IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Review of Computer Engineering Studies
Vol. 6, No. 2, June, 2019, pp. 29-38
Journal homepage: http://iieta.org/journals/rces

Implementation of Entropy-based Distributed Denial of Service Attack Detection Method in Multiple POX Controllers

Mahmood Z. Abdullah, Nasir A. Al-awad, Fatima W. Hussein*

Computer Engineering Department, College of Engineering, Al-Mustansiriyah University, Baghdad, 10001, Iraq

Corresponding Author Email: fatima.wadaa.hussein@gmail.com

https://doi.org/10.18280/rces.060201

Received: 26 March 2019
Accepted: 10 June 2019

Keywords:
attack traffic, DDoS, Mininet, normal traffic, sFlow-RT, software defined networking (SDN)

ABSTRACT

Software Defined Networks (SDN) differs from traditional networks, it splits the control from the data plane. The controller is an essential and important part in SDN architecture; and it is exposed to many potential security challenges, one of such critical challenges is the impact of Distributed Denial of Service (DDoS) attack. The objective of this study is to implement an Entropy-based detection algorithm for DDoS attacks in SDN networks using POX controller to improve the network security, and to test the performance of this algorithm using POX controller in different topologies and different number of controllers. In this paper the Entropy-based detection algorithm was carried out in different tests that include connecting POX controller to single topology with 64 hosts and then different number of POX controller had been connected to linear topology with 64 hosts. Through this study, it was found that the Entropy detection method works better in lightly loaded network and the results indicate that increasing the number of controllers can improve the security of the network. This paper introduces a new contribution in implementing a statistical DDoS detection method in multiple POX controllers and in different topologies to improve the security of the SDN network.

1. INTRODUCTION

Different from the traditional architecture of network devices where data and control plane are joined in the same device, a new networking architecture called SDN appeared in the last ten years and separated the control plane from the data plane. In SDN, network devices like switches and routers reside in the data plane and they become simple packet forwarding devices, and network intelligence in the form of software control program, named the controller, reside in the control plane [1]. Figure 1, shows the difference between the architecture of SDN and traditional network.

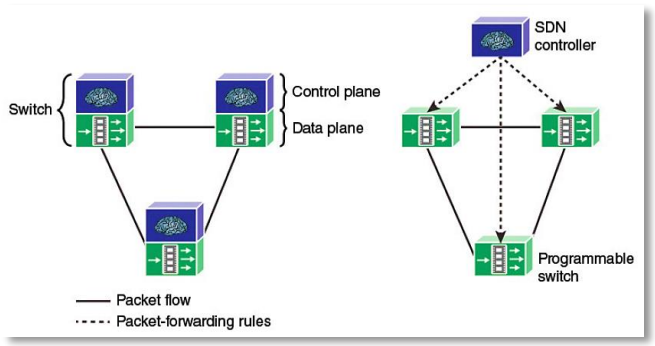


Figure 1. Traditional network vs. SDN [2]

SDN is introduced to support the continuous evolutions of current networking architectures and to achieve new requirements. The big aim behind SDN is to give an open interface to enable the development of software that controls the connectivity among network resources and flow of network traffic [3].

In SDN, network applications are not residing on the actual devices, and must only interface with them through the controller, for these centralized applications the network appears to be one big switch/router, there could be 3 devices on the network or 30,000, it's all the same. This provides simpler changes, upgrades, additions, and configurations than before [4]. For all the properties that SDN offers, both industry and academia have become interested in it [5].

However, the SDN suffers from problems that is common to any new technology, SDN centralized controller have many challenges that need to be addressed like scalability, performance, and high availability [6]; in addition, SDN introduces a new security challenges because of properties of decoupling the planes and migrating the functionality of control plane, and SDN may not mesh well with existing network security approaches, one of these security challenges is DDoS attack [2].

DDoS attacks is dangerous threat to network security, it is disastrous and can bring down a server or network very fast, and it forms a challenge to network availability [7-8]. In DDoS, the victim is attacked by two or more compromised devices, the goal of this attack is to make a machine or network resource unreachable to its legitimate user [9-10]. As shown in Figure 2, DDoS attack consists of four elements [11]:

(1) The attacker that is behind the attack.
(2) The master or handler that control multiple zombie hosts and instruct them to perform malicious activities.
(3) The zombie hosts or agents or bots that run the attack and generate packet streams destined for the victim.
(4) The targeted victim.

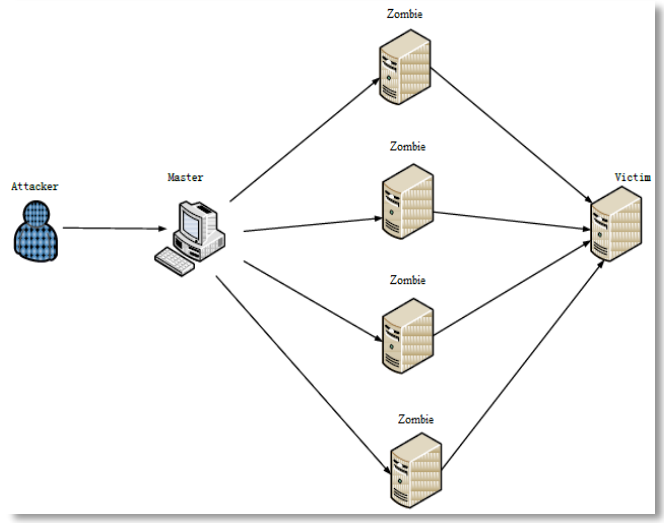


Figure 2. DDoS attack [7]

In SDN, DDoS attack is a big security challenge [12], it mainly occurs due to the flooding of traffic packets from the attacker to the victim to either decrease its performance or stop its service and it will be unavailable for future connection [13-14]; usually, the source addresses of these incoming packets are spoofed, in this case, the switch is not going to find a match in the flow table, and therefore, it will send the packet to the controller; the resources of the controller will be chained into continuous processing by the combination of DDoS spoofed and legitimate packets until they are completely exhausted; this can bring down the controller causing the loss of the SDN architecture and the newly arrived legitimate packets will not be able to reach the controller [7].

There are different kinds of DDOS attacks, such as Hypertext Transfer Protocol (HTTP) flood, Domain Name System (DNS) reflection, ping of death, Transmission Control Protocol (TCP) Synchronize (SYN) flood, and other attacks [15-16]. However, among these DDoS attacks, User Datagram Protocol (UDP), TCP SYN, ICMP, and HTTP flooding are most commonly used [8]. All of these attacks have a mutual aspect of sending large number of traffic loads to the victim and exhausting its resources.

The remainder of this paper is organized as follows: Section 2 presents some previous related works, Section 3 shortly reviews DDoS detection using Entropy, Section 4 shows the methodology of the implementation of the detection method, Section 5 presents the simulation results, and at last, in Section 6 conclusion is presented.

2. RELATED WORKS

A review of some related works that focuses on SDN security against DDoS attacks is presented in this section.

S. Mousavi, in [17] proposed a lightweight and simple DDoS detection mechanism based on Entropy variation of destination Internet Protocol (IP) address for the new coming Packet_In messages, in order to protect the SDN controller. the proposed mechanism was implemented using Mininet and POX controller that was connected to a tree topology and the effectiveness of the method was showed through extensive test cases. This method was able to detect DDoS in its early stages within the first five hundred packets of the attack traffic.

