2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI) | 978-1-6654-3843-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ACMI53878.2021.9528288

Towards Autonomic Entropy Based Approach for DDoS Attack Detection and Mitigation Using Software Defined Networking

Md Taef Uddin Nadim
Department of ECE
Khulna University of Engineering & Technology
Khulna, Bangladesh
taef.nadim@gmail.com

Md. Foysal
Department of ECE
Khulna University of Engineering & Technology
Khulna, Bangladesh
mdfoysal@ece.kuet.ac.bd

***Abstract*— Software defined networking (SDN) architecture frame- work eases the work of the network administrators by separating the data plane from the control plane. This provides a programmable interface for applications development related to security and management. The centralized logical controller provides more control over the total network, which has complete network visibility. These SDN advantages expose the network to vulnerabilities and the impact of the attacks is much severe when compared to traditional networks, where the network devices have protection from the attacks and limits the occurrence of attacks. In this paper, we proposed an entropy based algorithm in SDN to detect as well as stopping distributed denial of service (DDoS) attacks on the servers or clouds or hosts. Firstly, there explored various attacks that can be launched on SDN at different layers. Basically DDoS is one kind of denial of service attack in which an attacker uses multiple distributed sources for attacking a particular server. Every network in a system has an entropy and an increase in the randomness of probability causes entropy to decrease. In comparison with previous entropy based approaches this approach has higher performance in distinguishing legal and illegal traffics and blocking illegal traffic paths. Linux OS and Mininet Simulator along with POX controller are used to validate the proposed approach. By conducting pervasive simulation along with theoretical analysis this method can definitely detect and stop DDoS attacks automatically.**

Keywords*— SDN, DDoS attack, Control Plane, Entropy, Data plane, Cloud servers, Traffic.

I. INTRODUCTION

Now-a-days cloud based services and cloud based organizations are expanding day by day. People are entering into cloud computing structures as this cloud computing scheme is becoming more and more default option for many apps and software companies. The software vendors are increasingly offering and motivating customers to use and implement their services over the internet rather than the physical standalone products. These companies are basically trying to switch to a subscription model rather than a buying and selling model. The cloud based software distribution approach is an approach in which software runs at a hosting provider which is situated at the cloud service [1]. The Cloud servers faces some serious risk, threat and vulnerabilities such as:

- Abuse of cloud computing resources,
- Data breaches: Malicious insider,
- Data breaches: Online Cyber Theft,
- Cyber Security Attacks.

The key Cyber Security threats and Abuse of cloud computing resources [2]:

- Denial of Service (DoS)
- Distributed Denial of Service (DDoS)
- Insider threats
- Hijacking accounts
- Insecure Applications
- Inadequate Training

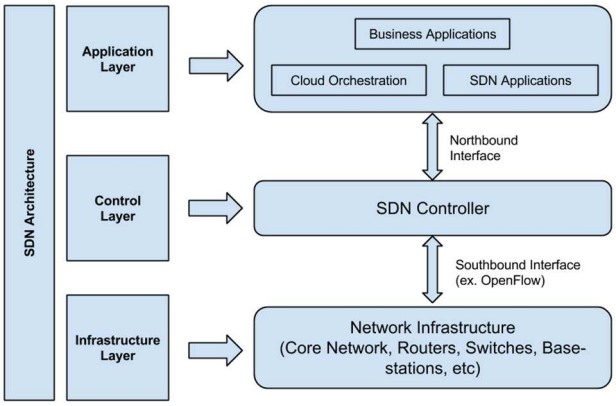

Fig. 1. Basic SDN Architecture

SDN is the software enablement of the infrastructures. Fig. 1 shows the basic SDN architecture. It is a part of the software defined datacenter theory that basically decouples the hardware form the software module and also in place of complex hardware devices it deploys layers of software which are changeable. Specifically, SDN detaches the control panel having the software interface from a node or network points forwarding devices such as routers and switches. Furthermore it establishes the control software remotely on a cloud based local server. This helps enormously in the field of securing cloud computing architectures. When considering to deploy SDN on a cloud based network it should be thought over its orchestration capabilities. Basically effective and useful orchestration is one of the key factors to be successful in dealing with a difficult infrastructure that consists of millions of software transactions. These can link an SDN solution with direct

978-1-6654-3843-8/21/$31.00 ©2021 IEEE

application. However the cloud architecture vulnerabilities or the threats that were discussed earlier can be enormously minimized using SDN systems. As the forwarding devices connected with the end devices can be controlled by the programmable software interface of a control plane. Each of them can be provided with feasible security measures that can prevent most of the deadly and serious cyber-attacks towards the cloud server or the local server.

