International Journal of Advances in Applied Sciences (IJAAS)
Vol. 8, No. 3, September 2019, pp. 232~242
ISSN: 2252-8814, DOI: 10.11591/ijaas.v8.i3.pp232-242

Performance analysis of security framework for software defined network architectures

D. Arivudainambi, K. A. Varun Kumar
Department of Mathematics, Anna University, India

Article Info

Article history:

Received Apr 29, 2019
Revised Jul 26, 2019
Accepted Aug 14, 2019

Keywords:

Bandwidth
Denial of service attack
Software defined network
Traffic inspection
Virtualization

ABSTRACT

Software defined data centers (SDDC) and software defined networking (SDN) are two emerging areas in the field of cloud data centers. SDN based centrally controlled services takes a global view of the entire cloud infrastructure between SDDC and SDN, whereas Network Function Virtualization (NFV) is widely used for providing virtual networking between host and Internet Service Providers (ISP's). Some Application as a Service used in NFV data centers have a wide range in building security services like Virtual firewalls, Intrusion Detection System (IDS), load balancing, bandwidth allocation and management. In this paper, a novel security framework is proposed to combat SDDC and SDN based on NFV security features. The proposed framework consists of a Virtual firewall and an efficient bandwidth manager to handle multiple heterogeneous application requests from different ISPs. Real time data were taken from an experiment for a week and A new simulation based proof of concept is admitted in this paper for validation of the proposed framework which was deployed in real time SDNs using Mininet and POX controller.

Corresponding Author:

K. A. Varun Kumar,
Department of Mathematics,
Anna University,
Chennai, Tamil Nadu, 600025, India.
Email: kavaruncse@gmail.com

1. INTRODUCTION

Present day network infrastructure is fully loaded with network devices such as router, switches etc. Most of these devices constitute dedicated hardware and require manual configuration. Such configurations can lead enterprises to failure during provisioning and de-provisioning. Change management is a time consuming task that requires halting of adequate resources to make a change in a single hardware (network devices). Advanced technologies such as Cloudification of things, Internet of Things etc., force the networking organization under high pressure to be more efficient. As noted, the use of traditional Application Specific Integrated Circuits (ASIC) does not have the scalability to handle these technologies. Switching between traditional hardware centric data network (ASIC) is very resilient and time consuming. Hence, for overcoming the above mentioned issues in the traditional ASIC based network hardware, a software defined controller specific network architecture called Software Defined Network and Network Virtualization has been designed. Software Defined Network is a dynamic updation framework which controls and manages network devices, network related services using high level languages and application program interfaces. SDN architecture is a centralized platform that can economize on the Information Technology (IT) infrastructural cost. Hence SDN supports different kinds of network component for better flexibility and agility. In traditional networking device, a packet that arrive at the switches has some proprietary rules that takes the packet forwarded to its assigned destination. Switches treat all the packets arriving in the same allotted way. SDN has certain rules defined programmatically in the controller for packet handling. Network

administrator has complete control over the switches. A dynamic change of rules is possible where traditional network devices do not have provision for dynamic change management. SDN has an additional support for packet Prioritizing, deprioritizing and restriction which gives granular level of control to the administrators. This could help administrators taking care of the traffic related issues in the network and to manage the traffic loads in a flexible manner. The SDN Security vector architecture is divided in to four sections namely southbound, northbound, eastbound and westbound as shown in Figure 1.

Southbound Section is popular for its interaction between controllers and switches. Open Flow protocol is a southbound interface to the controller representing the bridge which connects the controller and forward plane such as switches. SDN architecture has an open standard non - vendor protocol which allows all the vendors to use of an open architecture. Hypervisor plays the key role in the present day SDN architecture which gaps the control between the controller and the protocols both Southbound and Northbound.

Northbound Section is the most critical API in the SDN environment. A majority of networking components exist in this section. Pyretic and frenetic is a SDN specific policy program language which communicates with controllers in the northbound section. All the security applications such as virtualized firewalls, intrusion detection system have a common API for interacting with the controller. Some hacking methods may compromise API's which cause intentional risks to SDN because of their API's communication policy.

East and West Section: Management of SDN architecture is done by east and west bound sections. The management plane is controlled by the distributed architecture in which instructions through the controller for managing data. The distributed architecture has some important functionalities including control, management, monitor and task distribution for different low level instances. This section has also to support multiple controllers where they can share the same tasks and nodes.

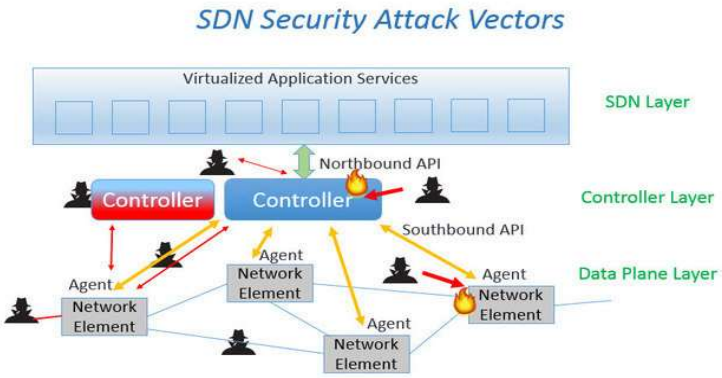

Figure 1. Software defined network security attack vectors

A new paradigm architecture is introduced in SDN for attaining a centralized management of the entire network that may also offered potential risk on network security problems like DDOS attack. Attacks attempt at making a service unavailable to their legitimate users by tiring the network resources. DDoS attack is witnessing a rapid growth in frequency despite various solution found by researchers. Some effort in network security research community has been made to give the appearance as the most effective for data in potential issues or opportunities to observe DDoS attacks with in the new enterprise network setting that adopts SDN.

