markov's Chain Rule analysis which will be discussed in next chapter. In security, Chaos Theory suggested in work of [9], [22], [23] and [24] is a very fundamental and prudent methodology towards DDOS mitigation. Exponential variable algorithms such as "Lyapunov Exponent" are developed incorporating operator-user needs based SDN security. The outcome of exponential, or other such algorithms designs, is to support entropy or anomaly detection solutions to detect flooding attack abnormalities in a SDN. Scalable and self-assessing solutions offer reliable and affordable performance as demonstrated by these papers and provide less computational power with reduced manpower for the system. Flooding attack mitigation by using CAs is discussed in next topic.

2.4.2. CAs past solutions against DDOS

Basic security measures formulated on simple algorithm designs are now inadequate. Thus, due to the mutating nature of cyber security threats, the intrusion detection database of signatures and anomalous behavior patterns of cyber threats needs to be autonomously and adaptively updated in a manner that requires intelligence beyond that provided by conventional algorithms [9]. CAs do so by dynamically changing the outcome to meet adaptive network requirements against varying attack strengths.
Chaos theory is a popular idea used for anomalous behavior recognition using entropy-based tools. As explained in [9], Chaos theory is a branch of mathematics that deals with the study of the stability of dynamic systems and is distinguished from nonlinear stochastic processes where variables are random in nature. Research methods in [9] use Lyapunov Exponent to calculate

anomalous network traffic which provided result of 66% detection. Lyapunov exponent fundamentals are shown below in (2) and (3).

$$\lambda \approx \frac{1}{t} \ln \frac{|\Delta x(t)|}{|\Delta x(0)|} \qquad (2)$$

$$\lambda \approx \underset{\Delta x(0) \to 0}{Limit} \frac{|\Delta x(t)|}{|\Delta x(0)|} = J_{i,j}\left(x(0)\right) \qquad (3)$$

Lyapunov Exponent [9]

(2) represents nonlinear dynamic system of Lyapunov exponent where (3) is matrix form of the equation. λ represents the divergence of the of the signature-based detection of the system. It evaluates and determine an attack based on time variation of data packets represented by $\Delta x(t)$. λ represents limit solution where packets are assessed on attack patterns. Entire spectrum of exponents cannot be evaluated [12] which leaves room for MiM and Eavesdropping threats. Despite the high detection output the algorithm designed in this paper for DNS flooding attacks required a lot of memory as attack files had to be broken down and analyzed per packet. Set-ups such as this require off line data processing units working in support of online devices and makes real time DDOS filtration cumbersome for system.

Network Anomaly Prediction Algorithm (NAPA) solution presented in [22] proposes divergence-based anomaly detection based on Chaos Theory. The anomaly is based on the Lemda (λ) variable which represents orbital divergence from the stable traffic. Under Chaos theory umbrella, [23] proposes "Time Range Analysis" using "Network Anomaly Detection Algorithm" (NADA) for mitigating DDOS attacks. This paper has utilized autoregressive model – time series analysis for cumulating averaging traffic variation. Both papers are supported by anomaly detection using variance algorithms. However, the anomalous detection in [22, 23] is less than 66% and is not designed for instantaneous traffic processing. System analysis with real time traffic would generate lag due to processing needs and would require expensive set-ups to keep with growing flooding traffic. Lag in sudden growing DDOS traffic in a network attack is inevitable, even for most advance systems. Therefore, this thesis recommends dynamic memory expansion of the system, allowing IPS and IDS to perform calculations while maintaining operation across network.

An entropy solution in [24] is customized using Lyapunov exponent to detect DDOS attack. Exponent detection algorithm and its separation is performed using equation (6), where (4) and (5) represent entropy limits which calculate the variation in traffic. It calculates source and destination IP entropy in time to predict traffic anomaly.

$$H_q = \frac{1 - \sum_{i=1}^{N} p_i^q}{q-1} \qquad (4)$$

$$H_q^{max} = \frac{1 - N^{1-q}}{q-1} \qquad (5)$$

General and Maximum Entropy Calculation [24]

$$\lambda_k \approx \frac{1}{t_k} \ln \frac{\widehat{H_s}(k)}{\widehat{H_d}(k)} \qquad (6)$$

Trajectory Calculations via Exponent Separation [24]

H_s for source IPs, and H_d for destination IPs. In order to generate dynamical equations, we herein use AR model to pre-process these sequences. And lastly t is notation for time. H_q is a Tsallis entropy where q is entropic parameter [24]. It detects traffic anomaly by dividing the probability factor p_i^q by total packets N, providing the threshold limits of the entropy-based system. The solution in [24] is lightweight solution which is very basic towards detection of DDOS. Far more advance detection methods exist with greater computational output and efficiency. Solutions suggested in [22, 23] and [24] allow anomaly detection by traffic analysis. These solutions are not flexible in SDN or any networking environment where hosts in subnetworks are evolving. Changing traffic volume generating legitimate traffic could be flagged as a potential anomaly. This thesis suggests adaptive algorithm towards ICMP flood, in an SDN environment where traffic changes and algorithms adapt, compensating legitimate session requests towards a host.

2.5. CA based solution for SDN

2.5.1. SDN Security Architecture and Challenges

The main features supported by cloud networks and SDN are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service [5]. SDN control and data plane separation is the biggest leap towards the security enforcement as mentioned in section 2.2. Currently most research work focuses on exploiting SDN to improve network security by using monitoring systems by inserting security policies to dynamically detect and mitigate suspicious traffic during live network operations [2]. The major approach towards securing a network is by performing security mitigations at the switch, governed by the controller. Work done in [2], [4], [26] and [27] stress on security checks and balances to be performed through switch under the centralized controller network. This alleviates controller to perform higher functions and prevents it from a potential bottleneck by flowing raw data through it. Instead the raw data is fed to switch for filtration through already provided flow controls via controller, and only flagged data (as error) is sent for higher processing such as Deep Packet Inspection (DPI).

The security architecture in [2] defines the threats towards SDN platform, which are common to traditional networking but their profile and level changes with SDN architecture. The threat level

for certain spoofing and eavesdropping attacks increases due to the remote nature of SDN. For example, the controller is under the constant threat of being hijacked, to redefine the entire network to hacker's will, initiating stealth attacks such as eavesdropping. Work in this paper recommends packet data scan detection (PDSD) to locate worm and Trojan like threats. It solves the issue for secure communication link between the data plane and the controller using southbound API. However, PDSD is a cumbersome process with the tradeoff between time and higher processing. System cost and higher processing requirements make this an expensive set up for commercial use.

A systematic approach using SDN specific features (programmability, dynamic flow etc.) can yield higher security mitigation against threats as mentioned in [26]. It provides a holistic hierarchal approach where a switch acting as an actuator senses and provides information performing the grunt work for controller, which in turn assesses the threats and responds with security mitigation. From the detection of a threat towards the mitigation act generated by the controller, a time lag is often observed which can eventually overwhelm the system. This particularly is a problem towards high volume flooding attacks. However, algorithms designed towards countering DDOS have been hinted in [26] and their importance is highlighted. Researchers in [27] proposed an event-based security measure using "Bro" for security enforcement. The architect for the solution is shown in figure 2.3 below.

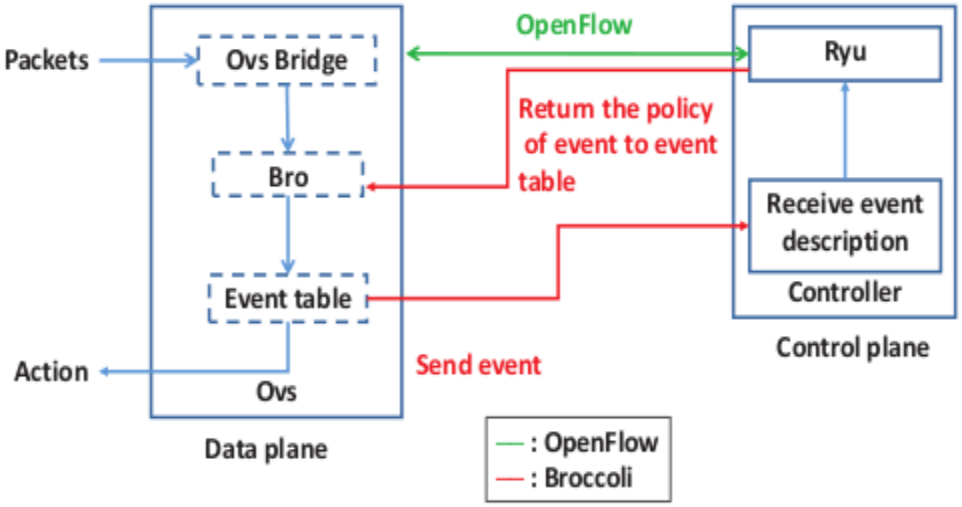

Figure 2.3 Bro Security System [27]

An application or control defined network based on algorithm design is implemented through "Bro" to make controller as light as possible. As indicated by figure 2.6, this is made possible by performing security checks for data packets at data plane on open vSwitch. Thus, the overall central processing unit (CPU) saw a significant drop in its workload, making it available to tackle large volume attacks like DDOS.

Outsourcing security procedures to data plane devices like vSwitches decreases the workload for a controller, making it more capable towards handling a DDOS. This thesis entails this idea and

executes it to measure the response of this under a CA and dynamic memory allocation to minimize sudden network crashes caused by DDOS attacks.

2.5.2. Benefits of Security through SDN

SDN security solutions have provided promising results for security implementations for networks. Centralized network control allows for new features to be added. Dynamic access control functions can be implemented which was previously done by independent middleboxes (e.g., firewalls) to achieve in-line access control [26]. Work in [2], [4] and [26] highlights some of the basic and vital advantages of SDN over a traditional routing network. Aside from the reduced cost of the overall system, it offers easily applicable, reliable security architecture. Some of the important features are detailed below:

- Dynamic Flow Control: allows for systematic and dynamic flow of control throughout the network as mentioned in [26].

- Centralized Control: Centralized administrative control over network performance and security, for various devices and hosts.

- Network Programmability: Network application programmability, data plane reconfiguration and multiple vendor operability provides customized solution tailored for customer satisfaction.

- Open Source Platform: Programmability integrated with open sourcing opens the door towards customized solution removing hinderance of proprietary device management. This offers a customizable, centralized solutions to be accepted universally in a single SDN system.

- Network Security: Controlling network flows dynamically provides many new possibilities in network security functions [26]. Network security is increased using diverse granularity and better NAC is enforced by centralized monitoring of the system.

- Compatibility: programmable centralized network offers greater compatibility across a variety of vendors which are used according to operator's preference.

- Cost Efficient: Overall cost of the system is cheaper due to lesser "purpose specific networking devices [26] and reduced manpower for network checks and maintenance.

Cognitive Algorithm uses programmability to execute NAC against DDOS attacks at the data plane or controller depending upon the performance preference of operator. This thesis suggests using a CA as a filter towards growing DDOS attack at network devices to minimize Flow table or cache memory overflow.

2.6. Chapter Summary

This chapter has discussed the basic architecture of SDN and various security solutions associated with it in depth. Existing security infrastructure of SDN for threat analysis and IPS are studied particularly towards flooding attacks. SGuard, IPS and IDS are analyzed in existing solutions for DDOS attacks. A dynamic memory limiter is suggested as a valuable asset for memory overflow limitation. In depth analysis of defense against DDOS through CAs is carried out. Chaos Theory has served as a remarkable tool for mitigating DDOS attacks. This theory is based on mathematical algorithms based on exponential variables such as Lyapunov exponent. Lastly, in section 2.5 we summarize how CA – based solutions through SDN are used in the past as effective methods in DDOS defense.

CHAPTER 3

SDN BASED COGNITIVE ALGORITHM DEPLOYMENT

3.1. Introduction

Software Defined Networking (SDN) has made easy network deployment possible, with centralized control through a controller. However, some of the new features such as remote and centralized data control exposes the network to traditional network threats. Moreover, there are some new security challenges such as bypassing predefined mandatory policies by overwriting flow entries and data eavesdropping by inserting fraudulent flow entries [2]. Cognitive Algorithm (CA) grants better security to the data and control plane to provide reliable, scalable and robust security solutions. Primary security of the data plane through CA, also provides a set up for better data and memory resource management. This provides a single Software as a Service (SaaS) platform which is a multi-purpose tool for administrative and security management. Through the centralized controller we can easily implement a network load balancing function that is not readily and cheaply solved with existing techniques [26]. This thesis recommends the use of CA to establish a self-adjusting count check for an ICMP-DDOS attack. The proposed CA count check is deployed in parallel with a dynamic memory limiter to support SDN functionality in an event of a DDOS attack. The CA solution proposed here, needs to be expanded and executed in an SDN framework to demonstrate the Network Access Control (NAC).

SDN offers programmability and reliance in the system where CA can be deployed for improved network services. SDN thrives on the concept of separated control and data plane functionality and programmability in the network [4]. Application and security controls can be assembled in a single plane to offer Software as a Service (SaaS) for the entire network. Centralized control of the network through a single controller aids in SaaS protocol implementation on multiple devices with uniformity and total compliance. The CA framework presented in this chapter illustrates how SDN and CA can be integrated to form a Line of Defense (LOD) for the entire network and how it can contribute towards memory management to counter flooding attacks. The framework discussed would formulate the foundation for the proposed solution of CA-SDN against an ICMP-DDOS attack.

The remainder of the chapter is distributed as follow: Section 3.2 discusses the proposed security solution and the algorithm used for its deployment, particularly Markov's Chain Rule. Section 3.3 describes the details of data plane and how the security solution deployment at the infrastructural layer reduces workload and potential bottleneck at controller. A brief introduction about the SDN Controller is made in section 3.4 to highlight its functions. The development of security solutions in the application layer is discussed in section 3.5 in relation to the past solutions. Finally, the chapter is summarized in section 3.6.

3.2. Cognitive Algorithm based SDN Framework

As mentioned in previous chapter CA is self-assessing logical solution, which works for improvement of overall functionality of the purpose-built system. The algorithm is designed to manage specific functionality of the system and challenges accompanied with it. Security analysts and administrators can utilize its flexibility and recurrence analysis to their advantage to save money, time and hardware-based resources. Security equipment manufacturers have already started to develop and sell commercial products based on SDN concepts [3]. Instead of purpose built IPS and IDS devices, CA can be deployed at data plane devices through the centralized controller. The majority of threat mitigation is handled by CA based in data plane, which grants flexibility to the administrator to perform higher level security tasks. The integration of CA in SDN for this thesis is illustrated in figure 3.1

The figure shows multiple layers of the SDN network set up much like figure 2.1 in previous chapter. Multiple Open Virtual Switches (OVS) can be deployed in a cluster that are connected to each other forming data connectivity in data plane. This allows operators to rapidly send security policies through north bound API to the controller, which can instantaneously deploy these policies at OVS. Rapid deployment of OVS also reduces the cost and hardware which in a conventional network would have to be physically deployed into the network. The OVS is an open flow switch which resides in data plane layer connected to VMs, acquiring instructions from application layer [29]. Communication between application-control plane and control-data plane through north and south bound APIs respectively, is what controls the OVS as shown in figure 3.1

The proposed network security solution in this thesis is in fact a Network Function (NF) which is being implemented at data plane level. After being defined at application layer as a NF, the CA solution is executed in OVS, the functionality of which resides in application plane and is executed in data plane. The SDN controller in early solutions was used to handle the security enforcements which developed a lot of traffic towards the controller from north and south bound APIs. This particularly was a challenging issue for DDOS mitigation as the network could not cope with growing network traffic. Besides a potential flooding attack, a growing SDN with more hosts experienced delays between issuing of control flow protocols and implementing them. Another option was to delegate as much computation as possible to switches without compromising their performance, letting the controller being only in charge of mitigation [3]. VMs and OVS allow this feature to be deployed with ease, which is now quickly becoming a norm, delegating power to switches. This thesis deploys CA supporting this norm in order to make controller work load as light as possible. Dynamic memory limiter and count check based on CA can sustain a flooding attack at data plane while mitigation policies are enforced and updated LOD is implemented. Compromise of the controller is of grave concern as discussed in [3], [12] and [30] as it can isolate data plane from rest of the network, making security mitigations impossible to implement among other network functions. However, redundancy and failsafe to the system can be introduced by adding a backup controller, with same plane view of the network as the primary as shown in figure 3.1. It offers better network management at the cost of more resources. The next topics discuss the deployment and effects of CA in each individual plane of SDN.