The deadly attacks like DDoS can be prevented by deploying SDN and programming the control plane accordingly. In this work, an algorithm is proposed to prevent DDoS attacks by implementing the method on the SDN control plane. The end level switches can be authorized accordingly to detect and stop cyber-attacks like DDoS by employing such scripts which eventually will make the network administration way easier and feasible. The main contributions of this work are summarized as:

- Employment of Hash table for entropy measurement of the destination IP addresses.
- Applying the adjustable entropy threshold and time interval between packets for rapid detection of DDoS attack.
- A robust algorithm is proposed for detection as well as mitigation of DDoS attack.

The rest of the script is organized as follows. Section II presents the related state of the art works. Section III describes the proposed methodology clearly and section IV provides the experimental result with discussion. Finally the section V concludes the whole work as well as future scope of research.

II. Related Works

Recently many research works are done in this area of DDoS attack detection. Singh *et al.* [3] proposed a method to detect DDoS attack that includes Multilayer Perceptron (MLP) classification algorithm and along with Generic Algorithm (GA). They basically used an entropy based machine learning technique to detect DDoS attack on a particular host. They calculated the HTTP GET request's entropy, count and variance for each connection. Devi *et al.* [4] proposed a machine learning approach which is comprised of online monitoring system (OMS) and hop count inspection algorithm (HCF) which is eventually coupled with support vector machine (SVM). The OMS provides with the impact measurement of DDoS which is in real time by observing the degradation and lessening in host and also the network performance metrics. The spoofed flow detection module associates HCF to check the uniqueness of the incoming packets based on their IP address and the required hops to the target victim. Then HCF is coupled with the Support Vector machine which increases accuracy. The Interface Based Rate Limiting (IBRL) algorithm stops the traffic sums at the target router whenever it is suppressing the system limits for providing BW for remaining flows.

Bawany *et al.* [5] introduced a robust framework on SDN for detecting and mitigating DDoS attack. The proposed framework is capable enough for application specific. They used SNORT which is a well-known intrusion detection system and network intrusion prevention system. The framework named ProDefense or ProActive DDoS defense network occupies the programming and the dynamic nature of SDN and establishes a changing DDoS mechanism. Yadav *et al.* [6] proposed a packet entropy algorithm technique for detecting and mitigating the DDoS attack in a SDN based network architecture. There basically used three different algorithm for generating normal and malicious traffics and also a DDoS detection and mitigation scheme for stopping the attack. The simulation was also carried on mininet emulator and Pox controller.

Sahay *et al.* [9] proposed a priority based DDoS mitigation system. Their method is based on ISP assigned priority on flow considering customer's claim whether it is legitimate or not. Dalou *et al.* [10] designed mechanism is based on the entropy values of source and destination IP addresses of flows observed by the SDN controller which are compared to a preset entropy threshold values that change in adaptive manner based on network dynamics. Kalkan *et al.* [11] presents a mechanism of congestion avoidance technique called FlowFence mainly based on bandwidth controlling. Yaegashi *et al.* [12] proposed lightweight DDoS attack mitigation method utilizing the unused queue resources by randomly shuffling the queue allocation.

From the literature review, it is seen that various techniques such as machine learning, bandwidth controlling and utilization of unused queue are used to detect the DDoS attack. There also some works are done based on entropy measurement techniques. We proposed a novel entropy based scheme which is robust and having fast detection and mitigating capability.

III. Methodology

A. Hash Table

The main backbone point of the work is to create an automated system on the interface of the SDN controller which will measure the entropy of the incoming packets in the switches that are in the topology or network. It will calculate the entropy of the each incoming packets of each of the switches individual ports which are basically controlled by the SDN controller application (i.e. POX controller [7] in this case). So for the easiness of the calculation these port's entropy of the switches will be deployed with a hash table. The size of the hash table will be 50. In this hash table whenever a new packet arrives, first it checks out if the packet was arrived before according to its destination address. If not, the value of the frequency of that destination address is set to 1. In this way the frequencies of individual destination addresses are computed. Whenever the no. of packets reaches 50 the hash table is full and the controller forwards these packets into their destination addresses accordingly. Table I shows the hash table of size 10.