M. Kia, in [13] proposed a method to detect and mitigate DDoS attacks at its early stages by using the Entropy variation, the rate of Flow initiation, and the Flow specifications. The algorithm was implemented using Mininet and POX controller. The high detection rates for different traffic patterns in the results show that the algorithm was able to perform well under different network conditions and it was not limited to a specific network condition.

M. Badrinath et al., in [18] provided an approach to detect DDoS attack using the concept of Entropy that was determined using flow statistics from the switches, and implemented multiple measures to prevent these attacks, like, Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA), installation of a drop entry for blacklisted IP's, and honeypots mechanism. The results showed that the algorithm can detect the attack at early intervals and the attack solution was scalable and optimal for a campus network.

P. Lin et al., in [19] applied Entropy-based DDoS attacks detection method and presented a solution based on Network Function Virtualization (NFV) and SDN, to leverage available resources in a data center to mitigate these attacks. The solution features a strategy to recognize anomalies packets and allocate functions to minimize the latency and footprint of the network. a proof-of-concept design was also simulated in Mininet to demonstrate the feasibility of this solution.

B. Al-mafrachi, in [20] considered multiple forms of threats against SDN controllers initiated from DDoS attacks and conducted a made a comparison between a set of statistical methods for the purpose of DDoS attack detection and identification of switch interfaces that is involved in the attack, and publicly available Defense Advanced Research Projects Agency (DARPA) datasets were used to evaluate these methods.

3. DDOS DETECTION USING ENTROPY

Entropy-based algorithms can be used for the detection of attacks in communication networks. This statistical approach can detect several types of attacks including DDoS, by identifying the randomness of the incoming traffic [21]. The higher the Entropy is, the higher the randomness of the traffic flow; on the contrary, the lower the Entropy, the higher the determinacy of the traffic flow [7].

This paper uses Entropy-based detection algorithm for DDoS attack proposed by [17] and apply it on different test cases. Entropy in this detection algorithm uses two components to calculate the randomness of the incoming packets; the first component is the window size, which represent how many incoming new packets are used in measuring Entropy and it is set to 50; and the second component is the threshold which is set to 1, and it is compared with the Entropy value to decide whether it is an attack or not.

For every new Packet_In message that arrive in the network its header is parsed for the destination IP address; and a hash table of this address and its number of occurrences is created. The hash table can be represented by Eq. (1), where W represents a window with n elements (n equals 50), x represents the destination IP address, and y represents the number of times it appeared. For each destination IP address, the probability is measured using Eq. (2).

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

$$p_i = \frac{x_i}{n} \quad (2)$$

Then after a window of 50 packets, the Entropy (H) for that window is calculated using Eq. (3).

$$H = -\sum_{i=1}^{n} p_i \log p_i \quad (3)$$

If each incoming packet had a distinct destination IP address, the Entropy will be at its maximum; while in case the attack is directed to a single host, massive number of packets is sent to it, as a result the window will be filled by these packets and the number of unique destination IPs in the window will be reduced, and as a result the Entropy will be reduced.

The calculated Entropy is then compared to a threshold, if it is lower than the threshold, a counter is incremented until it reaches five consecutive lower-than threshold entropies, which is considered as a DDoS attack. Otherwise, the counter is cleared, and there is no attack. Detection within five Entropy periods is 250 packets in the attack. These five consecutive periods have the lowest false positive for early detection.

4. IMPLEMENTATION OF THE DETECTION METHOD

In this paper the Entropy detection algorithm has been implemented in 8 GB Laptop with installed Windows 8.1 (64-bit); and Ubuntu 14.04 (64-bit) virtual machine were installed in Virtual Box with 5000 MB base memory; and a package of Mininet was installed in this virtual machine as the emulation tool. Mininet is a network emulator, it has the advantage of being simple, flexible, open source, and available for free. Mininet is widely used in research where several topologies that contains hosts, forwarding devices, and controllers, can be created and customized, and apply different tests on them in a simple manner. Also, the controller in which the algorithm was run is POX; POX is an open source controller written in python programming language and developed by Nicira, it is used mainly in research by many developers and SDN engineers, and it comes already included within the Mininet VM.

Different tests have been done to implement the Entropy detection algorithm, the setup of the network used in the first test is shown in Figure 3 where POX controller (c0) was connected to single topology with one switch (s1) and 64 hosts (h1-h64).

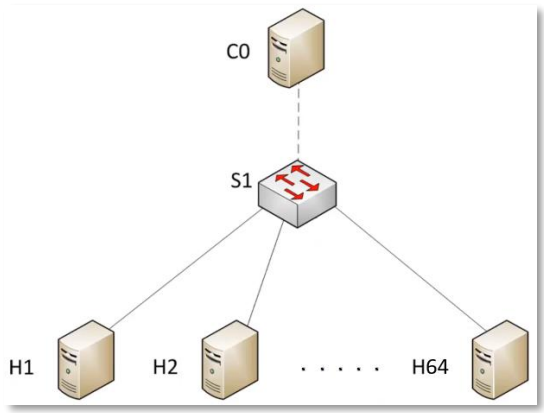


Figure 3. Single topology with 64 hosts

Then, the setup of the network used in further tests is shown in Figure 4 where each of (1, 2, 4, 8, and 16) POX controllers was connected to linear topology with 64 hosts. The integer (N) represents the number of controllers and the topology are divided equally between these controllers, for example, if N=2 then two controllers (c0, c1) will be connected to 32 switches for each one of them. The (j) in the figure represents the reduplication of (64/N).