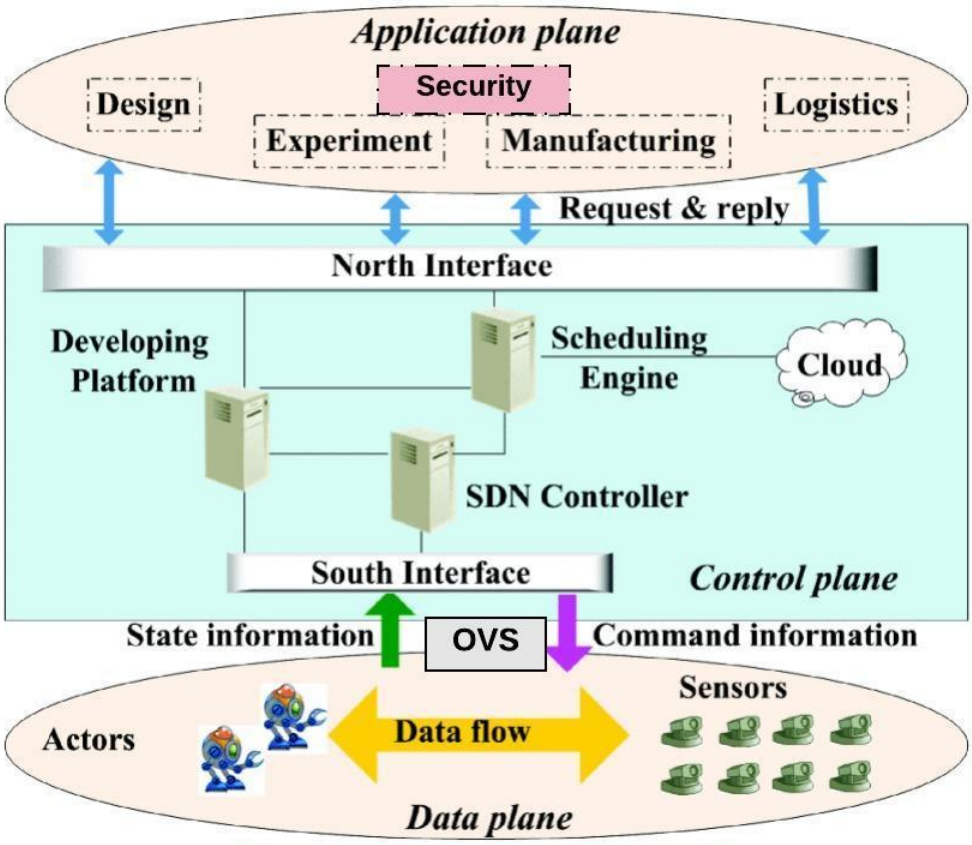

Figure 3.1 SDN Plane View Architecture [31]

3.2.1. Markov's Chain Rule

The general concept of cognitive and higher power algorithms is formulated on a cyclic loop feedback system. This thesis utilizes the Discrete-Markov's Chain Rule to develop and deploy the cognitive algorithm through an SDN controller. This is a memory dependent algorithm which resides on the outcome of past calculations, unlike Bernoulli and Poisson equations which are stateless [34]. Markov's Chain Rule solution implied for SDN-CA model in this thesis is represented by figure 3.2. A state of constant update and monitoring is demonstrated through the figure where existing flow rules are assessed; represented by "A11" and "B12". OVS performs checks through flow tables to locate existing solutions in A11 and B12 which are represented by state of mitigation actions A12 and B11 respectively. An error or prompt update request is sent to the ODL Controller should "A11 and B12" are unable to locate a flow rule for the scenario at hand. Using OVSDB (OVS – Database Management), southbound API constantly updates the controller for requesting new flow rules or updating the blacklist for the entire network. The entire network is monitored by the administrator at the application layer which has direct access over the ODL controller. The controller sends updated blacklist to OVSs connected to the network and requests new flow rule policies from the administrator, which are represented by control actions Cnt 11 and Cnt12 respectively. This promotes a centralized command and access control over the SDN. The

loopback system of Markov's Chain provides feedback which is used in this thesis to detect sudden ICMP growth in network. This provides an early DDOS detection and executes mitigation strategies.

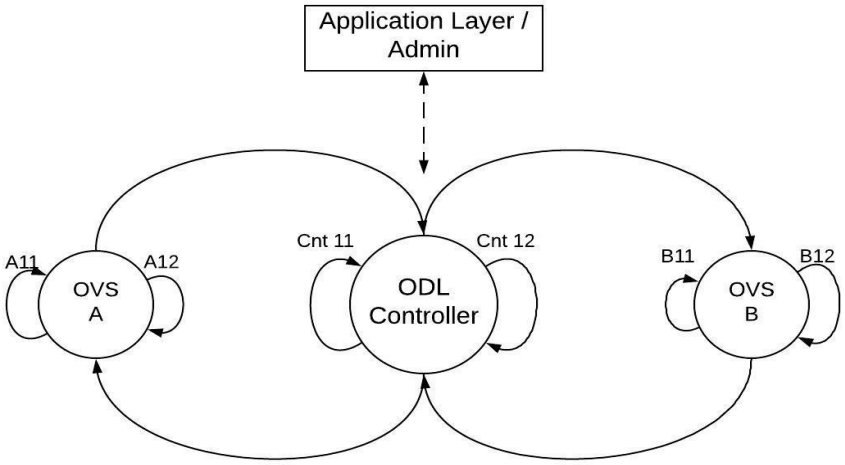

Figure 3.2 Markov's Chain Rule for SDN-CA Deployment

Figure 3.2 provides a system specific view of memory state function [34] algorithm, which in this thesis is Markov's Chain Rule. Equation (4) and (5) cited in [24] are an entropy-based solution, which are one of many ways to detect and address growing traffic in a network. The generalized concept behind the working of such algorithms is shown in figure 3.3 below. Figure 3.3 holds true for traditional network and SDN where anomaly detection using Lyapunov exponent [9], [24] could be achieved at the input. However, SDN allows for remote and easier deployment of such solutions as compared to a traditional network.

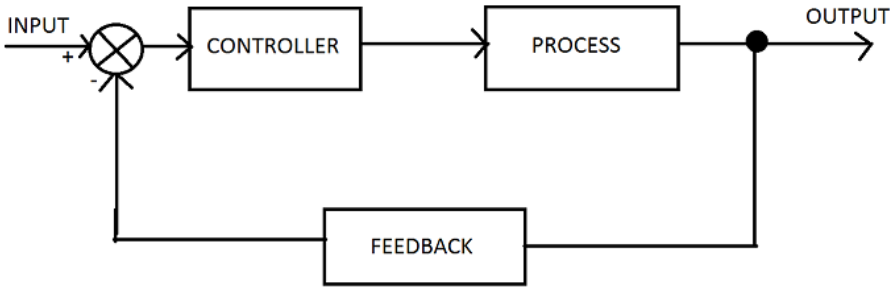

Figure 3.3 Generalized Algorithm Deployment

3.3. Data Plane in CA-SDN Security

OVSs and physical interfaces in a cluster form the data plane layer of SDN. VMs resources are managed at application plane by the administrator, controlled by software entities called

Hypervisors. Open vSwitch leverages OpenFlow and the Open Virtual Switch Database (OVSDB) management protocols, which means it can operate both as a soft switch running within the hypervisor, and as the control stack for switching [29]. Memory and resource management is controlled through the hypervisor which resides over VM. On practical bases VM and hypervisor are referred to as a single logical entity which is configured by administrator for rapid network deployment. On demand of the user, the SDN operator leases the VM resources which are utilized by host for the session withinthe network. The functionality of OVS is of huge importance as it is the first entry point into the data plane and bifurcates the controller and network devices as shown in figure 3.1. The data plane supports network devices through design implementations shown in figure 3.4. The hypervisor hosts the OVS and VM which communicate via Virtual Interface (vIF) and Virtual Network Interface Controller (VNIC) as described in figure 3.4. Hypervisors, VMs and OVSs are integrated to form a virtual network which is connected to physical switch through vIF created in OVS.

OVS provides a universal logical solution deployment as it is not bound by proprietary network device limitations, which are seen in vendor only devices such as CISCO. As mentioned before, this provides customizable security solutions to be implemented at data level and OVS can be updated at the operator's discretion with the latest security methodologies of industry and academia. This thesis exploits that very concept to introduce a CA-based solution against flooding attacks at the OVS. The basic principle of OVS through sflow or other such protocol dictates that all NBT and SBT to flow through the switch. Dynamic traffic flow management at the OVS allows not only for control of incoming traffic but can prohibit an internal threat from one of the hosts as stated in [11]. Internal network security offers prevention measures in isolating a network within a network. This is shown through figure 3.5 where CA security measures can prevent flooding attacks within the network.

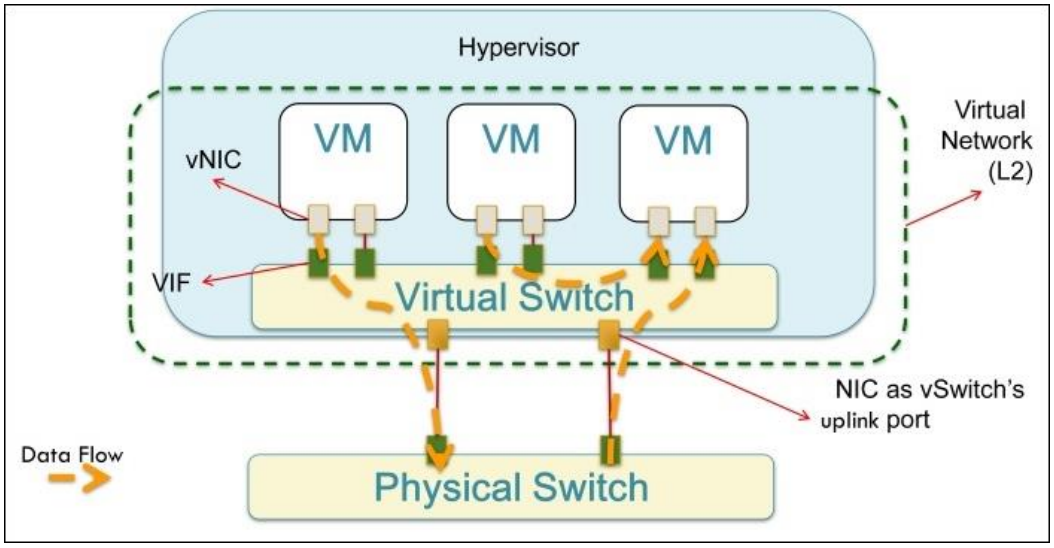

Figure 3.4 OVS and pIF Setup [32]

Multiple planes of SDN shown in figure 3.5 are secured through the following:

- Each plane communicates via secure API mediums which ensure data confidentiality.

- The data packets are secured through hash functions, data off sets and other measures.

- Finally, access to the application, control and data plane itself is encrypted to ensure maximum data integrity and, intrusion detection.

This provides localized security zones within a SDN, enabling layer by layer security as shown in figure 3.5. Dynamic memory allocation solutions such as critical amount guaranteed memory allocation (CAGMA) presented in [21] are fast becoming standards in network security against flooding attacks. The need for a system to be operational during an external threat is paramount whilst the operator or service provider deploys countermeasures. Most dynamic memory solutions, function on a mathematical-based algorithm which allots memory dynamically based on the need of the system. Memory resource management in SDN is controlled through the hypervisor – VM, which is directly controlled by the operator. The administrator can deploy CA, which allocates memory dynamically to an OVS, depending on the resources available. Flooding mitigation, dynamic memory allocation and limitation within a threshold can all be integrated into a single CA, developed and updated at the application layer by operator to be implemented at data plane in OVS. The result would be a sustainable SDN ecosystem which can withstand a potential flooding attack without depleting its memory resources. The logical details of the solution presented in this thesis will be discussed in proof of concept (PoC).

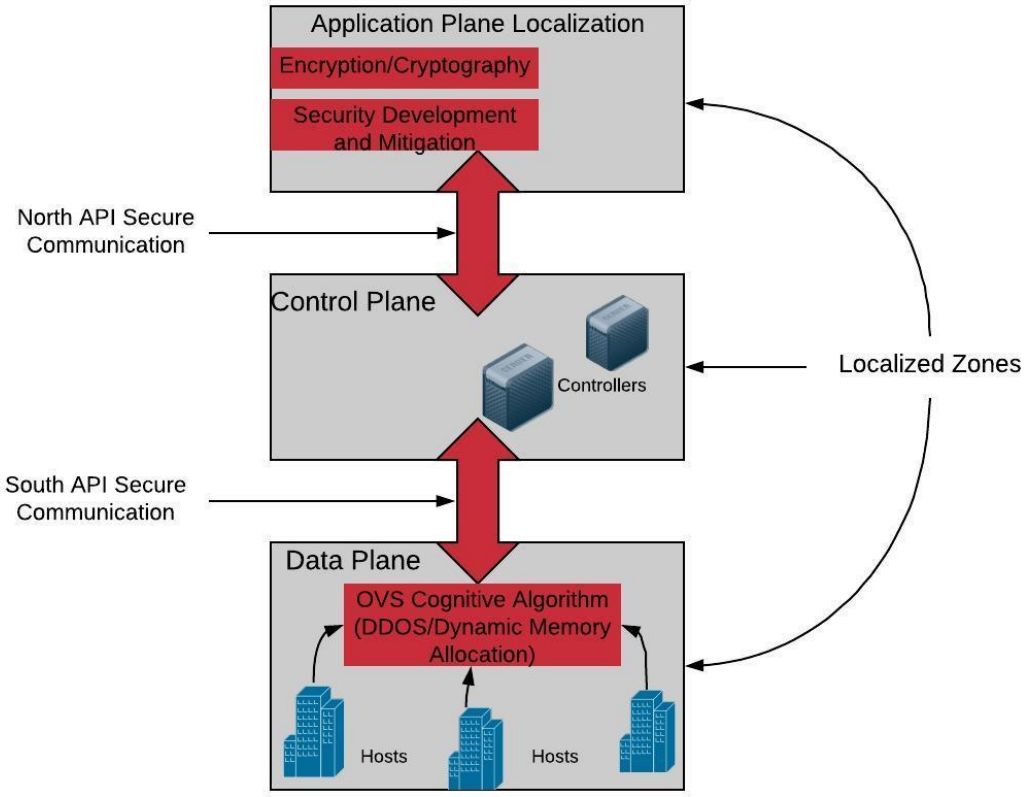

Figure 3.5 Localized Security Zones in SDN

3.4. Controller plane in SDN-CA deployment

The control plane houses the controller which resides over data plane to oversee the functions of the network devices. The controller is operated by the administrator to optimize network functionality for providing reliable and consistent services to the host. The control plane is made of SDN-administrative virtual functions which govern the control flow of the network. The controller is a vital part of the SDN environment as it not only acts as a mediator for traffic communication but is often optimized for performing security functions of network. However, security algorithm and performance evaluation for this research are based in the data plane for distributed work load across network. Research papers mentioned in section 3.3 support LOD to be established at the data plane due to its stability, flexibility and workload-ease towards the controller. For improved network performance, multiple controllers can be connected for higher data bandwidth and reliability towards overall SDN. The overall working concept for the experiment is provided below.