Let 'x1' packet arrive at an individual port of a switch. The controller (which is remote in this case) checks if it arrived before or not according to same destination. Let's it is not arrived earlier so its frequency count of that destination IP increases to one. Then let 'x2' packet arrives after the 'x1'. Then 'x2' is also included at the hash table. But if in 'x2' the destination address is same then the count value (number of occurrences) of that destination IP will be 2. In this way 50th packet arrives that is also included in the hash table. If the packet count reaches 50 then the hash table

is full and these packets will be parsed for their destination IP addresses.

TABLE I. A HASH TABLE SIZED OF 10

Packet	Destination IP	No. of Occurrence
x1	10.0.0.56	1
x2	10.0.0.51	1
x3	10.0.0.56	2
x4	10.0.0.56	3
x5	10.0.0.1	1
x6	10.0.0.56	4
x7	10.0.0.4	1
x8	10.0.0.6	1
x9	10.0.0.51	2
x10	10.0.0.56	5

B. *Destination Address based Entropy Measurement*

From the Table I, it is seen that the destination IP addresses are repeated more than once in the hash table. Window size or hash table size can be represented by equation (1) which has x_n destination IP and y_n source IP addresses.

$$W = (x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n) \quad (1)$$

So for calculating the entropy of the hash table, the probability of the individual events i.e. the probability of the same destination address happening in the hash table is calculated using the probability equation (2)

$$P_i = \frac{x_i}{N} \quad (2)$$

Where N is the size of the hash table or window size, $i = 1$ *to* N and x_i is the number of occurrence. After the sum of the entropy is calculated by using the equation (3) where E is the entropy.

$$E = \sum_{i=1}^{N} P_i log P_i \quad (3)$$

So for each of the hash table this entropy value is calculated. If one of the destination IP happens more enormously than others then the entropy E of that hash table at that individual switch port decreases as the individual probability increases.

C. *DDoS detection based on Entropy and Mitigation*

Fig. 2 shows the proposed algorithm to detect and mitigate the DDoS attacks. This phenomena of Entropy measurement is employed for detecting the DDoS attack. For evaluating the DDoS attack we set a count value C. The count value C counts 1 whenever the entropy falls below a predefined threshold (which is 0.5 in this case). If the Entropy value E falls below the threshold value for equal to or greater than 5 times consecutively then the C reaches 5 or more. Whenever the C reaches 5 the controller will detect it as a DDoS attack. If the entropy value E again reaches beyond threshold before making it consecutive 5 times, then the count C will be cleared and it will look for the new packet incoming and the corresponding hash table entropy. After detecting the DDoS attack the controller detects the corresponding switch port that is causing the entropy down the threshold and eventually blocks the switch port that is connected to the attacker PC. For those switch port the entropy decreases to zero after blocking as no packets are coming from them. The other ports having the normal PC sending normal traffics will not be blocked.

