2022 International Conference for Advancement in Technology (ICONAT)
Goa, India. Jan 21-22, 2022

Mitigating DoS Attacks in Software Defined Networks

Vasantharaj Karunakaran
Research Scholar,
Dept of CSE
Hindustan Institute of Technology and Science
rs.vk0819@hindustanuniv.ac.in

Dr. Angelina Geetha
Professor/CSE,
Dean(E&T)
Hindustan Institute of Technology and Science
angelinag@hindustanuniv.ac.in

***Abstract*: Software Defined Network became a popular technology which yields the key features of control, tractability and scalability. The controller of SDN suffers from different types of harmful issues such a Spoofing, flooding and Denial of Service attacks (DoS). Among them DoS gives more threat to the system, because the opponent cannot be easily traced. There should be no delay when packets starts traveling from source host to destination host. Flow rules are installed in the centralized SDN controller, which the paths are established in the switches. In this paper, a new method is approached to mitigate the DoS attacks, so as to avoid the timeouts, dropping of the packets and also mitigate the traffic and Controller's utilization. With our method, the response time and flow request of the controller has been increased. Finally the results are simulated and the performances are measured in terms of response time and flow request of the controller.**

Key terms: Software Defined Networks, Vulnerable, Denial of Service attacks, Mitigate

I. INTRODUCTION:

Nowadays, network networks consistently favor Software Defined Networks (SDN) as a result of their simplicity of arrangement and straightforward setup. The Controller gives incorporated command over network traffic, while the bundle gets moved from source to objective. It can deal with all organization traffic with various conventions like Internet Control Message Protocol(ICMP), Transmission Control Protocol(TCP), User Datagram Protocol(UDP), Internet Protocol v6(IPv6), Internet Protocol v4(IPv4) [1], and so forth, gathered from various sources. The paper is coordinated as Software Defined Networks (SDN) and Denial of Service attacks (DoS) which clarify the working of SDN and DoS attacks [2]. The following segment "Related works" depicts the various strategies to deal with DoS assaults in SDN. The following segment of the paper manages the "Proposed technique" the proposed model, proposed calculation, and so forth, area "Experiment and result" which assesses the exhibition of the calculation and proposed model. At last "Conclusion and Future work" finishes up the paper and represents possible directions.

II. SOFTWARE DEFINED NETWORKS:

SDN has raised as a new network paradigm which decouples both Control planes and Management lanes from Data plane by implementing a controller. The controller is responsible for fixing the path direction of the packets. SDN controllers in particular holds enormous of applications [3] [4]. Default it will be a remote controller, but we can install any of the controller such as RYU, POX, Flood light, Daylight etc. Some may be designed by using Python, some controllers may be used Java to implement. Controllers play a vital role as tit is responsible for fixing the direction of the packets[5] [6]. The main purpose of SDN is the architecture can be virtually organized or customized by the developers. The customized SDN architecture developed is shown below in Fig 1.

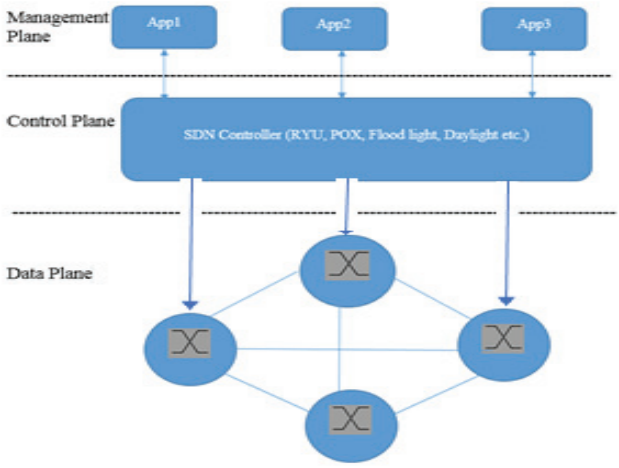

Fig. 1. Architecture of SDN

Fig 1. Shows the easy design of SDN Architecture. The bottom of the architecture consists of Data Plane which holds standard devices that may act as Routers, Switches and Access focuses. This relies upon how the software engineer is modified. A sending table from a Controller is accessible toward the finish of the gadget. Presently an information parcel shows up at the gadget, checks whether there is a passage for a bundle to be moved from source host to destination host. If the answer is "Yes" then the packet will be forwarded, else the packet will be dropped. Necessary action may be taken for the dropped packet.