- The algorithm is generated and forwarded by control plane with "Python" programming language to the SDN Controller. The Opendaylight (Helium) Control plane from Open Software Network allows Python script to work interchangeably with L2/L3 switching devices.
- Controller acts as the central core system for maintaining and updating security CA policies. Virtual switches running on Hypervisor resources utilize the security policies and memory allocated by the CA via controller.
- DMA resides with controller based on constant update from the Virtual Switches. Memory increase during attack, isolation and filtration is performed when controller is updated of potential threat via L2/L3 devices.

Due to limited processing and memory unit of switches; bandwidth limitation, feedback to SDN controller and DMA updates are the core responsibilities of the switches. As memory allocated by SDN Controller configures the data management, that process becomes automated. Hence, our primary concern is the Controller's ability to quickly manage and enforce security policies. But our primary result for filtering DDOS packets is attained from the L2 devices as controller does not handle the data packets directly, due to a single bottleneck point in SDN.

3.5. Application plane in SDN-CA deployment

Mutually agreed operator-user preferences in a network are developed in the application plane for a SDN. It hosts security, management and monitoring applications which allow for overall management of the network. Work in this thesis suggests developing a CA security solution for DDOS mitigation and dynamic memory allocation at application layer, which can be deployed at data plane layer at OVS for host and network security. SDN privacy and safeguards developed are configured by the administrator, as the sole control of the system resides with that person. Many of the security solutions including DDOS mitigation strategies are developed and deployed through this plane. This thesis proposes a similar approach to develop a security solution using CA

as a tool to provide flooding isolation and system operation via dynamic memory allocation. The operator can provide variable solutions, distinguishing the traffic need of a host, set of hosts or a sub network leased to different users and organizations.

The entire network can be viewed as “localized logical security zones” which provides security mitigation through every step of the way. Control flow commands, security policies and administrative decisions for the OVS are securely transferred through northbound API to the controller as shown in figure 3.5. Controller relays the information through a secure channel using south bound API to VMs and OVSs deployed in data plane. As highlighted before, the control plane acts as a filter for secure communication, making sure only authentic data packets are processed in either direction. This localized security adds check points along the network adding extra redundancy for authentic traffic communication. In support of the secure communication, APIs provide timely information on threats against a particular IP in the pool or a compromised OVS. Security deployed at OVS can monitor outbound traffic for an insider threat, which is another focus of this research. This provides isolation in an organizational network where multiple SDN networks can have independent safeguards for preventing a total system crash. Flow table residing in a OVS can log IP addresses of malicious hosts and can inform the administrator to apply desired prevention measures. Incoming traffic towards a network is subjected to the same checks and procedures ensuring network wide integrity of data and strengthens NAC.

As compared to a traditional network, SDN planes offer flexibility in terms of logic implementation. For instance, instead of proprietary only security implementations, application plane allows integration of multiple programming and security software directly. This flexibility makes the operator better capable of providing custom solutions according to user preferences. CA proposed in this thesis suggests a security system where the algorithm should adapt to the user’s traffic and a flooding attack independently and simultaneously. This grants security analysts or administrators to only perform higher level functions which is why IPS, IDS and middle boxes are upgrading towards CA as mentioned in [6].

3.6. Chapter Summary

To deploy a scalable and security solution for a network, SDN and CA can be integrated for optimization. As discussed in this chapter, many algorithms-based security systems are utilized for security which are deployed across SDN. Traditional SDN security deployments at controller are fast moving towards OVS as a better and efficient option. SDN offers flexible deployment of network security where LOD can be implemented remotely by the administrator without physical intervention in contrast to traditional L2 IPS and IDS. The controller coordinates the data plane through instructions managed by administrator at the application layer. Communication between different planes of SDN is carried out through secure APIs. The next chapter will discuss how these policies can be implemented to provide DDOS mitigation and memory resource management.

CHAPTER 4

PROOF OF CONCEPT IMPLEMENTATION

4.1. Introduction

A Proof of Concept (PoC) for the proposed Software Defined Network – Cognitive Algorithm is presented in this chapter. The algorithm is designed for DDOS – ICMP detection and mitigation as well as Dynamic Memory Allocation (DMA) in a SDN. Opendaylight (ODL) is used as open source project for defining and implementing the CA, application layer and API's, and ODL Controller to manipulate Open vSwitch (OVS). The proposed framework in the previous chapter is implemented by integrating OVS, VMs and Opendaylight. DDOS mitigation is achieved by implementing a python script algorithm, supported at application layer, which is managed and executed by the ODL controller at the OVS switch. This ensures Network Access Control (NAC) while providing total control over the SDN. An ideal Matlab simulated solution for non-lossy packet environment is also presented which is used for result analysis in next chapter.

This chapter is organized as follows: Section 4.2 elaborates the software tools utilized for SDN deployment and the technologies supporting the set up. ODL platform, OVS and controller integration are primary software through which SDN is created, for the proposed method. SDN-CA security in light of the created set up is discussed to highlight its efficiency as a security service. This section is concluded on the API's used for the data transfer between host and admin at data and application plane respectively. The Design and implementation of the network (via tools described in 4.2) is demonstrated in section 4.3. A comprehensive network configuration is elaborated, where CA for mitigation and DMA is implemented. Lastly, the chapter is concluded in the summary.

4.2. Associated Software and Technologies

This section gives in depth details of the various software platforms and technologies, integrated to provide the SDN based CA solution for security. ODL platform and controller is established to provide application and control plane organization which oversees the tasks carried out in infrastructure (data plane) layer devices, OVSs and VMs. The VMs hosted by VirtualBox are connected to OVS to complete the SDN. The PoC is presented by integrating these three software platforms.

4.2.1. OpenDayLight (ODL)

OpenDaylight (ODL) is a modular open platform for customizing and automating networks of any size and scale which arose out of the SDN movement, with a clear focus on network programmability [7]. It is a software platform supported by The Linux Foundation, openly

available to the public for academic and industrial research. Figure 4.1 illustrates some of the existing projects under ODL Helium. The following subsections will discuss the various components and functionalities of ODL in detail.

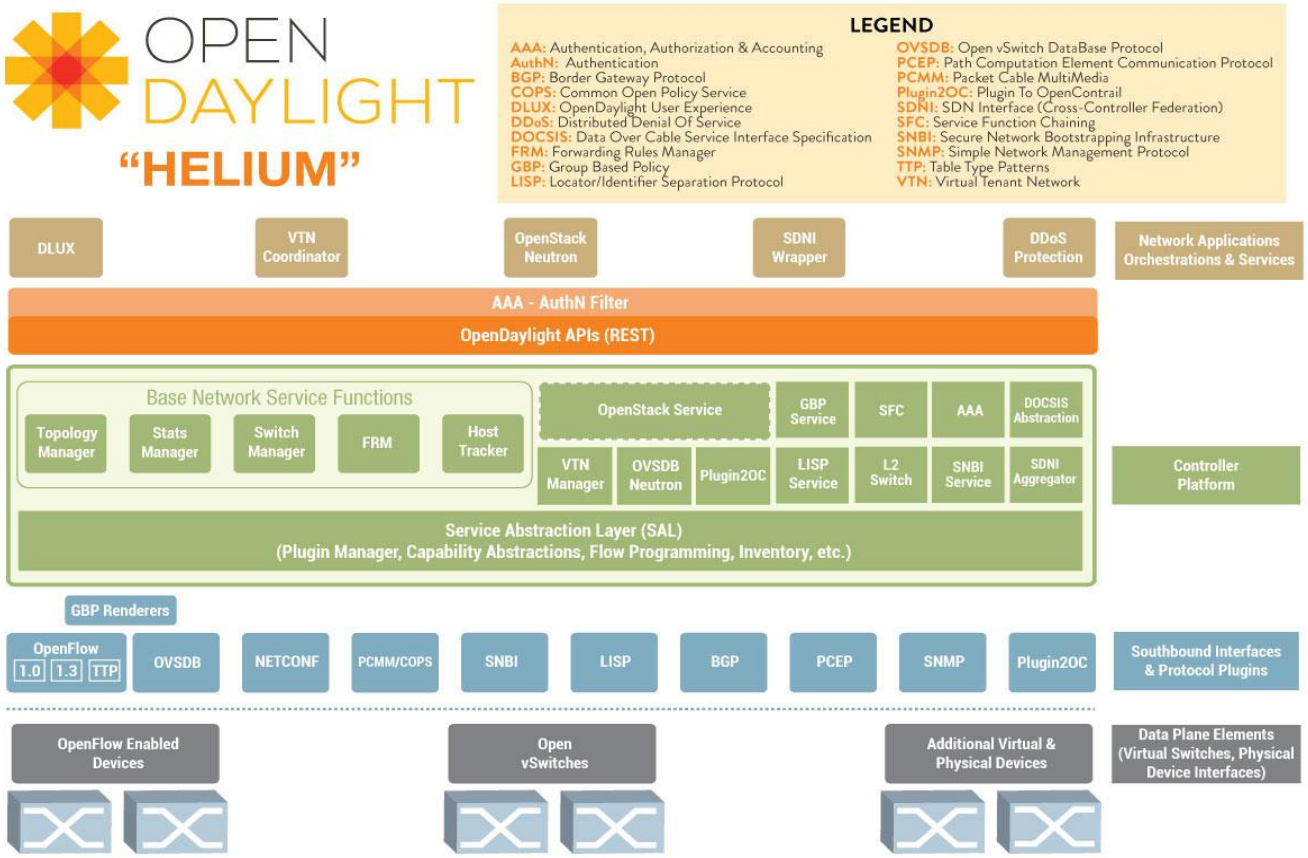

Figure 4.1 ODL Helium Projects [7], [33]

4.2.1.1 ODL Controller

The controller is launched as a distinguished software entity which is connected to the ODL. The controller uses a Java Virtual Machine to operate and is connected to the OVS and ODL application through various APIs. The ODL Controller utilizes Representation State Transfer REST-API for its northbound communication towards application layer [7]. This offers a secure state transfer from applications layer to the controller to further develop and deploy, infrastructure layer. The southbound API by default is configured through OpenFlow (OF) and Border Gateway Protocol (BGP) as stated in section 3.1. Conventional routing and network device configuration with BGP makes easier integration for the southbound API. As shown in figure 4.1, Service Abstraction Layer (SAL) is connected to these modules to provide Software Platform as a Service (SPaaS) as demonstrated in [30].

4.2.2. Open vSwitch (OVS)

Open vSwitch (OVS) is a widely used software switch which uses a tuple space search algorithm for packet classification, and an exact match cache (EMC) scheme for caching most frequently used flows [36]. OVSs are created to bridge the gap between the VM, hypervisor and the ethernet connection to the internet, in a SDN and cloud environment where network is ever expanding. The hypervisor containing the VMs have finite numbers of Network Identity Cards (NIC) which provides limited connectivity for hosts. Due to this, each subnetwork is limited to few connections due to limited NIC as shown in figure 3.4. The OVSs bridges that gap by connecting VMs to a Virtual Interface (vIF) which is connected to physical interface (pIF) commonly eth0, for connection to the rest of the network. Like traditional networking devices containing routing tables and protocols, OVS houses flow tables to execute open flow actions by identifying MAC addresses, IPs and ports of the source and destination. The flow table manager is configured to manage searching, adding, updating and deleting of flow entries in the flow table [2]. Architectural design of OVS is shown in figure 4.2. Proprietary L2/L3 switches and routers are introduced within the SDN for resource management and compatibility. Resource management simply implicates that OVS and associated SDN resources are limited and are combined with proprietary physical hardware technology to optimize the network for efficiency and NAC

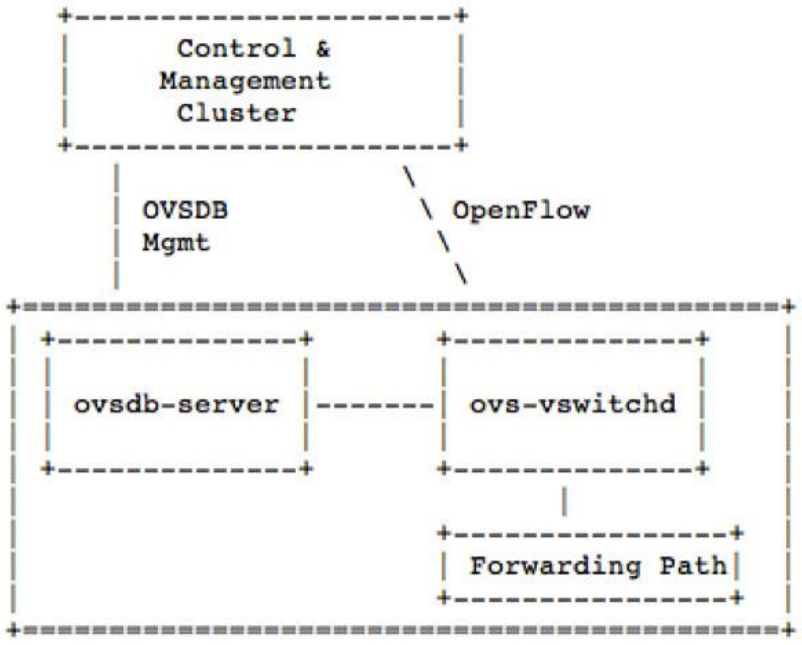

Figure 4.2 OVS Architecture [33]

4.2.3. Virtual Box

VirtualBox is an Oracle supported platform designed to be a virtual test-bed for the proposed framework. It is designed to enhance academic and industrial research, without the need for actual

hardware deployment. The suggested SDN-CA solution is implemented on a Linux Operating System (OS). Hypervisors, VMs and multi-tenant network are established using Mininet on Virtual Box platform, where a localized subnetwork is connected to OVS using southbound BGP and open flow APIs.
Multiple Operating Systems (OS) could be used in Virtual box depending upon compatibility of the system. Linux OS is chosen owing to its ease and user familiarization with the system. Report in [30] discusses the effects of DOS attack on SDN Controllers in Virtual Box simulation. Similar methodology using ODL controller has been employed in this thesis where VMs and hosts are housed in mininet.

4.2.3.1. Mininet

The Mininet is a system that allows rapidly prototyping large networks on a single computer and it creates scalable SDNs using lightweight virtualization mechanisms, such as processes and network namespaces [37]. A small network of 3 hubs and multiple users are deployed as tenants for the proposed framework. Users in the subnetwork are connected to OVS by the southbound API sflow. IP and MAC for each user is registered as flow entry maintained by flow table in OVS, which connects to controller in ODL via REST-API. This ensures user connectivity to SDN and the internet which is demonstrated in section 4.3 of the PoC.

4.2.4. SDN-CA as a Security Service

The Cognitive Algorithm solution makes scalable security solutions possible by limiting the user interference to a minimum. SDN-CA can be used to establish a flexible generic solution, adaptable to the DDOS attack, in data plane OVS. The SDN-oriented network is self-managed and self-controlled where the controller can detect and mitigate attacks automatically [2]. The SDN REST, OVSDB, sflow and open flow – API also provides authentication, integrity, and privacy of communications between a client and server implementing this protocol [38], which secures SBT and NBT. Few of the basic features of CA in SDN are highlighted below:

- Pre-processing Network Traffic Prediction Method: Using Time Range analysis, a cumulative average of the traffic can help predict the future traffic patterns [23].