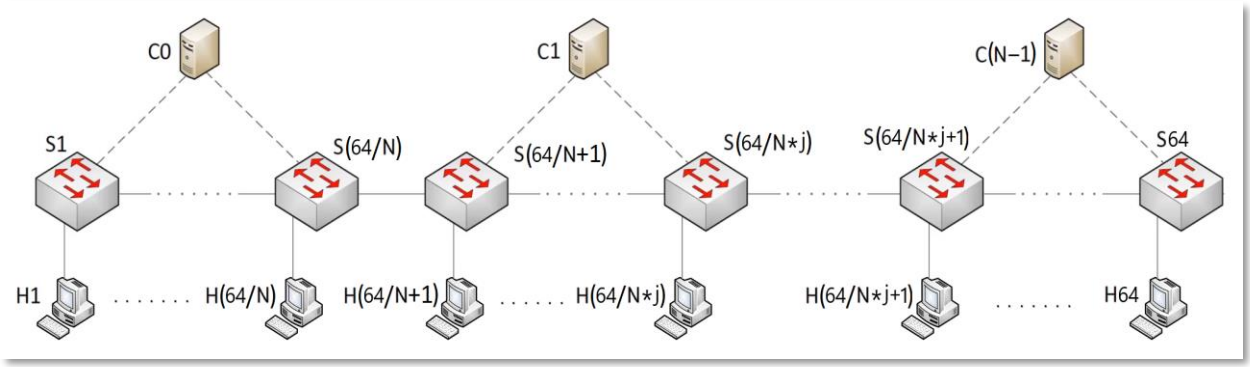


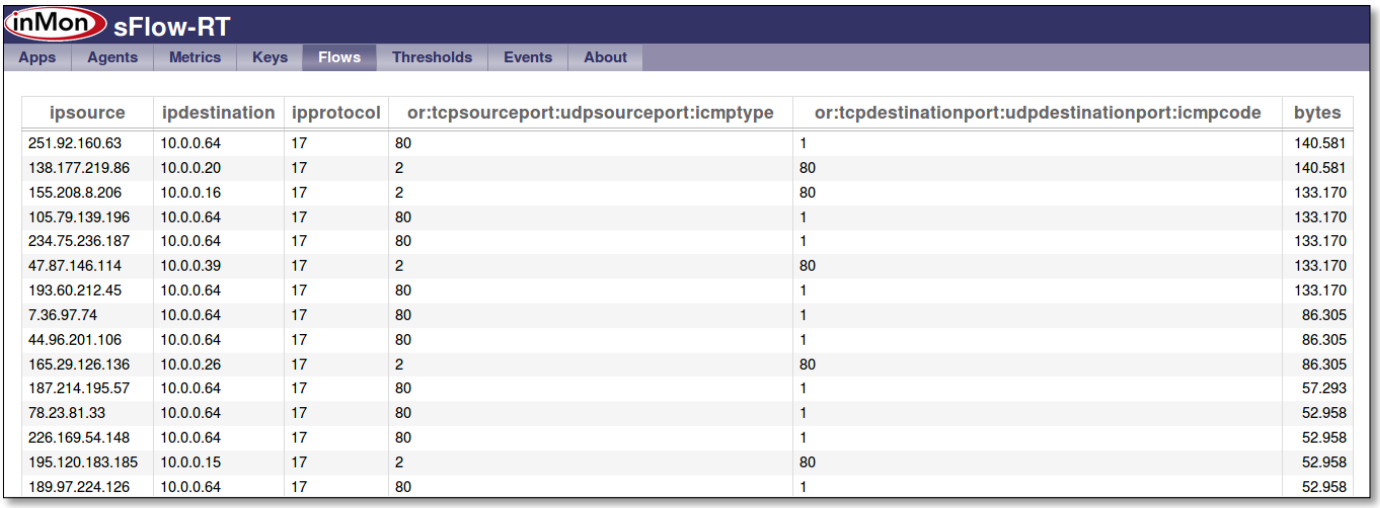
Figure 4. Setup of the designed network

In these tests, POX controller was modified to call the detection algorithm in a new module called "l3_editing"; also, host (h1) was used to generate the normal traffic and host (h2) was used to generate DDoS attack traffic. The idea behind using these tests is to show the behavior of the networks when it deals with normal and attack traffic and when different networks and different number of controllers is used.

Normal and DDoS attack traffic are generated using Scapy [22] which is a powerful tool for packet manipulation, Scapy is started programmatically by two different python codes for normal and attack traffic flows, these codes generate large number of random spoofed source IP addresses, and these codes are provided by [23]. In normal traffic, packets from spoofed IP addresses are sent to all hosts in the network. While in attack traffic, all packets from the spoofed IP addresses are sent to victim host (h64) that has the IP (10.0.0.64). The attack traffic has a higher rate than normal traffic and the type of all traffic packets is UDP.

In all tests, both normal and attack traffic will run in the same time, to see how the Entropy detection method handles both traffic; and the network traffic is monitored using sampled Flow-Real Time (sFlow-RT) [24], which is a tool for

traffic monitoring in SDN as shown in Figure 5; in this figure it can be seen that the source IP addresses are all spoofed, and packets are sent mostly to victim IP (10.0.0.64) in the destination IP addresses.



ipsource	ipdestination	ipprotocol	or:tcpsourceport:udpsourceport:icmptype	or:tcpdestinationport:udpdestinationport:icmpcode	bytes
251.92.160.63	10.0.0.64	17	80	1	140.581
138.177.219.86	10.0.0.20	17	2	80	140.581
155.208.8.206	10.0.0.16	17	2	80	133.170
105.79.139.196	10.0.0.64	17	80	1	133.170
234.75.236.187	10.0.0.64	17	80	1	133.170
47.87.146.114	10.0.0.39	17	2	80	133.170
193.60.212.45	10.0.0.64	17	80	1	133.170
7.36.97.74	10.0.0.64	17	80	1	86.305
44.96.201.106	10.0.0.64	17	80	1	86.305
165.29.126.136	10.0.0.26	17	2	80	86.305
187.214.195.57	10.0.0.64	17	80	1	57.293
78.23.81.33	10.0.0.64	17	80	1	52.958
226.169.54.148	10.0.0.64	17	80	1	52.958
195.120.183.185	10.0.0.15	17	2	80	52.958
189.97.224.126	10.0.0.64	17	80	1	52.958

Figure 5. Normal and attack traffic flow

5. SIMULATION RESULTS

The results of implementing the Entropy detection algorithm in POX controller using different test is presented in this section.

In all tests, when running normal and attack traffic, the number of occurrences of destination IP (10.0.0.64) will be higher than other destination IPs in the window, therefore the Entropy value will decrease and when it is less than the threshold for five consecutive times it is considered an attack and the packets from h2 is blocked and only normal traffic from h1 keeps running.

If the attack packets overcomes the normal traffic, there will be 50 packets in the window with only one destination IP (10.0.0.64) and the Entropy value for that window will be zero because the probability of that destination IP will be (50/50) which is 1 and since log(1) = 0, the Entropy in this case will be zero which is less than the threshold and the port is blocked.

5.1 Results of first test

Figure 6, presents a combination of attack and normal traffic captured by sFlow-RT, when the attack traffic is detected by the algorithm, the number of bytes returns to the normal traffic rate.

Figure 7, presents a window of 50 packets where destination IP (10.0.0.64) has higher number of occurrences due to attack traffic and as a result the Entropy value will be less than threshold value.

5.2 Results of second test

In this test, POX controller is connected to linear topology, Since the number of switches in linear topology is more than in single topology, the load on the controller will increase, and the detection of attack traffic will be slower than in single topology as presented in Figure 8.

5.3 Results of third test

In this test, the load of the linear topology will be decreased and distributed between the two POX controllers, thus the detection of attack traffic will be faster than in previous section as presented in Figure 9. The detection of attack traffic in the window of 50 packets at the two POX controllers is shown in Figure 10.

5.4 Results of fourth test

In this test, the load of the linear topology will be decreased more than in the previous test as presented Figure 11. The detection of attack traffic in the four POX controllers is shown in Figure 12.

5.5 Results of fifth test

In this test, as presented in Figure 13, the load of the linear topology will be decreased, and number of bytes is nearly less than 2K. The detection of attack traffic in the eight POX controllers is shown in Figure 14.

5.6 Results of sixth test

In this test, as presented in Figure 15, the load of the linear topology will be decreased, and number of bytes is less than 1.5K. The detection of attack traffic in the sixteen POX controllers is shown in Figure 16.

Finally, Figure 17 presents a comparison between the traffic generated in each test that was captured by sFlow-RT.

