Advances in Research

21(8): 1-9, 2020; Article no.AIR.57485
ISSN: 2348-0394, NLM ID: 101666096

Detection and Mitigation of Flood Attacks in IPv6 Enabled Software Defined Networks

O. Ashimi Quadri[1] and Adeniji Oluwashola David[1*]

[1]Department of Computer Science, University of Ibadan, Ibadan, Nigeria.

Authors' contributions

This work was carried out in collaboration between both authors. Author OAQ design the study and perform statistical analysis. Author AOD managed the analyses of the study and literature review. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AIR/2020/v21i830221
Editor(s):
(1) Dr. Ali Said Mohamed Al-Issa, Sultan Qaboos University, Oman.
(2) Dr. Martin Kröger, Swiss Federal Institute of Technology, Switzerland.
(3) Dr. Jose Eduardo Serrao, Federal University of Viçosa, Brazil.
(4) Pradip K. Bhowmik, University of Nevada, Las Vegas, USA.
(5) Dr. Michel Mainguenaud, LITIS Laboratory, National Institute of Applied Sciences of Rouen, France.
(6) Dr. Shi-Hai Dong, CIDETEC, National Polytechnic Institute, Mexico.
Reviewers:
(1) Gheorghe Grigoras, Gheorghe Asachi Technical University of Iasi, Romania.
(2) Ali El Ksimi, University of Hassan II Casablanca, Morocco.
(3) Sridevi, Karnatak University, India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/57485

Original Research Article

Received 05 April 2020
Accepted 11 June 2020
Published 10 July 2020

ABSTRACT

Software-defined networking (SDN) is an emerging technology, which provides network architecture that decouples the control plane from the data plane. Due to the centralized control, the network becomes more dynamic, and the network resources are managed in a more efficient and cost-effective manner. The centralization of the control plane requires robust and real-time security techniques. The security Techniques will protect it from any sign of vulnerabilities associated with the network such as a distributed denial of service (DDoS) attacks. The problem of the data-plane is that the attack is hard to be tracked by the SDN controlling plane. This makes the switches to be more susceptible against these types of attacks and hence it is very important to have quick provisional methods in place to prevent the switches from breaking down as soon as first signs of an attack are detected. To resolve this problem, the research developed a mechanism that detects and mitigates flood attacks in IPv6 enabled software to define networks. An experimental testbed was developed using sFlow technique, floodlight controller, and OpenFlow version 1.3. A mitigation algorithm was also developed and was tested with a simulation tool Mininet. The real network traffic

**Corresponding author: E-mail: sholaniji@yahoo.com, od,adeniji@ui.edu.ng;*

was tested on the testbed to investigate the effective mitigation of a DDoS attack. The mitigation time performance for IPv6 was 46.6% while IPv4 was 66.6%. Also, The result gathered from the experiment showed that both the response and detection times were 4 secs while the mitigation time was 7secs respectively. The overall control time being 11 secs. The experimental Testbed result shows that the developed testbed outperformed the previous methods with the ability to detect threats on the network faster. The result from the IPv6 testbed is a probable solution to mitigate the threats posed by DDoS attacks on the IPv6 enabled SDN network resources.

Keywords: IPv6; openflow; openvswitch; mininet; sFlow; DDoS; flood attack; SDN.

1. INTRODUCTION

The ability to abstract the control functions from the forwarding elements has turned the tide in networking and has led to the emergence of Software-defined networking (SDN). SDN is changing the way IT networking infrastructures are controlled, managed and configured. Network administrators see SDN has a breakthrough for having total remote access and control of the network while network providers and operators believe that SDN will reduce capital expenditure (CAPEX), operational expenditure (OPEX) and energy consumption. Their arguments are all true as SDN allows the programmability of the network, the modification of routing rules or data flows according to real-time necessities, monitoring of network performances from a dashboard and the abstraction of the control plane from the data/infrastructural plane. Other benefits include the implementation of new network services such as quality of service (QoS), enforcement of new policies, accessontrol, traffic engineering, security enhancement and bandwidth management.

Software-Defined Networking (SDN) is a new networking approach that is introduced to simplify the network management by separating the data and control planes. SDN has brought with itself programmability in the network control plane. The shift of the control logic from networking devices, such as switches and routers, in traditional networks to a centralized unit known as the controller permits the physical network hardware to be detached from the Control Plane. This separation simplifies the design of new protocols and implementation of new network services such as access control, QOS, enforcement of new policies, bandwidth management and traffic engineering. No longer does every small change need to come at the cost of reconfiguring all the network devices [1].

