Journal of Computer Engineering
ISSN 10003428

Original Research Article
Vol 12, Issue 4, 2023 | ISSN: 1000-3428

Optimizing SDN Traffic Management with Automated Flow Header Information Extraction

Amr A. Hassan, Nurul A. Rashid, Mohd H. A. Bahari

Department of Computer Science, Universiti Teknologi Malaysia; Department of Electrical Engineering, University of Malaya

***Abstract*—**Software Defined Networking (SDN) is a paradigm designed to facilitate the way of controlling the network dynamically and with more agility. Network traffic is a set of flows, each of which contains a set of packets. In SDN, a matching process is performed on every packet coming to the network in the SDN switch. Only the headers of the new packets will be forwarded to the SDN controller. In terminology, the flow header fields are called tuples. Basically, these tuples are 5-tuple: the source and destination IP addresses, source and destination ports, and protocol number. This flow information is used to provide an overview of the network traffic. Our module is meant to extract this 5-tuple with the packets and flows numbers and show them as a list. Therefore, this list can be used as a first step in the way of detecting the DDoS attack. Thus, this module can be considered as the beginning stage of any flowbased DDoS detection method.

I. Introduction

N recent years Software Defined Networking has completely changed our point of view to networking industry.. Researchers at Stanford are credited with the idea of bringing the tenets of virtualization to networking and thus creating the SDN market. Traditional networking uses integrated software and hardware to direct traffic across a series of routers and switches. The original use case for SDN was to virtualize the network by separating the control plane that manages the network from data plane where network flows [1].

The OpenFlow switch is one of the SDN components. It consists of flow table(s) besides other components which perform packet lookups and forwarding. An OpenFlow switch has OpenFlow channel(s) to the external controller. Via OpenFlow switch protocol, the controller manages the switch and the switch communicates with the controller. Both reactively and proactively, in response to packets, the controller can add, update and delete flow entries in the flow tables using the OpenFlow switch protocol. Within each OpenFlow switch, there is a flow table(s). This flow table is used to define what actions should be applied to packets that enter this particular SDN switch. Readers interested in extracting these statistics from its counters using POX controller can refer to our previous work [7]. Matching process is starting at the first flow table and it may continue to the next flow tables in the switch. In priority order, flow entries match packets with the first matching entry in each table being used. The instructions associated with the specific flow entry will be executed if a matching entry is founded. The table-miss is configured to forward the packet to the controller via the OpenFlow channel if no matching is founded in the flow table. When the matching process is over in the flow table and the packet is forwarded to the controller, another matching process is conducting in the table-miss. If no match is founded in the table-miss, the packet will be dropped [4].

For all network administrators,, one of the important security matters is how to protect the network from the internal and external attacks. In order to achieve the purpose, many attack detection systems have been proposed.

Traditionally, the approach used to detect the network attack is to inspect the contents of every packet (payload). However, in the high-speed networks, traditional approach cannot be easily performed. Therefore, alternative inspecting approaches have been suggested such as flow-based detection approach. This type of detection is based on the headers of the flows instead of the individual packet contents [2].

In the subsequent sections, we will talk about some components of flow tables and the table-miss flow entry, along with the matching process in the flow table.

II. Flow Tables

This section provides an overview of the flow tables and the flow entries inside these tables. Also, this section gives an idea of how flow tables do matching with incoming packets in the OpenFlow switch as well as the role of the table-miss flow entry if no match is founded.

A. Flow Tables and Flow Entries

An OpenFlow switch consists of flow tables and each flow table consists of flow entries. In OpenFlow switch terminology, the set of linked flow tables is called Pipeline. These linked tables provide matching, forwarding and packet modification.

The flow entry is an element used to match and process the packets in the flow table. Table I shows what each flow entry contains as stated in OpenFlow Switch Specification Version 1.5.1 [4].

