An Entropy-Based DDoS Defense Mechanism in Software Defined Networks

Yajie Jiang[1(✉)], Xiaoning Zhang[1], Quan Zhou[1], and Zijing Cheng[2]

[1] Qingshuihe Campus of UESTC, No. 2006, Xiyuan Ave, West Hi-Tech Zone, Chengdu 611731, Sichuan, People's Republic of China
jiangyj319@163.com, xnzhang@uestc.edu.cn, 634466414@qq.com

[2] Beijing Institute of Information Engineering, Beijing, People's Republic of China
linuxdemo@126.com

Abstract. The issue on defensing against Distributed Denial of Service (DDoS) attacks in Software Defined Networks (SDN) has been highly concerned by academe and industry. The existing studies cannot eliminate the false positives by using the simple classification algorithms. In this paper, we analyze the essential difference between DDoS attacks and flash crowds which causes some similar consequences to DDoS. Accordingly we design a novel effective Entropy-based DDoS Defense Mechanism (EDDM) running on the SDN controller, which including a two-stage DDoS detection method. Compared with the existing works, the EDDM avoids the dropping of legitimate packets and minimizes the losses of legitimate users. Simulations demonstrate that the EDDM can distinguish the DDoS attacks from flash crowds, find the locations of bots, and block attack packets at source effectively.

Keywords: DDoS defense · Flash crowd · SDN · Entropy

1 Introduction

In the Distributed Denial of Service (DDoS) [1] flooding attacks, the attacker takes control of some hosts distributed in the network which are called zombie hosts or bots, then instructs them to send huge attack packets with legitimate looking. The target will be busy processing the attack traffic, cannot deal with the legitimate packets, meanwhile, network bandwidth is occupied by the attack traffic. From the users' perspectives, the target is unable to provide service as usual. In addition, it is difficult to trace to the zombie hosts, as the attack packets are filled with forged source IP addresses to conceal zombies' locations. With the development of network technology, the cost of DDoS attacks is getting lower and lower, while the attacker hides more easily. Enterprises and Operators have always been focusing on the topic *How to effectively defend against DDoS attacks*, due to the significant losses caused by DDoS attacks.

In traditional networks, the security strategies are complex, not easy to manage, and the upgrading is time-consuming, which needs the participation of vendors. The

Q. Chen et al. (Eds.): ChinaCom 2016, Part I, LNICST 209, pp. 169–178, 2018.
DOI: 10.1007/978-3-319-66625-9_17

emerging network architecture, Software Defined Network (SDN), has the characteristics of centralized control and programmable logic, which makes the upgrading quite easy.

There are emerging studies on DDoS attacks defense methods for SDN [1–10]. However, these studies didn't give a simple, complete and feasible defense mechanism against DDoS attacks. Most of them are unable to distinguish the DDoS attacks traffic from the flash crowd traffic.

Our contributions can be summarized as follows: (1) we examine the essential difference between DDoS attacks and flash crowds. (2) We propose a two-stage detection method based on the entropy of the destination and the essential difference mentioned above, which can detect DDoS accurately. (3) We present a novel DDoS defense mechanism (EDDM) deployed in the SDN controller, thus can trace to the bots, block attack traffic at source, and prevent the network from being overwhelmed by the attack traffic.

The rest of the paper is organized as follows. Section 2 summarizes some related studies. Section 3 describes the principles of entropy-based DDoS detection methods, analyzes the essential difference between DDoS and flash crowds. Sections 4, 5 present the proposed mechanism and simulation results, respectively. Finally, Sect. 6 summarizes our conclusions.

2 Related Work

To minimize the impacts on the SDN controller and the network bandwidth from attack traffic, the attack packets should be blocked at source after a rapid detection of DDoS. Therefore, the key of mitigating DDoS attack is how to distinguish attack packets from legitimate packets.

The existing DDoS detection schemes in SDN can be divided into two categories: pattern matching and anomaly-based detection.

Pattern matching methods extract the features of input flows, and match with the attack features library. Once a match is found, the flow must belong to DDoS attack traffic. Pattern matching methods [2, 3] will never lead to false positives, but they are invalid against the variants beyond the library.