A. Control Plane:

The middle layer of the Architecture is the control layer which holds the responsibilities of the controller. The Controller is liable for the conduct of the information bundles dependent on the network topology.

B. Management Plane:

The Management plane identifies the network applications such as load balancing, firewall etc.

2022 International Conference for Advancement in Technology (ICONAT) | 978-1-6654-2577-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICONAT53423.2022.9726071

There are Plane interfaces in the SDN architecture interact with each other through the standard interfaces.

(i)North Bound Interface: This North bound interface allows a particular network component to communicate with a higher level component.

(ii) South Bound Interface: This South Bound Interface allows a particular network component to communicate with a lower-level component.

C. *Denial of Service Attack:*

A Denial of Service (DoS) attack means it will disrupt the entire network, making it tedious for its envisioned users. In simple terms, the DoS attack denies its deserved users by interrupting the services. Victims of DoS attack are often targeted are web servers of high profile organization such as Department of Defense (DoD) etc. DoS attack does not result in any loss of money or any information, but results the victim a great loss of time and may lead to disruption of their infra-structure.

The Controller being a unified power keeps up with the refreshed data around the entire organization. The directing system is likewise overseen by the Controller with the assistance OF messages, like Packet-In, Packet-Out, Flow-Add, and so on [2]. Switches contain stream tables to store stream leads for a brief time and forward data as demonstrated by these standards. The regulator keeps the organization geography in its data set to give effective and ideal directing to the associated hubs. The directing applications/modules in notable open-source SDN regulators, for example, Floodlight [13], ONOS [14],Ryu (simple_switch, simple_switch_12, simple_switch_13) [15], and POX (L2_learning, L3_learning) [15] and so forth, the stream rules are introduced on OF switches in a straight or tree fashion(creates huge geography), or a solitary fashion(for more modest geographies) where each switch in the way between the source and the objective sends a Packet-In message toward the regulator to get the stream rule to advance the bundle to their recently showed up information.. In these case, if existence of any DoS attack may lead to dropping of the packets which is shown in Fig 2

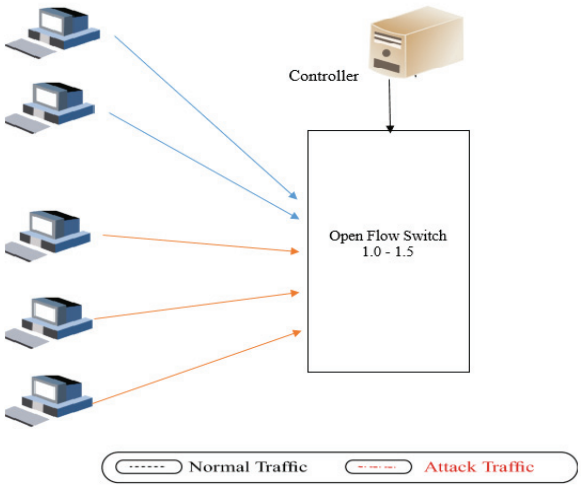

Fig. 2.

From the fig 2, the blue lines show the normal traffic, in which the packets are successfully transferred. The red lines shows that there may be a traffic attack, it means there may be an intruder, which leads to DoS attack. This may also leads to dropping of the packets.

III. RELATED WORKS:

Though the Control Plane and Data Plane are segregated, the Controller is still vulnerable to type of attacks such as DoS(Denial of Service) attacks, Spoofings etc. A Parallel flow Installation which was projected by Imran *et al* [1] is supposed to overcome the above said Vulnerability DoS in particular because this may lead to huge traffic, even we drop malicious packets. The proposed model of the SDN Controller, capable to monitor all the switches if and only if all the paths of the switches are connected to the controller. The pathless switches are ignored. So there is no possibility of delayed packets or any malicious behavior.

SDN not only a modern technology, but a cutting edge technology which decouples Control Plane and Data Plane. Lima *et al* [2] introduced a mechanism to mitigate the Dos attacks just because of its vulnerability. This overcomes the Statistical and traffic entropy flood attacks like SYN and UDP.

