Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 06:2023
DOI 10.17605/OSF.IO/D26U9

DDoS ATTACK DETECTION IN SDN – ENTROPY BASED APPROACH

HITESH KUMAR

Faculty of Engineering Science and Technology, Hamdard University, Karachi, Pakistan. *Corresponding Author Email: Hitesh.ned@gmail.com

ADNAN AHMED SIDDIQUI

Faculty of Engineering Science and Technology, Hamdard University, Karachi, Pakistan.

SYED SAJJAD HUSSAIN RIZVI

Department of Computer Science, SZABIST, Karachi, Pakistan.

Abstract

Software Defined Networking (SDN) is the most recent, evolving, and emerging technology nowadays in computer networks. It has replaced the traditional networks in which the control and data planes were tightly coupled with decoupling the control and data planes. SDN provides complete network visibility, centralized management, a global view of the network, the programmability of the network devices, and dynamic updates of forwarding rules. Although SDN has provided a great advantage, there are many security issues like data modification, data leakage, configuration issues, denial of service (DoS), distributed denial of service (DDoS) attacks, and unauthorized access to network devices. DDoS is the most lethal, restricting authorized users from gaining access. In this paper, a high-rate DDOS attack is detected by using an entropy-based approach. Mininet emulator used for creating topology and defining rules. Furthermore, attack traffic was generated from different sources on a single destination. In the future, attack traffic will be blocked by turning off the incoming port on the switch, and attack traffic rules will be deleted from the flow table using a suitable algorithm.

Keywords: SDN, DoS, DDoS, Entropy, Mininet

1. INTRODUCTION

In this current era of cloud computing, cloud, and network service providers must fulfill network requirements, i.e. network and cloud security, QoS (Quality of Service), bandwidth, and reliability. Network architecture should be highly flexible and elastic according to the need to achieve these requirements. However, in traditional networks, commonly used devices are routers, and switches have many drawbacks, such as integrating software and hardware with the same device. They are tightly coupled, most devices are manufacturer proprietary, and we cannot program or change their functionality; complicated protocols are integrated into the devices. All these drawbacks make traditional networks inappropriate for fulfilling cloud service providers' requirements.

SDN (Software defined networking) is the evolving cloud and computer networking technology. The main idea of SDN is the decoupling of the control plane and data plane. The centralized controller takes all the decisions, and instructions are forwarded to the data plane. Network devices (Routers and Switches) in the data plane are not as intelligent as in traditional networks, and their role is to deliver the data as per the instruction given by the centralized controller placed in the control plane.

SDN is under core consideration from both industry and academic points of view as it has gained significant importance due to its unique characteristics. SDN definition, according to ONF (Open Networking Foundation), is “In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications [1].

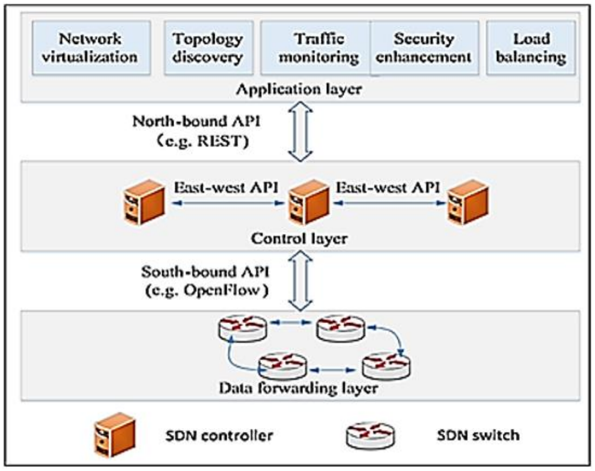


Figure 1: SDN Architecture [2]

The architecture of SDN consists of three main layers: Application Layer, Control Layer, and Infrastructure Layer:

Application Layer: It consists of various SDN applications like Network Security applications, routing services, network management policies, Quality of Service policies, traffic monitoring applications, and load balancing.