Anomaly-based DDoS detection methods [4–10] can distinguish abnormal network traffic, but they may cause false positives. For example, flash crowds caused by the network hot events may be misjudged as DDoS attacks. Among anomaly-based detection methods, the flow-based attack detection [4–6] mechanisms deployed in the controller without proper aggregation of network traffic could overload the communication among control and data plane [8]. The existing entropy-based detection methods [7–10] take the entropy values as the sole indicator, which can also cause false positives. So we propose the EDDM to eliminate false positives.

3 Principle of Entropy-Based DDoS Detection in SDN

Compared with traditional networks, SDN has the characteristics of centralized control and programmable logic, which makes the upgrading and maintaining of the network strategies easy.

3.1 SDN Profile

In SDN networks, switches only need to forward data flows quickly following the flow table entries which are issued by controller through secure channel [11].

When a packet enters into OpenFlow switch, it is matched with the flow table entries and follows the entry with the highest priority. Otherwise, when the packet doesn't match any existing entries in OpenFlow switch, the switch will send a Packet-in to controller. Then controller determines forwarding strategy according to network state, and sends Packet-out to instruct relevant OpenFlow switches to establish corresponding flow table entries.

So, when controller receives a Packet-in, that means a newly launched flow wants to enter into the network. If there is DDoS attack in SDN network, the controller will be busy dealing with the attack flows, the secure channel will be occupied by Packet-ins of attack flows, and the legitimate flows will be delayed or dropped.

3.2 Principle of Entropy-Based DDoS Detection

In information theory, the entropy is a measure of the uncertainty associated with a random variable [12]. Suppose X represents a random event, it has some possible results $\{x_1, x_2, \ldots, x_n\}, x_i$ has a probability $p(x_i)$ to occur, then the entropy value of X is given by the formula:

$$H(X) = E[I(x_i)] = E[\log_2^{p(x_i)}] = -\sum_{i=1}^{n} p(x_i) \log_2^{p(x_i)}. \quad (1)$$

Formula (1) validates that the bigger of the event's uncertainty, the higher of its entropy value.

In normal network communications, a host may connect to any other host distributed in the whole network. Each IP has the same probability to appear in the destination IP field of packet-in. But if the network is under a DDoS attack, the SDN controller can receive a large number of *Packet-ins*, the target host's IP will appear in destination IP field with a high frequency.

Thus, if we calculate the entropy of destination IP address, with a specific window size, it will be found that the entropy value descends visibly under a DDoS attack [7].

3.3 The Essential Difference Between DDoS Attacks and Flash Crowds

However, some network hot events may cause many hosts communicating with the server, which is called flash crowd [13], has the similar characteristics of the soaring network traffic, with the declining entropy value of the destination IP addresses. Hence, using entropy variation of the destination IP addresses as sole indicator in DDoS attacks detection may lead to false positives.

Accordingly, we investigate the essential difference between DDoS attack packets and flash crowd packets. We found that the flash crowd packets, with real source IP, are from legitimate hosts distributed in the whole network, but the DDoS packets, usually with huge forged source IP, are from specific zombies. Therefore, the mapping

relationships from the source MAC to the source IP, flash crowd packets use the actual source IP with the actual MAC, while DDoS packets use many spoofed source IPs with the actual MAC.

4 Proposed Mechanism: EDDM

The EDDM is deployed in the SDN controller. There are three phases in EDDM, *Window Construction Phase*, *DDoS Detection Phase* and *DDoS Mitigation Phase*. The flow chart is shown in Fig. 1.

4.1 Initialization and Window Construction Phase

Initialization and Window Construction Phase collects and extracts some relevant arguments from *Packet-ins*. It is shown as Initialization and Step 1 in Fig. 1.