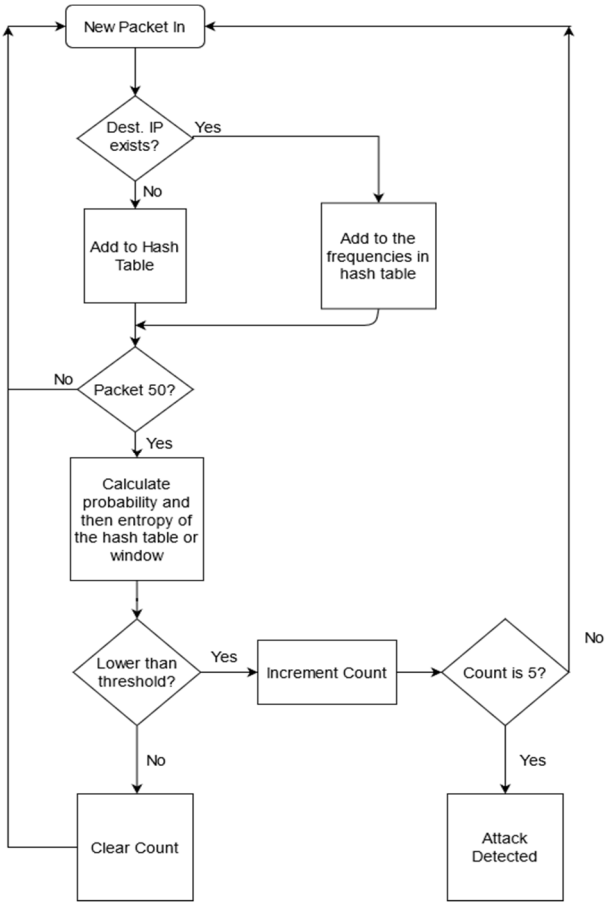

Fig. 2. Proposed Algorithm

IV. EXPERIMENTAL RESULTS AND DISCUSSION

The main topology for simulation in Fig. 3 is created by Mininet emulator [8]. Fig. 4 shows the topology creation process. The topology contains 9 switches, 64 hosts and a remote controller which is Pox controller. Here every host was assigned with a corresponding IP address of its host number like 10.0.0.1 to 10.0.0.64 and the remote controller has the IP address of 127.0.0.1 which is connected at port 6633. Generation of packets is done with the assistance of Scapy tool. Where Scapy is utilized for generation of normal and abnormal packets, sniffing, checking, producing of the parcel and attacking. Here Scapy is utilized for creation of UDP parcels and satirizing the source IP address of the bundles.

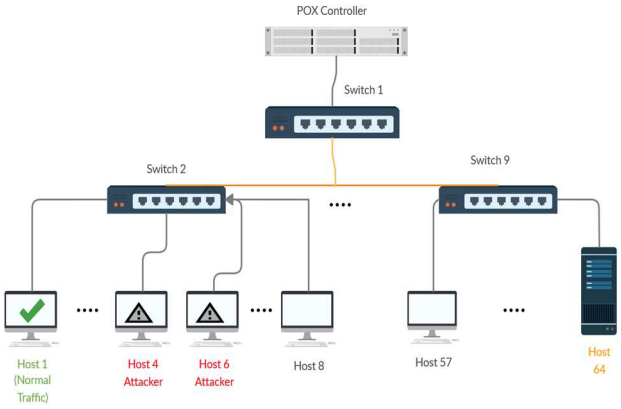

Fig. 3. Dedicated Topology for simulation.

```
63 h64
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s1, s5) (s1, s6) (s1, s7) (s1, s8) (s1, s9) (s2, h1)
 (s2, h2) (s2, h3) (s2, h4) (s2, h5) (s2, h6) (s2, h7) (s2, h8) (s3, h9) (s3, h1
0) (s3, h11) (s3, h12) (s3, h13) (s3, h14) (s3, h15) (s3, h16) (s4, h17) (s4, h1
8) (s4, h19) (s4, h20) (s4, h21) (s4, h22) (s4, h23) (s4, h24) (s5, h25) (s5, h2
6) (s5, h27) (s5, h28) (s5, h29) (s5, h30) (s5, h31) (s5, h32) (s6, h33) (s6, h3
4) (s6, h35) (s6, h36) (s6, h37) (s6, h38) (s6, h39) (s6, h40) (s7, h41) (s7, h4
2) (s7, h43) (s7, h44) (s7, h45) (s7, h46) (s7, h47) (s7, h48) (s8, h49) (s8, h5
0) (s8, h51) (s8, h52) (s8, h53) (s8, h54) (s8, h55) (s8, h56) (s9, h57) (s9, h5
8) (s9, h59) (s9, h60) (s9, h61) (s9, h62) (s9, h63) (s9, h64)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18 h19 h20 h21 h22 h
23 h24 h25 h26 h27 h28 h29 h30 h31 h32 h33 h34 h35 h36 h37 h38 h39 h40 h41 h42 h
43 h44 h45 h46 h47 h48 h49 h50 h51 h52 h53 h54 h55 h56 h57 h58 h59 h60 h61 h62 h
63 h64
*** Starting controller
c0
*** Starting 9 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 ...
*** Starting CLI:
mininet>
```

Fig. 4. Topology creation process in Mininet.

Normal Traffic Generation:

Function 1: Traffic launch

Input: Set the start and end of the random source IP addresses excluding the values [1, 2, 10, 127, 169, 172, 192] for the first octet of the IP address. Output: Traffic launch in the network to random destination addresses in the fixed range.