Control Layer: This is the core layer of SDN infrastructure. It runs the Network Operating System (NOS) and contains a logically centralized control framework. The control layer takes all the core and forwarding decisions. SDN is a logically centralized controller that is the main device working on this layer and has flow table policies, flow table manager, SDN policies, network security monitoring, and security privileges. The infrastructure layer contains forwarding/hardware devices like switches and routers. All the end devices (Computers, laptops, printers, scanners, servers, and wireless devices) are connected to these switches. These switches and routers forward the incoming traffic to the SDN controller for delivering decisions. Delivering devices contain Flow tables and forward traffic as per the table. These forwarding devices in SDN are less intelligent than those in traditional networks and, hence, cost-effective.

Infrastructure layer: This layer contains forwarding/hardware devices like switches and routers. All the end devices (Computers, laptops, printers, scanners, servers and wireless devices) are connected to these switches. These switches and routers forward the incoming traffic to SDN controller for forwarding decisions. Forwarding devices contain Flow tables and forward traffic as per the table. These forwarding devices in SDN as not so intelligent as compare to the devices in traditional networks and hence, are cost effective devices. SDN has characteristics like a Simplified network, a Logically centralized controller, an Open, programmable interface, a Switch management protocol, Virtualized logical network, Centralized monitoring units, a Global network view, and Increased control capabilities. The list of SDN challenges consists of Vulnerable Controllers, Risk caused by open, programmable interfaces, Switch design, Controller platforms, Resilience, Scalability, Performance evaluation, Security, and Migration to SDN [2].

- **Potential security threats in SDN networks are:**
  - **Malicious applications:** Controller provides an abstraction between the application and the data plane; in this way, 3rd party applications are integrated into the SDN network. A malicious application can take control of the SDN network or add vulnerabilities to the network.
  - **Data Leakage:** An attacker can leak the data by gaining additional information about the network. An attacker can analyze the packet processing time and specific actions applied to the packet and inform the data/information regarding the packet.
  - **Unauthorised access:** The 3rd party applications running over the control plane have the accessibility of the data plane through the controller, and this can provide unauthorized access to the network if an attacker impersonates an application or controller.
  - **Denial of Service (DoS):** An unauthorized user tries to exhaust system resources and make the system unavailable for actual users. This attack can be performed by a single host or multiple hosts sending requests simultaneously. This attack hits on system availability and makes the system unavailable for authorized users [3].
  - **Configuration Issues:** Misconfiguration or incorrect use of policies can severely impact SDN infrastructure as attackers continuously try to find vulnerabilities in the network. Network policies and security policies are also constantly evolving to tackle the attacks. If these security policies are misconfigured and wrongly implemented, then attackers can take advantage of these policies and can attack the network.
  - **Data Modification:** SDN controller has the authority to control the whole SDN infrastructure and takes all the controlling and forwarding decisions. Attacking and taking control of the controller will logically provide complete control of the SDN network. In this way, an attacker can inject forwarding rules of its own interest, forward all the data to itself, and modify it [4].
- **DDoS Attack**

DDoS is the most severe network attack in which network resources are exhausted by sending thousands of fake packets, and system resources cannot process legitimate or authorized traffic flows [5]. The reactive packet processing mechanism is introduced by open flow protocol in SDN. Available flow switches maintain a flow table, and packets are processed using the flow tables [6]. The button takes action once the entry is matched in the flow tale, and the packet is forwarded to the controller as packet-in for delivering a decision in case the package is not fit in the switch flow table. The controller calculates the flow rule against each packet-in message and installs that rule on open flow switches with a flow-mod message. Packets are stored in switch buffer memory when there is no match inflow table, and only packet headers are forwarded to the controller. These hundreds of thousands of packet-in messages can increase the computational power of the controller to process these requests, and controller performance can be degraded, the communication channel can be congested and switch buffer memory can be exhausted [7].