Apart from the segregation of Control plane and Data plane, the centralized SDN control leads to various types of attacks such as topology poisoning and side-channel attacks. To overcome this Shang *et al* [3] proposed a framework called Flowkeeper Framework which provides a countermeasure for the above kind of attacks.

From Rajat *et al*[4] point of view as large scale of deployments can be done, lot of security issues had been arised. This paper also discusses types of attacks such as attacking the control plane Bandwidth and attacking switch's flow table. This authors overcome the issue by adjusting the configurations(i.e) Bandwidth value and tuning the network requirements to those DoS attacks.

Though SDN Controllers play a very important role, it leads to an important issue in which it will get compromised of threats. To prevent the controllers getting compromised, the Open flow switches between Controller and hosts uses packet traces from Data plane. Deviation occurred because SDN Controllers got compromised by malwares. Anand *et al*[5] proposes a solution of creating a large volumes of Open flow traces and also studying a Machine Learning based detection technique for compromised controllers.

An Open flow mechanism will diminish the amount of control messages between controllers and switches. The congestion in control plane will be decreased. This limitation is proposed by Imran *et al*[6] as SDN is very much vulnerable because of its key features such as Flooding, Spoofing and DoS attacks.

SDN always easy for monitoring traffic flow, analyze threats and pullout or change security policies. This leads to security challenges. Two Proposals given by Yum *et al*[7]to get rid of security challenges such as Flowsec and Blackbox, where the flowsec mitigates an attack on the Controller Bandwidth by setting limitation on the number of packets that

can be used to sent to the controller, where Blackbox mainly keeps track of security threats.

There is possible for an attacker to produce a large amount of table-miss packet. Wang *et al*[8] introduced a solution called FLOODGUARD in which runtime of the controller is always updated and protects the controller from being overloaded. So the packets can be protected from the attackers. But this is applied only for limited protocols.

The two important security challenges while implementing AVANT-GUARD mechanism is i) intrinsic communication between control plane and data plane and ii) DoS can produce more potent impacts on Open flow networks. Shin *et al*[9] get rid of those two challenges by introducing the connection migration technique embedded in the above said mechanism. This technique reduces the amount of data-to-control plane interaction that arise. Also the mechanism evaluates the performance to overcome security challenge issue.

A single point of failure really makes sense in centralized SDN controller point of view. The possibility of chances persists when segregating Control plane and Data plane. Yung *et al*[10] prescribed a solution to detect such kinds of attacks.

SDN has rapidly emerged a promising technology because of its centralized controller. But issues arises when attacks such as DoS and Spoofing which makes the controller vulnerable and packets transferred to destination becomes un-authentic. Tao *et al*[11] proposed SDN Manager which is said to be a fast detection method of DoS attacks. To implement the fast detection method, five components such as Monitor, Forecast engine, Checker, Updater and Storage service plays an important role in uncovering and extenuation of DoS attacks.

SDN always provides high programmability to control and manage networks. A source host can transfer packets to destination. But the packet transfer can be delayed because of DoS attacks. It also leads to packet loss rate by using machine table miss-packets to jam link between the two hosts. Shang *et al*[12] proposed a solution FloodDefender a framework for SDN open flow networks, which identifies and efficiently mitigate DoS with a very little overhead.

While adopting SDN in any Organization or in any Institutional zones, security breaches have always been a big issue. Shravanya *et al*[13] urged to secure the SDN architecture against DoS attacks. The proposed architecture includes Big data techniques to build architecture that is prone to prevent network traffic.

Lot of Researchers accepted SDN security, while building SDN architecture it is important to build the features such as scalability, bottleneck and load balancing particularly in distributed environement. Pratima *et al*[14] undergone various SDN security issues, different scenarios and provided a brief case study on that.

Inside DoS attacks, there are lot of attacks which includes IP/MAC spoofing and Bulky/Garbage message. All these types of attacks can be overcome and similar solution is implemented in the network simulation environment Mininet 2.3.2. Oktian *et al*[15] also provided more supplementary method to mitigate DoS attacks.

Apart from DoS attacks, SDN has a lot of vulnerability issues such as scalability which becomes severe vulnerability in highly dynamic large scale networks, whereas the forwarding rule must be updated by a centralized controller. Qing *et al*[16] proposed a Dynamic flow rules which enables the network elements to change their forwarding behavior locally, as per the pre-defined instruction defined by SDN controller. This is also applicable for different SDN use cases.