- Network Anomaly: Using (NAPA) [22] and (NADA) [23], we can identify a potential DDOS attack and mitigate it before network failure.

- Data Acquisition: Generic and cognitive algorithms operate at the core of data mining and cleaning, to obtain required outputs in SDN [6]. SDN-CA security can provide fine combing of data to detect malicious packets for malware and IP spoofing.

- Resource Management: Resource allocation, particularly of memory involves algorithm deployment. Critical Amount Guaranteed Memory Allocation (CAGMA) presented in

[21] and secure memory allocation through “Secure Memory Controller (SecMC)” demonstrated in [20] both work against catastrophic system failure with algorithm solutions.

4.2.4.1. Python Script

Python project was started in the early years of 1990 decade by Guido Van Rossum in the CWI in Netherlands [40]. It is a multipurpose programming language, with a huge compatibility factor with numerous applications. As cited in [40], the language is flexible with diversity towards structure, objective oriented and other types of coding requirements. Cognitive Algorithm proposed here is written in a python script. The ODL Helium platform supports python script for administrative actions related to log management, routing and security allowing easy writing, debugging and execution of the script. Python version 2.0 runs in ODL software platform using REST API to coordinate and control data plane through ODL Controller [38].

4.2.5. API’s

Section 3.3 discussed in detail the importance of APIs for a SDN. A communication between the controller and the application plane is established using northbound API and, between controller and the data plane using southbound API. APIs used in this PoC are described below:

- REST API: Possibly one of the most important APIs, it is used to establish and maintain connection between application and control plane. The controller and software platforms run on different address space for which REST provides integration. REST commands include GET, PUT, DELETE, POST which are used to request, create/update, remove and repeat respectively [40].

- OpenFlow: It provides communication between controller and OVS. The control logic from controller is provided to OVS which is stored in its flow table (in form of flow rules). Openflow is often extended to support security such as packet data scan detection [2]. This thesis suggests the use of flow rules to implement CA.

- OVSDB: This is a data base management protocol that allows configuration of vSwitches as a southbound API.

4.3. Design and Implementation

This section provides a detailed description of how the software platforms and technologies discussed in section 4.2 have been used to present PoC in this thesis. The various sections covering different topics in this section are Network Setup, CA deployment, Dynamic Memory Allocation deployment, DMA and CA integration, OVS flow rule configuration and flow table reconfiguration.

4.3.1. Network Setup

Figure 4.3 demonstrates the network deployment of OVS, vSwitches and multiple tenant network in the SDN data plane. OVS provides security checks for traffic growth between multiple users, in and out of network, and relays the information to the controller for analysis. A small network of two vSwitches s1 and s2, hosting multiple tenants ranging from h1 to S3, S4 and S6 respectively, are created in Mininet. Software-defined networks using lightweight virtualization mechanisms, such as processes and network namespaces, permit the Mininet to create, interact, customize and share the prototypes quickly [37]. Both vSwicthes are created under independent VMs in Virtual Box so that two vSwitches cannot communicate directly without an OVS as shown in figure 4.3. This is to ensure the flow of network traffic in data plane through OVS. Set up for the vSwicthes is explained in Appendix A. VirtualBox allows for automatic port mapping within the network which is confirmed by establishing connection between two users of the same network. Throughout the setup ping command is used to establish network connectivity [42] and as means of DDOS attack [12], [8] for testing the network for vulnerabilities.

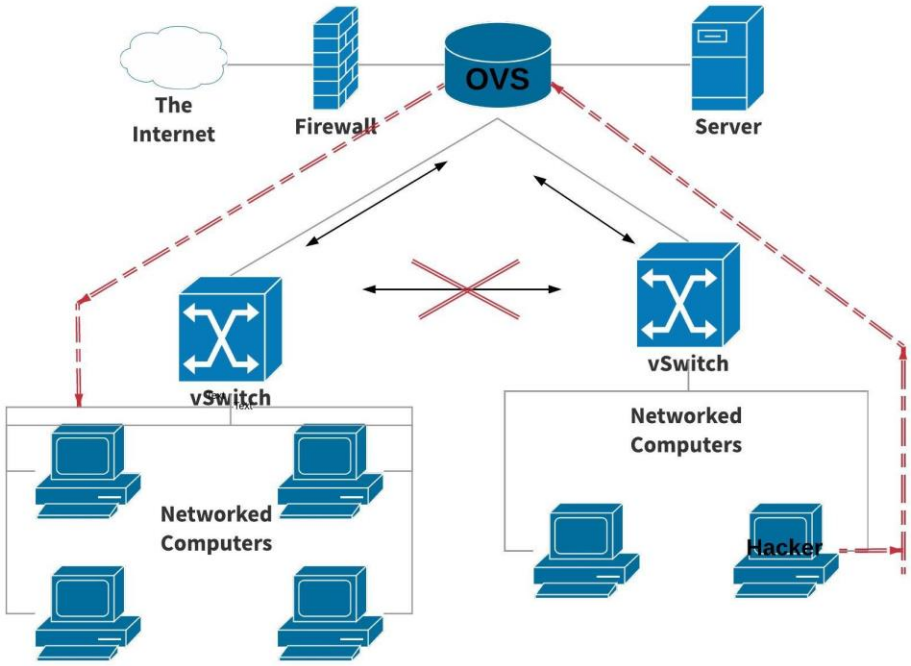

Figure 4.3 Data Plane deployment

The Shell – CLI of Debian based OS allows for easy installation and configuration of OVSs in the OS as demonstrated in Appendix A. An OVS bridge named “OVS Link” using “ovs-vsctl” command [33] is built as shown in figure 4.3. By default, OVS is connected to the IP stack of DHCP, which is connected to the pIF of the system (eth0). In order to establish connection to other networks (internet), OVS link’s internal port is made a DHCP client, with a valid IP address and default gateway as shown in figure 4.4 [33]. Network connection is verified with ping to “google” to confirm network traffic flow. The sub network hosting tenants and vSwitches is connected to OVS using virtual tap ports.

The OVSDB protocol establishes data flow from OVS to the subnetwork vSwicthes and hosts [36]. The OVSDB management protocol is intended to allow programmatic access to the OVS database [38]. Port connection from the OVS to the VMs is configured through a bridge network adaptor in virtual box which facilitates a valid network setup shown in Appendix A. OVS and VMs connections in figure 4.4 are supported by open flow protocols which support SDN configuration, necessary for this thesis. Connections between vSwicthes-vSwicthes and vSwitches-eth0 are tested to ensure intra and inter – network wide connectivity respectively. By default, OVS is configured to forward packets as a normal layer 2 OSI model switch [33], which enables data flow without the use of a SDN controller as demonstrated so far. This practice allows for backward compatibility in a traditional network system with an OVS.

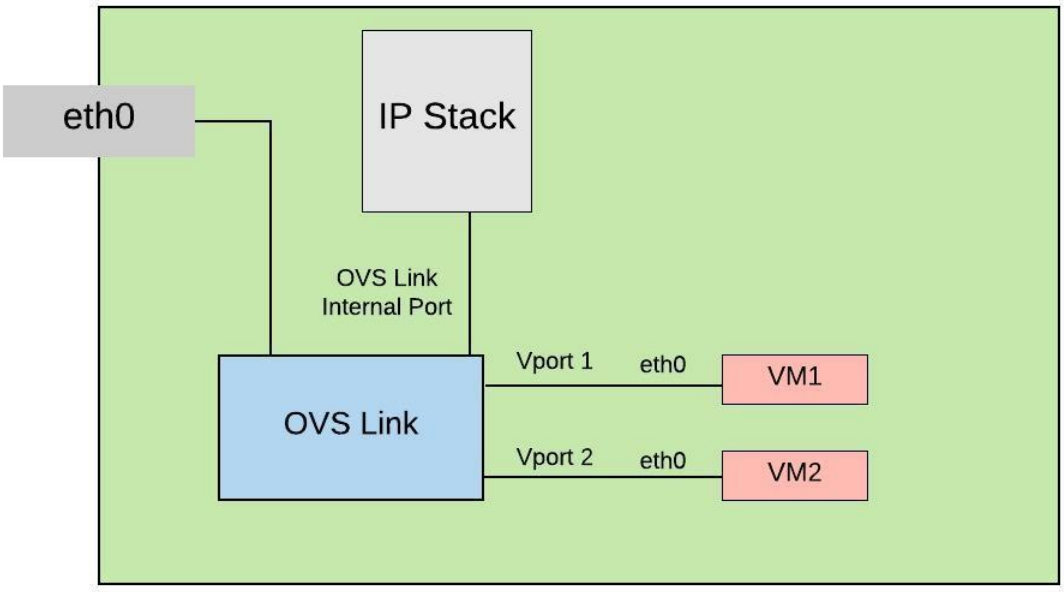

Figure 4.4 OVS and VM Interface Integration [33]

The ODL controller is used in this thesis for relaying control information to OVS as discussed in section 4.2.1.1. The controller is established as a node in the ODL platform where it is assigned MAC address and designated port for connection with OVS. MAC, IP and port address allows for a SSH connection between the OVS and the Controller. Automatic port mapping using open flow allows for default port connection between a remote controller and opensource application platform in most generic controllers. However, a manual connection using port, MAC and IP information is established in this thesis.The packet flow behavior between OVS and controller is cited in [12] as shown in figure 4.5

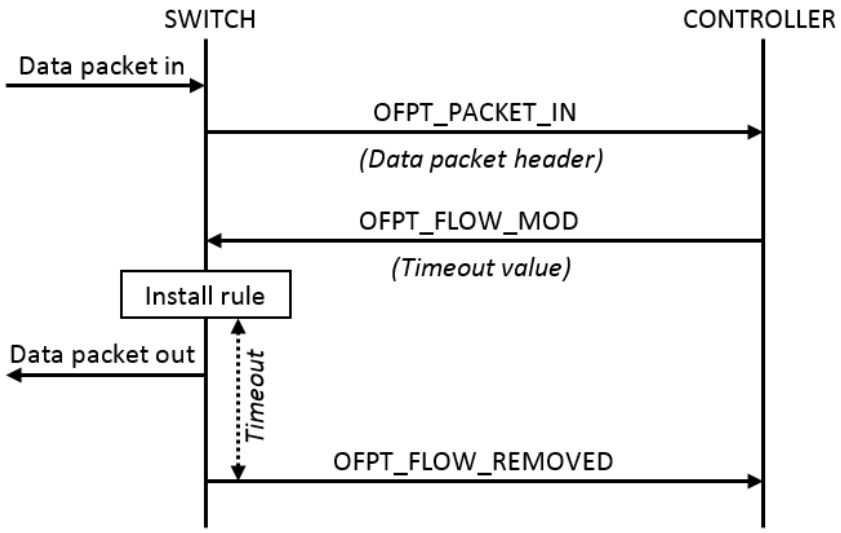

Figure 4.5 OVS – ODL reactive flow [12]

Figure 4.5 demonstrates the constant update of controller by a switch as cited in [12]. Using Open Flow Packet Transfer (OFPT) data packet is sent to controller where new flow rule is sent and replaced at the switch. The effects of a DDOS attack in [12] are assessed by OVSs for which a new flow entry using "OFPT_PACKET_IN" is sent by the controller. The same methodology is used in this thesis where CA is deployed using python in application layer. Algorithm is converted to flow entries which are forwarded to controller for deployment using REST-API. The ODL Helium allows default data packet forwarding to the ODL controller using REST API.

4.3.2. Cognitive Algorithm Deployment

The Cognitive algorithm is deployed as a python script which allows for troubleshooting and compatibility in the ODL environment. A similar approach in [27] is used where an algorithm is used for an event-based security system in data plane. The Ryu controller used in that work supports python scripts for security implementation. CA is developed in application plane by the operator, and is transferred to the ODL controller for processing via REST API. The proposed model here is to deploy security measures at the OVS to alleviate controller from managing large traffic. Experiments performed in [2], [26] and [27] are based on algorithm-based security which is deployed at OVS or data plane layer. The algorithm is deployed through a secure API at OVS. The flow table manager is configured to manage searching, adding, updating, and deleting of flow entries in table and OVSDB confirms that new security policies are correctly inserted [2]. The security measures are stored as flow rules which are used as LOD against DDOS attacks for detection and mitigation. The security solution deployed at the data plane also generates less traffic due to little required communication. Variables involved for detection, mitigation and tracing such as source IP, MAC address all reside in flow tables. Each ICMP request source information is stored before stripping the headers as it moves for forwarding, which is overseen by OVSDB in real time. Data packets from north and south bound traffic are incremented for ICMP flooding. This prevents a potential DDOS attack from within the network as discussed in section 3.2.

4.3.3. DDOS Mitigation and DMA – CA

The ODL – cognitive algorithm framework, for ICMP DDOS mitigation as well as DMA is presented below. The ODL application layer structures the CA and then transfers it to ODL controller. The algorithm is built on Markov's Chain Rule [34] as discussed in section 3.2.1. The ODL controller assigns CA for the required OVS subnetwork. The user specific solution is provided by the controller to cater for varying organizational or individual traffic requirements respectively. NAPA and NADA cited in [22] and [23] respectively, are based on computational algorithm where probabilistic traffic analysis helps determine if the network is under attack or not. CA in this thesis is based on dynamic count check to analyze a potential increasing ICMP traffic disguised as a DDOS attack. The IP and MAC identity of the source is processed against growing traffic to isolate the attacker and enforce mitigation policy flow rule. Following a potential red flag:

- Malicious source is isolated.
- Additional memory is allotted to the OVS.
- Data dump for malicious source is performed in flow tables.
- And new flow rules are requested by OVS flow tables through ODL Controller.

Controller manages memory allocation and instructions using OVSB and OVSDB which relays information to VMs using virtual ports. CA allows for constant check and balance for an overflow while updating its flow table for new policies simultaneously, in real time.

ODL Cognitive Algorithm for DDOS Mitigation & DMA

Task: Store ICMP requests against Source_IP in OVS flow table;
Perform count check for ICMP increment; Consult flow table for new policies.

Action: Locate appropriate policy for scenario; Install security policy; Update flow table or request controller for new mitigation policies.

```
1.  For all ICMP_Requests do;
2.      Perform count check against source IP          %Metering Algorithm per IP%
3.  Time = time++; Countcheck = 0; Increment = contt    %Count check measure %
4.  If Time ≥ contt3
5.      If Countcheck ≥ initial value then;            %Condition verification%
6.          action ← perform concerned mitigation.
7.      Where action → Block IP_Source;
8.                     Initiate Source_IP trace;
9.                     Initiate Source_IP memory dump;
10.     Else;
11.       Forward packet to ODL Controller for policy update;   %New security policy%
12.  End If;
13.     Memory = Memory + Increment;
14.     Countcheck = Countcheck + Increment1;
15.     Add new flow rule
```

16. End for;

4.4. Chapter Summary

To validate the SDN-CA security proposed ideas of chapter 3 are executed in PoC. Software and technologies required for the emulation of the proposed model are discussed. ODL and ODL Controller in helium-based architecture of platform offers integration with python and plug ins, with ease and reliability. Mininet and Virtual Box are integrated for Linux based system to establish VM and hosts in data plane layer. Integration of these two technologies offer scalable and rapid deployment of sub network in this SDN platform. Installation and deployment of OVS is carried through Linux Shell and is connected to internet and VMs through pIF and vIF respectively. OVS is linked to controller using OVSDB and OVSB to establish secure connection. ODL and ODL controller are linked through REST API. This chapter further introduces CA used for security and DMA which is presented as a scalable modern solution for DDOS prevention and detection.