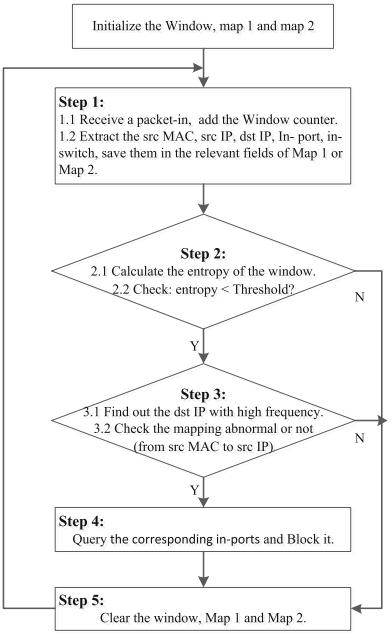

Fig. 1. The main flow chart of EDDM

The principles of entropy-based DDoS attacks detection, the essential difference between DDoS attacks and flash crowds, are introduced in the previous section. For the purpose of detecting DDoS attacks accurately, we propose a new two-stage detection method in the DDoS attack detection phase (see the next subsection), which need to collect and save the relevant arguments in Map with the customized value field (as shown in Fig. 2).

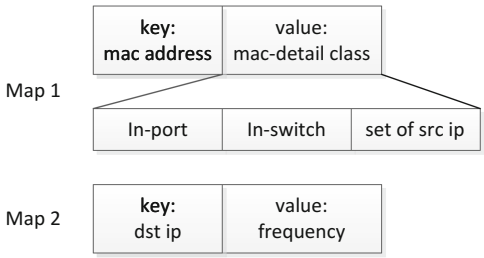

Fig. 2. Data structures of Map 1 and Map 2

The EDDM detects DDoS attacks by analyzing header information (such as the source IP, the destination IP and the source MAC) of *Packet-ins*. As proved in [9], we calculate the entropy of the destination IP address with a specific **Window** size 50. As shown in Fig. 2, we use two kinds of Map to store different arguments for different purposes in the SDN controller. Each pair in ***Map 1*** is used to store arguments of the packet(s) with the same source MAC address. The *key* field is filled with the source MAC address, and the *value* field is customized with the *mac-detail class* which is defined with three instance variables, *In-switch, In-port,* and *set of src ip*. *In-switch* represents the number of the OF switch where the *Packet-in* sent from. *In-port* represents the ingress port of the flow on the *In-switch*. The *set of src ip* stores the source IP address(es) with the same source MAC address, which is the critical indicator in the second stage of the detection phase (the Step 3 in Fig. 1) because the attack packets from a zombie host always have various source IP addresses. Each pair in the ***Map 2*** is used to store the destination IP addresses and their frequencies in the *Window*. The *Map 2* is used to calculate the entropy value of the *Window* in the first stage of *Detection Phase* (the Step 2 in Fig. 1).

When controller receives a *Packet-in*, it will check whether the *Window* is full or not. If the *Window* is not full, controller will extract relevant header information and store them in the relevant fields in *Map 1* or *Map 2*. When the *Window* is full of 50 *Packet-ins*, controller will go to the *Attack Detection Phase*.

4.2 DDoS Attack Detection Phase

DDoS Detection Phase detects the DDoS attack accurately by the two-stage detection method. Step 2 and Step 3 in Fig. 1 are included in this phase.

The SDN controller can receive many *Packet-ins* with the same destination IP when DDoS attack is in progress. But the flash crowd can also cause similar consequence. It's inaccurate to take the entropy of destination IP as the sole indicator in DDoS detection. After found the essential difference between DDoS attack packets and flash crowd packets, we propose a two-stage detection method as the second phase of EDDM.

In the first stage (see the Step 2 in Fig. 1), controller calculates the *entropy* of the destination IP using the arguments in *Map 2* (according to the mathematical *formula* (1)). If the entropy is higher than the preset **Threshold** (we chose 1.31 as proved

in [9]), the flows which are represented by the *Packet-ins* in the *Window* are legitimate, and controller will execute the Step 5. If not, they are abnormal flows which need to go to the second stage for further detection.

In the second stage (see the Step 3 in Fig. 1), controller finds out the destination IP with a high frequency in the *Window*, and checks whether the corresponding *Packet-ins* have informal mapping relationships (from the source MAC to the source IP) by checking the *size* of *set of src ip* (the instance variable of the *mac-detail* class customized in the *value* field of *Map 1*). If the *size* is more than two (which means the host sends packets with more than two source IPs), controller can determine that the corresponding *In-port* of the *In-switch* (both in the *value* field of *Map 1*) is connected to a zombie host. Otherwise, that means there is a flash crowd caused by some hot events in network.