The data plane consists of devices that contain tables of flow entries. These devices use a secure transport layer protocol to securely communicate with a controller about new entries that are not currently in the flow table. Each flow entry includes matching rules and actions that guide the data plane on the action to take for the matched flow. When a packet arrives at a switch the header fields will be matched against the matching rules available in the flow table and if a match is found the action specified by the flow entry will be taken by the switch otherwise the packet header will be forwarded to the controller for further process. The controller then processes the header and constructs a flow rule to be installed in the flow tables of the switches along the chosen path.SDN solves the issue of proprietary in the traditional network. It has proven to be the required technology to make networking more scalable, dynamic and removes the data plane devices vendor proprietary [2]. SDN abstracts the control functions from the switches or routers as against the traditional network, rendering the data plane devices as a merely forwarding agent [3,4]. The forwarding rules are logically generated by the controller and communicated to the switches through the OpenFlow protocol [3].IPv6 is the most recent generation of the Internet Protocol (IP) defined by the Internet Engineering Task Force (IETF). The resource management of Multihoming in nested mobile network raises new issues in the host mobility of ipv6 network.in [4].This new version, previously called IP The Next Generation (IPng), incorporates the concepts of many proposed methods for updating the IPv4 protocol. IPv6 is designed to have minimal impact on upper-layer and lower-layer protocols and to avoid the random addition of new features. peer-to-peer communications, each end acts as both client and server and, therefore, peers separated by NATs might not operate correctly and must be modified for NAT awareness. Eventually, Internet Corporation for assigned Names and Numbers (ICANN) added the addresses of IPv6 to its DNS server in 2004.An internet service-driven network is a new approach to the provision of network computing that concentrates on the services you want to provide as adopted in [5]. One of the main

targets of IPv6 is to increase address space so that NAT will become needless and to bring back end-to-end communication over the internet, it also aims to minimize the time and cost people spend while configuring new devices, it uses auto configuration when a new device is to be configured. Another goal of IPv6 is IPsec to prevent unauthorized user and transmission interception [6]. Software-defined network (SDN) is a networking approach that allows network administrators to programmatically initialize, control, change, and manage network behaviours dynamically via open interfaces such as OpenFlow protocol.The first model he proposed was Secure Architecture but the significant roles of encryption algorithms are numerous and essential in information security [7] *in* Comparative Study of Symmetric Cryptography Mechanism. OpenFlow initializes the rules of communication between data plane and a centralized control plane while it enables the entire network to be controlled through written software applications (APIs) [8].The prediction of incoming attacks is achieved promptly which enables security professionals to install defense systems in order to reduce the possibility of such attacks [9] Zero Day attack Prediction. if the attack is launched from multiple sources as in DDoS attacks, the network may still be overwhelmed[10,11,12].. A set of security threats, such as unauthorized access and data leakage, are enumerated and mapped against each of the five SDN layers and interfaces.[13,14] from Chungwa Telecom Co. proposes an OpenFlow DDoS Defender that monitors flows on an open flow switch. It is of very low standard and quality, little or no integrity, very easy to forge in [15] and If the number of packets received in 5 seconds exceeds 3000 then the number of packets will be studied in per second duration [16]. If the number packets per second exceed 800 for 5 continuous times then an attack is detected and the DDoS defender will start dropping the incoming packets until the flow entry times out [17].

2. MATERIALS AND METHODS

The developed experimental testbed in this research consists of Mininet. Mininet is an emulator software tool, which allows an SDN experiment to be emulated on a single computer. It uses lightweight process-based virtualization (Linux network namespaces and Linux container architecture) to run many hosts and switches on a single OS kernel. It can create kernel or user-space OpenFlow switches, controllers and hosts to communicate over the emulated network and connects switches and hosts using virtual Ethernet pairs. It simplifies the initial development, debugging, testing, and deployment process.The default mininet option installs openVswitch, wiresharks, floodlight controller. The specific option to be install in Mininet is given by the following commands below.

- ✓ Git clone git://github.com/minnet/mininet
- ✓ Cd mininet
- ✓ Mininet/util/install.sh –help
- ✓ Mininet/util/install.sh -n

sFlow is an open source statistical time-based and real-time traffic sampling technology used for monitoring traffic in data networks at high speed. When a new packet is transmitted on the network, sFlow agent counts the number of packets and check if the packet is an exempted one which it should ignore or a sample to be collected. If it is an exempted packet it will wait for the next batch (as defined by **I**) of samples. If not, it will take a quantity (as specified by S) and the packet is assigned a source and destination address before being sent to its destination.