IV. Proposed Method:

Present flow rules with a steadily expanding number of coordinating with fields for fine-grained the leaders and seeing of information insights. Accordingly, the OF explicit depicts a defaulting set of 45 matching fields. Besides, as the amount of organizing with fields for the streams increases, more traffic will stream toward the controller as Packet-In messages and the controller will utilize more resources for handle them. During normal traffic, SDN ends up incredible with extra planning with fields; in any case, when there is a DoS attack in progress, then, the quantity of groups sent over the control channel will increment radically. The attack on SDN will bring about the increment of the reaction time from the objective side, Controller's usage will expand, Controller's channel traffic increment, and so on

The packet from source to destination will happen because of the request of the data packet to be transmitted. The DoS attack happened on the same way, where the request of the packet will be fake one. Now the packet going to be transmitted from source to destination will result in generation of fake packets created by DoS attacker.

The plan of Controller is obligatory which will introduce flow rules. The Open flow determination in the Control field characterizes a default set of coordinating with fields of 40 to 50 identical fields. Besides, as the quantity of coordinating with fields for the streams increments, more traffic will stream toward the regulator as Packet-In messages. Assuming there is ordinary traffic, then, at that point, this will be fine. Be that as it may, in the event of more traffic (or) Denial of Attack (DoS) attack in progress, then, at that point, the number of bundles will increment dramatically, which influence SDN by the accompanying ways.

- Acknowledgement acceptance time of the packet by destination will increase.
- Increase of Controller’s CPU usage.
- Possibility of traffic getting increased.

The above fig 3 shows the working of mitigating Denial of Service(DoS) attack. The term prevention of DoS attack means transferring the packet from source host to destination host without any disturbance. If the packet transferred at the maximum speed to the destination, then it is the mitigation. Once the circuit is simulated, the links for the packet transfer will be created in the following ways. The hosts that are configured by the circuit are h1, h2, h3, h4, h5, h6, h7, h8 and the links (h1, s1) (h2, s2) (h3, s3) (h4, s4) (h5, s5) (h6, s6) (h7, s7) (h8, s8) and the switches are (s2, s1) (s3, s2) (s4, s3) (s5, s4) (s6, s5) (s7, s6) (s6, s7). Presently, to diminish the effect of the DoS attack, it is obligatory to minimize the amount of

control messages, as the amount of control messages is direct relating to the Controller's CPU.

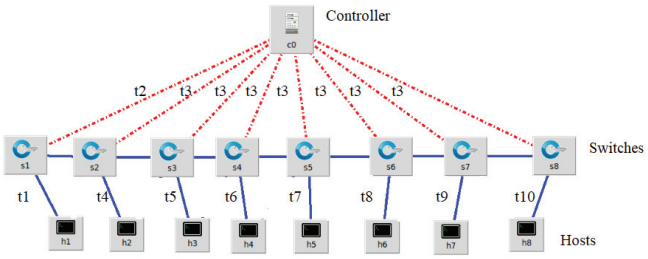

Fig. 3. Working of the Circuit

The proposed model in fig 2 shows the time interval to transmit the packet from source host(h1) to destination host(h2). At time t1, host1 sends the packet to switch1 which searches the flow table whether it is installed for the possible existence of matching flow rule. If it is installed, then the packet will be transferred to host8 i.e the destination host. Any debacle in observing the flow rule, switch1 sends a bundle in the message to the Controller at time t2. The Controller computes the total way from source to objective and sends stream rules to all changes from switch s1 to switch s8. The Controller will send the stream rule to the way among source and objective. From the above Fig 3, the blue lines shows the path from source host to destination host, i.e. flow rules are installed from the controller. The red line shows that the flow rules are not installed; hence the controller will be installing the flow rules to all the possible switches, which are inside the path of the circuit.

If the switch is not included in the path between source and destination, then that path will be discarded. This is well explained by the following algorithm.