How to cite: Amr A. Hassan, Nurul A. Rashid, Mohd H. A. Bahari (2023). Optimizing SDN Traffic Management with Automated Flow Header Information Extraction. Journal of Computer Engineering, 12(4), 14–17. DOI: https://doi.org/10.5281/zenodo.19348297

TABLE I
FLOW ENTRY COMPONENTS

Match fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

B. Matching

Matching is a process of comparing the match fields of a flow entry with a set of header fields and pipeline fields. Match fields are a part of the flow entry that is used against packets to know which packet is matched. It can match various packet header fields such as Ethernet, IPv4, and IPv6. Match fields can also match pipeline fields such as packet ingress port, the metadata value, and other pipeline fields. It is important to underline the difference between the packet header fields and the pipeline fields. The header fields are the values extracted from the packet header after the packet header is parsed which are matched against the corresponding match fields, whereas the pipeline fields are the values attached to the packet during the pipeline processing.

As shown in Fig. 1, on receipt of a packet, the OpenFlow Switch begins performing a table lookup in the first flow table (table 0) using packet header fields. When packets are matched by some or all flow entries in the flow table, counters will update and a set of instructions will execute. Otherwise, the table-miss flow entry specifies how to process these packets. The table-miss flow entry may send these packets to the controller or may drop them [4].

C. Table-Miss

A table-miss flow entry must be supported by every flow table. In most ways, the table-miss flow entry behaves like any other flow entry in the flow table. By default, the table-miss flow entry does not exist in the flow table; it may be added or removed by the controller at any time. By its match and its priority, the table-miss flow entry is identified. Among other flow entries, the table-miss flow entry must have the lowest priority which is zero. If the table-miss flow entry exists in the flow table and it matches the unmatched packets by other flow entries, packets will be sent directly to the controller using the CONTROLLER reserved port; otherwise these packets will be dropped (discarded) [4].

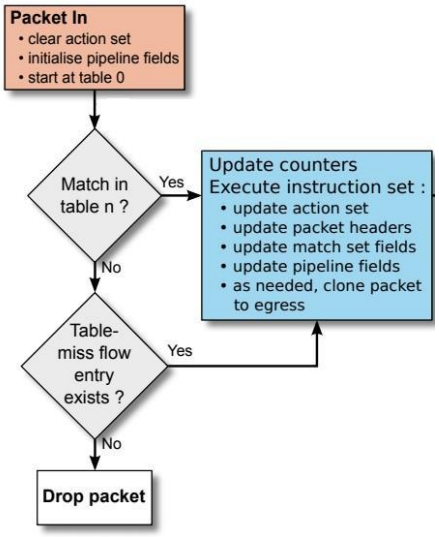

Fig. 1 Matching and table-miss flow entry. This flowchart has taken from the OpenFlow Switch Specification version 1.5.1 file

III. DESIGN

Due to the design of switch flow tables, packets will be forwarded to the controller if a table-miss flow entry matched the unmatched packets by other flow entries in the flow table. When we say packets we mean the headers of these packets only, not the packet's payload. In this paper, a Python module is designed to produce a list of flow headers contents as well as the number of packets and flows. This work is aimed to extract this information from the SDN traffic using POX controller.

In order to achieve the proposed design, Python language is used. This module is provided with a flow information extractor function to prepare the list and display its contents periodically. Our module is implemented on the OpenFlow controller "POX".

IV. EXPERIMENT AND RESULTS

Creation of a realistic virtual network and running endhosts, switches, routers and application codes on a single Linux kernel is allowed by Mininet network emulation software [6]. Mininet can create capable network topologies in SDN. In Mininet Virtual Machine or Mininet VM, a single command can be used to launch all controllers, switches, and hosts.

Mininet Command Line Interface (CLI) is used to create the topology in the SDN environment. Our topology in this experiment is containing one OpenFlow controller, one OpenFlow switch and four end-hosts as presented in Fig. 1.

POX is one of the SDN controllers and it is originated from NOX where NOX is the original OpenFlow controller. The POX repository has multiple branches. For this experiment, we use POX (eel) branch which is the current active branch [3]. When POX controller works, the

OpenFlow switch is connected to the network to be ready to receive the instructions or the modification messages as shown in Fig. 2.

Our module (flow_info_extractor.py) is run along with the forwarding.l2_learing component.