2.1 Open Flow Packet Flow Algorithm for Packet Matching

The Fig. 1 was used for the packet matching when a packet arrives at a port for the first time, the switch checks its flow table for matching, if it matches any rule in its entry, it will act upon the instruction, if not, it will forward it to the controller through the packet in message and wait for the controller to tell it what to do (forward or drop) with it via the packet out message. It then acts upon the instruction and saves the new rule in its flow table for future use.

The following procedure was carried out in the implementation in the testbed using Fig. 2.

- For each N packet transiting the network, a normal of one of these packets is sent to the sFlow collector for analysis. These examples are gathered from every one of the switches on the network by the sFlow agents which are pre-installed on them already. Sampling rate and polling intervals are assigned on sFlow to gather the samples. Sampling rate (S): The number of packets that should be sent for sampling from every packet transiting the network.
- When a new packet is transmitted on the network, sFlow agent counts the number of

packets and check if the packet is an exempted one which it should ignore or a sample to be collected. If it is an exempted packet it will wait for the next batch (as defined by I) of samples. If not, it will take a quantity (as specified by S) and the packet is assigned a source and destination address before being sent to its destination.

2.2 Developed Data Sampling Algorithm for IPv6 in SDN

IPv6 mitigation Algorithm was developed as discussed below. Total number of ipv6 packet and the sampled ipv6 packet was flooded into the Algorithm for investigation in the emulator.

The testbed develop a IPv6 mitigation Algorithm. IPv6 mitigation Algorithms use threshold value, T, can also be set to ensure that the number of packets per second, P, flowing through the switch/es is below the threshold value, T. If the traffic surpasses the threshold, T, the modified mitigation algorithms triggers an alarm and generates traffic handling rule as defined in the script which will be communicated to the controller. The controller, on the other hand, sends an update on the new forwarding rules (to drop packets from the infected port/s) to the switch/es through OpenFlow protocol. The effect of the attack will soon be minimized and with a further update from the analyzer and the controller, the attack will be eliminated, and the network will be protected. The Mitigation Algorithm of IPv6 SDN is shown in Fig. 3.