Proposed algorithm:

---

1. *Src* -> new *Pckt*;
2. *Send(Pckt) -> InPort*;
3. **If** *Pckt.Header ϵ Flow_Table* **then**
4. Update_Counters();
5. Send(Pckt) -> Output.Port
6. **else**
7. *send(Pckt_In) -> Controller;*
8. *Identify next hop and add Flow rule;*
9. *Flow_rule.Output_Port <- next_hop;*
10. *Send(Flow_rule) -> switch;*
11. *Update_Counters();*
12. *Send(Pckt) -> output_Port;*
13. endif
14. *Output_Port -> Dst;*

A. Experimental setup and results:

The proposed model is executed by the transmitting mechanisms of RYU(Simple_switch_13) and POX(Simple_switch_12) Open flow types 1.3 and 1.1 separately. To survey the proposed model, the analysis is performed by following an organization geography as displayed in Fig3. An ordinary traffic is produced at each finish of host, so any host might go about as source or objective host. A DoS attack is dispatched by arbitrarily pick a host, which continues to request another standard from the Controller. As the stream rules are now introduced in the regulator, this is an avoidable work of continuing to introduce the stream rules. The presentation of the Controller can be determined by the directing component of RYU (Simple_switch_13) and POX(Simple_switch_12).

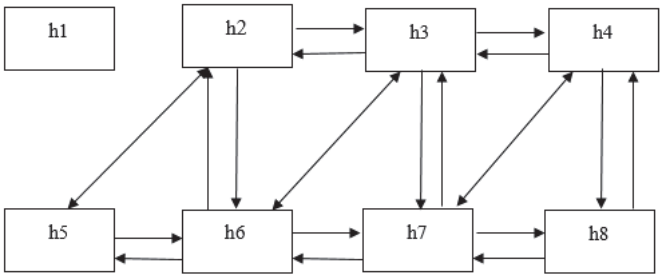

Fig. 4. Dropping of Packets due to DoS attack

From the Fig 4, it shows the dropping of Packets due to DoS attacker. A DoS attack is launched by the attacker, which issues requests to the controller, to install the flow rules. Due to the elapsing of time, the hosts(h1) forgets it path as it is unable to recognize the switch, whether the flow rules are installed in its respective switch. The delay in tracing the path, leads to DoS attack.

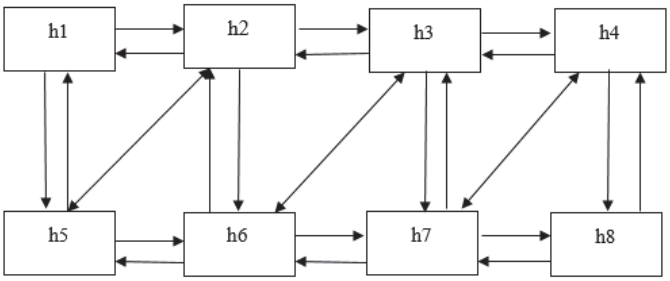

Fig. 5. Successful transmission of Packets

The above Fig 5 depicts the successful transmission of packets. The figure represents any hosts may act as source host and destination host. If the flow rules are installed perfectly, then the switches will complete the role of successfully transferring the packets. The experiment produces a total of 7.9s to complete all the hosts to transfer the packets. As there is no delay in the packet, the packet from the source to destination is transferred at its best, as shown in the proposed algorithm.

The topologies widely used are Single, Linear and Tree. Both the RYU and POX controllers adopt these topologies. Single topology uses less number of hosts, so it takes very less time probably 2s. The very familiar used topology will be linear topology, more hosts can be added. Performance of tree topology takes more time and a bit of tedious while

approaching tree topology. All the three topology's performance are calculated in the fig 6.

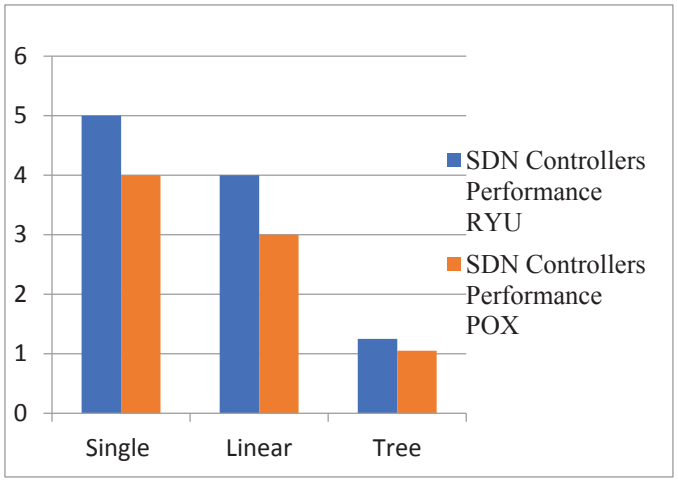

Fig. 6. Performance of the Controller's Topology

The above Fig 6 represents the graphical work of the SDN Controllers. The topologies that are familiarly used by the SDN Controllers are Single, Linear and Tree. The X-Axis of the graph shows the types of Topologies of SDN Controllers and the Y-axis represents the values measured in terms of Throughput and time in seconds. Linear topology is always suggested by most of the SDN researchers.

