Firewall Application for ONOS SDN Controller

Nan Haymarn Oo, Aung Htein Maw
University of Computer Studies, Yangon
nanhaymanoo@ucsy.edu.mm, ahmaw@ucsy.edu.mm

Abstract

Firewall provides filtering packets among network devices according to the security policy. As the firewall is an essential devices in traditional network, it is also a crucial application in the security of modern network, Software Defined Network (SDN), which is the physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices by controller. This paper focus on developing a Ping Flood and Ping of Death attacks protection firewall application running on Open Network Operating System (ONOS) SDN controller. The firewall application enhances the oneping application in order to permit ping packet's source and destination only predefined in the list of security policy prohibiting the reconnaissance from attackers. Hence, this application overcomes the weakness of existing oneping application that allow any ping once.

Keywords: *Firewall, SDN, ONOS, Ping of Death, Ping Flood*

1. Introduction

Most organizations need to manage their incoming and outgoing traffic of their networks: to prevent attacks against their computer systems, to prevent attacks originating from their network against other organizations, to prevent attacks originating from inside of the organization against other parts of the organization [insider threat - i.e a user in Student network trying to get into the Faculty network], and to confirm with various policy choices. The firewall is the primary control point for these tasks.

Firewalls can be divided into two categories: stateless firewall, and stateful firewall depending on whether the firewall keeps track of the state of network connections or treats each packet in isolation. Stateless firewall keeps no state. The filtering decision is made separately for every packet, and does not take into account any earlier decisions made on related packets. Thus, this type of firewall needs to write two filtering rules in order to allow both directions of traffic through it. Stateful firewall keeps track of established flows, and all the packets that belong to an existing flow, in both directions, are allowed to cross it. Hence, a single rule is enough to describe a flow, whereas stateless firewall need two rules for the same flow.[3]

Software defined networking (SDN) is one of the recent trend in network technology. It emphasizes the separation of the network management and data forwarding layer. It is also called control plane and data plane respectively. Data plane is only responsible for packet forwarding in the network. Control plane is responsible for policy creation and its implementation, based on predefined rules. Any packet coming into the network is examined by the control plane and a decision is taken on whether to drop the packet or forward it to the next host. Thus, instead of each networking device independently forwarding packets to the next hop, the controls are centralized on SDN controllers. The communication between control plane and data plane is performed by Open Flow protocol. It carries the message between SDN controllers and the underlying network infrastructure, bringing network applications to life.

The more common SDN security concerns include attacks at the various SDN architecture

layers. There are attacks at data plane layer, control plane layer and application layer. Attackers could target the network elements from within the network itself. An attacker could theoretically gain unauthorized physical or virtual access to the network or compromise a host that is already connected to the SDN and then try to perform attacks to destabilize the network elements. This could be a type of Denial of Service (DoS) attack or it could be a type of fuzzing attack to try to attack the network elements.

Controller is also an attack target. Attackers would try to target the SDN controller for several purposes. The reason is that, if they can handle the controller, the attacker would have complete control of the network. An attacker might try to perform a DoS of the controller or use another method to cause the controller to fail.

If the attacker could leverage the vulnerable northbound API, then the attacker would have control over the SDN network through the controller. If the controller lacked any form of security for the northbound API, then the attacker might be able to create their own SDN policies and thus gain control of the SDN environment. [8]

Ping command is used to mostly check if a server or a gateway is up and running. However, ping command can also be used for some other purposes. A ping packet can also be malformed to perform Denial of Service (DoS) attack by sending continuous ping packets to the target IP address. A continuous ping will cause buffer overflow at the target system and will cause the target system to crash. A ping flood(PF) attack typically occurs when ICMP echo requests overload its victim with so many requests that it expends all its resources responding until it can no longer process valid network traffic. [2]

Increasing the size of ping packet unnaturally, forming a malformed ping packet to attack a computer system, this type of attack is called Ping of death (PoD) attack. PoD is a type of DoS attack in which an attacker attempts to crash, destabilize, or freeze the targeted computer or service by sending malformed or oversized packets using a simple ping command.