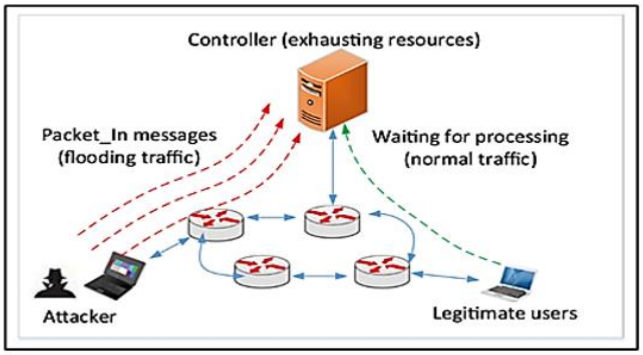


Figure 2: DoS/DDoS on SDN Controller

Most network devices in network architecture are resource constraints; hence it is the main reason for the DDoS attacks on network architecture. These attacks mainly exhaust resources like Processing power, memory & bandwidth of the links. Different DDoS attacks have other purposes, as many speedily grow in today’s internet era [8].

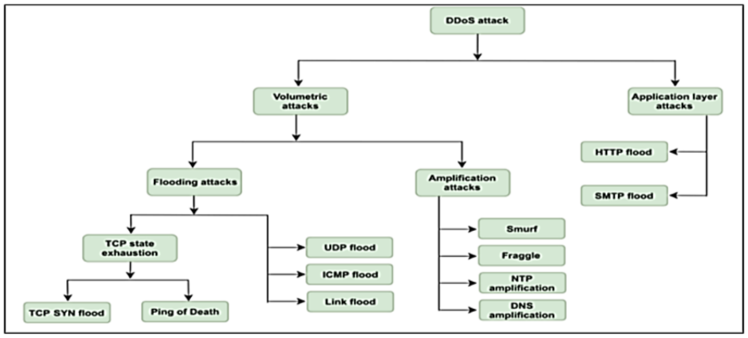


Figure 3: DoS/DDoS Attack Taxonomy [9]

Many authors have discussed SDN & DDoS attacks in detail and presented different algorithms and techniques to detect and mitigate the low-rate and high-rate DDoS attacks on SDN networks. All the algorithms/techniques have different approaches like Entropy-based algorithms, machine learning algorithms, deep learning neural networks, intrusion detection system (IDS) and intrusion prevention system (IPS) models etc., to detect and mitigate DDoS attacks efficiently.

2. RELATED WORK

In [10] author has proposed Multi-Controller based solution to detect and mitigate DDoS attacks in SDN. Two methods are presented to handle DDoS attacks: Entropy of destination IP packet and Window initiation rate for early detection of DDoS attack. The simulation tool used is Mininet, and Controller used in the topology is Floodlight.

In [11], Renyi Joint Entropy-based solution is proposed to protect SDN controllers from DDoS attacks. A dynamic threshold algorithm with a new rule-based detection mechanism is proposed. The author has used the Mininet simulation tool with a POX controller to produce the results.

In [12], Cyber-attacks, especially DDoS attacks, is the major challenge as they restrict system availability. An online entropy-based system is proposed to detect flooding attacks. The offered modules consist of 5 modules: Feature Extraction, Suspicious Activity Detection, Attack detection, alert generation, and threshold update.

In [13], DDoS attack in the Covid-19 scenario for small entrepreneurs is discussed. Small entrepreneurs need more security resources to detect DDoS attacks. The author has proposed a filtering mechanism that efficiently identifies DDoS attacks in COVID-19 scenarios. Entropy and machine learning-based techniques are proposed. A cost-effective framework with a 92.8% accuracy rate is presented. Support Vector Machine (SVM), a machine learning-based approach, is offered.

In [14], DDoS attack has been detected in Open Flow-based networks. DDoS attack in SDN is the major challenge due to centralized controller. This paper proposes a method using entropy and flow statistics obtained from Switch's flow table with the help of the Mininet simulation tool and POX controller.

In [15], DDoS is the most lethal attack in SDN networks. An Entropy-based DDoS attack detection scheme is proposed with a Light and effective method to detect DDoS attacks early by calculating the entropy of the destination IP. The author has used the Mininet simulation tool with the RYU controller in the paper.

In [16], TCP SYN Flood DDoS Attack Detection and Mitigation in SDN Environment has been discussed in the paper. The author has proposed lightweight Entropy based statistical approach to detect and mitigate TCP SYN Flood DDoS attacks. Three-phased detection schemes to minimize false positive rates have been used. The author uses the Mininet Simulation tool to produce the desired results.