1) Import Scapy for packet generation;
2) Open required module to open exact interface;
3) Generate UDP packets having the fixed destination;
4) IP addresses range randomly;
5) Send those packets to random hosts by using send() between a fixed intervals.

Attack Traffic Generation:

Function 2: Attack Launch

Input: IP address of the target host. Output: UDP packets with random source IP addresses having single destination IP address of the target.

1) Import Scapy;
2) Open required module to open exact interface;
3) Generate UDP packets having same destination IP address;
4) Send those packets to the target host by using send() between a fixed intervals.

Attack Detection and Mitigation:

Function 3: Entropy based detection and mitigation

Input: DDoS attack to a single host. Output: Calculate hash table entropy according to timer function and based on entropy block that port which is flooding UDP packet.

1) For every new packet in event, check if destination IP repeats and increment Packet count accordingly;
2) If packet count = 50, Calculate the Entropy E;
3) Else Clear Entropy value and go to step 1;
4) If E < threshold then Increment Count
5) If Count is less than Count threshold then normal execution;
6) Else Attack detected and Block the port;
7) Else if E >threshold;
8) Clear Count
9) Clear Entropy and Go to Step 1.

Fig. 5 shows the scheme for first 1000 packets coming from each hosts the entropy was calculated for each set of 50 packets individually. From the graph we can see that for Host1 as it was sending normal traffic throughout the time. The packets coming from Host1 has the most variation in the destination addresses its entropy is always higher than the threshold: which is actually greater than 1. For Host4, it is seen that the entropy is greater than the threshold value of 0.5 for 500 packets. But after the passing 500 packets, some of the destination addresses of the packets coming from Host4 starts repeating. So the variation in the hash table of the switch port connected with the Host4 decreases. For this reason entropy decreases. We can see at the 550th packet entropy decreases to less than 0.5. Then the counter increases its count to 1. As we can see after that the entropy of the Host4 cannot cross the threshold up again, the counter value starts increasing.

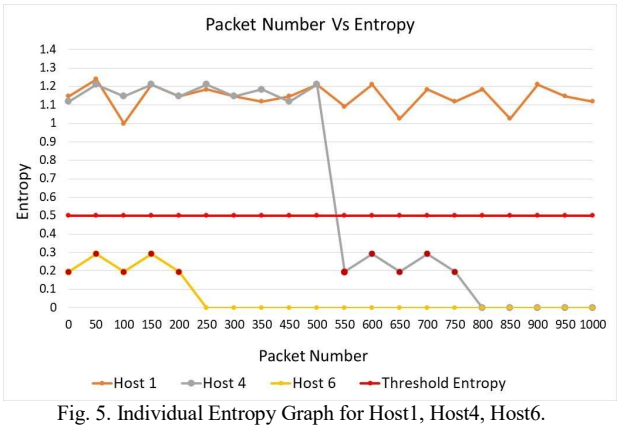

Fig. 5. Individual Entropy Graph for Host1, Host4, Host6.

Whenever the counter value reaches 5 the controller detects it as a DDoS attack and the switch port carrying the packets of Host4 is blocked. So that the entropy decreases to zero. If we look at the Host6 we also can see the same phenomena. For Host6 it starts sending malicious packets from the very beginning. So that after consecutive lower entropy value than 0.5 i.e. 5 counts the port is blocked just after sending 250 packets. The time interval counted between individual packets coming from Host1 is kept 0.1 and the time interval of the individual packets coming from Host4 and Host6 is kept 0.025 sec. Here an analytical comparison also presented in Table II and from the comparison proposed method shows promising results in this particular field.

TABLE II. A COMPARATIVE STUDY WITH RECENT WORKS

Research	Method	Comparative outcomes
Singh *et al.* [3]	MLP classification and GA algorithm	Detects the attack by analyzing HTTP GET requests variation
Yadav *et al.* [6]	Entropy threshold	Blocks the particular port for a certain time
Dalou *et al.* [10]	Adaptive Entropy Threshold	Blocks traffic from the attacking subnet
Hong *et al.* [13]	Counting incomplete HTTP requests	Blocks attack by changing the flow rules
Buragohain *et al.* [14]	Comparing with the legitimate traffic flow	Blocks attack after some repeated flow comparison
Ours	**Hash table based entropy measurement**	**Instantly blocks the suspected port more precisely**