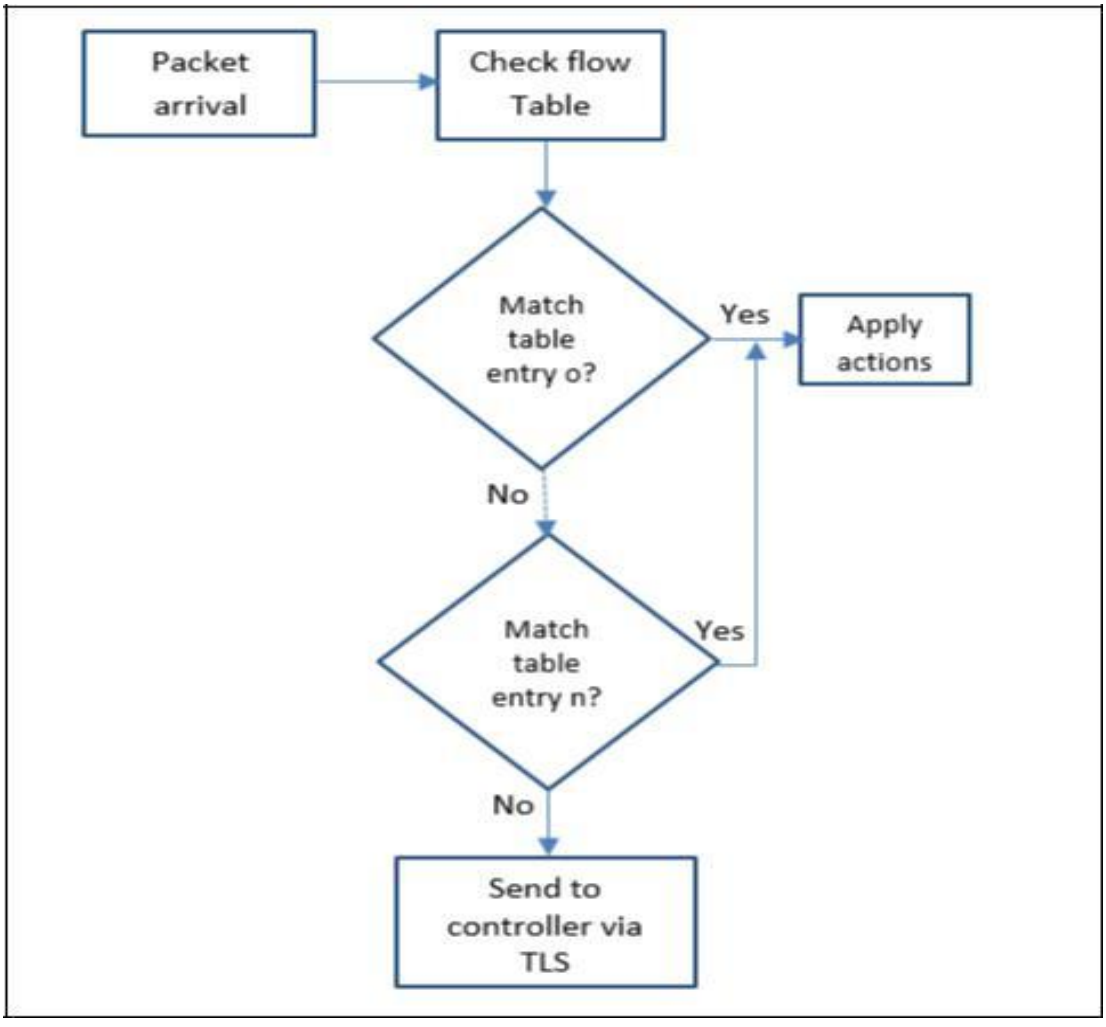

Fig. 1. Open flow packet flow algorithm

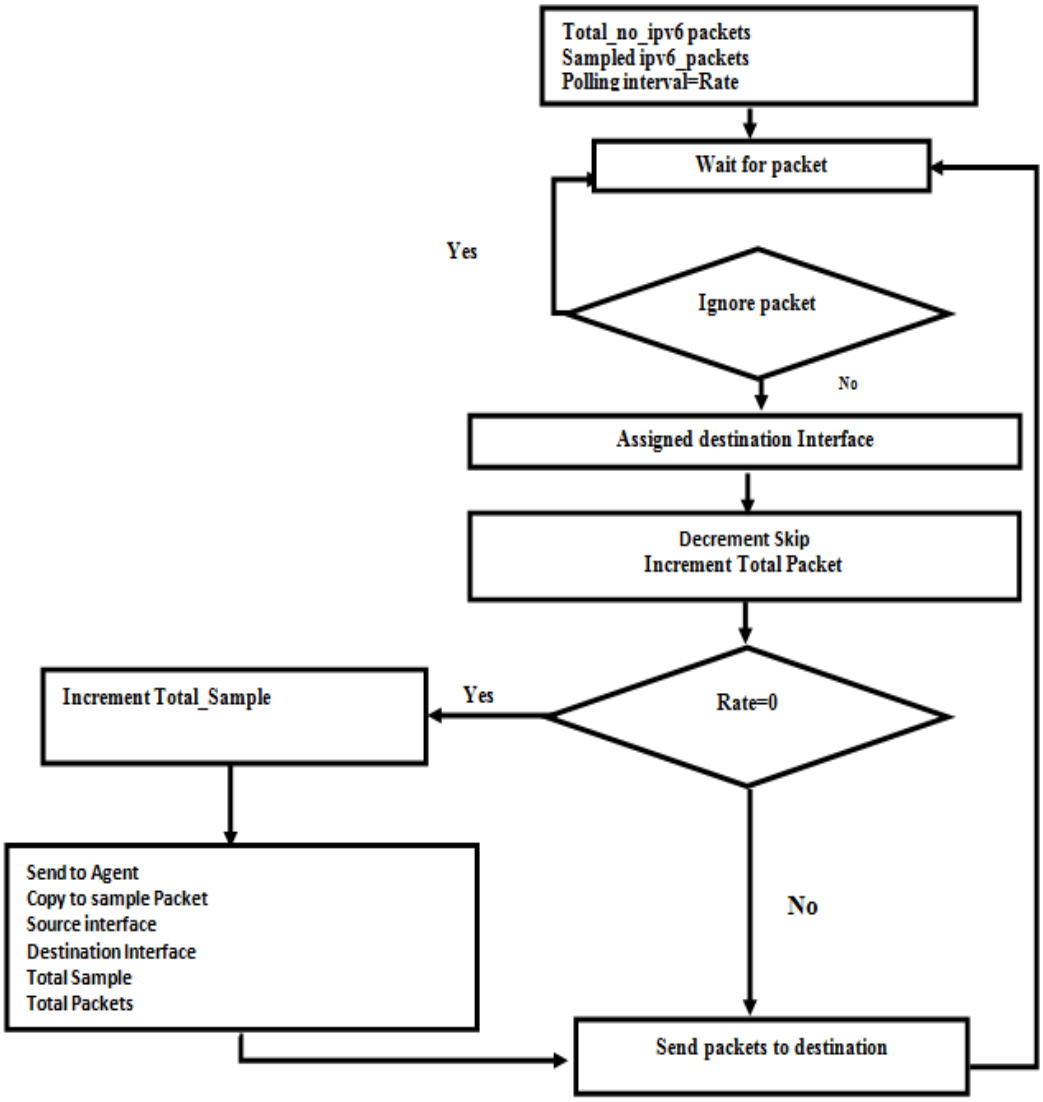

Fig. 2. Data sampling algorithm for IPv6

The characteristic of the testbed component is shown below in Table 1. Software define network(SDN) network topology was emulated on mininet using MiniEdit, MiniEdit can build and configure topologies including their network elements. In addition, it can save and run the topology for emulation.To start miniEdit, mininet must be running on and the following command is executed to call or open the mini Edit GUI.

The emulated network is a tree topology consisting of one controller, fifteen OpenVSwitches and sixteen hosts as shown in Fig. 4 Each switch is connected (logically) to the controller (c0) and some switches are connected to other switches. However, the switches are configured to avoid redundancy. Fig. 4 shows the *IPv6 Network topology testbed.*

3. RESULTS AND DISCUSSION

The tests were carried out in three phases and sFlow collected samples of each packet for analysis areas follows:

Phase 1: Host (h1) was pinged from other hosts(h2,h3,h4,h5,..h16) on the network when there was no attack on it and it was proved reachable. The Fig. 5 shows that the traffic rate is less than 2000pps while there was no attack on the host (h1).