3. METHODOLOGY

Information theory is the most widespread method to detect DDoS attacks. Entropy is the measure of randomness in the field of information theory. Entropy provides a fast & convenient manner of filtering suspicious flows. It is a lightweight method and makes it possible to identify DDoS attacks early. In entropy, the randomness of a packet with a destination IP address in the network is calculated. Windows Size and the threshold value are the two main components to detect DDoS attacks using entropy in SDN infrastructure. Shannon equation is used in this paper to calculate entropy:

$$H(X) = -\sum_{i} P(xi) \log P(xi)$$

H is the Entropy value, and P is the probability of each IP address.

First of all new packets will enter into the system whenever a host tries to communicate with another host in the network. The switch will check the destination IP address in its forwarding table, if the destination IP exists, the count is incremented in the forwarding table of the Open flow Switch. If the value does not exist in the forwarding table then a new entry will be added in the table. Windows Size (No. of packets in a Window) after adding an entry in the table will be checked, if it is equal to or greater than 50 then entropy will be calculated for the window size. If entropy is lower than the threshold value then counts (it is a simple integer and is incremented every time when entropy is lower than the threshold value) is incremented and if count becomes 5 then a DDoS attack is detected, if count is less than 5 than it will be treated as normal or legitimate traffic. If entropy is greater than the threshold value then it will also be treated as legitimate traffic and the whole process will restart from New Packet-In.

4. SIMULATION & RESULTS

- **Topology**

The author has used the Mininet simulation tool to create and deploy network topology.

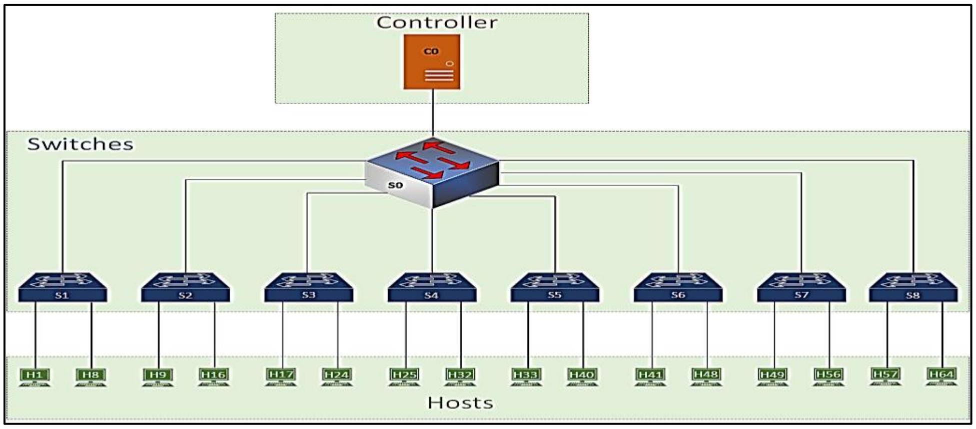


Figure 4: Network Topology

System Specification:

Processor: 2.4 GHz

Hard Disk: 150GB

RAM: 6GB

Operating System: Linux (UBUNTU 16.04)