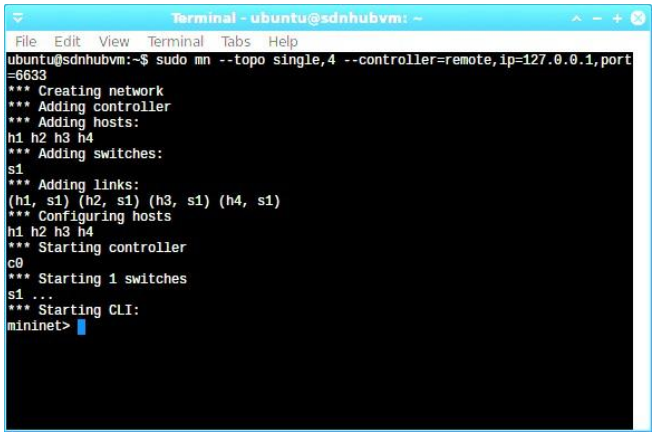

Fig. 2 Topology used in the experiment

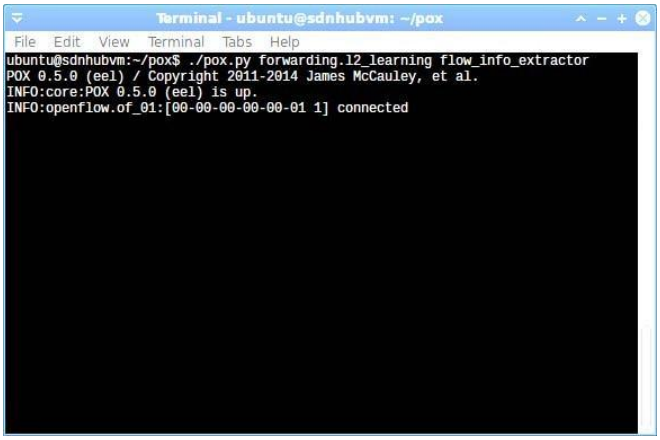

Fig. 3 Switch connected and ready to receive instructions

On the terminal appeared in Fig. 2, the topology consists of four hosts, one switch and one controller. Once the switch is connected, the hosts begin the traffic generation.

As presented in Fig. 4, once the traffic starts generating between hosts our POX controller module begins displaying the contents of the list. As it is appeared, the list contents are:

- Source and Destination IP addresses
- Source and Destination port numbers
- Number of packets
- Number of flows

Based on [5], the numbers of protocols that appeared in Fig. 4 are belonging to TCP, ICMP and IGMP protocols. The protocol number (6) refers to TCP protocol and the numbers (1) and (2) refer to ICMP and IGMP protocols. We are grouping ICMP and IGMP protocols together and separate TCP protocol from them because TCP is one of the main protocols in the internet protocol suite, while ICMP and IGMP are a part of internet protocol itself. ICMP and IGMP are both protocols that operate within the sphere of IP. Technically, ICMP and IGMP are layer 3 protocols but they have not the same functionalities. TCP is used to provide host-to-host connectivity but ICMP and IGMP are used to send error messages and operational information and facilitate management of multicast groups between cooperating routers and switches. For clarification, protocol (6), which is the TCP protocol, has a source and destination port numbers while protocols (1) and (2) have not as manifested in Fig. 4.