If there is no zombie host found in the *Window*, controller will clear the *Window*, *Map 1* and *Map 2* to prepare to build the next *Window*. Or else it goes to the *Attack Mitigation Phase*.

4.3 DDoS Attack Mitigation Phase

DDoS Mitigation Phase blocks the attack flows. It is shown as the Step 4 in Fig. 1.

In theory, the best way to mitigate DDoS attacks is blocking the attack packets at source, which minimizes impacts on the network bandwidth and the SDN controller. The existing studies have not considered tracing to the attack sources effectively, in this way dropping packets is a general method to mitigate DDoS attacks, but they may drop some legitimate flows that cause QoS and SLA degraded. Besides, the zombie hosts send legitimate looking packets with forged source IP addresses to conceal their locations, so tracing to the bots and blocking attack packets at source is difficult to achieve. The inevitable result is massive packets with forged source IP entering into the OpenFlow switches. As a result, the SDN controller is busy in processing attack flows, cannot deal with legitimate flows normally.

However, the EDDM can find out the bots and query corresponding *In-port* of the *In-switch* (stored in the *value* field of the *Map 1*). Then, the SDN controller can issue *Packet-outs* to relevant switches to establish corresponding flow table entries, and the ingress ports connected to those bots will be blocked, the attack packets cannot enter into the network to affect the SDN controller and the target host.

5 Simulation Results

This study presents EDDM developed with the Floodlight [14], a Java-based SDN controller, which is one of the mainstream SDN controllers at present. In order to evaluate the proposed novel EDDM, a virtual scenario is simulated. Mininet [15] is a great way to develop, share, and experiment with OpenFlow and SDN systems. It can create a realistic virtual network, running real kernel, switch and application code, on a single machine (VM, cloud or native), in seconds, with a single command. Besides, sFlow [16] can display and monitor the traffic on the network, which is used in virtual networks but also in real ones.

The simulations use a custom Mininet script for the network topology. The test topology configuration is shown in Fig. 3. We test the effectiveness of EDDM which is running over the Floodlight controller in a small SDN network, which contains a Floodlight controller, 3 OpenFlow switches and 12 hosts. The bots (signed with "B") are generating huge traffic with forged source IP addresses towards the target nodes (signed with "T"). sFlow is used to display the real-time traffic at some specific ports to illustrate the traffic state in the network.

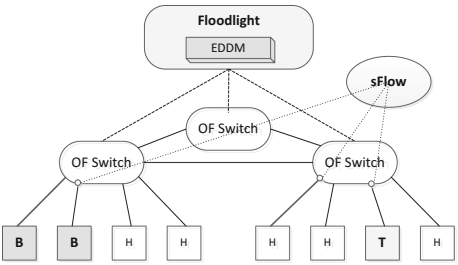

Fig. 3. The test network topology

In order to verify the impacts of the EDDM on DDoS defense, we simulate the following three scenarios: *DDoS without any defense mechanisms*, *DDoS with EDDM*, *the flash crowd and DDoS with EDDM*. We use sFlow to display the packets rate of specific ports connected to the attack zombie host, the legitimate host and the target host (see Fig. 3), and the situation of each scene can be compared visually.

5.1 DDoS Without Any Defense Mechanisms

For a better comparison, we set a reference basis. So we first simulate that the DDoS attacker attacks the SDN network without any defense mechanisms in Floodlight.

The attack packets are sent from specified bots at a high transmission rate with forged source IP addresses. The transmission rate remains high for a long while, as shown in Fig. 4, because there is no defense mechanism in the SDN network.

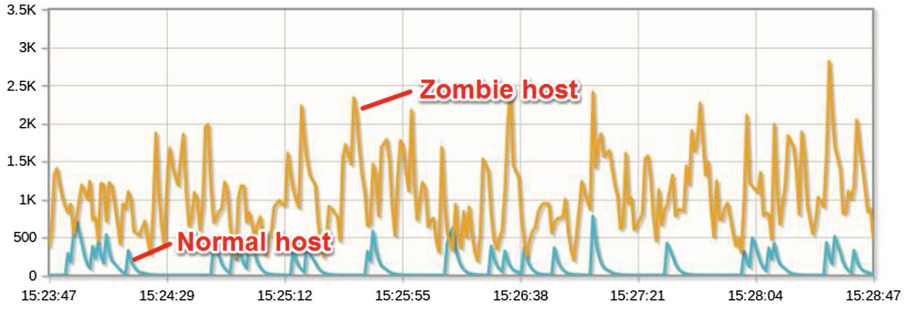

Fig. 4. The packet rate of the zombie host and the normal host without EDDM

We use the Wireshark [17], which is a free and open source packet analyzer, to capture the attack packets at the ingress port connected to a zombie host. As shown in Fig. 5, the destination IPs in the fourth column are all the same, but the source IPs (shown in the third column) are various fake IP addresses.