Programming Language: Python

Simulation Tool: Mininet

Controller Type: POX Controller

No. of Controller = 1

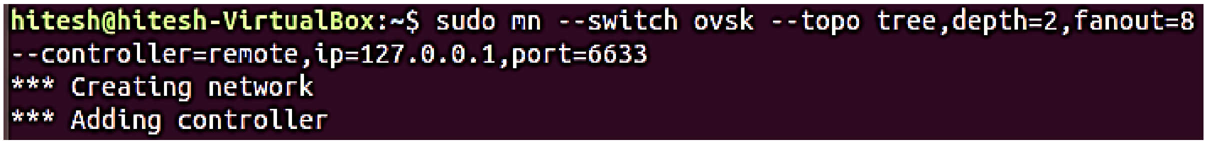
No. of Switches = 9

No. of Hosts = 64

Windows Size = 50

Entropy Threshold Value = 0.5

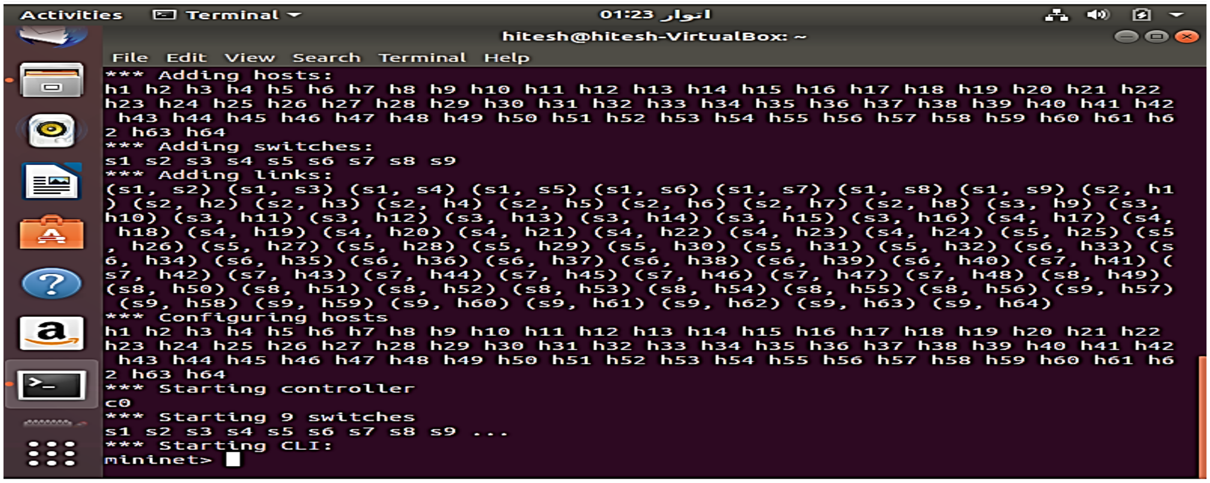
In this paper, the author has created a tree topology on Mininet using one controller, nine switches, and 64 hosts :



```
hitesh@hitesh-VirtualBox:~$ sudo mn --switch ovsk --topo tree,depth=2,fanout=8
--controller=remote,ip=127.0.0.1,port=6633
*** Creating network
*** Adding controller
```

Figure 5: Topology Creation

Hosts, Switches, Links, and Controllers are added to the topology. Hosts are configured, and switches and controller are started:

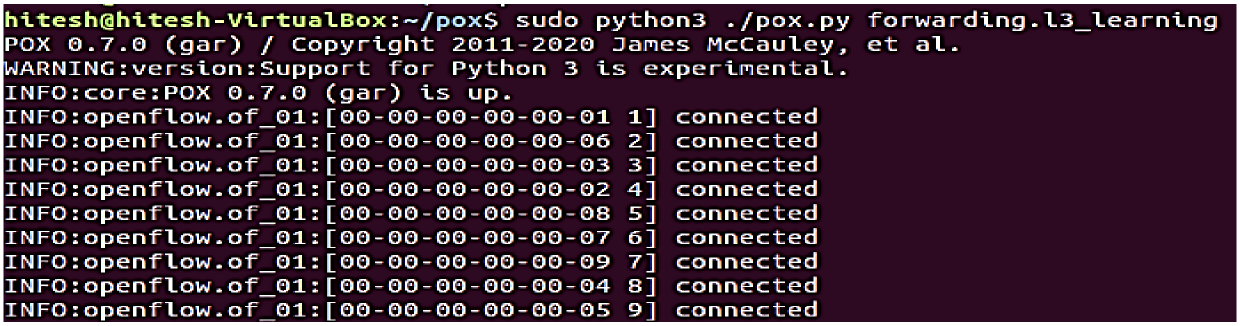


Activities
Terminal
01:23 انوار
hitesh@hitesh-VirtualBox: ~
File Edit View Search Terminal Help