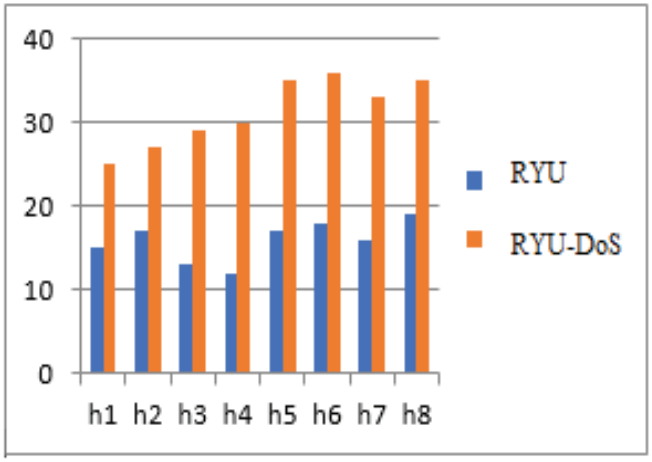

Fig. 7. Comparison of Packets of RYU and RYU-DoS

The above Fig 7, depicts the comparison of packets using RYU controller. The packets are compared when RYU controller with DoS attack and without DoS attack. From the Fig 7, it shows RYU-DoS shows, the packets are getting delayed and because of this there may be chances of the packets getting discarded. RYU without DoS shows the packets are getting immediately transferred to its destined location. X-Axis of the Fig shows the number of hosts, totally 8 hosts. Y-Axis of the Fig shows the transfer rate of the packets which is measured in terms of milliseconds.

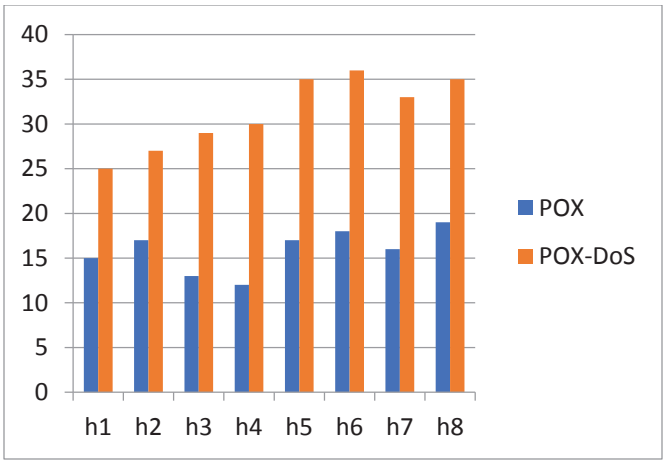

Fig. 8. Comparison of Packets of POX and POX-DoS

The above Fig 8, depicts the comparison of packets using POX controller. The packets are compared when POX controller with DoS attack and without DoS attack. From the Fig 7, it shows POX-DoS shows, the packets are getting delayed and because of this there may be chances of the packets getting discarded. POX without DoS shows the packets are getting immediately transferred to its destined location. X-Axis of the Fig shows the number of hosts, totally 8 hosts. Y-Axis of the Fig shows the transfer rate of the packets which is measured in terms of milliseconds.This experiment can be executed on Ubuntu Platform using Ubuntu 18.0 or 20.0 and then mininet 2.3.2.

V. Conclusion and Future Plan:

DoS attacks on SDN environment makes it harmful which can disrupt the entire network, even though if working in Ubuntu platform. Existing techniques counter DoS counter attacks in SDN, but it requires additional equipment or switches, hence it becomes very tedious. Also this leads to issues in routing process. All these issues are taken into consideration and a new proposed model had been developed which provides more tolerance to DoS attacks. Time expiry had been overcome by the proposed model which is available for legitimate users. The results and experiments of the proposed model proves the reduction of time that establishes a path between two nodes, lowers the controller's processing and decreases channel's traffic. Working in Ubuntu makes the Researchers very comfortable. In future a complex structured topology will be designed and developed. The key point is efficiency which will be checked for DoS attack detection and mitigation.