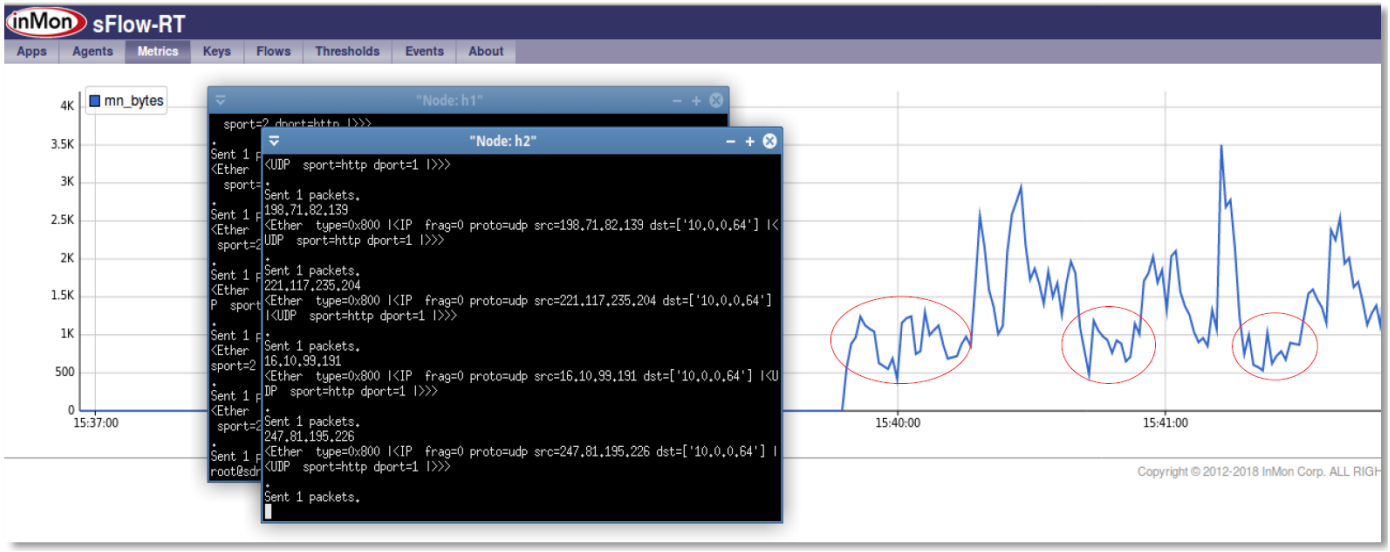


Figure 6. Captured traffic of first test

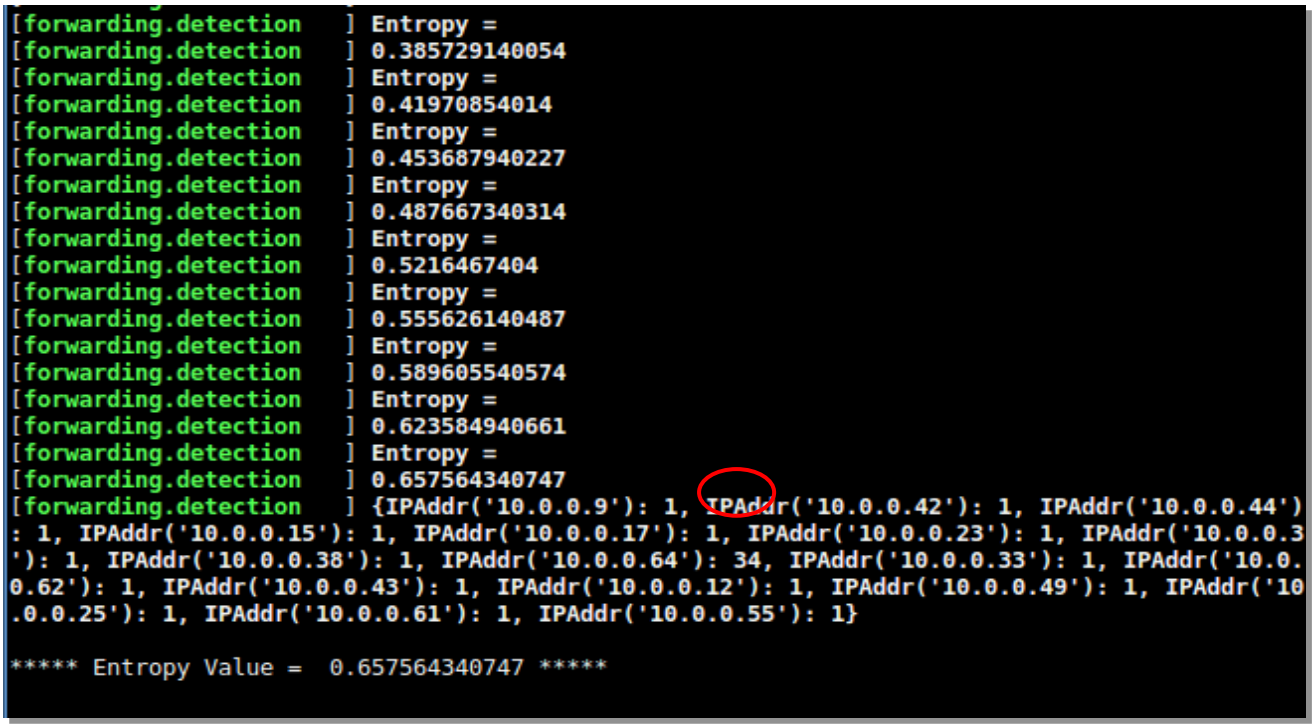


Figure 7. Output of the detection function in first test

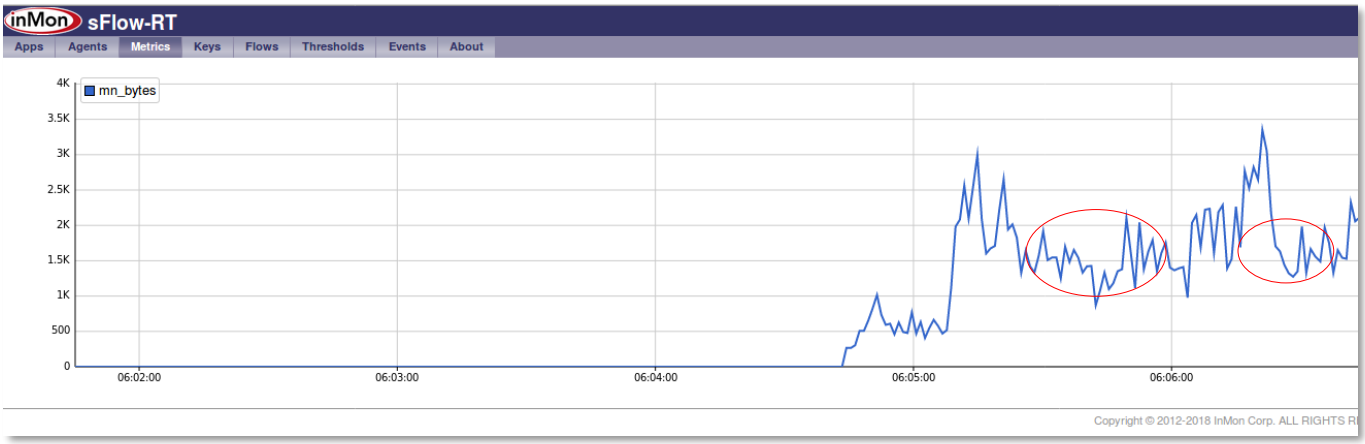


Figure 8. Captured traffic of second test