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

Figure 6: Network Configuration

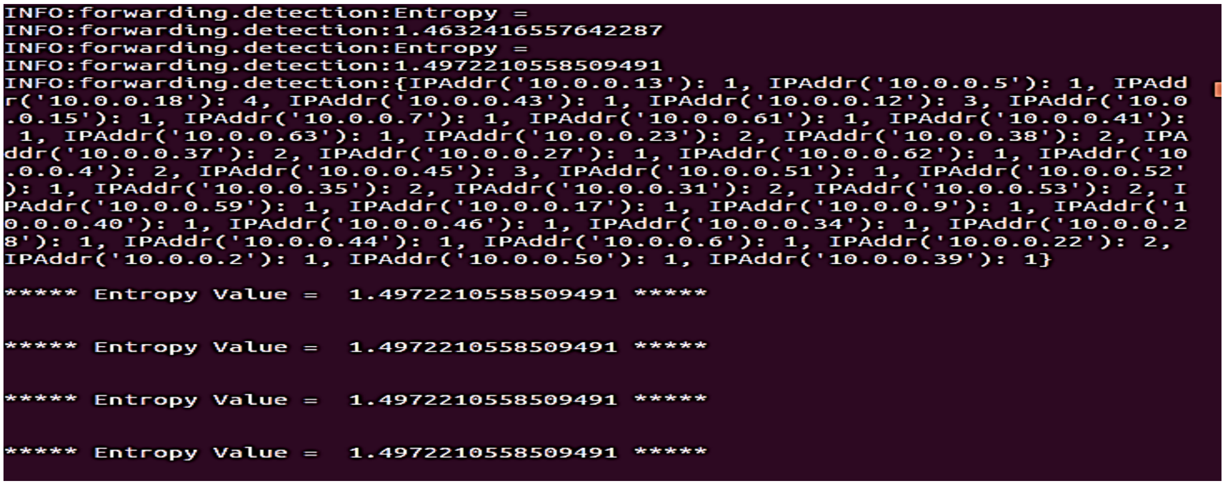
POX controller is started, and nine switches are connected to the controller:



```
hitesh@hitesh-VirtualBox:~/pox$ sudo python3 ./pox.py forwarding.l3_learning
POX 0.7.0 (gar) / Copyright 2011-2020 James McCauley, et al.
WARNING:version:Support for Python 3 is experimental.
INFO:core:POX 0.7.0 (gar) is up.
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:openflow.of_01:[00-00-00-00-00-06 2] connected
INFO:openflow.of_01:[00-00-00-00-00-03 3] connected
INFO:openflow.of_01:[00-00-00-00-00-02 4] connected
INFO:openflow.of_01:[00-00-00-00-00-08 5] connected
INFO:openflow.of_01:[00-00-00-00-00-07 6] connected
INFO:openflow.of_01:[00-00-00-00-00-09 7] connected
INFO:openflow.of_01:[00-00-00-00-00-04 8] connected
INFO:openflow.of_01:[00-00-00-00-00-05 9] connected
```