Implementation of SDN could provide defense against the DDoS attacks in many cases. But it may have certain challenges too (i) Slow packet forwarding (ii) Dynamic network topology. The data plane in SDN normally forwards packets on the basis of the policies assigned by the control programs. In traditional network, Packet forwarding is done by a piece of hardware. But, in contrast, it is done by software in the SDN. So there is a delay in the forwarding of packets due to traffic overhead and network delay in communication between control programs. Migration of virtual machines plays a vital role in SDN. The main role of virtualizing network is the allocation of the VM's to their clients, which needs to updation of the entire network topology after the occurrence of every migration. So, it needs Dynamic network topology for its operational model. But it becomes highly vulnerable. DDoS attacks can be performed easily because, In SDN, separation of control plane and data plane turns out to be a major considering that data plane usually

asks the control plane to obtain a flow rule. When the data plane receives a new packet, it does not normally know how to handle it. This is a key feature which helps attackers to strike the target in the sdn with flooding of request. This leads to DDoS attack. So, the performance of the sdn network and incoming traffic is analyzed to search to discover any abnormal behoviorial changes.

The rest of this paper is organized as follows. Related works have been studied and compared with the work in Section II. Analysis of the effect of SDN on DDOS attack through motivation towards this work has been explained in section III. Section IV elaborates on the proposed security framework and its main components. A description of the operations of security framework is provided. In other words the experimental setup and evaluation of the framework in terms of scalability and performance is done. Section VI provides the conclusion.

2. RELATED WORK

Zhou et al [1] has proposed a method to detect compromised SDN devices by introducing Backup controllers to the network by recognizing the inconsistent behavior of Primary controllers and Switches when they are trustless. A backup controller plays an auditor role where the information of any state is recorded. Varadharajan et al [2] has constructed an architechture for securing end-to-end services across multiple SDN Domains by the policy language where security policies are used to control the flow of information. It also defines the path and flow based security policies which are significant for the services. It demonstrates intra and inter domain communications with multiple SDN Domains. Fawcett et al [3] has delivered the concept of virtualization in SDN for effective detection and protection. TENNISON a novel distributed SDN Security framework which is capable of monitoring, rapid detection and remediation and creates an insight into the multiple controllers of the network attack. Liu et al [4] has designed a SDN-based data transfer security model Middlebox-Guard (M-G) which reduces network latency and manages the dataflow of the network to enusure it run safely based on Intergral Linear Program (ILP) algorithm to tackle switch volume constraints. Kalkan et al [5] has described and evaluated joint entropy-based security scheme (JESS) to enhance security against DDos Attacks where it detects and mitigates these kind of hazards. Since it relies on a statistical model, it mitigates not only known attacks but also unfamiliar attacks with efficient manner. Bing et.al. [6] have presented a DDoS attack mitigation system for both cloud and SDN computing where attack detection is done with the help of high programmable network monitoring. They also have proposed a graphical model that deals with the data shift problem for a new network paradigm. Shin et.al. [7] have provided a new technique to fingerprint the SDN network and launch the DDoS attack using the resources of the entire network or part thereof in the SDN. They have introduced the SDN scanner and feasible defense mechanism techniques. Chun et.al. [8] Presented an attack mitigation system for virtual machines. First, they started with the generation of an attack graph analytical model to enable counter measure selection and vulnerability detection. Their proposed frameworks significantly improve the effectiveness of attack detection. Kim et.al. [9] have focus on issues in network management. They have identified three major problems, namely dynamic network changes, support providing for network configurations in high level languages, control over performing network diagnosis and troubleshooting. Their prototype working in SDHN and Campus network states points out to the numerous advantages over deploying a SDN rather than the traditional network. Wang Yang et.al [10] have presented an Open Scass: An open chain as a service platform, in which they have integrated the Software Defined Network and Network function virtualization for the enforcement of a service chaining policy, which enables the improvement of the scalability, auto provisioning by integrating SDN and NFV. Banikazemi et.al. [11] designed a meridian framework which supports dynamic updates to virtual network and organization of different task on a large number of devices, irrespective of whether they concentrate on automatic controller of network devices through application Interface with the help of SDN. It can be integrated with many cloud controllers. Wang et.al. [12] have applied SDN technology to the cloud center to propose a dynamic load balancing of the cloud center. SDN monitors the network traffic, task scheduling and balances the load condition of the network over a long period of time and increases the throughput. Richard et.al. [13] Proposed a Unified Server Resource Management for Information and Communication technology (ICT), It is a Dynamic method for allocation of VM's to cloud providers and their clients. This method can reduce cost by migrating 50% of virtual machines. Yan et.al. [14] have discussed the impact of DDoS attacks over SDN in cloud computing. They have discussed the implications of features in SDN like centralized controller, dynamic rule updation, software controlled traffic analyzer that play a vital role in defence against DDoS attacks in SDN. Munoz et.al. [15] present orchestration of both SDN and NFV for abstraction of virtualizing the tenant network. On the one hand, SDN slices the physical network into multiples of virtual network and on the other, NFV drives all hardware devices to virtualize. Then the virtualization of SDN controllers update their tenant instances to work independently with Individual controller that connects within a second.