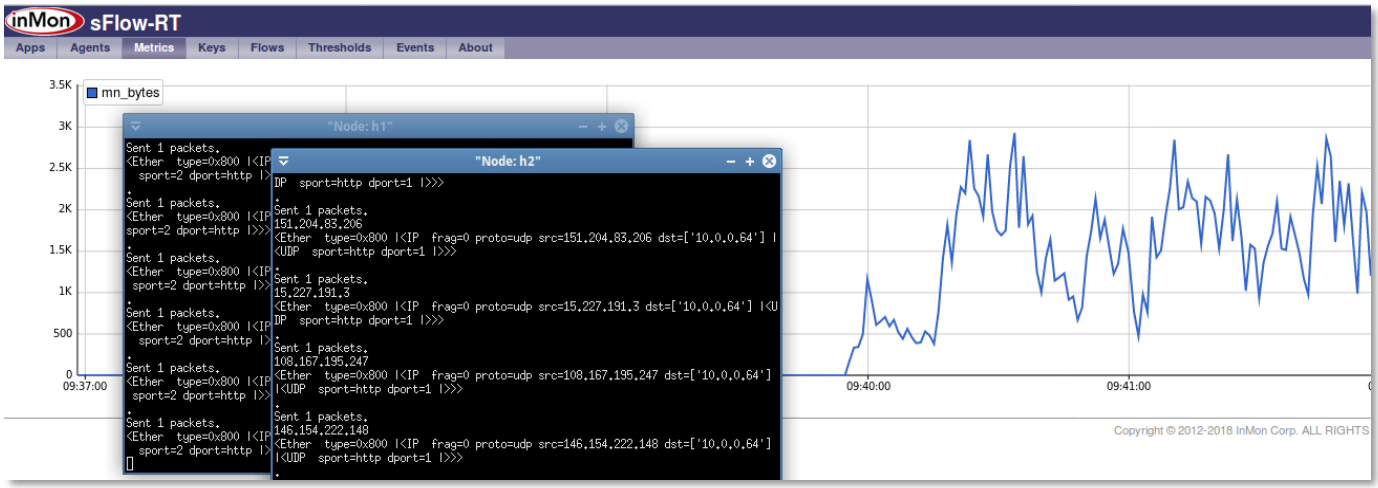


Figure 9. Captured traffic of third test

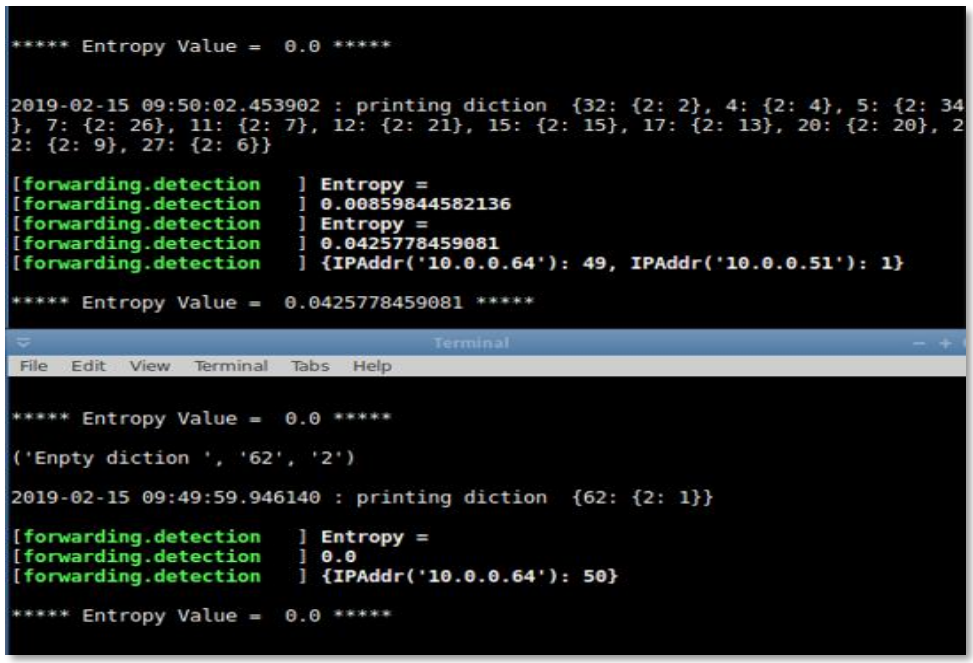


Figure 10. Output of the detection function in third test

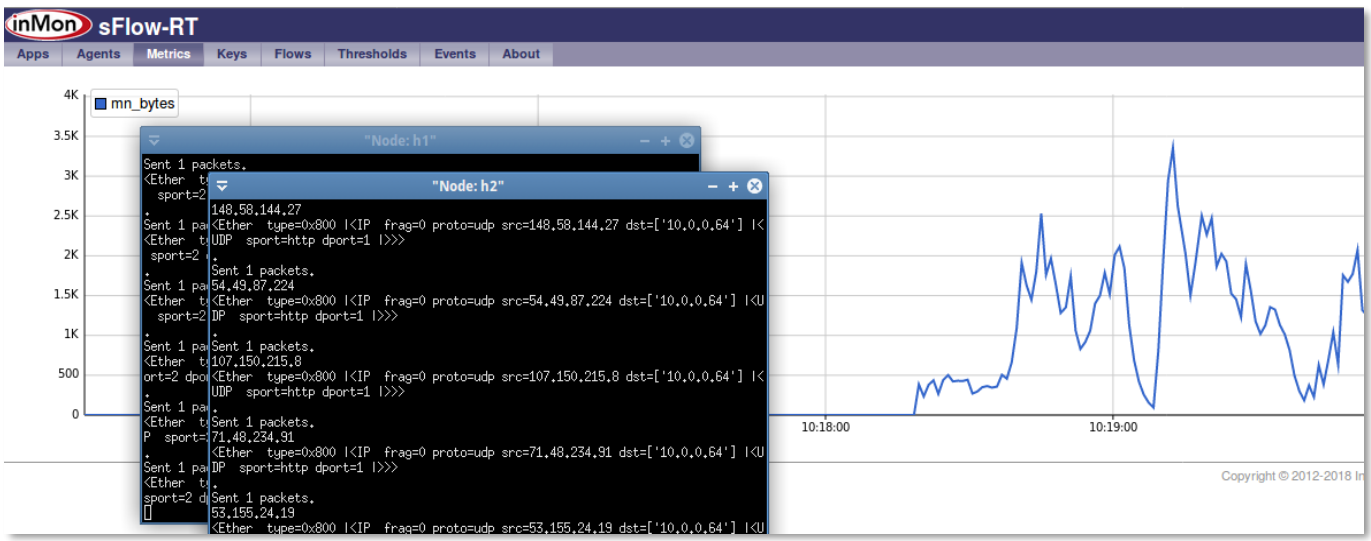


Figure 11. Captured traffic of fourth test

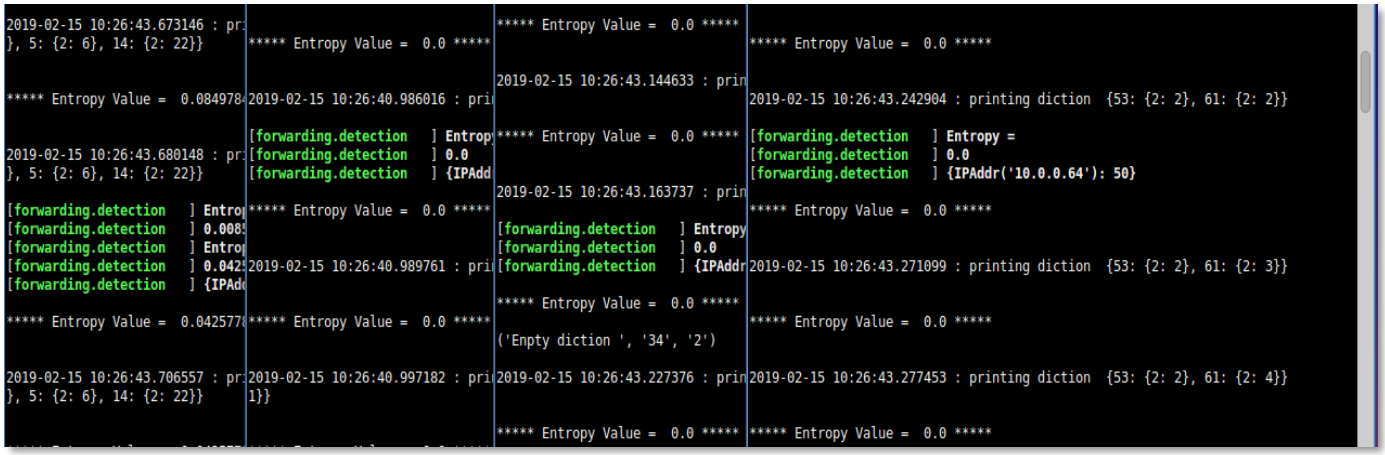