Figure 7: Network configuration of customized topology

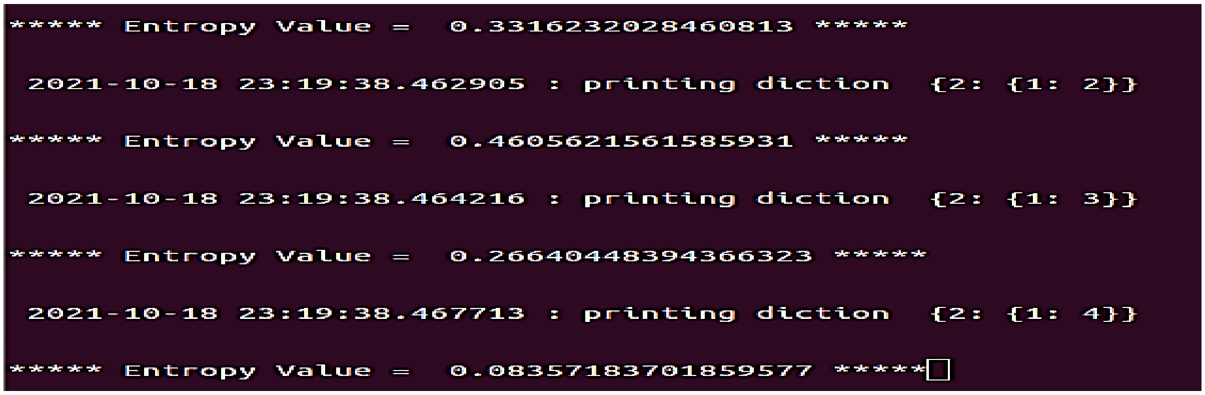
Traffic is generated from one of the hosts to the random destinations, and entropy is calculated against each packet.



```
INFO:forwarding.detection:Entropy =
INFO:forwarding.detection:1.4632416557642287
INFO:forwarding.detection:Entropy =
INFO:forwarding.detection:1.4972210558509491
INFO:forwarding.detection:{IPAddr('10.0.0.13'): 1, IPAddr('10.0.0.5'): 1, IPAdd
r('10.0.0.18'): 4, IPAddr('10.0.0.43'): 1, IPAddr('10.0.0.12'): 3, IPAddr('10.0
.0.15'): 1, IPAddr('10.0.0.7'): 1, IPAddr('10.0.0.61'): 1, IPAddr('10.0.0.41'):
 1, IPAddr('10.0.0.63'): 1, IPAddr('10.0.0.23'): 2, IPAddr('10.0.0.38'): 2, IPA
ddr('10.0.0.37'): 2, IPAddr('10.0.0.27'): 1, IPAddr('10.0.0.62'): 1, IPAddr('10
.0.0.4'): 2, IPAddr('10.0.0.45'): 3, IPAddr('10.0.0.51'): 1, IPAddr('10.0.0.52'
): 1, IPAddr('10.0.0.35'): 2, IPAddr('10.0.0.31'): 2, IPAddr('10.0.0.53'): 2, I
PAddr('10.0.0.59'): 1, IPAddr('10.0.0.17'): 1, IPAddr('10.0.0.9'): 1, IPAddr('1
0.0.0.40'): 1, IPAddr('10.0.0.46'): 1, IPAddr('10.0.0.34'): 1, IPAddr('10.0.0.2
8'): 1, IPAddr('10.0.0.44'): 1, IPAddr('10.0.0.6'): 1, IPAddr('10.0.0.22'): 2,
IPAddr('10.0.0.2'): 1, IPAddr('10.0.0.50'): 1, IPAddr('10.0.0.39'): 1}

***** Entropy Value =  1.4972210558509491 *****

***** Entropy Value =  1.4972210558509491 *****

***** Entropy Value =  1.4972210558509491 *****

***** Entropy Value =  1.4972210558509491 *****
```

Figure 8: Entropy Calculation of Normal Traffic

Now, attack traffic is generated. In the attack scenario, the Entropy value is decreased than the threshold.



```
***** Entropy Value =  0.3316232028460813 *****

 2021-10-18 23:19:38.462905 : printing diction  {2: {1: 2}}

***** Entropy Value =  0.4605621561585931 *****

 2021-10-18 23:19:38.464216 : printing diction  {2: {1: 3}}

***** Entropy Value =  0.26640448394366323 *****

 2021-10-18 23:19:38.467713 : printing diction  {2: {1: 4}}

***** Entropy Value =  0.08357183701859577 *****
```

Figure 9: Entropy Calculation of Attack Traffic

5. CONCLUSION

In this paper, the author has demonstrated an implementation of the Entropy-based SDN network to detect DDoS attacks using the Mininet Simulation Tool. The simulation result shows that the Entropy value is greater than the threshold in the case of normal traffic, and the Entropy value decreases than the threshold value when there is an attack on the SDN network. Attack traffic was generated from different sources on a single destination. Attack traffic was blocked by turning off the incoming port on the switch. Attack traffic rules were deleted from the flow table.