References:

[1] Imran, M., Durad, M.H., Khan, F.A. et al. Reducing the effects of DoS attacks in software defined networks using parallel flow installation. Hum. Cent. Comput. Inf. Sci. 9, 16 (2019). https://doi.org/10.1186/s13673-019-0176-7

[2] S. Gao, Z. Peng, B. Xiao, A. Hu, Y. Song and K. Ren, "Detection and Mitigation of DoS Attacks in Software Defined Networks," in IEEE/ACM Transactions on Networking, vol. 28, no. 3, pp. 1419-1433, June 2020, doi: 10.1109/TNET.2020.2983976.

[3] S. Gao, Z. Li, B. Xiao and G. Wei, "Security Threats in the Data Plane of Software-Defined Networks," in IEEE Network, vol. 32, no. 4, pp. 108-113, July/August 2018, doi: 10.1109/MNET.2018.1700283.

[4] R. Kandoi and M. Antikainen, "Denial-of-service attacks in OpenFlow SDN networks," 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, Canada, 2015, pp. 1322-1326, doi: 10.1109/INM.2015.7140489.

[5] Anand Narayanan, Sarath Babu, Manoj BS, "On Detecting Compromised Controllers in Software Defined Networks", Computer Networks, March 2018

[6] Imran, M., Durad, M.H., Khan, F.A. et al. "Towards an Optimal Solution against Denial-of Service attacks in Software Defined Networks", Future Generation Computer Systems, Vol 92, March 2019, Pages 444-453

[7] Y. Tian, V. Tran and M. Kuerban, "DOS Attack Mitigation Strategies on SDN Controller," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0701-0707, doi: 10.1109/CCWC.2019.8666456.

[8] H. Wang, L. Xu and G. Gu, "FloodGuard: A DoS Attack Prevention Extension in Software-Defined Networks," 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Rio de Janeiro, Brazil, 2015, pp. 239-250, doi: 10.1109/DSN.2015.27.

[9] Seungwon Shin, Vinod Yegneswaran, Philip Poras, Guofei Gu, "AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software Defined Networks", Nov 2013, DOI: 10.1145/2508859.2516684

[10] H. Wei, Y. Tung and C. Yu, "Counteracting UDP flooding attacks in SDN," 2016 IEEE NetSoft Conference and Workshops (NetSoft), Seoul, Korea (South), 2016, pp. 367-371, doi: 10.1109/NETSOFT.2016.7502468.

[11] Tao Wang, Hongchang Chen, Guozhen Cheng, Yulin Lu, "SDNManager: A Safeguard Architecture for SDN DoS Attacks Based on Bandwidth Prediction", Security and Communication Networks, vol. 2018, Article ID 7545079, 16 pages, 2018. https://doi.org/10.1155/2018/7545079

[12] G. Shang, P. Zhe, X. Bin, H. Aiqun and R. Kui, "Flood Defender: Protecting data and control plane resources under SDN-aimed DoS attacks," IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, Atlanta, GA, USA, 2017, pp. 1-9, doi: 10.1109/INFOCOM.2017.8057009.

[13] S. G., S. N. H., R. P. Rustagi and O. Sharma, "Securing Distributed SDN Controller Network from Induced DoS Attacks," 2019 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bengaluru, India, 2019, pp. 9-16, doi: 10.1109/CCEM48484.2019.000-4.

[14] Prathima Mabel J., Vani K.A., Rama Mohan Babu K.N. (2019) SDN Security: Challenges and Solutions. In: Sridhar V., Padma M., Rao K. (eds) Emerging Research in Electronics, Computer Science and Technology. Lecture Notes in Electrical Engineering, vol 545. Springer, Singapore. https://doi.org/10.1007/978-981-13-5802-9_73

[15] Y. E. Oktian, S. Lee and H. Lee, "Mitigating Denial of Service (DoS) attacks in OpenFlow networks," 2014 International Conference on Information and Communication Technology Convergence (ICTC), Busan, Korea (South), 2014, pp. 325-330, doi: 10.1109/ICTC.2014.6983147.

.