Chapter 5

Simulation Evaluation and Result Analysis

5.1. Introduction

This chapter discusses the outcomes and result comparison analysis of the experimental set up explained in previous chapter. The outcomes for the projected model with and without enforcement of CA are compared. Network analysis of CA in the given state of the device is explained in detail. This chapter focuses on the architectural networks deployment of the system and its configuration with SDN Controller; DDOS attacks are performed for output comparison; and Network Access Denial also serves as a test to conduct CA simulation. Further it broadly specifies the mathematical model (formula and code) implementation of the CA. The employed algorithm output is to be evaluated against count variables (ICMP). Finally, the graphical output for packet drop percentage and filtration is studied to conclude the thesis. The key factor associated with network filtration is RTT to isolate the source. A Wireshark supported packet trace, for advance countermeasures is suggested as well.

5.2. Attack Initialization

In this section we will discuss the types of DDOS attacks used, their impact on SDN without CA deployment and resource utilization. As elaborated in section 3.2, CA is designed to monitor traffic flowing in and out of OVS. Security systems, by default, are configured for monitoring traffic flowing into the network. The attack is deployed by the host within the network towards the victim to test the algorithm. The feedback from OVS to the ODL Controller in the control plane verifies the validity of the detection and countermeasures. ODL Controller relays the information to application layer which provides new security flow rules, implemented by the controller.

5.2.1. Types of DDOS Attacks

A Denial-of-Service (DoS) attack in OpenFlow SDN networks involves overwhelming computing or networking resources such that an OVS is unable to forward packets as expected [12]. As mentioned in literature review, classification and type of DOS attack varies ranging from TCP, Smurf, ICMP and many others depending upon intent and priority of the attacker. An ICMP flood attack is used in this thesis to attack a host within a network. The traffic is generated from user s1, s2 and s3 residing under switch 1 directed towards user (victim) in switch 2, as shown in figure 5.1. The traffic is routed through OVS to have a centralized network data flow point in the entire data plane. Centralized data flow for each subnetwork allows for filtration through CA. Attack methods used in here are ping of death [47], Hping3 [30] and LOIC [45].

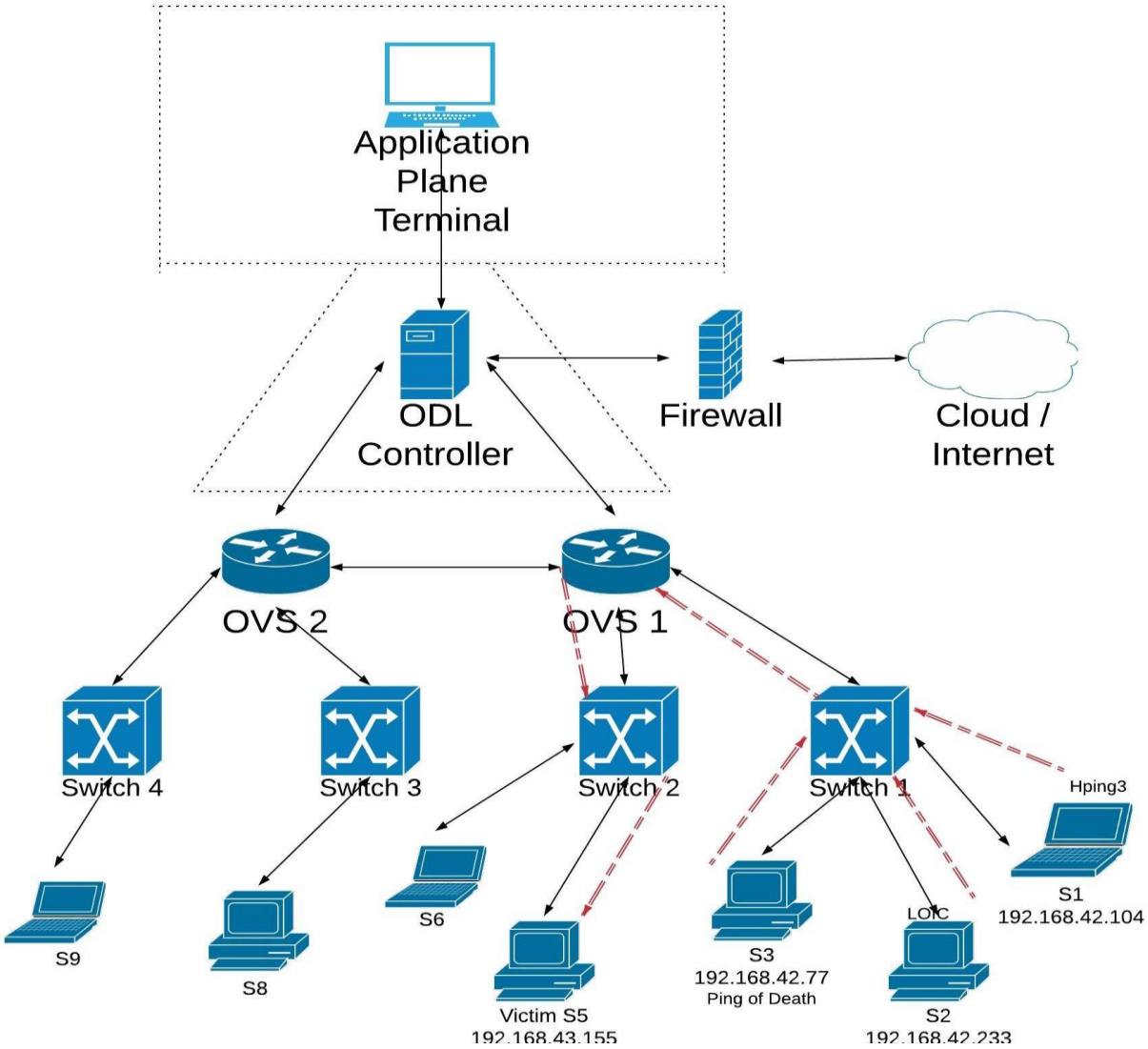

Figure 5.1 Network Topology of ODL SDN

5.2.2. Attack Simulation

Hping3 is a packet generator and analyzer for the TCP/IP protocol. This tool can be used for security auditing, testing of firewalls and networks [30]. Hping3 offers multiple attack techniques as shown in [30]. For simplification, known IP of S5 is flooded with attack packets to simulate a DOS attack. The packets are flooded at a rate of 60,000 packets/sec as a single attack source. The attack path and the sub network data traffic – flow is shown in figure 4.3. Hping3 is a software platform that utilizes the same principle as ping of death attack. The Ping of Death is a typical TCP/IP implementation attack. In this assault, the DDOS attacker creates an IP packet that exceeds the IP standard's maximum 65,536-byte size [5]. When the system cannot assemble the large packet quickly enough, it either crashes or reboots. S3 uses static IP to send ICMP flood attack packets towards S5 at a slow rate of 40,000 packets/sec.

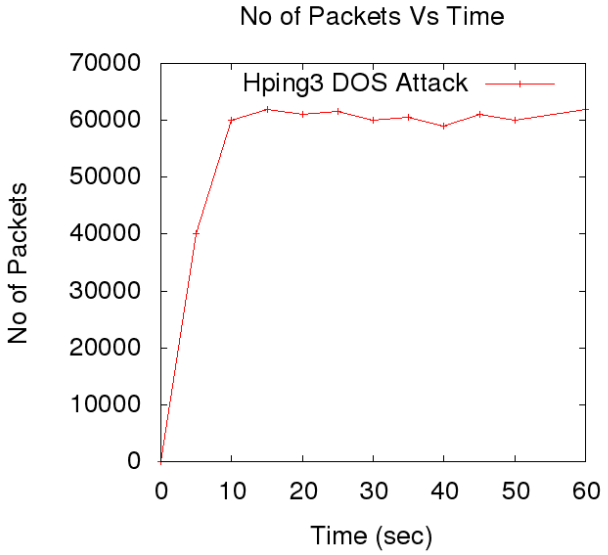

Figure 5.2 Hping3 Throughput

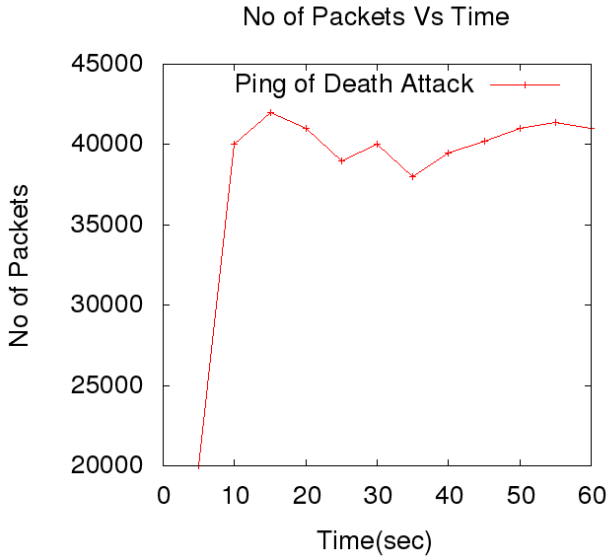

Figure 5.3 Ping of Death Throughput

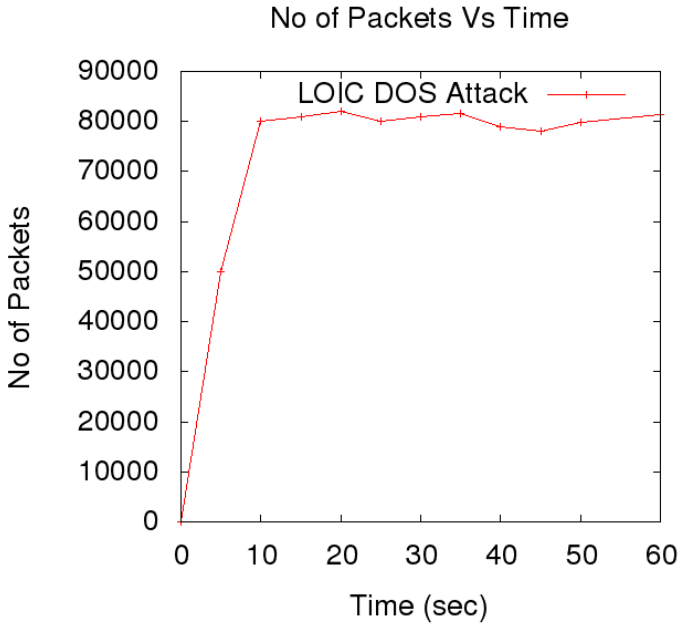

Figure 5.4 LOIC Throughput

Slow DOS attacks induce delays in the network and memory consumption due to response. This type of attack is usually combined with high bandwidth attacks such as Hping3 and LOIC to amplify the strength of a DDOS. Some of these newer DDoS tools, such as Low Orbit Ion Cannon (LOIC), were originally developed as network stress testing tools but were later modified and used for malicious purposes [45]. This is a GUI software platform that allows for high volume data packet to be sent per second. An attack rate of 80,000 packets per second is initiated using LOIC as shown in figure 5.4. The software allows for target IP to be flooded by a desired number of

packets per second as shown in figure 5.5 [45]. DOS attack for each type is shown in figures 5.2, 5.3 and 5.4 respectively.

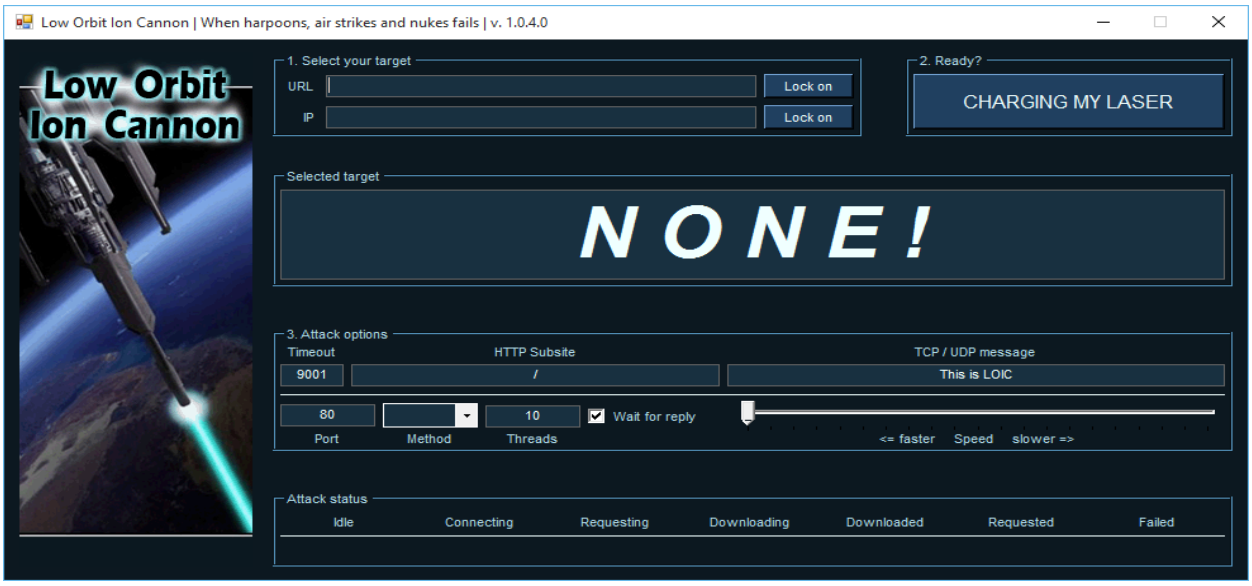

Figure 5.5 LOIC Interface [46]

5.2.3. Simulation Analysis

DDOS attack strength is a measure of memory exhaustion, network intrusion and packet or information loss from that network. ICMP in an SDN network is particularly designed for packet loss, and memory exhaustion which is the measure of DDOS attack strength for this research. Table 5.1 shows the packet loss, power consumption and memory utilization for individual attacks as well as DDOS.