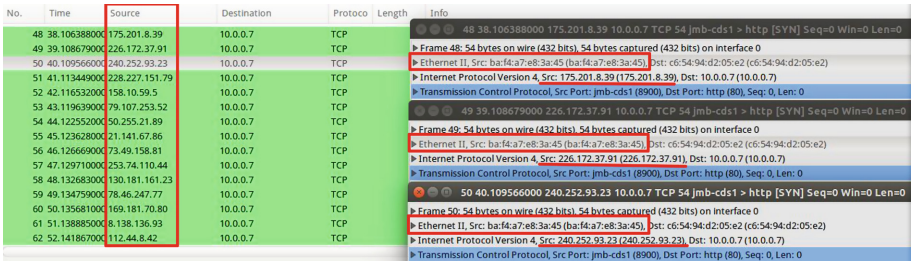

Fig. 5. The packets captured at the attack port in DDoS

5.2 DDoS with EDDM

When we run the EDDM in Floodlight, the defense mechanism can detect DDoS attacks accurately, trace to the bots and block the attack traffic at source.

Compared with the scene without any DDoS defense mechanisms, the attack packets are blocked at the ingress port of access OpenFlow switch almost immediately, as shown in Fig. 6. As a result, the attack packets cannot be delivered the *Packet-ins* to burden the controller through the secure channel between controller and switches, and cannot occupy the network bandwidth too.

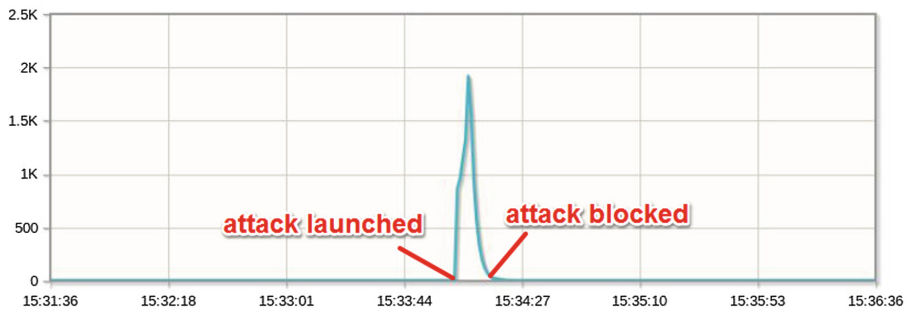

Fig. 6. The packet rate of a zombie host in DDoS with EDDM

5.3 The Flash Crowd and DDoS with EDDM

Moreover, in order to prove that EDDM is able to distinguish DDoS traffic from flash crowd flows, we simulate the flash crowd and DDoS with the EDDM. Every host in the network sends packets to the server with a high transmission rate, 3 hosts of them as the bots sending spoofed packets, while the remaining 8 hosts send legitimate packets.

As shown in Fig. 7, with the EDDM running on Floodlight, the attack flows are all blocked in the access switch, while the legitimate flows belonged to flash crowd are forwarded normally, and the packets rate of the legitimate host is unaffected. It demonstrates that the proposed EDDM can protect flash crowd flows from being misjudged as the DDoS attacks, and avoid the dropping of flash crowd packets.