V. CONCLUSION AND FUTURE WORKS

In this paper, a dedicated topology was created having a SDN POX controller and nine switches and those are connected to 64 hosts. This topology was made for simulation of the proposed scheme. The switches were OpenFlow virtual switches and the emulator used for creating topology was Mininet which has a Python API. For detecting and stopping malicious traffic flow or the DDoS here an algorithm was used which is based on entropy measurement. Here we used a hash table based entropy measurement system that enhanced the efficiency in detecting and mitigating the DDoS attack. The developed script was run on the API of the SDN controller that has access to the switches which are connected to the controller. After that we observed the performance of the host's packet sending capability where two of them were malicious traffic sender. We have seen that the method detected the malicious traffics efficiently and blocked the senders immediately. But there can be limitations such that if a sender sends legitimate packets having the interval of 0.025 seconds to the same destination, then it will be detected as DDoS attack. In future we want to deploy more than one controller in a dedicated topology for detecting and mitigating the DDoS attack. We also want to make the system more robust that it will only block the criminal IP for a certain hour. We also want to evaluate the performances of using more than one SDN controller in the network.

REFERENCES

[1] S. Karimunnisa and V. Kompalli, "Cloud computing: Review on recent research progress and issues," *Int. J. Adv. Trend. Comp. Sci. Eng.,* vol. 8, No. 2, pp. 216-223, 2019.

[2] M. Kazim and S. Y. Zhu,"A survey on top security threats in cloud computing," *Int. J. Adv. Comp. Sci. Appl.,* vol. 6, no. 3, pp. 109-113, 2015.

[3] J. k. Singh, K. Thongam and T. De, "Entropy-based application layer ddos attack detection using artificial neural networks," *Entropy,* vol. 18, No.10, Art. 350, 2016.

[4] B. S. K. Devi, G. Preetha, G. Selvaram and S. Mercy Shalinie, "An impact analysis: Real time DDoS attack detection and mitigation using machine learning," in *International Conference on Recent Trends in Information Technology*, Chennai, India, pp. 1-7, 2014.

[5] N. Z. Bawany, J. A. Shamsi, and K. Salah, "Ddos attack detection and mitiga- tion using sdn: Methods, practices, and solutions," *Arab. J. Sci. Eng.,* vol. 42, pp. 425-441, 2017.

[6] S. K. Yadav, P. Suguna and R. L. Velusamy, "Entropy based mitigation of Distributed-Denial-of-Service (DDoS) attack on Control Plane in Software-Defined-Network (SDN)," in *10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kanpur, India, pp. 1-7, 2019.

[7] H. M. Noman and M. N. Jasim, "Pox controller and open flow performance evaluation in software defined networks (sdn) using mininet emulator," in *IOP Conf. Ser.: Mater. Sci. Eng.* vol. 881, Baghdad, Iraq, 2020.

[8] K. K. Sharma and M. Sood, "Mininet as a container based emulator for software defined networks," *Int. J. Adv. Res. Comp. Sci. Soft. Eng.,* vol. 4, pp. 681-685, 2014.

[9] R. Sahay, G. Blanc, Z. Zhang and H. Debar, "Towards autonomic DDoS mitigation using software defined networking," in *Proc. of 2015 NDSS workshop on Security of Emerging Networking Technologies (SENT 2015),* San Diego, Ca, United States, 2015.

[10] J. Dalou, B. Al-Duwairi and M. Al-Jarrah, "Adaptive entropy-based detection and mitigation of ddos attacks in software defined networks," *Int. J. Comp.,* vol. 19, pp. 399-410, 2020

[11] K. Kalkan, G. Gur and F. Alagoz, "Defense Mechanisms against DDoS Attacks in SDN Environment," in *IEEE Communications Magazine*, vol. 55, no. 9, pp. 175-179, Sept. 2017.

[12] R. Yaegashi, D. Hisano and Y. Nakayama, "Light-Weight DDoS Mitigation at Network Edge with Limited Resources," in *IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, pp. 1-6, 2021.

[13] K. Hong, Y. Kim, H. Choi and J. Park, "SDN-assisted slow HTTP DDoS attack defense method," in *IEEE Communications Letters*, vol. 22, no. 4, pp: 688-691, April 2018.

[14] C. Buragohain, and N. Medhi, "FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers," in *Proc. of the 3rd International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, pp. 519-524, February 2016.