Type of Attack	CPU Utility	Memory Unit	Duration of Attack	Packet Loss Rate of Victim
Hping3	67%	63%	45 sec	74.48%
Ping of Death	55%	45%	60+ sec	18.5%
LOIC	89%	82%	35 sec	92%
Collective / DDOS	96%	86.63%	------	93.35%

Table 5.1 Resource Utilization During DDOS

Maximum packet loss is observed across DDOS attack where 93.35% of the packets were dropped. As the attack strength of DDOS varies, so does the CA response time to it. CA is designed to detect and isolate data traffic after certain data limit is reached. This defines the duration of attack as large data packets with higher bandwidth would reach limit quicker contrary to low bandwidth, small data packets. The SDN, without CA is subjected to heavy packet loss and high memory consumption. Graphical output for combined DDOS attacks is shown in figure 5.6

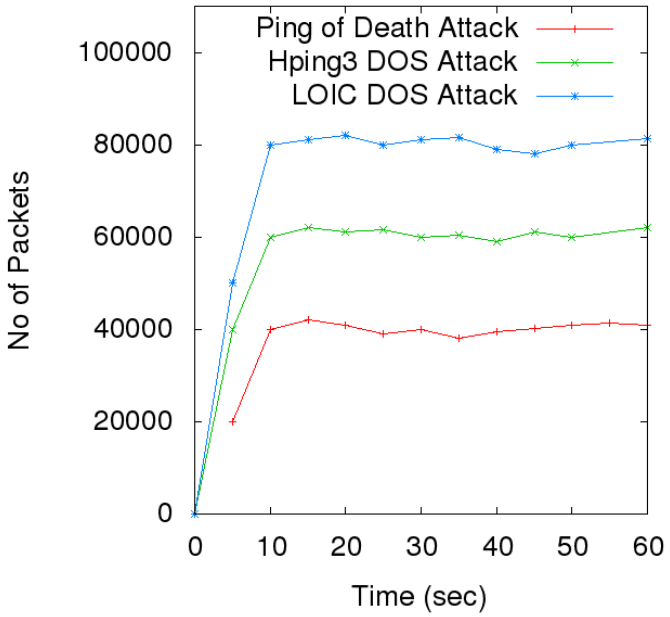

Figure 5.6 DDOS Attack Throughput

5.3. Network Reconfiguration

This section covers the SDN changes implemented in the network after CA deployment. Security management functions are developed and updated by application layer. The cognitive algorithm developed in section 4.3.3 is programmed in application layer. Application layer allows for the algorithm to be defined by one of the two ways:

- Dev API
- REST API

Dev API lets direct design of the algorithm or routing configuration through flow rules. It is primarily used by developers to directly install new flow rules into the controller through live application platform. REST API is used in this thesis which allows for development, deployment and update configuration of the CA. Protocols like “Simple Network Management Protocol” (SNMP) [38] developed through open standards, deliver routing and security policies. SNMP and Northbound APIs deliver the updated security measures to the control plane where they are implemented. Another protocol incorporated for managing and delivering security solution is Network Configuration Protocol (NETCONF). NETCONF is an IETF network management protocol that provides mechanisms to install, manipulate, and delete the configuration of network devices [38]. Numerous other such open source protocols exist, configured to the requirement of the developer software platform. This provides easy automation of the system on the go and reduces administrative workload.
The SDN-CA is developed using python 2.0 application in application/management layer. Figure 5.7 given below when supported by table in section 4.3.3, provides an elaborated flowchart of the network configuration.

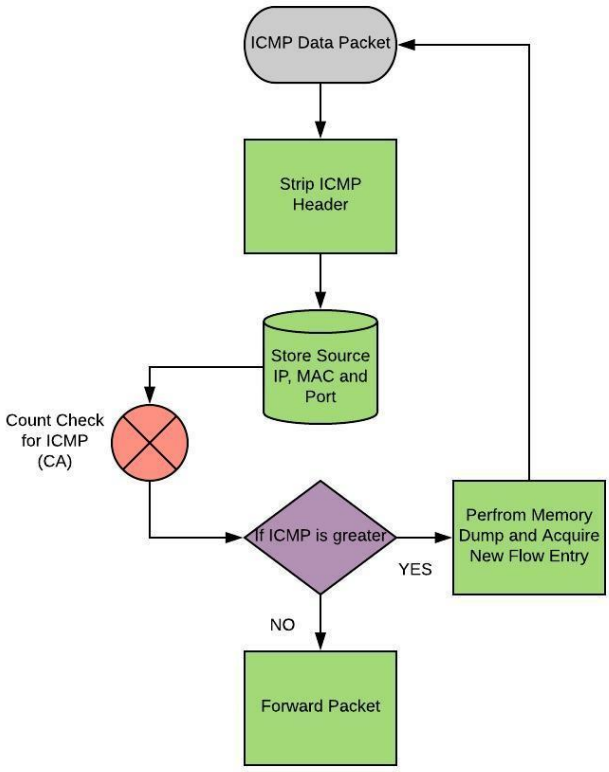

Figure 5.7 Network Reconfiguration

When received, OVS decapsulates the packet for ICMP header and adds it in an incrementing count check. Each OpenFlow-compliant switch maintains one or more flow tables, which are used to perform packet lookups [38]. The count check is performed by a "Metering Algorithm" which keeps track of ICMP entry against different IP's. Indicated attack source IP for thesis simulation are 192.168.42.104, 192.168.42.233 and 192.168.42.77 shown in Appendix. Flow rule determines the maximum number of allowed ICMP requests per user based on the algorithm. The Little's Law determines the memory allocation of the SDN, based on the resources available. Memory allocation is described in detail in the next section. In case of a potential DDOS attack, OVS uses Southbound API to inform ODL Controller. This is to ensure network wide security through constant update of control plane. Isolated IP or MAC source is blacklisted and a flow rule update, is sent to every OVS in the network. Prevention of simultaneous attacks on multiple OVS of the network is paramount, for which control plane has to constantly update the infrastructure layer. ODL Controller uses control logic to request new flow rules by the application layer. The application layer provides a security solution based on the existing CA scheme or, prompts the administrator to develop an updated version.

During the setup of the lab for simulation of this experiment, no major compatibility issues were faced. Minor issue like NETCON, Data ping transmission and protocol implementations were presented but were resolved. Having said that, a different scenario may present its own unique challenges for compatibility, which leaves room for future experimentation.

5.3.1. Maximum Memory Enforcement

5.3.1.1. Little's Law

Little's Law reasons average waiting time and number of items and relates these two metrics via the average rate of arrivals to the system [43]. This law is often employed to calculate throughput of a network, with many more applications for time and count dependent variables. Depending on the variables involved, a maximum threshold could be determined for a system. Most common use of this law in networks is for determining the window size for incoming traffic. Mathematically it is represented as below

$$\lambda(T) = \frac{N(T)}{T} = arrival\ rate\ during\ time\ period\ T, \quad (7)$$

$$L\ (T) = \frac{A(T)}{T} = average\ queue\ length\ during\ time\ period\ T, \quad (8)$$

$$W(T) = \frac{A(T)}{N(T)} = average\ waiting\ time\ in\ the\ system\ per\ arrival\ during\ T. \quad (9)$$

$$A\ slight\ manipulation\ gives\ L(T) = \lambda(T)W(T) \quad (10)$$

Little's Law Mathematical Analysis [43]

The maximum memory allocation for this thesis considers constant number of hosts, with potential for scalability. Little's Law allows for scalable solution for a dynamic system.

Number of Users = x
ICMP Requests = y(t)/sec
Maximum memory available = m
Maximum memory threshold = Sys = m – (m/100)*15
User Threshold = U = [Sys * (1/y(t)*x)] * percentage

The above equation shows the memory allocation for each individual user in relationship to the ICMP requests being processed. The system has a memory allocation denoted by "m" at any given instance t. However, this does not represent the maximum memory, which can be expanded to "Sys" under large amount of ICMP requests. "Sys" represents the maximum memory resource available for a System (Hypervisor), with the exception of 15%, which is only made available as a redundancy. Finally, each individual user is allocated a memory threshold out of "Sys" for ICMP management which is denoted by "U". The vital notation "U" is the primary focus, as it represents user memory and is utilized as a cut off parameter for ICMP DDOS mitigation. 'x' determines the memory distribution, for given ICMP requests. This allows for flexible system where additional memory and users are automatically accommodated. HP 5406zl mentioned in [12] is used for small network deployment with little memory requirements for the system. This reduces the chance for a catastrophic system failure under a large DDOS attack.

5.4. Analysis of Results

The proposed CA solution reduces the chances of DDOS attack overwhelming an SDN where multiple tenants could be potential targets. The simulations demonstrate effectiveness of CA and

DMA in an SDN, where in presence of no security measure network could be brought down easily. Figure 5.9 shows the response of the system under DDOS with security measures. The bandwidth of the system is quickly overwhelmed by the attacker and legitimate packets are dropped as a result. The defense against a malicious source shows promising results after the deployment of SDN-CA. The ICMP packet strength is measured per second against source which provides real time update of the flow table. Figure 5.9 shows the results of the CA when deployed against DDOS. Table 5.2 represents the packet filtration percentage for the amount of packets generated in figure 5.6. Mathematically it is calculated by:

$$P_{f=} \frac{C_p * 100}{A_p} \%$$

Packet filtration P_f is calculated using, "C_p" calculated packet response and $A_p$which is actual packet response of the system.

Type of Attack	Calculated Filtration Packet vs Time	Actual Filtration Packets vs Time	Packet Filtration Range (%)
LOIC	2.5 Million Packets / 32 sec	3.05 Million packets / 37 sec	62.22 – 70.21%
Ping of Death	2.6 Million Packets / 65 sec	2.69 Million Packets / 66 sec	65.87 – 73.38%
Hping3	2.7 Million Packets / 45 sec	2.88 Million Packets / 48 sec	78.51 – 85.97%

Table 5.2 Packet Response of the System

Immediately it shows 60-75% packet restriction and IP source isolation by blocking any further traffic. Despite increasing strength of the attacks, memory allocation prevents from network wide failure, which is confirmation of the proposed algorithm. It can therefore be concluded that as the attack is being isolated, memory response for the system increases and ICMP traffic is filtered and restricted. As the response is instantaneous for the number of packets generated, no need for mitigation response calculation was necessary as no catastrophic CPU or memory loss occurred.

Wireshark was used to monitor and track the amount of data flowing through the network as well as tracking legitimate packet drop/send. This part can be covered in threefold.

- The I/O graph enables to measure data packet bandwidth against time which was used before and after CA implementation.
- The data packets in Wireshark based on their source and destination IP can be monitored. Wireshark in this simulation had administrative viewing privileges, so that it can monitor all the traffic generated. This allowed confirmation for dropped and received packets during an attack as shown in Appendix A8.
- Finally, Wireshark is useful when it comes to calculating RTT of the data packet.

At any given instant the memory allocated and buffer for data handling is fixed for an OVS. So avertedly the measure of handling a DDOS attack falls on SDN Controller and how fast it responds and how quickly it allocates memory. So, the major contributing factors under consideration were

SDN Controller CPU consumption and DMA consumption at OVS. Both are mathematically calculated and presented in table 5.1.

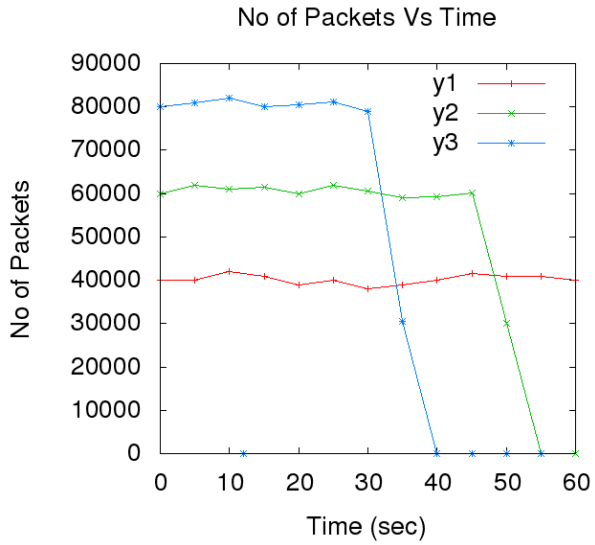

Figure 5.9 CA Deployment Results against DDOS

Figure 5.9 and table 5.2 confirm that high volume attacks with larger bandwidths have a slower reactive response to them. Confirmation of sudden attack takes time and delay in security measure response. Also, it shows the bandwidth and ICMP packets per second decrease after the enforcement of the security measures. A similar attack performed on a different OVS would act as a confirmation of isolation of attacker in entire SDN. Attack deterrence also alleviates memory and power resources of the system as indicated in table 5.1. For further network deterrence and evasive measures, more complex algorithm can be developed for the system. However, this would have to be accompanied by larger SDN resources, more efficient network solutions and advance programmability on the developer's part. The results further pave way for the following solutions.

5.4.1. Round Trip Time and Packet Trace

After successful isolation of an IP source, productive offensive measures against the attackers could be taken. Cybercrime like any other dictates mitigation at first and then tracking and locating the party involved. The use of Berkeley Packet Filter (BPF) has proven to be an effective method for packet trace as mentioned in [44]. The eBPF program traces network packets and extracts data from the header of the traced packet data at kernel space and passes extracted data to the "BPF Compilation Collection (BCC)" program again [44]. IP source is already available as the packets associated with ICMP requests require so. However, compromised sources and bot attacks allow hackers to work from the shadows, while leaving little or no traces. In most cases if not the hacker, compromised source/bot could be located and informed of the system breach.

Routing path in support of packet trace can also be aided by round trip time (RTT) calculation. This provides the geographical overview of the network being utilized and the associated core routers. Security organizations can share the information to prevent intra network breaches. RTT is used to verify the availability of network to send/receive data packets. At any given instance

RTT delay directly correlates to the available network bandwidth. RTT is a tool used to measure the strength of the DDOS attack. Increasing strength of attack would result in longer time delay for packet transfer in either direction and longer round-trip time. The concept of packet tracing is used to validate the malicious host isolation as a way of mitigation. It is back-up policy for multiple DOS attacks incase simple blacklisting and filtration is not sufficient, and an offensive security response is needed.

5.5. Chapter Summary

This chapter presented the simulation and experiment analysis that was discussed in the PoC for SDN-CA security solution. The CA was evaluated for response measure and isolation for security against multiple DOS attack sources. Evaluation analysis was made on the bases of ICMP flooding response in comparison to the expected calculated response. The test results show that higher bandwidth attacks pose a graver threat in an SDN where response measure is taken into consideration. CA deployment and security update of ODL controller allows logical isolation of the attack source. It prevents system wide failure or resource depletion and DMA ensures system wide function even during attack. It also elaborates high bandwidth likes LOIC and Hping3, which demand more rapid response and cause overhead delays if not dealt with quickly enough. The network is reconfigured and the isolation and mitigation against the attack are calculated as well. It shows that the CA deters 60-75% of the attack immediately, with high effectiveness and accuracy. Lastly it presented isolation and tracking parameters for counter, using flow table black list and RTT, respectively.

Chapter 6

Conclusion

6.1. Introduction

In this chapter we are going to discuss the overall findings of the thesis – research and potential areas for its growth and development. In addition to findings, this chapter also discusses the future related trends in relation to CA security and what betterments can be made for future solutions.

6.2. Conclusion

Cognitive Algorithm has tremendous scope and application in network management solutions and security development. CA especially benefits SDN as both represent the flexible and dynamic network deployment and control. Cybercriminals and cyber-attack techniques are evolving every day and they require more advance countermeasures to deal with. Cloud network providers often provide a complete network infrastructure solution which can be used to deploy different other solutions. SDN offers network as a solution (NaaS) and CA supports NaaS to provide a software as a solution (SaaS). The resulting network offers programmability, dynamic control and robust network.

Software Defined Network despite its many advantages provides a number of challenges, particularly in security sector. The dynamic nature and multiple plane configuration presents new and improvised security challenges towards existing networks. This thesis has prioritized and discussed security measures deployed against DDOS attacks. DDOS attacks are the most common and widely popular type of attack amongst hackers. Work in thesis illustrates how merging two technologies (DMA and Security Algorithm) under a single umbrella (CA) offers NAC, network reconfiguration, DDOS mitigation and attack isolation.

The DMA is modified by Cognitive algorithm and is at the disposal of user for alteration. This is achieved by providing software solution in application plane of SDN. The script is written in python to provide a HLPO solution for the memory resource management. This ensures proper memory allocation to the network and prevents depletion during a DDOS attack. Furthermore, CA is not limited to security and resource management. Enhanced HLPO with developing solutions through AI offer effective and dynamic network routing configurations. Such configurations have been addressed as future for many mitigation techniques, which will be discussed in the next section. This thesis discusses existing DMA and algorithm-based security solutions for SDN and DDOS mitigation. The solutions are assessed for their weaknesses and strengths to cultivate a new improved security response. The network security techniques presented in past are reviewed and carefully analyzed. After examining the security systems, their dependencies and proposed past solutions, a CA is proposed for security measure.