Figure 12. Output of the detection function in fourth test

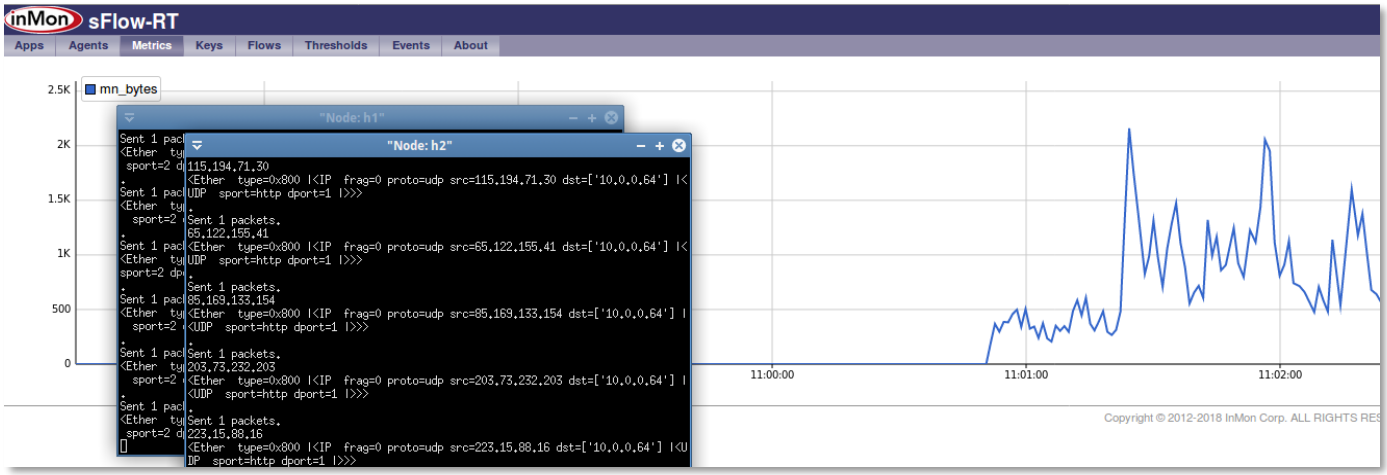


Figure 13. Captured traffic of fifth test

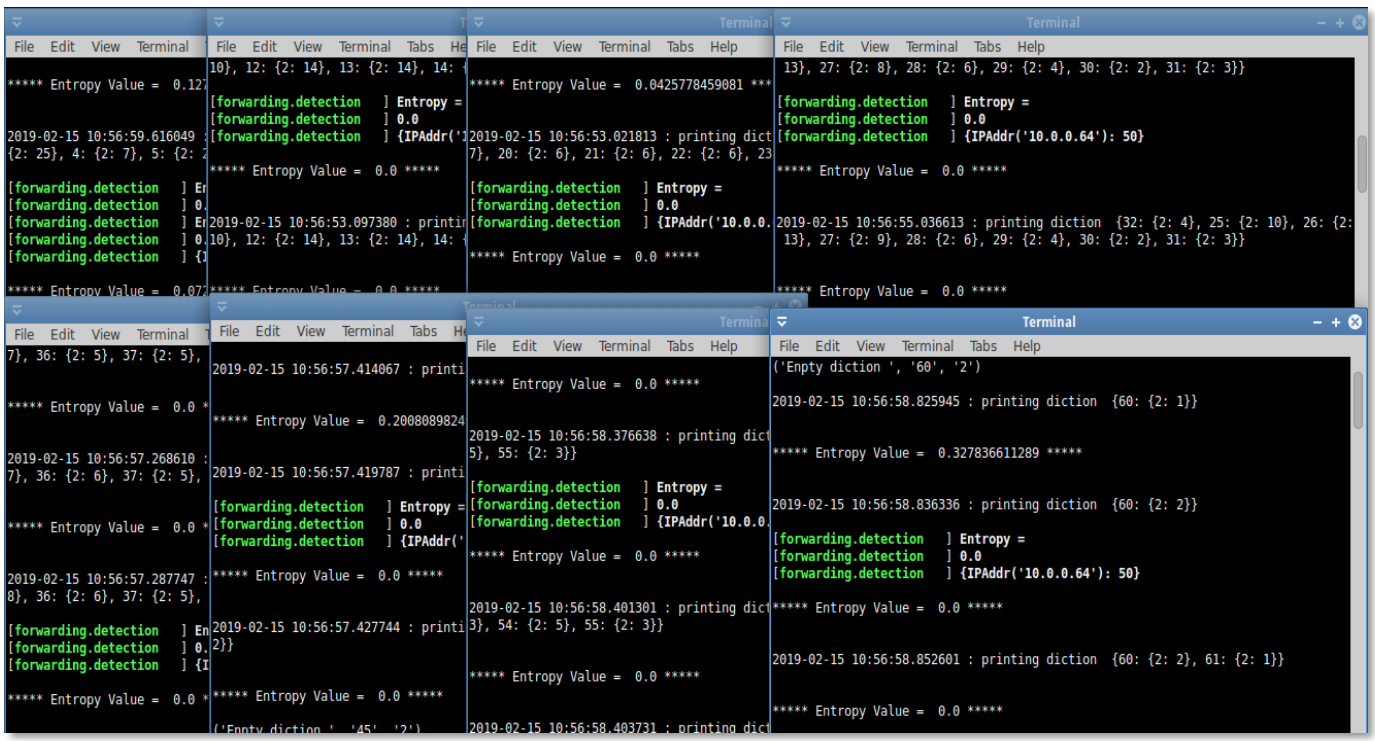


Figure 14. Output of the detection function in fifth test

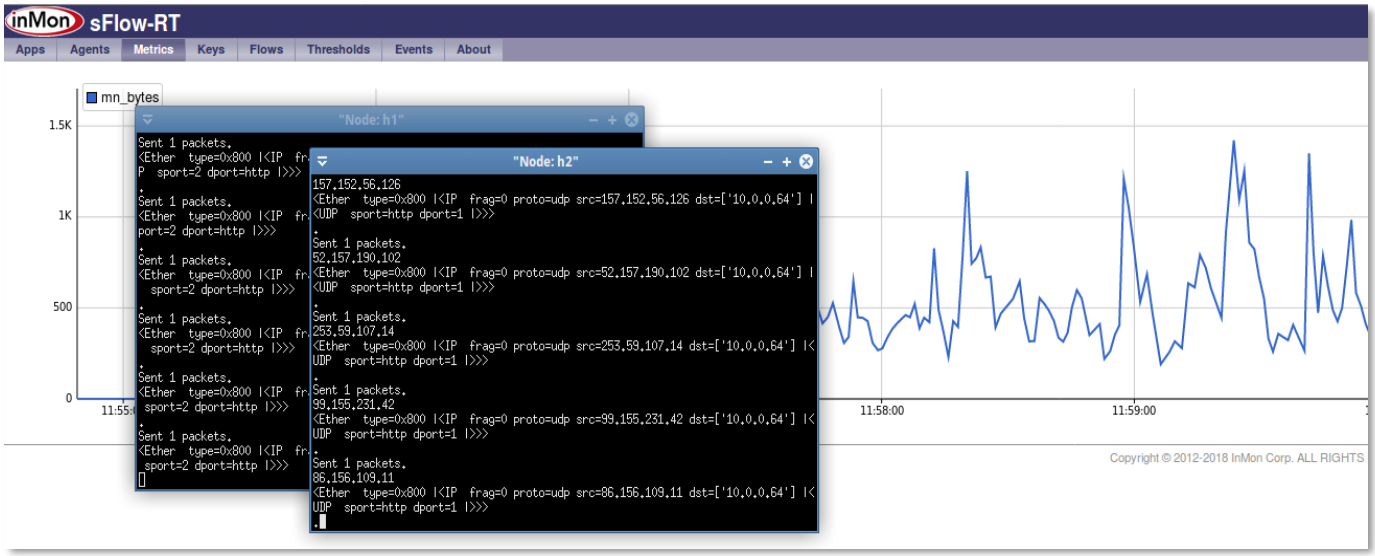