Li et.al. [16] have discussed a controller minimization problem in SDN. They state that distributed controller allocation is more efficient than a general controller that is allocated to each VM. They have proved the presence of better security for VM's through a distributed controller compared to a general controller. Izzat et.al. [17] have provided a widespread study on SDN security. They discuss security threats and their effects in SDN i.e., Ip Spoofing, Data Tampering, Refutation, Data Leakage, Denial of Service attacks, and Privilege Escalation. The authors have also surveyed different SDN security controls, such as intrusion detection systems/intrusion prevention system, firewalls, access control, network assessment, and policy management. They part out to several pathways of SDN evolving. Fei et.al. [18] made a complete study of the significant areas in SDN/OpenFlow operation, including the basic concepts, applications, virtualization, controller, language abstraction, security, Quality of Service (QoS), as well as its incorporation into wireless and optical networks. They discuss the advantages and disadvantages of different schemes, and discuss the future research developments in this area. They have expressed their optimism of the survey assistance to both industry and academia and Research and Development (R&D) people to realize the latest development of SDN/OpenFlow designs.

Shin et.al. [19] have presented different vulnerabilities of SDN. The attackers could mount an inundation attack on the SDN controller with the goal of weakening the entire SDN. Therefore, the controller must have a suitable architecture to survive such an attack while the current operation is in progress. Scheduling based architecture is designed for the SDN controller that hints to effective attack quarantine and network defense during denial of service attacks.

3. MOTIVATION

SDN is divided into three main efficient layers, namely organization layer, control layer, and application layer, as shown in Figure 1. The possibility of deadly DDoS attacks thrown on these three layers of architecture does exist. Based on the potential targets, the DDoS attacks thrown on SDN can be categorized, namely control layer DDoS attacks, application layer DDoS attacks and infrastructure layer DDoS attacks.

3.1. Control layer DDoS attacks

These controllers could possibly be seen as Single Point of Failure hazards for the network. So, they are a predominantly striking target for DDoS attack in the SDN design. The following approaches can launch flooding of control plane DDoS attacks: which can affect all the API's. In a real time scenario, many contradictory flow rules from different requests may cause DDoS attacks on the control layer. Inside the operation of SDN, data plane normally make a request to control plane to follow flow rules when the data plane gets new network packets which is not known to the controller that how to handle. There are two possibilities for the managing a new flow when flow match does not exist in the flow table: either the entire packet or a portion of the packet header is communicated to the controller to give solution to the query [20]. With a huge capacity of network traffic, transmitting the complete packet to the controller would lodge high bandwidth [21].

3.2. Application layer DDoS attacks

DDoS attacks in application layer are of two types, first one is to attack some application within the SDN and (b) to attack north bound API. There is no clear separation of API's and resources used in the SDN [22]. Hence DDoS attacks launched over the particular application in SDN can affect all other applications in SDN.

3.3. Infrastructure layer DDoS attacks

There are two possible types of data plane DDoS attacks. The first is to attack some switches, once the switches get compromised, automatically all the resources in the SDN architecture become vulnerable. The other type is to attack south bound. For example, the packets must be stored in node memory and only header information of packets is transmitted to the controller until the flow table entry is returned. In this scenario, the attacker has an option to execute a DDoS attack on the node by recognizing the number of new and unknown flows.

To reveal the viability of DDoS attacks, a scanning tool named SDN scanner has been introduced in to clearly exhibit the network that set up SDN. This technique can be easily performed by adjusting current network scanning tools (e.g., ICMP scanning and TCP SYN scanning). The attack can be directed to SDN network by a distant attacker, and can suggestively reduce the performance of an SDN network without needful high performance or high capacity devices [23]. One clear case of DDoS that can directly disturb the controller is flooding the controller with header information packets (new packets). Mostly a new packet that

does not have the match in the flow table would be sent to the controller for processing. Most DDoS attacks use unidentified source addresses obtained through the spoofing technique, which interprets new incoming packets at the switch. There is also main drawback seen when the number of new incoming packets is larger than the secure channel's bandwidth and reduces the controller's handling power. In DDoS attacks, flooding of packets are directed to a host or a group of hosts in a network. If the source addresses of the incoming packets are unidentified, the switch does not find a match and has to advance the packet to the controller. Hence for addressing the problem referred to above efficient Software defined Intrusion Detection System is desirable.