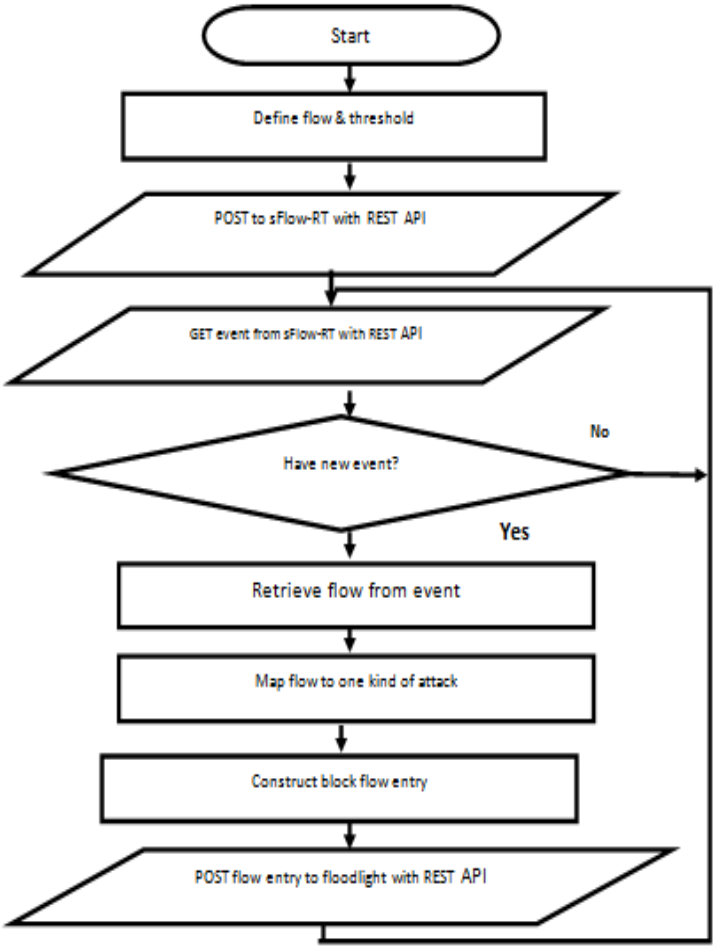

Fig. 3. Mitigation algorithm of IPv6 SDN

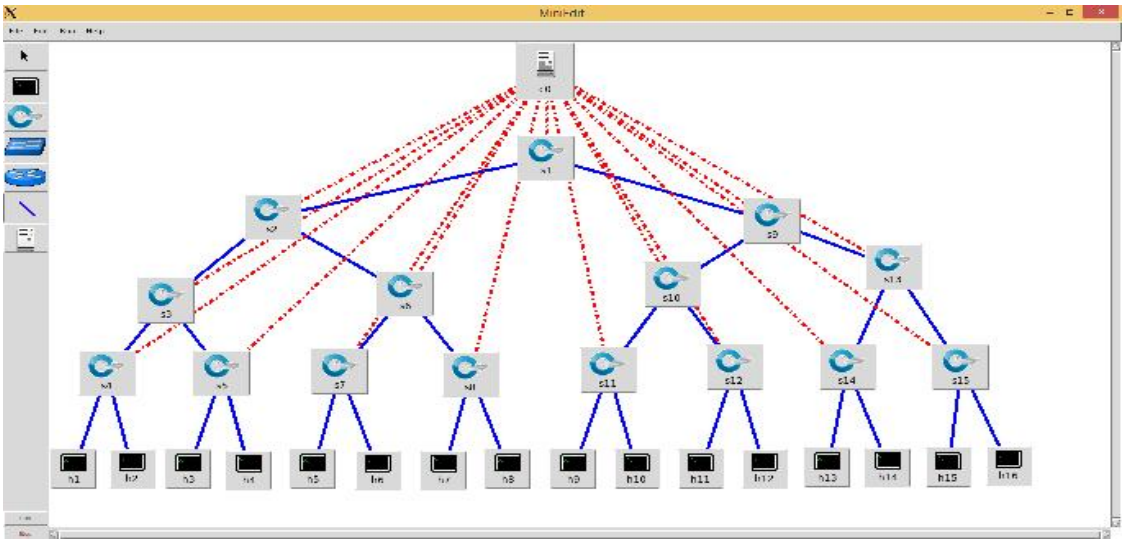

Fig. 4. IPv6 network topology testbed

Table 1. Characteristics of IPv6 SDN experimental test bed component

Model	CPU/speed	RAM	OS	Protocol	Emulator
G820 with Processor Intel (R) Core (TM) i5	CPU @ 2.50GHz 2.50GHz,	16.00 GB (15.9GB usable)	Linux Ubuntu 32bits	Floodlight 2.2.1 and OpenVswich 2.9.2.	MININET version 2.2