Figure 15. Captured traffic of sixth test

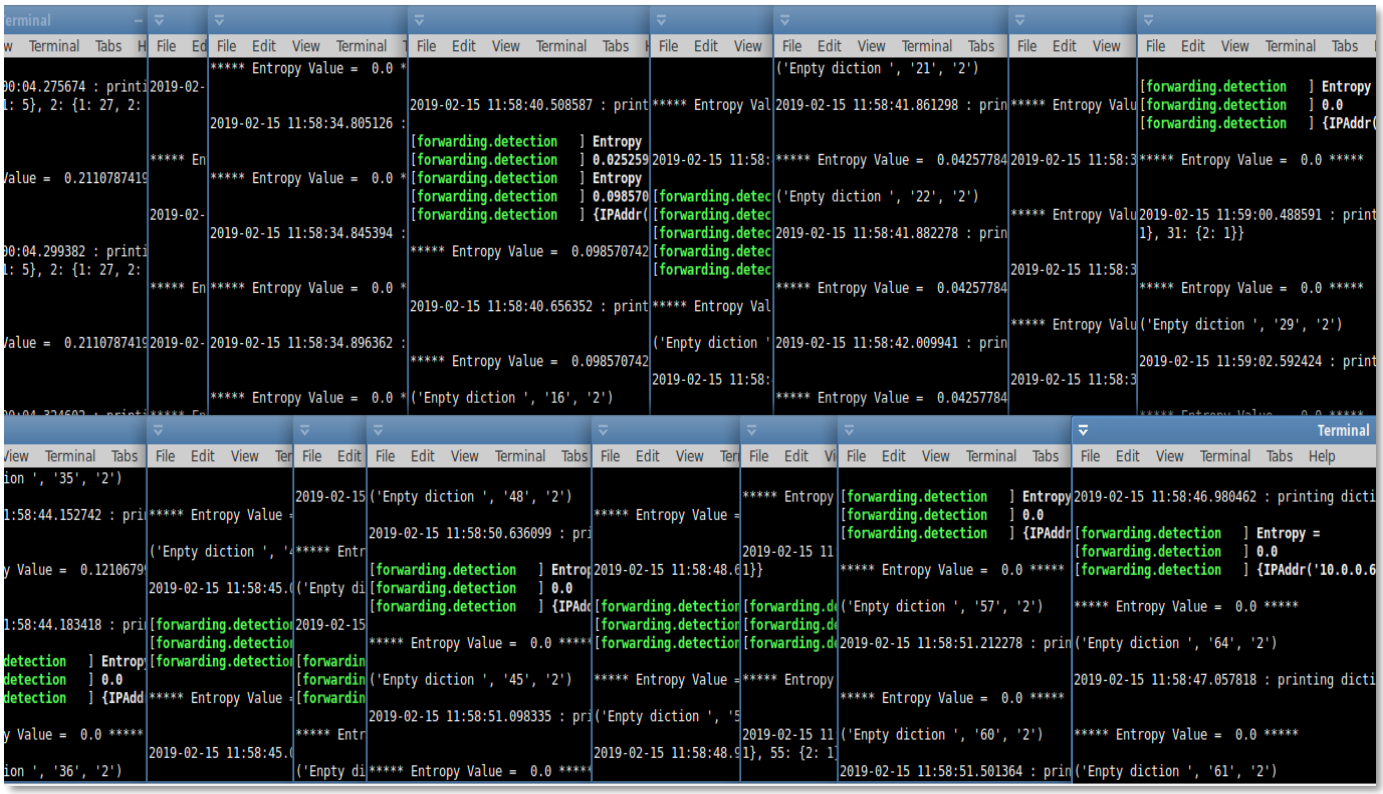
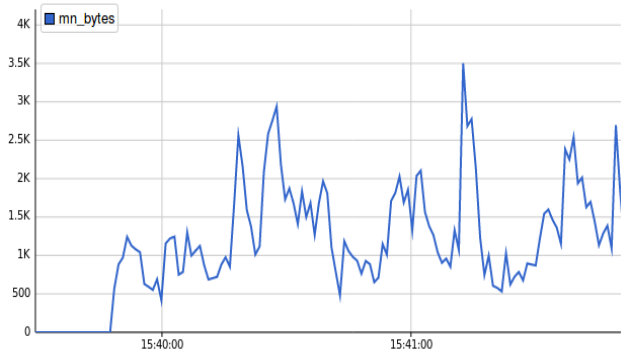
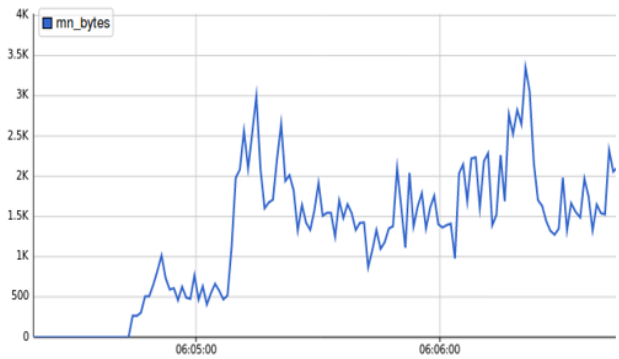


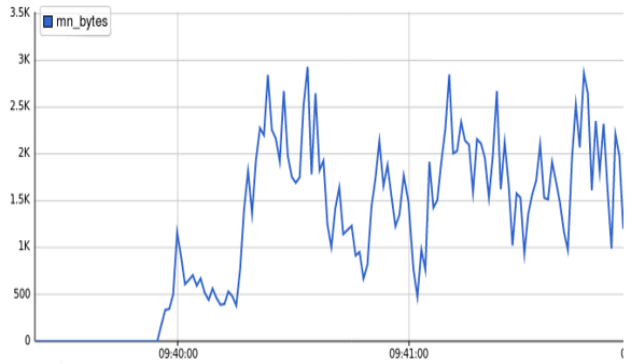
Figure 16. Output of the detection function in sixth test