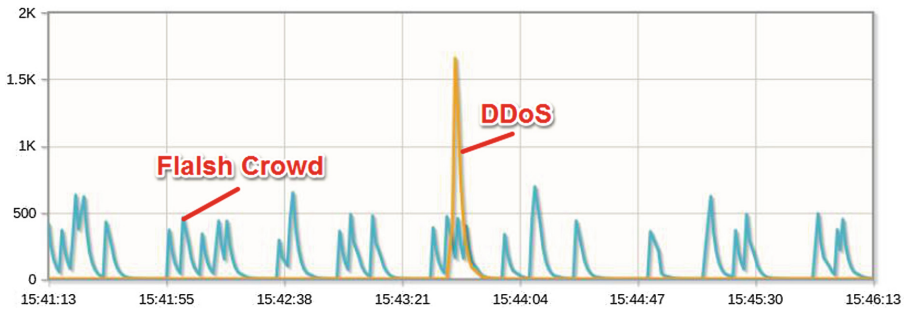

Fig. 7. The packet rate of a zombie host and a normal host with EDDM

5.4 Discussions

In [7], they confirm the DDoS attacks after the abnormal entropy in five consecutive windows, but we can detect the abnormal flows according to the entropy of one window, our decision process is quite short. In [9], they try to use the conditional entropy as a sole measure to distinguish DDoS attack from flash crowd. But they misunderstand the essential difference. The false alarm rate was up to 8.4% in the simulation. The EDDM can defense DDoS attacks with forged source IPs, distinguish DDoS attacks from flash crowd, and block attack packets at source. Thus it is able to protect the bandwidth of the secure channel, and the processing capacity of controller. Therefore, it can minimize the losses of legitimate users in the SDN networks.

6 Conclusion

In this paper, we analyze the essential difference between DDoS packets and flash crowd packets, design a two-stage DDoS detection method included in EDDM. The simulation results demonstrate that the EDDM can distinguish the DDoS from flash crowd, find out the locations of bots, and block the attack packets at source effectively. The only flaw is the processing delay is undesirable.

In the future, we will focus on the optimization of processing delay, investigate on the influences of the EDDM using different topologies with larger scale, and test it on a real physical network.

References

1. Vizváry, M.: Mitigation of DDoS attacks in software defined networks
2. Braga, R., Mota, E., Passito, A.: Lightweight DDoS flooding attack detection using NOX/OpenFlow. In: 2010 IEEE 35th Conference on. IEEE Local Computer Networks (LCN) (2010)
3. Van Trung, P., Huong, T.T, Van Tuyen, D., et al.: A multi-criteria-based DDoS-attack prevention solution using software defined networking. In: International Conference on Advanced Technologies for Communications (ATC), pp. 308–313. IEEE (2015)
4. Lim, S., et al.: A SDN-oriented DDoS blocking scheme for botnet-based attacks. In: 2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN). IEEE (2014)
5. García de la Villa, A.: Distributed denial of service attacks defenses and OpenFlow: implementing denial-of-service defense mechanisms with software defined networking (2014)
6. Dong, P., Du, X., Zhang, H., et al.: A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows. In: IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2016)
7. Mingxin, W., Huachun, Z., Jia, C., et al.: An entropy based anomaly traffic detection approach in SDN. Telecommun. Sci. **31**(9), 2015217 (2015)
8. Mousavi, S.M., St-Hilaire, M.: Early detection of DDoS attacks against SDN controllers. In: 2015 International Conference on Computing, Networking and Communications (ICNC). IEEE (2015)
9. Shu, Y., Mei, M., Huang, H.: Research on DDoS attack detection based on conditional entropy in SDN environment. Wirel. Internet Technol. **5**, 75–76 (2016)
10. Jantila, S., Chaipah, K.: A security analysis of a hybrid mechanism to defend DDoS attacks in SDN. Proc. Comput. Sci. **86**, 437–440 (2016)
11. Hwang, T., Liu, J., Wei, L.: SDN Core Principles and Application Practice, 36–37. Post & Telecom Press, Beijing (2014)
12. Information Entropy. https://en.wikipedia.org/wiki/Entropy_(information_theory)
13. Luo, K., Luo, J., Yi, M.: Survey on distinction between flash crowd and DDoS attacks. Comput. Sci. **11A**, 313–316 (2015)
14. Floodlight. http://www.projectfloodlight.org/floodlight/
15. Mininet Study Guide. http://www.sdnlab.com/11495.html
16. sFlow Overview Documents. http://www.sflow.org/about/index.php
17. About Wireshark. https://www.wireshark.org/#learnWS