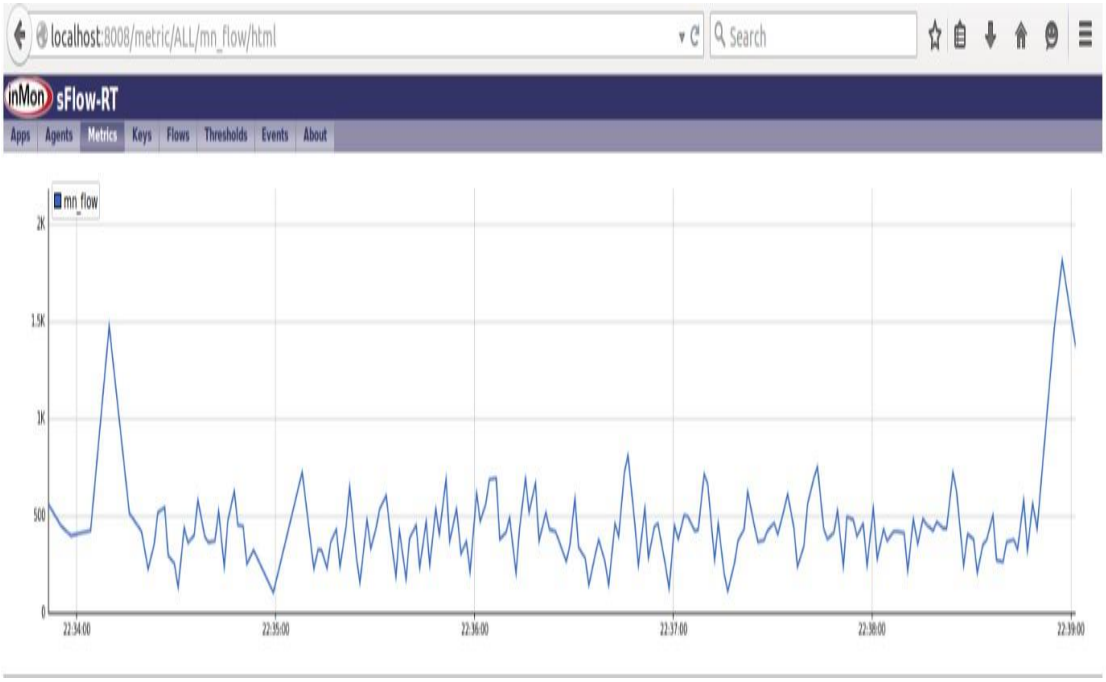

Fig. 5. Phase 1 testing for reachability on host h1

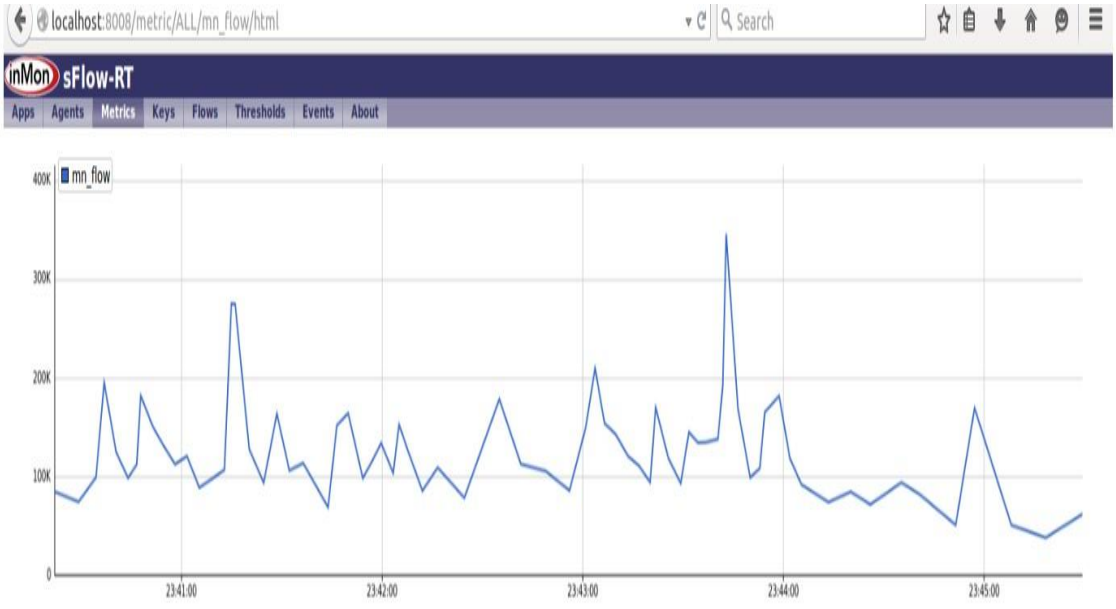

Fig. 6. Phase 2 ICMP flood attack

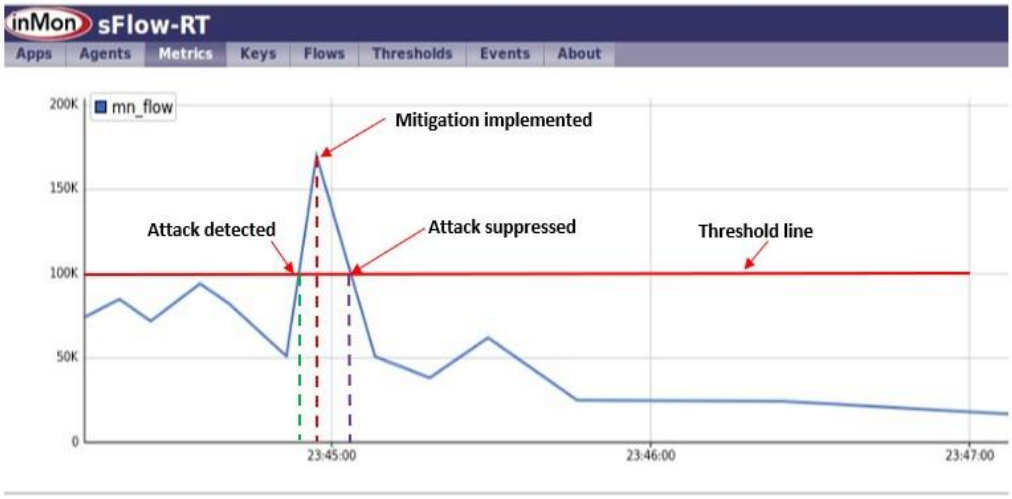

Fig. 7. Phase 3 attack detection and mitigation

Phase 2: ICMP flood attacks were launched from other hosts on the network to host (h1). This generated a huge amount of traffic that could overwhelm the network resources. The continuous request was sent to the controller for the inclusion of missing packets in the flow tables. the controller became so busy that it could not respond to requests from other hosts on the network and it started dropping packets from eligible users or hosts. Fig. 6 shows the Phase 2 *ICMP Flood Attack.*

Phase 3: At this phase, sFlow was launched to detect and mitigate the effect of the attack on the network. A threshold of 100,000 pps was set on sFlow at a sampling rate S of 10 (1 in every 10 packets should be sent for analysis) and a polling interval I of 5 (samples should be taken every 5 seconds). It can be seen from Fig. 6 that after a sharp rise was discovered by the analyzer, with the traffic surging to almost 700K pps, sFlow quickly generates the handling rule for the controller and it informs the switch/es to update its flow table; then packets were dropped from infected ports and the attack was mitigated within a period of ~7-11 seconds of implementation and the traffic went below the threshold line as shown in Fig. 7.