4. PROPOSED FRAMEWORK FOR COMBATTING DDOS IN SDN

In general, data plane plays the role of handling and forwarding data packets to destination & Control plane is a key player in handling the logical functionalities of router firmware such as load balancing, virtualization and firewall. This design in firmware decelerates the performance of hardware and leads to a major time consuming process for solving the problem referred to above, horizontal integration called Software Defined Network has been introduced. The main property of SDN is a controller based dedicated software architecture which separates the control plane from the data plane. However, this central software management is highly prone to cyber-attacks due to its single umbrella property of control and handles components. Here, a novel security framework stated in Figure 4 is proposed to provide security by combining virtual firewall and bandwidth analyzer & Ford-Fulkerson algorithm [24] is applied to obtain the maximum flow between two access points in the network. The algorithm is as follows.

Let G be a Graph with vertex and Edges G (V,E) and for each edge from each vertex, the flow be f(u,v) and capacity of the network be c(u,v).

a. Capacity

The flow along an edge cannot exceed its capacity as stated

f (u, v) ≤ c (u, v), (u, v) ϵ E

b. Skew symmetry

The net flow from u to v must be the opposite of the net flow from v to u

(u,v) ϵ E f(u,v) = - f(u,v)

c. Flow conservation

The net flow to a node is zero, except for the source, which "produces" flow, and the sink, which "consumes" flow.

u ϵ V : u ≠ s and u ≠ t => $\sum_{w \in V} f(u,w) = 0$

d. Value

The flow leaving from s must be equal to the flow arriving at t

$\sum_{(s,u) \epsilon E} f(s,u) = \sum_{(v,t) \epsilon E} f(v,t)$

Inputs Given a Network G = (V, E) with flow capacity c,a source node s,and a sink node t
Output Compute a flow ∫ from s to t of maximum value

f (u,v) ← 0 for all edges (u,v)
While there is a path p from s to t in Gf , such that cf(u,v) > 0 for all edges (u,v) Ɛ p:
Find cf (p) = min { cf(u,v):(u,v) Ɛ p}
For each edge (u,v) Ɛ p
f (u.v)←∫(u,v) + cf(p) (send flow along the path)
f (u,v)←∫(u,v) – cf(p)(The flow might be returned later)

For each pair (u,v) access points the path which gives the maximum flow has a entropy tag $E_{u,v}$
Define E = MAX $E_{u,v}$

(u, v) Ɛ p

Fixing a threshold 0.5, if E > 0.5 then the corresponding path is suspicious. The algorithm is applied to maintain each flow within the network. The below mentioned tuple table is maintained after each of algorithm execution.

4.1. Virtual firewall

Controller is one among the control components responsible for transforming Open flow switch (high level) logics into IP table rules. Firewall as shown in Figure 2 implemented in the security framework is a stateful firewall where it monitors all the active connections and performs both ingress and egress filtering on the basis of the rules defined in IP tables. The deployed firewall stated in Figure 3 can distinguish between the data packets coming from legitimate source or not. A sample set can be stated as follows Rule set1. ufw allow protocol tcp from IPTable_allow_list to any port 8008.

4.2. Bandwidth analyzer

Attacks performed on top of layer 3 and layer 4 focus on bandwidth and resource utilization which leads to compromise in performance and security. Most of the attacks launched over are volumetric and stealthier target specific. Hence an optimized bandwidth analyzer is required for monitoring the bandwidth periodically or intervals. The bandwidth analyzer consists of a significant module called Traffic Inspection Model (TIM). TIM helps incisive inspection of the packet and extracts possible information from the packets.