Within the confines of SDN architecture consisting of Application, Control and Data plane; a new mitigation technique using CA solution is proposed. Security solution using application plane software extension (Python 2.0) is proposed and developed at application layer of ODL. This is presented as a PoC where security script is developed by the user to be transferred to the Control plane using APIs. ODL controller maintains the flow rules updates and security implementation controls to the OVS. For every new threat against a DDOS attack, an alternate security rule is updated by the controller. Attack source isolation is directed by controller to the OVSs for NAC and ICMP flooding prevention. OVS flow rules for flooding prevention ensure the default isolation of outgoing flood attacks as well. This ensures security measures against compromised sources within the network. The constant feedback form the OVS maintains a valid network traffic flow management. The work done here is derived from research conducted in past which showed promising results towards the deployment of CA at OVS (data plane layer), which is overseen by Control plane for evaluation and action execution.

The proposed tools in PoC were used to set up the experiment, and the CA security solution. A DDOS attack was performed on the SDN without security measures to analyze the system network traffic. An attack scenario helped in understanding the effects of DDOS on resources such as CPU utilization, memory usage and packet loss of the user. This also formed bases for comparison with an SDN with security CA deployed. The CA showed that high bandwidth attacks require rapid response for traffic block and isolation from the source. ICMP flood with increasing bandwidth and sudden data injection demands quick reaction which otherwise compromises network resources. CA deployment resulted in isolation and data loss recovery of 65-70% of the traffic. The DMA showed the optimization of the system under LOIC attack, which prevented a system failure due to memory resource depletion. All these solutions and attack analysis are overseen by the administrator in real time, which allows him to take timely and informed decisions. All the attack attempts are recorded, and cooperating ISPs can be informed for isolation of compromised tenant IP source.

The security policy had to be enforced through a High-Level Programmable Output (HLPO) at application plane of SDN. The HLPO is written in Python in Helium SDN open platform. Helium and other Open source platform support Python extension for scripting and allow customization of code. Python is used in this thesis to develop a dynamic security algorithm to self-adjust itself based on input acquired from devices. The concept has been highlighted by many authors before which are mentioned in this thesis. This thesis has simply combined those two ideas and provided with a DDOS mitigation strategy. The idea of combining the two concepts is fairly new and can offer a lot of applications in future cloud and SDN solutions.

6.3. Future Solutions

CA and more advanced AI based solutions represent the future of SDN security. The solutions can be integrated with variety of security techniques to offer wide spectrum of attack mitigation. Proposed solution in this thesis can be integrated with Honeypot solution which offers a dummy target source for the attacker to target. This will provide security through obscurity as the identity

of real source will be isolated. This is the ironic solution to the attack as bot systems allow hackers to conceal their true identity as well and makes it difficult to track them.

Dynamic nature of SDN could also be used in IP and resource allocation. CA provides a flexible and easy to deploy network solution. This solution can be further strengthened with “Moving Target Defense (MTD)” which offers more security to the SDN. MTD offers dynamic and random IP source allocation which makes the attacker’s job of victim isolation very difficult. Algorithms are designed to be configurable and easily compatible, which makes their integration to existing and new solutions like MTD and honeypot very easy.

6.4. Chapter Summary

This chapter has given the overview of the proposed and existing solutions. It revisited the PoC and the implemented results through CA and DMA for DDOS mitigation. Obtained results were shown to be effective and, future advance solution possibilities in relation to the research are suggested and discussed.

References

[1] "1. ETSI, 'Network Functions Virtualisation (NFV): Architectural Framework,' ETSI GS NFV, vol. 2, p. V1, 2013."

[2] Z. Hu, M. Wang, X. Yan, Y. Yin, and Z. Luo, "A comprehensive security architecture for SDN," *2015 18th Int. Conf. Intell. Next Gener. Networks, ICIN 2015*, pp. 30–37, 2015.

[3] J. Boite, P. A. Nardin, F. Rebecchi, M. Bouet, and V. Conan, "Statesec: Stateful monitoring for DDoS protection in software defined networks," *2017 IEEE Conf. Netw. Softwarization Softwarization Sustain. a Hyper-Connected World en Route to 5G, NetSoft 2017*, 2017.

[4] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN security: A survey," *SDN4FNS 2013 - 2013 Work. Softw. Defin. Networks Futur. Networks Serv.*, 2013.

[5] P. Rengaraju, R. R. V, and C.-H. Lung, "Detection and Prevention of DoS attacks in Software-Defined Cloud Networks," *Dependable Secur. Comput. 2017 IEEE Conf.*, pp. 217–223, 2017.

[6] K. Machova and J. Paralic, "Basic principles of cognitive algorithms design," pp. 9–11.

[7] "https://www.opendaylight.org/."

[8] T. Wang and H. Chen, "SGuard: A lightweight SDN safe-guard architecture for DoS attacks," *China Commun.*, vol. 14, no. 6, pp. 113–125, 2017.

[9] M. S. Khan, K. Ferens, and W. Kinsner, "A chaotic measure for cognitive machine classification of distributed denial of service attacks," *Proc. 2014 IEEE 13th Int. Conf. Cogn. Informatics Cogn. Comput. ICCI*CC 2014*, pp. 100–108, 2014.

[10] N. El Moussaid, A. Toumanari, and M. El Azhari, "Security analysis as software-defined security for SDN environment," *2017 4th Int. Conf. Softw. Defin. Syst. SDS 2017*, pp. 87–92, 2017.

[11] S. Sanyal, A. Shelat, and A. Gupta, "New Frontiers of Network Security: The Threat Within," *2010 Second Vaagdevi Int. Conf. Inf. Technol. Real World Probl.*, pp. 63–66, 2010.

[12] R. Kandoi and M. Antikainen, "Denial-of-service attacks in OpenFlow SDN networks," *Proc. 2015 IFIP/IEEE Int. Symp. Integr. Netw. Manag. IM 2015*, pp. 1322–1326, 2015.

[13] L. Ran *et al.*, "The research of OpenFlow management and control interface protocols based on SDN technology," *Proc. 2015 IEEE Int. Conf. Comput. Commun. ICCC 2015*, pp. 45–49, 2016.

[14] D. Gopi, S. Cheng, and R. Huck, "Comparative Analysis of SDN and Conventional Networks using Routing Protocols," 2013.

[15] "http://searchsdn.techtarget.com."

[16] M. Suh, S. H. Park, B. Lee, and S. Yang, "Building firewall over the software-defined network controller," *Int. Conf. Adv. Commun. Technol. ICACT*, pp. 744–748, 2014.

[17] M. Kuerban, Y. Tian, Q. Yang, Y. Jia, B. Huebert, and D. Poss, "FlowSec: DOS attack mitigation strategy on SDN controller," *2016 IEEE Int. Conf. Netw. Archit. Storage, NAS 2016 - Proc.*, pp. 7–8, 2016.

[18] G. E. Shaikh and U. Shrawankar, "Dynamic Memory Allocation Technique For Virtual Machines," pp. 1–6.

[19] D. Costa and R. Matias Jr, "Characterization of Dynamic Memory Allocations in Real-World Applications: An Experimental Study," *2015 IEEE 23rd Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst.*, pp. 93–101, 2015.

[20] Y. Wang, B. Wu, and G. Edward Suh, "Secure Dynamic Memory Scheduling Against Timing Channel Attacks," *Proc. - Int. Symp. High-Performance Comput. Archit.*, pp. 301–

312, 2017.
[21] J. Wu and S. L. Sun, “A Dynamic Memory Allocation Approach for Virtualization Platforms,” *Proc. - 3rd IEEE Int. Conf. Big Data Secur. Cloud, BigDataSecurity 2017, 3rd IEEE Int. Conf. High Perform. Smart Comput. HPSC 2017 2nd IEEE Int. Conf. Intell. Data Secur.*, pp. 213–218, 2017.
[22] A. Chonka, J. Singh, and W. Zhou, “Chaos Theory Based Detection against Network Mimicking DDoS Attacks,” *Communications*, vol. 13, no. 9, pp. 717–719, 2009.
[23] Y. Chen and X. Wu, “DDoS Detection Algorithm Based on Preprocessing Network Traffic Predicted Method and Chaos Theory,” vol. 17, no. 5, pp. 1052–1054, 2013.
[24] X. Ma and Y. Chen, “DDoS detection method based on chaos analysis of network traffic entropy,” *IEEE Commun. Lett.*, vol. 18, no. 1, pp. 114–117, 2014.
[25] K. Fiedler, “How to study cognitive decision algorithms: The case of the priority heuristic,” *Judgm. Decis. Mak.*, vol. 5, no. 1, pp. 21–32, 2010.
[26] S. Shin, L. Xu, S. Hong, and G. Gu, “Enhancing Network Security through Software Defined Networking (SDN),” *2016 25th Int. Conf. Comput. Commun. Networks*, pp. 1–9, 2016.
[27] P.-C. Lin, J.-C. Liu, and P.-R. Chiou, “An event-based SDN architecture for network security analysis,” *2015 Int. Carnahan Conf. Secur. Technol.*, pp. 159–164, 2015.
[28] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “Security in Software Defined Networks: A Survey,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2317–2346, 2015.
[29] “https://www.sdxcentral.com/cloud/open-source/definitions/what-is-open-vswitch/.”
[30] H. Polat and O. Polat, “The Effects of DoS Attacks on ODL and POX SDN Controllers,” pp. 554–558, 2017.
[31] J. Zhou, H. Jiang, J. Wu, L. Wu, C. Zhu, and W. Li, “SDN-Based Application Framework for Wireless Sensor and Actor Networks,” *IEEE Access*, vol. 4, pp. 1583–1594, 2016.
[32] and V. K. Rajdeep Dua, Santosh Kumar Konduri, *Learning Docker Networking*. 2016.
[33] “https://www.sdxcentral.com/.”
[34] P. Examples and L. Mejlbro, “Introduction to probability,” *Introd. Med. Stat.*, pp. 1–61, 2002.
[35] “http://openvswitch.org/.”
[36] Y. Wang, T. C. Tai, R. Wang, S. Gobriel, J. Tseng, and J. Tsai, “Optimizing Open vSwitch to Support Millions of Flows,” *Globecom*, 2017.
[37] R. L. S. De Oliveira, C. M. Schweitzer, A. A. Shinoda, and L. R. Prete, “Using Mininet for emulation and prototyping Software-Defined Networks,” *2014 IEEE Colomb. Conf. Commun. Comput. COLCOM 2014 - Conf. Proc.*, 2014.
[38] “https://tools.ietf.org/html/rfc7047.”
[39] M. Nosrati, “Python: An appropriate language for real world programming,” *World Appl. Program.*, no. 12, pp. 110–117, 2011.
[40] “https://code.tutsplus.com/tutorials/a-beginners-guide-to-http-and-rest--net-16340.”
[41] “https://www.virtualbox.org/.”
[42] L. Liang, K. Zheng, Q. Sheng, W. Wang, R. Fu, and X. Huang, “A denial of service attack method for iot system in photovoltaic energy system,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10394 LNCS, pp. 613–622, 2017.
[43] J. D. C. Little and S. C. Graves, “Chapter 5 Little â€™ s Law,” *Oper. Manag.*, vol. 115, pp. 81–100, 2008.

[44] J. Kim, “on Multiple Network Interfaces of Linux Boxes,” pp. 324–326, 2017.
[45] Radware, “DDoS Handbook: The Ultimate Guide to Everything You Need to Know About DDoS Attacks,” pp. 1–43, 2015.
[46] S. Security, “A Closer Look at the Low Orbit Ion Cannon and Distributed Denial of Service Main Form / GUI Code IRC And Hive Mode.”
[47] L. Gallon and P. Owezarski, “Network Security and DoS Attacks 0.,” no. April, pp. 1–24, 2005.

Appendix

Appendix A: Experiment

A1. Set up & Configuration

This section describes how the tools and software mentioned in PoC are actually used to implement and deploy SDN – CA. This is to present an insight to the reader through the various installation and configuration procedures.

A2. System Requirement

The ODL platform, virtual box and supporting software were implemented on a system with following specs: Core i5, 500 Gb hard disk, 16 Gb RAM with Linux Mint 18.3 Operating System. Mininet and Wireshark were configured as a part of virtual box platform and OVS switch was deployed using LINUX – CLI shell.

A3. Open Virtual Switch Installation and Set up

OVS 2.9.90 was installed using Open source platform through CLI shell of Linux. It was installed using the following command:

```
sudo apt-get install openvswitch-switch
```

The switch was given a port connection and assigned a DHCP IP so that all the traffic is routed through the OVS. Command used for this purpose in order are:

Commands
1. ovs-vsctl add-br mylink 2. ifconfig mylink up 3. ovs-vsctl add-port mylink enlpo2 4. ifconfig enlpo2 5. dhclient mylink 6. ip tuntap add mode tap vport1 %for each port% 7. ifconfig vport1 up 8. ovs-vsctl add-port mylink vport1 %for each port%

These commands initialize and establish an OVS switch with four tuntap ports which can be later connected to the Virtual box. The bridge is remote access and automatically configured in tree topology of the virtual box. Figures below show the OVS set up of the experiment.

```
cypher@cypher-Satellite-Pro-L850 ~ $ sudo ovs-vsctl show
[sudo] password for cypher:
1b5cf449-8bcd-4329-9157-840769f13835
    Bridge mylink
        Port "vport1"
            Interface "vport1"
        Port "vport4"
            Interface "vport4"
        Port "vport2"
            Interface "vport2"
        Port "enp0s20u1"
            Interface "enp0s20u1"
        Port "vport3"
            Interface "vport3"
        Port mylink
            Interface mylink
                type: internal
    ovs_version: "2.5.4"
cypher@cypher-Satellite-Pro-L850 ~ $
```