4. CONCLUSION

The different methods in the literature have large packet overheads that consume a notable fraction of the overall bandwidth of the controller. In contrast, the method used in this work does not install any extra device on the controller. It only generates rules to be implemented by the controller and communicates this through the existing northbound interface. This ensures easy deployment and scalability on the network.The sFlow technology was embedded with the controller and its packet overhead consumption is negligible about 0.02% of a 10 Gb Ethernet link yielding a very flexible and light solution in the testbed. The developed testbed can be used in traffic Engineering to secure network against vunerability such as DDOS attack. The future work is to apply the model on a robust machine learning approach that can distinguish a flash crowd from a DDoS attack on the SDN network.

ACKNOWLEDGEMENT

The authors wish to thank the Department of Computer Science, University of Ibadan for the support in this research work.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Braun W, Menth M. Software-defined networking using OpenFlow: Protocols, applications and architectural design choices, Future Internet. 2014;6(2):302-336.
2. Porras PA, Cheung S, Fong MW, Skinner K, Yegneswaran V. Securing the Software Defined Network Control Layer in NDSS; 2015.

3. Caraguay ALV, Lopez LIB, Villalba LJG. Evolution and challenges of software defined networking, in Future Networks and Services (SDN4FNS). 2013;1-7.
4. Adeniji SD, Khatun S, Raja RSA, Borhan MA. Design and analysis of resource management support software for multihoming in vehicle of IPv6 Network. Proceedings of the Fifth IASTED International Conference. 2008;607(089): 13.
5. Haleplidis E, Pentikousis K, Denazis S, Salim JH, Meyer D, Koufopavlou O. Software-defined networking (SDN): Layers and Architecture Terminology. 2015;2070-1721.
6. Adeniji SD, Khatun S, Borhan MA, Raja RSA. A design proposer on policy framework in IPV6 network. IEEE; 2008.
7. Hunag CY, MinChi T, YaoTing C, YuChieh C, YanRen C. A novel design for future on-demand service and security. IEEE. 2010;385-388.
8. Logunleko KB, Adeniji OD, Logunleko AM. A comparative study of symmetric cryptography mechanism on DES, AES and EB64 for information security. International Journal of Scientific Research in Computer Science and Engineering. 2020;8(1):45-51.
9. Abdou AR, van Oorschot PC, Wan T. Comparative analysis of control plane security of SDN and conventional networks. IEEE Communications Surveys & Tutorials; 2018.
10. Adeniji OD, Olatunji OO. Zero Day Attack Prediction with Parameter Setting Using Bi Direction Recurrent Neural Network in Cyber Security. International Journal of Computer Science and Information Security (IJCSIS). 2020;18(3):111-118.
11. Kreutz D, Ramos F, Verissimo P. Towards secure and dependable software-defined networks, in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM. 2013; 55-60.
12. Casado M, Freedman MJ, Pettit J, Luo J, McKeown N, Shenker S. Ethane: Taking control of the enterprise, in ACM SIGCOMM Computer Communication Review. 2007;37(4):1-12.
13. Vaughan-Nichols SJ. Open Flow: The next generation of the network?, Computer. 2011;13-15.
14. Zou T, Xie H, Yin H. Supporting software defined networking with applicationlayer traffic optimization," ed: Google Patents; 2016.
15. Eze Chika Victor, Adeniji Oluwashola David "Character Proximity for RFID Smart Certificate System: A revolutionary security measure to curb forgery Menace. International Journal of Scientific & Technology Research; 2014.
16. Olushola D Adeniji, Olubukola Adigun, Omowumi O Adeyemo. An intelligent spam-scammer filter mechanism using bayesian techniques. International Journal of Computer Science and Information Security. 2013;10(3):126.
17. Amer Nizar Abu Ali. Comparison study between IPV4 & IPV6", International Journal of Computer Science Issues. 2012;9(3).

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/57485