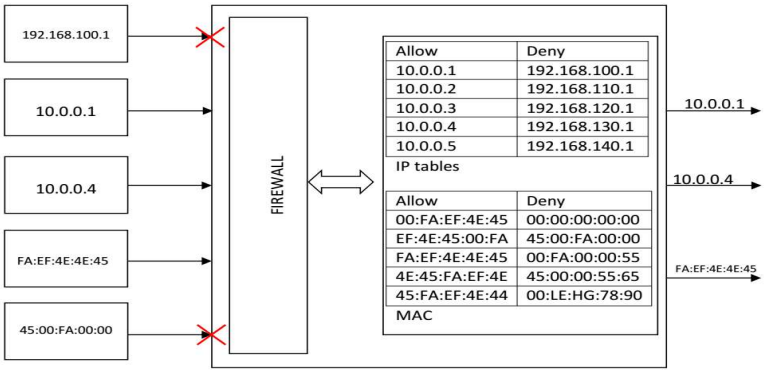

Figure 2. Proposed virtual firewall architecture

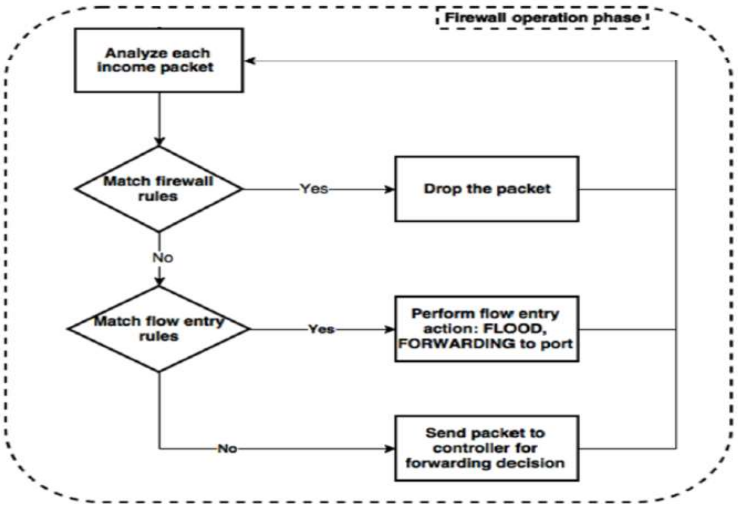

Figure 3. Firewall operation based on rule set

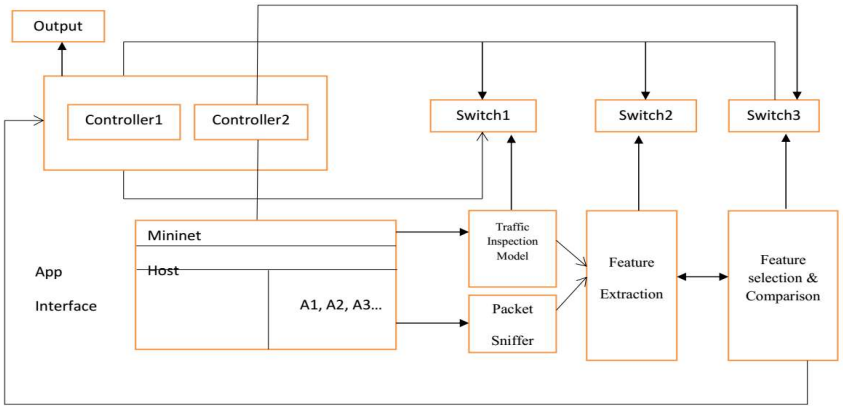

Figure 4. Architecture diagram of proposed SDN security framework

ALGORITHM

Controller

Define ports.
Define host policies (host name, pid.uid);
For each Switches S in network N do
Monitor-enable==Controller 1||Controller 2
If traffic –generating host==INTRA_HOST
Monitor-enable==Controller 1
Else if Traffic –generating host==INTER_HOST
Monitor-enable==Controller 2
For each host do
Inspect-packet==stateful
host.Traffic.match (true);
host.Uid.match (true);
host.Uid.match (true);
end
end
end

Detection

For each host in the network do
For each Switch in the network do
Packet.thershold==50
Packet.host==host.getUID, get NID
Packet. Network==get. Network by Switch (SID)
If packet. Generating! =host.getUID,get(host ID)
flag ("Malicious host")
elseIf (Packet.host exceeds threshold)
flag ("DDOS possible")
flag(host.getUID())
end
end
end

5. EXPERIMENTAL SETUP

The proposed security framework is tested and implemented in real time test bed. The proposed framework is developed with the complete use of python and deployed in a parrot operating system. SDN configuration as stated in Table 1 is implemented in the parrot Operating System (OS) using Mininet.

Table 1. Configuration of host, controller and link state

S.No	Host	Host address	Host MAC	Link state	Controller1 Ryu	Port	Controller2 Ryu	Port
1	AAh1	10.1.1.1	0A:0A:00:00:00:01	sAA	cA	6633	cB	6634
2	AAh2	10.1.1.2	0A:0A:00:00:00:02	sAA	cA	6633	cB	6634
3	ABh1	10.1.2.1	0A:0B:00:00:00:01	sAB	cA	6633	cB	6634
4	ABh2	10.1.2.2	0A:0B:00:00:00:02	sAB	cA	6633	cB	6634
5	BAh1	10.10.10.1	0A:0B:0A:00:00:01	sBA	cA	6633	cB	6634
6	BAh2	10.10.10.2	0A:0B:0A:00:00:02	sBA	cA	6633	cB	6634
7	BBh1	10.10.20.1	0A:0B:0B:00:00:01	sBB	cA	6633	cB	6634
8	BBh2	10.10.20.2	0A:0B:0B:00:00:02	sBB	cA	6633	cB	6634

5.1. Experimentation

Initially Mininet was executed to get virtual network with virtual hosts, virtual switches, virtual controllers, virtual links etc. The controller with proposed security framework was developed in python and deployed as individual component. A new topology with ovsf switch setup was created with limited number of switches and its associated hosts and links. Mininet was executed using custom topology with parameters. Open flow switch support was always enabled and ryu controller component was used for managing and handling the switch. A clear view on basic SDN experimental configuration is stated in Table 2. For example S is the switch, source host is represented as BBh1, Destination host as ABh1, attack type as SNMP, protocol as TCP. Packet captured in normal scenario is 6435 and the attack scenario is 1074.

5.2. Packet capture

TCP dump utility is installed and all the data packets moving in and around all interfaces were tapped and recorded into .pcap file [25]. These captured data helped extraction of the packets that deviated from the normal packets. The captured packets along with statistics (Normal and attack) are detailed in Table 2. Switch S22 was monitored for validating the experiments and they were considered for the attack scenario. Figure 5 shows the packet flow rate within the switch S22. The arrival of packet rate was estimated for every 15 minutes interval. In Figure 4, dark lines clearly show that number of requests is very high while comparing normal and attack scenario.