Journal of Computer Engineering
ISSN 10003428

Original Research Article
Vol 12, Issue 4, 2023 | ISSN: 1000-3428

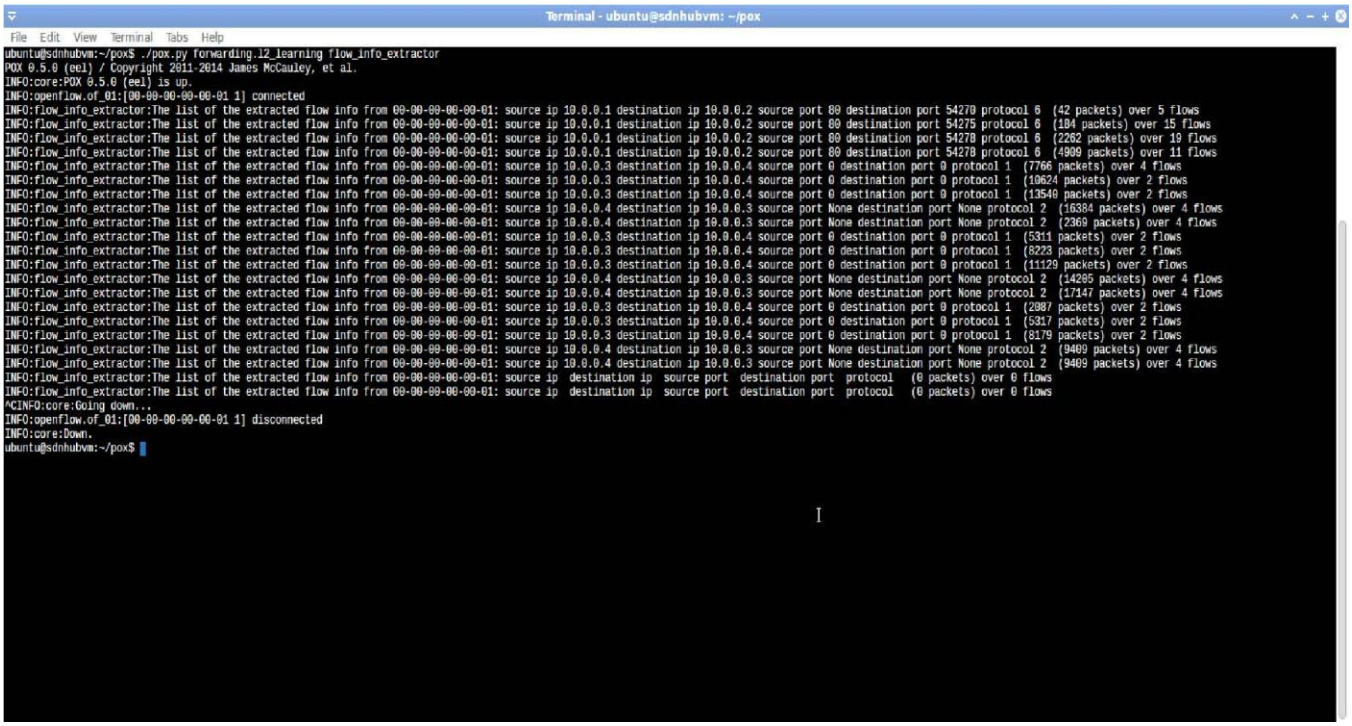

Fig. 4 Flow header information obtained by the POX controller module

V. Conclusion

This paper is aimed to produce a complete method to extract the information of the flow headers from the SDN traffic. This module is an attempt to provide the researchers who concern about developing and enhancing the Network Intrusion Detection Systems (NIDS) with a clear list of flow header information. In particular, this list can be used for the detection of DDoS attacks that uses flow-based methods.

References

Accessed at: 13/9/2017.

[4] OpenFlow Switch Specification Version 1.5.1. Available online: https://www.opennetworking.org/wp content/uploads/2014/10/openflow-switch-v1.5.1.pdf. Accessed on:

[5] Assigned Internet Protocol Numbers. Available at: https://www.iana.org/assignments/protocol-numbers/protocolnumbers.xhtml. Accessed on: 10/10/2017.

[6] Mininet. Available: http://mininet.org/ Accessed on: 3/9/2017.

[7] Muragaa, W. H., Seman, K., & Marhusin, M. F. A POX Controller Module to Collect Web Traffic Statistics in SDN Environment. World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering, 10(12), 2051-2056.

Open Networking Foundation "Software Defined Networks: The new Norm of Networks" White paper 2012 Available at: https://www.opennetworking.org/images/stories/downloads/sdnresources/white-papers/wp-sdn-newnorm.pdf. Accessed on: 22/9/2017. Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., & Stiller, B. (2010). An overview of ip flow-based intrusion detection. IEEE Communications Surveys and Tutorials, 12(3), 343-356.

POX controller. Available: http://www.noxrepo.org/pox/about-pox/

How to cite: Amr A. Hassan, Nurul A. Rashid, Mohd H. A. Bahari (2023). Optimizing SDN Traffic Management with Automated Flow Header Information Extraction. Journal of Computer Engineering, 12(4), 14–17. DOI: https://doi.org/10.5281/zenodo.19348297