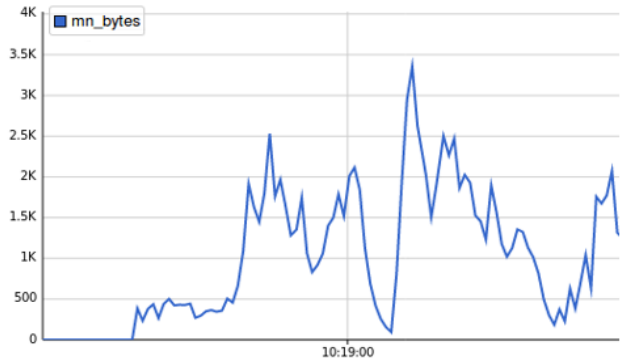
(a) Traffic of one POX connected to single topology



(b) Traffic of one POX connected to linear topology

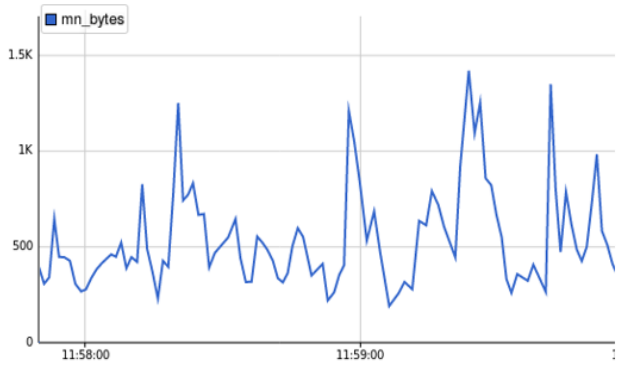


(c) Traffic of two POX connected to linear topology



(d) Traffic of four POX connected to linear topology

(d) Traffic of four POX connected to linear topology



(f) Traffic of sixteen POX connected to linear topology

Figure 17. Captured traffic of all tests

6. CONCLUSION

In this paper, Entropy-based detection algorithm was carried out using POX controller in different tests that include using different networks and different number of controllers. from the results it is noticed that the Entropy detection algorithm detects lower than threshold Entropy values and block attack packets and only keep normal packets. It is also noticed that detection algorithm works better in single topology than in linear topology, however single topology has the disadvantage of single point of failure. Finally, it is noticed that increasing the number of controllers in linear topology improves the security of the network, where the detection of the attack traffic is better in the case of sixteen controllers than the other number of controllers. Some suggestions for future works can include implementing the detection algorithm in another controller such as Ryu, or it can include adding a new function that uses a mitigation technique to mitigate DDoS attack after it's detection, or it can include detection of DDoS attack using other methods like machine learning.

REFERENCES

[1] Azodolmolky, S. (2013). Software Defined Networking with OpenFlow. Packt Publishing Ltd., UK.

[2] Stallings, W. (2016). Foundations of Modern Networking SDN, NFV, QoE, IoT, and Cloud. Pearson Education Inc., USA.

[3] Sharma, D.K. (2015). Training Report on Software Defined Networking. Florida International University, Florida, USA. https://doi.org/10.13140/RG.2.1.2444.3361

[4] Doherty, J. (2016). SDN and NFV Simplified Visual Guide to Understanding Software Defined Networks and Network Function Virtualization. Pearson Education Inc., USA.

[5] Qi, H., Li, K.Q. (2016). Software Defined Networking Applications in Distributed Datacenters. Springer, Switzerland. https://doi.org/10.1007/978-3-319-33135-5

[6] Göransson, P., Black, C. (2014). Software Defined Networks A Comprehensive Approach. Elsevier Inc., Massachusetts, USA.

[7] Zhu, W., Yi, X. (2017). A research review on SDN-based DDOS attack detection. 4th International Conference on Management Science and Management Innovation (MSMI), Suzhou, China. https://doi.org/10.2991/msmi-17.2017.33

[8] Bhattacharyya, D.K., Kalita, J.K. (2016). DDoS Attacks Evolution, Detection, Prevention, Reaction, and Tolerance. Taylor & Francis Group, LLC, USA.

[9] Yan, Q., Yu, F.R. (2015). Distributed denial of service attacks in software-defined networking with cloud computing. IEEE Communications Magazine, 53(4): 52-59. https://doi.org/10.1109/MCOM.2015.7081075

[10] Yan, Q., Yu, F.R., Gong, Q., Li, J. (2016). Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges. IEEE Communications Surveys & Tutorials, 18(1): 602-622. https://doi.org/10.1109/COMST.2015.2487361

[11] Douligeris, C., Mitrokotsa, A. (2004). DDoS attacks and defense mechanisms: Classification and state-of-the-art. Computer Networks: The International Journal of

Computer and Telecommunications Networking, 44(5): 643-666. https://doi.org/10.1016/j.comnet.2003.10.003
[12] Dayal, N., Srivastava, S. (2017). Analyzing behavior of DDoS attacks to identify DDoS detection features in SDN. IEEE 9th International Conference on Communication Systems and Networks (COMSNETS), Bengaluru, India. https://doi.org/10.1109/COMSNETS.2017.7945387
[13] Kia, M. (2015). Early detection and mitigation of DDoS attacks in software defined networks. M.Sc. Thesis. Ryerson University, Toronto, Ontario, Canada.
[14] Ertaul, L., Venkatachalam, K. (2017). Security of software defined networks (SDN). Proceedings of the 2017 International Conference on Wireless Networks-ICWN'17, Las Vegas, Nevada, USA.
[15] Barki, L., Shidling, A., Meti, N., Narayan, D.G., Mulla, M.M. (2016). Detection of distributed denial of service attacks in software defined networks. IEEE 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India. https://doi.org/10.1109/ICACCI.2016.7732445
[16] Geetha, V., Dheebalakshmi, R. (2018). Survey on solutions to DDoS attacks to controllers in software defined networks. International Journal of Pure and Applied Mathematics (IJPAM), 118: 1983-1990.
[17] Mousavi, S.M. (2014). Early detection of DDoS attacks in software defined networks controller. M.Sc. Thesis. Carleton University, Ottawa, Ontario, Canada.
[18] Badrinath, M., Parab, P., Mishra, R., Gupta, V., Chavan, V., Vasudev, V. (2016). Custom Network Security Implementation for Detection and Prevention of DDoS Attacks in SDN.
[19] Lin, P.C., Hsu, Y.T., Hwang, R.H. (2017). Detecting and preventing DDoS attacks in SDN-based data center networks. The 3rd International Conference on Cloud Computing and Security (ICCCS), Nanjing, China. https://doi.org/10.1007/978-3-319-68542-7_5
[20] Al-Mafrachi, B.H. (2017). Detection of DDoS attacks against the SDN controller using statistical approaches. M.Sc. Thesis. Department of Computer Science and Engineering, Wright State University, Dayton, Ohio, USA.
[21] Boite, J., Nardin, P.A., Rebecchi, F., Bouet, M., Conan, V. (2017). StateSec: Stateful monitoring for DDoS protection in software defined networks. 2017 IEEE Conference on Network Softwarization (NetSoft), Bologna, Italy. https://doi.org/10.1109/NETSOFT.2017.8004113
[22] Biondi, P. The Scapy Community. Scapy. https://scapy.net/, accessed on Feb. 15, 2019.
[23] GitHub Inc. Anandkumar26/DDOSAttack_SDN. https://github.com/Anandkumar26/DDOSAttack_SDN, accessed on January 4, 2019.
[24] InMon Corp. sFlow-RT. https://sflow-rt.com/, accessed on March 4, 2019.

NOMENCLATURE

H	Entropy
n	Elements of the window
pi	probability of each IP address
W	Window
x	destination IP address
y	number of times destination IP address appeared