Figure 7.1 OVS Port Connections

```
mylink    Link encap:Ethernet  HWaddr 0e:6d:1d:0f:4e:42
          inet6 addr: fe80::c6d:1dff:fe0f:4e42/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24374 errors:0 dropped:0 overruns:0 frame:0
          TX packets:18967 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:14562273 (14.5 MB)  TX bytes:2504691 (2.5 MB)

vport1    Link encap:Ethernet  HWaddr 3e:cd:e4:42:2a:5d
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:7993 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

vport2    Link encap:Ethernet  HWaddr ca:4f:16:ec:68:7e
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:7294 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

vport3    Link encap:Ethernet  HWaddr 22:22:e2:bd:34:cd
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:7941 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

vport4    Link encap:Ethernet  HWaddr 42:9d:63:cf:61:a3
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:11665 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlp2s0    Link encap:Ethernet  HWaddr 44:6d:57:ea:49:a7
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figure 7.2 OVS Virtual Tun Tap Ports and IPs

A4. Virtual Box Set Up

Linux platform support both Virtual box and VM Ware. For this experiment, Oracle Virtual Box 5.1 was used which is shown in figure below. Three independent Virtual Machines are initiated using bridged adapter, which are connected to tuntap ports of OVS. The fourth one is kept idle in case needed for monitoring or other such purposes. The network is configured in VM so that subnetworks or switches withinthe VM cannot send traffic directly to each other. This is to ensure unified traffic flow through OVS for filtration.

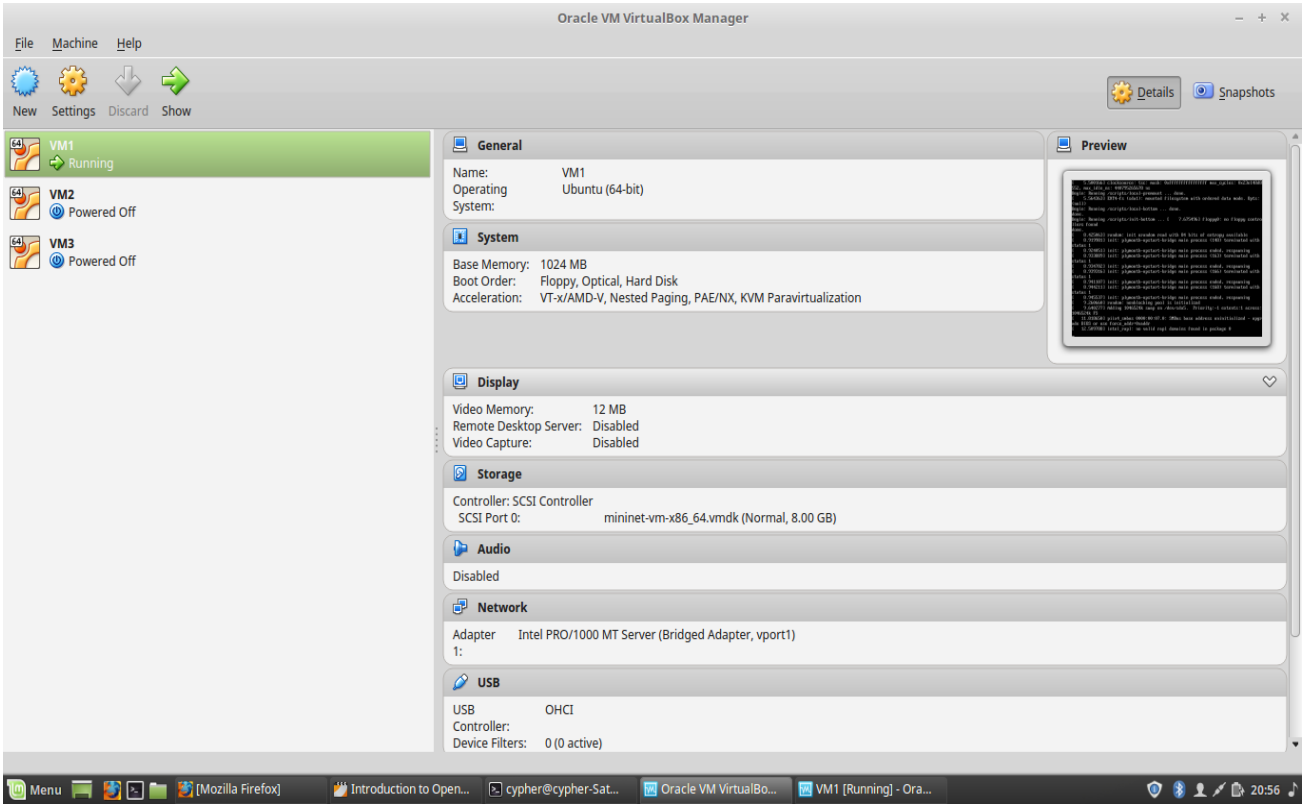

Figure 7.3 Virtual Box Set Up

A5. Mininet Network Set up

A Mininet OVF image file was downloaded using mininet.org. The file acted as the OS for Vitual Box platform for the network topology to be deployed. Network topology was deployed using the following commands:

```
sudo mn --switch ovs --controller remote --topo tree,depth=2,fanout=2
sudo mn --switch ovs --controller port 192.168.49.322:6634
```

The network set up is shown below in the following figures. These figures represent the attack source, their IPs and the target IP.

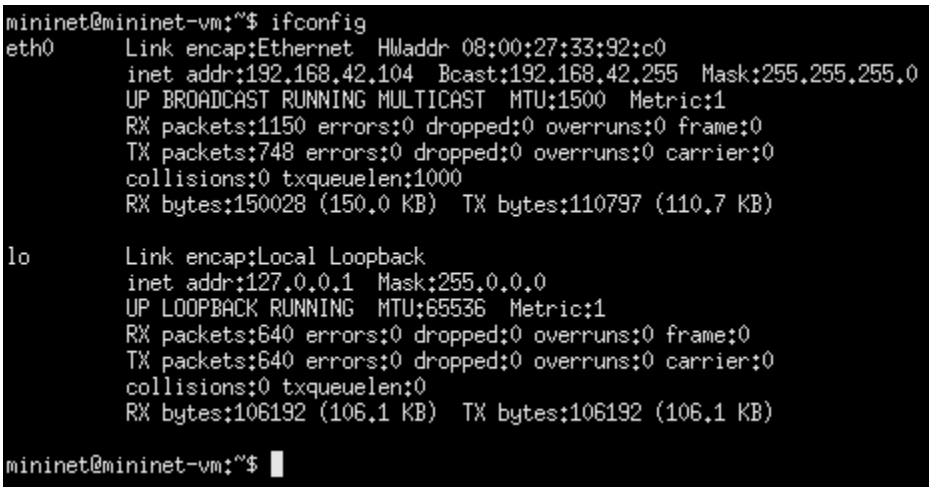

Figure 7.4 Mininet (Xterm) Malicious Source Configurations 1

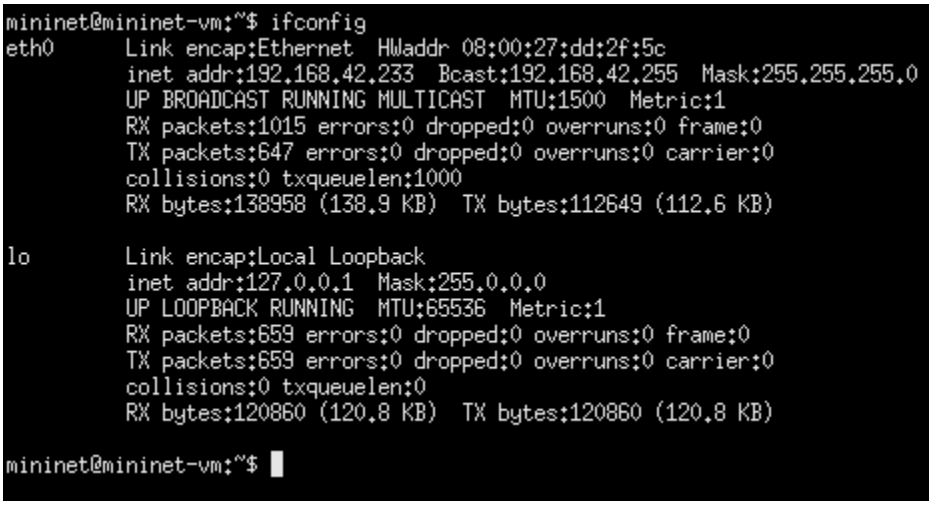

Figure 7.5 Mininet (Xterm) Malicious Source Configurations 2

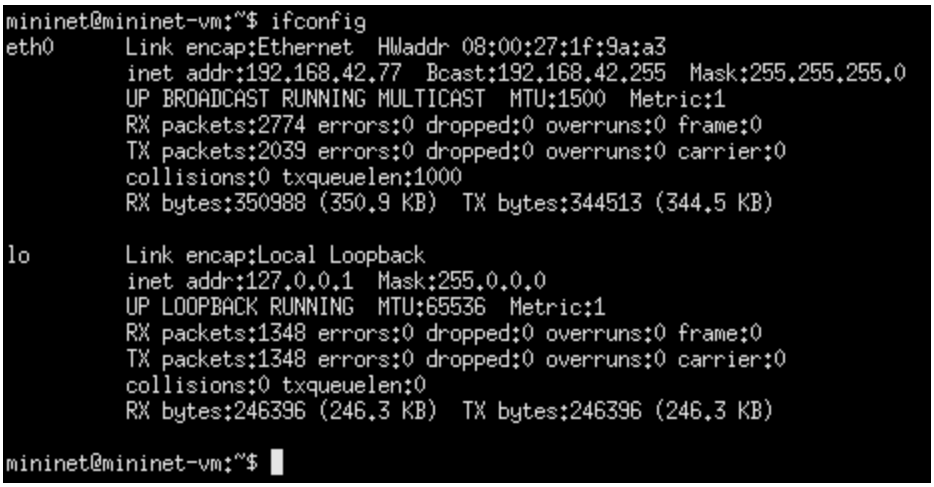

Figure 7.6 Mininet (Xterm) Malicious Source Configurations 3

A6. Opendaylight

Opendaylight is installed using the same way as OVS through CLI shell. On Linux OS, Java and other dependencies needed to be installed first.

```
# sudo apt-get install openjdk-7-jdk

# sudo add-apt-repository ppa:webupd8team/java -y

# sudo apt-get update

# sudo apt-get install oracle-java8-installer

# sudo mkdir -p /usr/local/apache-maven


# sudo service openvswitch-controller stop

# sudo service openvswitch-switch stop

# cd distributions/karaf/target/assembly/bin

# ./karaf -of13
```

Java, mininet and other such dependencies are necessary for the working of ODL controller. After the installation, ODL controller is set up and remote IP is assigned to it established by mininet in the Virtual Machine. Also, OVS port and IP are configured using APIs to direct traffic between OVS and ODL controller. The network set up is shown below in the figure.

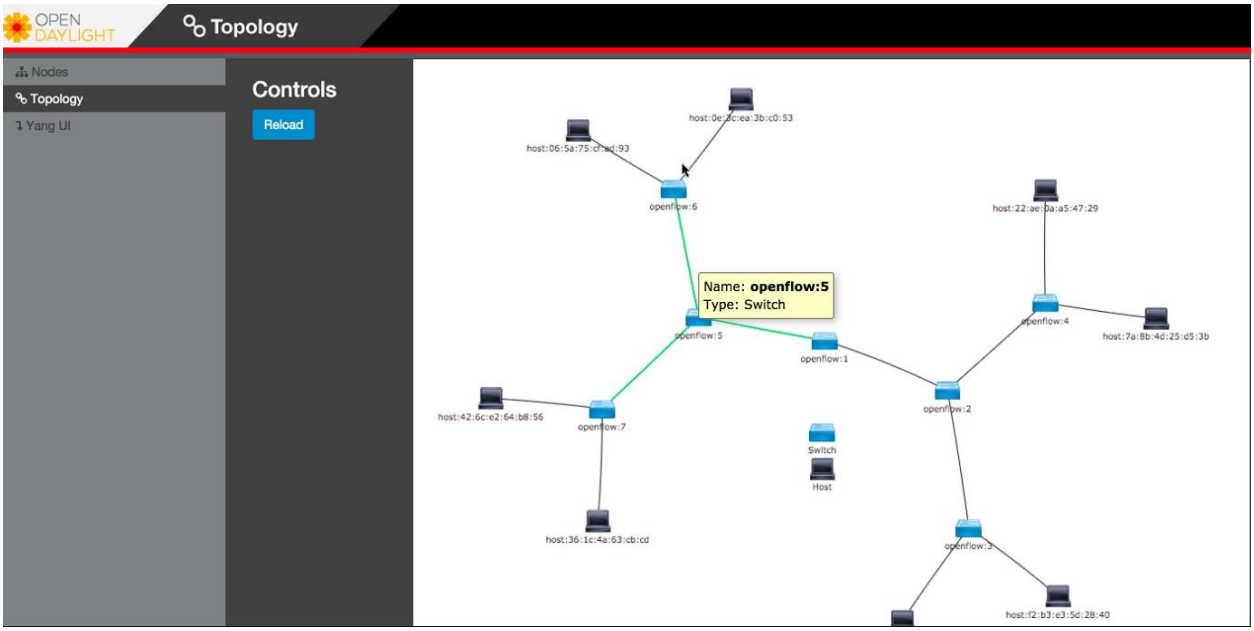

Figure 7.7 Opendaylight Network Topology

A7. Python 2.0

A python 2.0 extension in ODL is used for the development and deployment of the algorithm. The algorithm is configured in python script syntax which is converted into flow rules by ODL application platform REST API. Then, it is transferred to ODL Controller where it is managed and deployed at OVS.

A8. Wireshark

Wireshark is used in this thesis to monitor ICMP packet activity and to measure RTT of the packets. It is also used to measure the response of CA against DDOS attacks. Origin and destination of the packets are captured in Wireshark which are shown below in the figure:

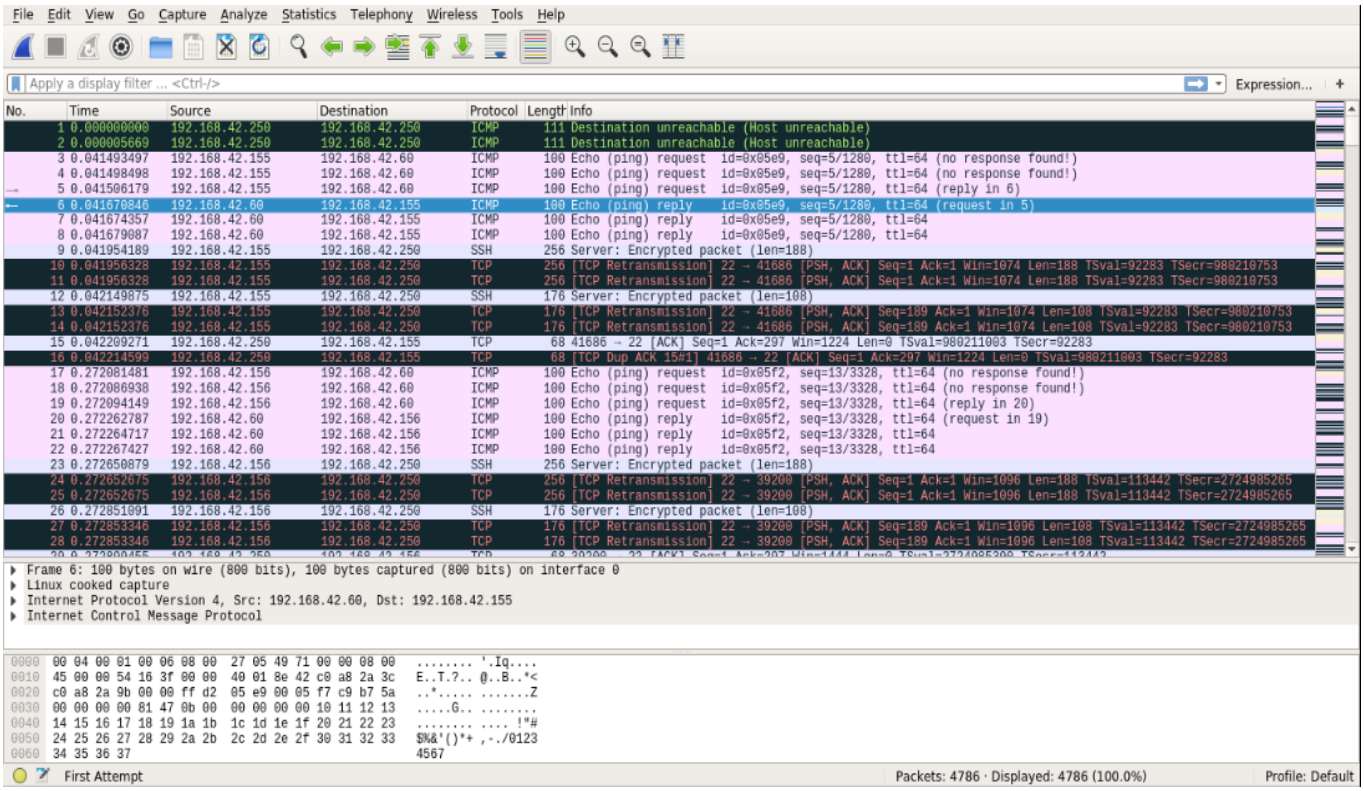

Figure 7.8 Wireshark Results and RTT for ICMP packet