6. FUTURE RECOMMENDATIONS

The controller is the key component in SDN infrastructure which controls the whole SDN network. The entire SDN network will become unavailable if the SDN controller becomes unavailable due to a DDoS attack. Therefore, the following research areas can be considered as future work:

- Multi-Controller based SDN infrastructure to protect SDN from DDoS attack
- In this paper, single node attack is presented, and the detection of a multi-node attack and controller performance is still a challenge
- Multiple attack scenarios can be considered to check controller performance and availability

References

1. Scott-Hayward, S., Natarajan, S., & Sezer, S. (2016). A survey of security in software-defined networks. *IEEE Communications Surveys & Tutorials*, *18*(1), 623-654.
2. Shu, Z., Wan, J., Li, D., Lin, J., Vasilakos, A. V., & Imran, M. (2016). Security in software-defined networking: Threats and countermeasures. Mobile Networks and Applications, 21(5), 764-776.
3. R. B. Shohani, S. Mostafavi, and V. Hakami, “A Statistical Model for Early Detection of DDoS Attacks on Random Targets in SDN,” *Wirel. Pers. Commun.*, no. 0123456789, 2021.
4. Nada M. AbdelAzim, Sherif F. Fahmy, Mohammed Ali Sobh, Ayman M. Bahaa Eldin “A hybrid entropy-based DoS attacks detection system for software-defined networks (SDN): A proposed trust mechanism”, N.M. AbdelAzim et al. / Egyptian Informatics Journal 22 (2021) 85–90
5. Shanshan Yu, Jicheng Zhang, Ju Liu, Xiaoqing Zhang, Yafeng Li, and Tianfeng Xu “A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN”, Yu et al. J Wireless Com Network (2021) 2021:90.
6. A. Banitalebi Dehkordi, M. R. Soltanaghaei, and F. Z. Boroujeni, *The DDoS attacks detection through machine learning and statistical methods in SDN*, vol. 77, no. 3. Springer US, 2021.
7. Hamidreza Lotfalizadeh, Dongso S. Kim, Investigating Real-Time Entropy Features of DDoS Attack Based on Categorized Partial-Flows, IEEE, 2020.
8. Swami, R., Dave, M., & Ranga, V. (2019). Software-defined Networking-based DDoS Defense Mechanisms. *ACM Computing Surveys, 52(2), 1–36. doi:10.1145/3301614*

9. Parisa Valizadeh, Ahmad Taghinezhad-Niar, “DDoS Attacks Detection in Multi-Controller Based Software Defined Network,” Proc. - 2022 8th International Conference on Web Research (ICWR) | 978-1-6654-6626-4/22/$31.00 ©2022 IEEE

10. Aladaileh, M.A.; Anbar, M.; Hintaw, A.J.; Hasbullah, I.H.; Bahashwan, A.A.; Al-Sarawi, S. “Renyi Joint Entropy-Based Dynamic Threshold Approach to Detect DDoS Attacks against SDN Controller with Various Traffic Rates” Appl. Sci. 2022, 12, 6127.

11. Loïc D. Tsobdjou, Samuel Pierre, and Alejandro Quintero “An Online Entropy-Based DDoS Flooding Attack Detection System With Dynamic Threshold,” IEEE Transactions on Network And Service Management, Vol. 19, NO. 2, JUNE 2022

12. Soodeh Asgari, Behzad Akbari, “DDoS Attack Detection in OpenFlow Based Networks,” 2022 27th International Computer Conference, Computer Society of Iran (CSICC) | 978-1-6654-8027-7/22/$31.00 ©2022 IEEE

13. Mayadah A. Mohsin and Ali H. Hamad, “Implementation of Entropy-Based DDoS Attack Detection Method in Different SDN Topologies,” American Academic Scientific Research Journal for Engineering, Technology, and Sciences ISSN (Print) 2313-4410, ISSN (Online) 2313-

14. Sehrish Batool, Farrukh Zeeshan Khan, Syed Qaiser Ali Shah, Muneer Ahmed, Roobaea Alroobaea, “Lightweight Statistical Approach towards TCP SYN Flood DDoS Attack Detection and Mitigation in SDN Environment,” Hindawi Security and Communication Networks Volume 2022, Article ID 2593672.

15. Akshat Gaurav, Brij B. Gupta, “A novel approach for DDoS attacks detection in COVID-19 scenario for small entrepreneurs,” https://doi.org/10.1016/j.techfore.2022.121554, Elsevier.

16. A. Mishra, N. Gupta, and B. B. Gupta, “Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller,” *Telecommun. Syst.*, vol. 77, no. 1, pp. 47–62, 2021, doi: 10.1007/s11235-020-00747-w.