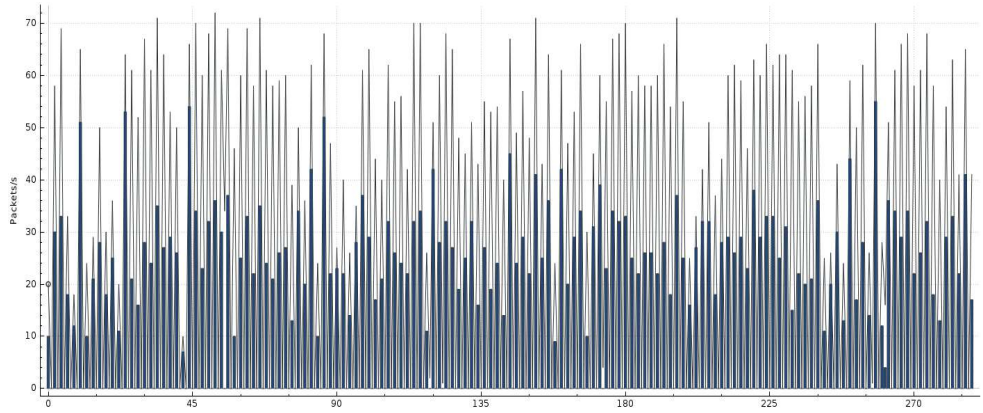

Figure 5. Packet flow rate within the switch S22

Table 2. Switch, host configuration, protocol and attack type of experimental setup

S.No	Number of Switch	Source Host	Destination Host	Attack Type DDoS	Protocol	Packet captured	
						Normal Scenario	Attack Scenario
1	S1,S2,S11,S12,S21,S22	BBh1	AAh1	SNMP	TCP	6435	1076
2	S1,S2,S11,S12,S21,S22	BBh1	AAh2	Land	TCP	6435	1076
3	S1,S2,S11,S12,S21,S22	BAh1	AAh1	Flood	Openflow	17000	15002
4	S1,S2,S11,S12,S21,S22	BAh1	AAh2	UDP flood	Openflow	17000	15002
5	S1,S2,S11,S12,S21,S22	ABh1	AAh1	Hping3	Openflow	17000	15002
6	S1,S2,S11,S12,S21,S22	ABh1	AAh2	POD	Openflow	17000	15002
7	S1,S2,S11,S12,S21,S22	ABh2	AAh1	Smurf	TCP	6435	1076
8	S1,S2,S11,S12,S21,S22	ABh2	AAh2	DR-DoS	TCP	6435	1076
9	S1,S2,S11,S12,S21,S22	BBh1	ABh1	SNMP	TCP	6435	1076
10	S1,S2,S11,S12,S21,S22	BBh1	ABh2	Land	TCP	6435	1076
11	S1,S2,S11,S12,S21,S22	BAh1	ABh1	Flood	Openflow	17000	15002
12	S1,S2,S11,S12,S21,S22	BAh1	ABh2	UDP flood	Openflow	17000	15002
13	S1,S2,S11,S12,S21,S22	ABh1	ABh1	Hping3	Openflow	17000	15002
14	S1,S2,S11,S12,S21,S22	ABh1	ABh2	POD	Openflow	17000	15002
15	S1,S2,S11,S12,S21,S22	ABh2	ABh1	Smurf	TCP	6435	1076
16	S1,S2,S11,S12,S21,S22	ABh2	ABh2	DR-DoS	TCP	6435	1076
17	S1,S2,S11,S12,S21,S22	BBh1	BAh1	SNMP	TCP	6435	1076
18	S1,S2,S11,S12,S21,S22	BBh1	BAh2	Land	TCP	6435	1076
19	S1,S2,S11,S12,S21,S22	BAh1	BAh1	Flood	Openflow	17000	15002
20	S1,S2,S11,S12,S21,S22	BAh1	BAh2	UDP flood	Openflow	17000	15002
21	S1,S2,S11,S12,S21,S22	ABh1	BAh1	Hping3	Openflow	17000	15002
22	S1,S2,S11,S12,S21,S22	ABh1	BAh2	POD	Openflow	17000	15002
23	S1,S2,S11,S12,S21,S22	ABh2	BAh1	Smurf	TCP	6435	1076
24	S1,S2,S11,S12,S21,S22	ABh2	BAh2	DR-DoS	TCP	6435	1076
25	S1,S2,S11,S12,S21,S22	BBh1	BBh1	SNMP	TCP	6435	1076
26	S1,S2,S11,S12,S21,S22	BBh1	BBh2	Land	TCP	6435	1076
27	S1,S2,S11,S12,S21,S22	BAh1	BBh1	Flood	Openflow	17000	15002
28	S1,S2,S11,S12,S21,S22	BAh1	BBh2	UDP flood	Openflow	17000	15002
29	S1,S2,S11,S12,S21,S22	ABh1	BBh1	Hping3	Openflow	17000	15002
30	S1,S2,S11,S12,S21,S22	ABh1	BBh2	POD	Openflow	17000	15002
31	S1,S2,S11,S12,S21,S22	ABh2	BBh1	Smurf	TCP	6435	1076
32	S1,S2,S11,S12,S21,S22	ABh2	BBh2	DR-DoS	TCP	6435	1076

5.3. Attack scenario

Once the topology was created using mininet, interfaces other than eth0 were monitored and recorded. Interfaces other than eth0 and interface start S1-eth1....Sn-ethn belongs to software defined switches and hosts associated with the switches and its links were recorded using tcpdump utility. Any interface which was up in the control plane was taken with unlimited requests to send data packets to an individual switch or its associated hosts. This led to the performance of DDoS over a specific host within the connected switch. From Figure 6, red spline which ranges from 15 – 105 packets per second states the general packet flow handled by the controller from switch S22. Blue explains the packet flow during DDoS detection which in turn clearly shows that the flow of packet rate is completely reduced and falls between 10-15 packets per second.

5.4. Firewall

The controller component consists of a major module firewall. Firewall is used for recording the internal host address (permanent and logical). Rule set is defined to the firewall for inspecting and detecting malicious traffic. Rule set is based on the behavior features of hosts and networks.

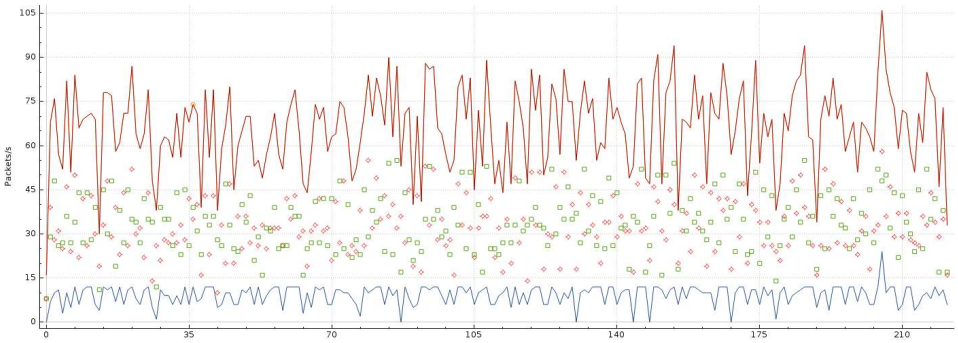

Figure 6. Detection rate vs. packet flow rate within switches

5.5. Training phase

The Experimentation is carried out and data transfer, routing, packet forwarding etc were performed within the hosts. This experimentation was carried out for a week with balanced payload and transfer. The recorded traffic was taken into consideration and processed for extracting the possible features from individual interfaces, connected switches, associated hosts, links. These extracted features were tabulated into a system with network level features and used for further classification. A final entropy was calculated and links with defined IP address and MAC address were registered in the IP tables (firewall rule table).

5.6. Testing phase

In testing phase, the attack module written in python was executed for creating suspicious traffic from other interfaces. ie. Host H1 associated to Switch S2 created the attack payload over H2 host associated with switch S3. Once the attack module got executed, all the traffic moving in and around the interfaces was tapped and features were extracted. Based on the extracted features, the entropy was calculated for the individual switch, if the entropy exceeded, the system flagged the traffic as suspicious and the IP address and MAC address were registered automatically in the IP tables for denial.

6. CONCLUSION

This paper has proposed a novel security framework to combat DDoS using a virtual firewall and a bandwidth analyzer. The model presented shows attacks launched over SDN as highly sophisticated and hard in terms of scalability. The proposed framework is highly reliable in detecting the suspicious traffic and handling that traffic within the medium without compromising on its performance. The proposed framework works fine for fixed topology. In future the work can be enhanced to fit changing topology. Some of the key concerns like topology changing firewall rules, automating rule updation etc. can be taken into consideration and a large volume of work spent on the above addressed issues.

REFERENCES

[1] Haifeng Zhou, Chunming Wu, Chengyu Yang, Pengfei Wang, Qi Yang, Zhouhao Lu, and Qiumei Cheng, "SDN-RDCD: A Real-Time and Reliable Method for Detecting Compromised SDN Devices," *2018 IEEE/ACM transactions on networking*, Vol. 26, No. 5, 2018.

[2] Vijay Varadharajan, Kallol Karmakar, Uday Tupakula, and Michael Hitchens "A Policy-Based Security Architecture for Software-Defined Networks," *2019 IEEE Transactions on information forensics and security*, Vol. 14, No. 4, 2019.

[3] Lyndon Fawcett, Sandra Scott-Hayward, Matthew Broadbent, Andrew Wright, and Nicholas Race, "TENNISON: A Distributed SDN Framework for Scalable Network Security," *2018 IEEE Journal on selected areas in communications*, Vol. 36, No. 12, 2018.

[4] Yanbing Liu, Yao Kuang, Yunpeng Xiao, and Guangxia Xu "SDN-Based Data Transfer Security for Internet of Things," *2018 IEEE Internet of things journal*, Vol. 5, No. 1, 2018.

[5] Kübra Kalkan, Levent Altay, Gürkan Gür, and Fatih Alagöz "JESS: Joint Entropy-Based DDoS Defense Scheme in SDN," *2018 IEEE Journal on selected areas in communications*, Vol. 36, No. 10, 2018.

[6] Bing wang, Yoa Zeng, Wenjing liu, and Thomas Hou "DDoS Attack Protection in the Era of Cloud Computing and Software-Defined Networking," *2014 IEEE 22nd International Conference on Network Protocols, Computer Networks 81*, pp. 308–319, 2015.

[7] S. Shin and G. Gu, "Attacking software-defined networks: a first feasibility study," *Proc. the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, pp. 165–166, 2013.

[8] Ying-Dar Lin, Po-Ching Lin, Chih-Hung Yeh, Yao-Chun Wang, and Yuan-Cheng Lai "An extended SDN architecture for network function virtualization with a case study on intrusion prevention," *IEEE Network*, Vol. 29, No. 3, pp. 48-53, 2015.
[9] Sungheon Lim, Seungnam Yang, Younghwa Kim, Sunhee Yang, and Hyogon Kim "Controller scheduling for continued SDN operation under DDoS attacks," *Electronics Letters*, Vol. 51, No. 16, pp. 1251–1261, 2015.
[10] Wanfu Ding, Wen Qi, Jianping Wang, and Biao Chen, "OpenSCaaS: an open service chain as a service platform toward the integration of SDN and NFV," *IEEE Network*, Vol. 29, No 3, pp. 30-35, 2015.
[11] Mohammad Banikazemi, David Olshefski, Anees Shaikh, John Tracey, and Guohui Wang, "Meridian: an SDN platform for cloud network services," *IEEE Communications Magazine*, Vol. 51, No. 2, pp. 120–127, 2013.
[12] Wang Yon, Tao Xiaoling, He Qian, and Kuang Yuwen, "A dynamic load balancing method of cloud-center based on SDN," *China Communications*, Vol. 13, No. 2, pp. 130-137, 2016.
[13] Richard Cziva, Simon Jouët, David Stapleton, Fung Po Tso, and Dimitrios P. Pezaros "SDN-Based Virtual Machine Management for Cloud Data Centers," *IEEE Transactions on Network and Service Management*, Vol. 13, No. 2, pp. 212–225, 2016.
[14] Qiao Yan, F. Richard Yu, Qingxiang Gong, and Jianqiang Li, "Software-Defined Networking (SDN)and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges," *IEEE Communications Surveys & Tutorials*, Vol. 18(1), pp. 602-622, 2016.
[15] Raul Munoz, Ricard Vilalta, Ramon Casellas, Ricardo Martinez, Thomas Szyrkowiec, Achim Autenrieth, Víctor Lopez, and Diego Lopez "Integrated SDN/NFV management and orchestration architecture for dynamic deployment of virtual SDN control instances for virtual tenant networks," *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 7, No. 11, pp. 62-70, 2015.
[16] He Li, Peng Li, Song Guo, and Amiya Nayak, "Byzantine-Resilient Secure Software-Defined Networks with Multiple Controllers in Cloud," *IEEE Transactions on C+loud Computing*, Vol. 2, No. 4, pp. 436-447, 2014.
[17] Izzat Alsmadi and Dianxiang Xu, "Security of Software Defined Networks: A Survey," *Computers and security, Elsevier*, Vol. 53, pp. 79-108, 2015.
[18] Fei Hu, Qi Hao, and Ke Boa, "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation," *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 4, pp. 1553-877X, 2014.
[19] Seungwon Shin and Guofei Gu, "Attacking Software-Defined Networks: A First Feasibility Study," *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pp. 165-166, 2013.
[20] Mayur Channegowda, Reza Nejabati, and Dimitra Simeonidou, "Software-defined optical networks technology and infrastructure: Enabling software-defined optical network operations," *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 5, No. 10, pp. A274–A282, 2013.
[21] C. Dixon, D. Olshefski, V. Jain, C. DeCusatis, W. Felter, J. Carter, M. Banikazemi, V. Mann, J. M. Tracey, and R. Recio, "Software defined networking to support the software defined environment," *IBM Journal of Research and Development*, Vol. 58, No. 2/3, pp. 3:1-3:14, 2014.
[22] Zilong Ye, Xiaojun Cao, Jianping Wang, Hongfang Yu, and Chunming Qiao, "Joint topology design and mapping of service function chains for efficient, scalable, and reliable network functions virtualization" *IEEE Network*, Vol. 30, No. 3, pp. 81-87, 2016.
[23] Shibo Luo, Jun Wu, Jianwua Li, and Longhua Gua, "A Multi-stage Attack Mitigation Mechanism for Software-defined Home Networks," *IEEE Transactions on Consumer Electronics*, Vol. 62, No. 2, pp. 200–207, 2016.
[24] Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Ford%E2%80%93Fulkerson_algorithm
[25] Haifeng Zhou, Chunming Wu, Ming Jiang, Boyang Zhou, Wen Gao, Tingting Pan, and Min Huang, "Evolving Defense Mechanism for Future Network Security," *IEEE Communications Magazine*, Vol. 53, No. 4, 2014.

BIOGRAPHIES OF AUTHORS

Arivudainambi D is currently an Associate professor in Department of Mathematics, Anna University, Chennai, Tamilnadu. He received his Post-Doctoral in University of Toronto in 2004. He received his Ph.D. degree from Anna University, in 2002 and His research interest includes Computer Networks, Queuing theory, Stochastic Processes and its applications, Operations Research, Cloud Computing, Wireless Sensor Networks, Evolutionary Algorithms, Adhoc Networks.

Varunkumar K.A. is currently pursuing his Ph.D. in Anna University, Chennai. His research interest includes Network Security, Malware Analysis, Cloud Security, Software Defined Network Security and Image Processing.