BlackNurse attack is able to bring down the firewalls or other network devices with small amount of traffic (15-20 Mbit of traffic with packet rates of 40,000 to 50,000). This type of attack is launched in 2016 and its method is relatede to both ping flood attack and ping of death attacks because it takes the ICMP ping unnatural packet rate and size.

ONOS controller has implemented oneping application. It allows ping one host to another only one time. After 60 seconds, it allows the ping again. The objective of the application is to protect PF attack. The disadvantage of this application is permitting any packets. As a result, attackers can reconnoiter the network. Moreover, currently new attack like BlackNurse attack deals with PF attack together with PoD attack. Therefore, this paper proposed a PF and PoD attacks protection firewall on ONOS controller by enhancing the existing oneping application.

The rest of the paper is organized as follows. Section 2 lists the related works with the firewall application. Section 3 describes theory backgrounds of this paper. Section 4 introduces the process of firewall application over ONOS controller. Section 5 presents the results of the implemented firewall application. Section 8 describes the conclusion and future works.

2. Related Works

M. Suh, et al [6] built firewall over POX SDN controller and used the VirtualBox and Mininet. They focused on simple access control list based SDN firewall and did not consider about the protection of any specific attack.

S. Morzhov, et al [9] created firewall application for Floodlight SDN controller. However this type of controller has already been implemented firewall application. Therefore they focused on building Prefirewall to avoid collisions, redundancy and overlapping rules by monitoring the adding new rules.

J Jeong, et al [5] proposed a framework for security services using SDN and specified requirements for such a framework. They described centralized firewall system and centralized DDoS-attack mitigation system and its limitations of legacy systems respectively. They did not implement any system and only proposed the frameworks.

A. Arins [1] proposed DDoS firewall as a service in SDN networks with ONOS controller. However, as ONOS aims to become the SDN system platform for controlling service provider networks, the author only focused on providing firewall service especially for ISP network.

This paper developed a firewall application that can be able to protect both PF and PoD attacks as well as prohibit the reconnaissance from attackers. In addition, this application especially takes into account the insider threats by setting the rule in the firewall with the addresses of the inside network.

3. Theory Background

There are three main parts in this theory background for firewall application over ONOS SDN controller. How general firewall application is working in SDN is described in the first part of the section. And the second one shows the differences between traditional firewall and SDN firewall. The final part briefly explains the architecture of ONOS controller and its existing application, oneping.

3.1. Firewall Application in SDN

Figure 1 shows the implementation scenario of a firewall application with SDN/ Openn Flow. It demonstrates using two firewall rules in security policies and showing the block packet.

Firstly, host A sends packet to the Open Flow switch and then it reports it to the controller if there is no flow rules in it and controller forwards this information to the firewall application. The firewall application parses the received packet and checks whether the incoming packet violates security policies or not. Then it enforces a flow rule based on the policies. And controller delivers the rule to the switch. Finally, the switch puts the delivered rule into its flow table.[10]

Generally, centralized firewall application in SDN controller is implemented as mentioned in the above scenario. The firewall application in this paper is developed based on oneping application. Hence, the application needs more processes than this scenario.

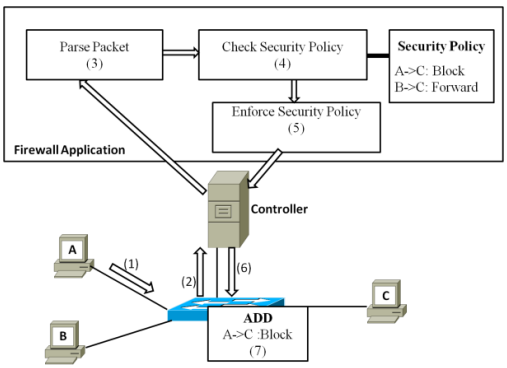

Figure 1.Firewall Implementation with SDN

3.2. Traditional Firewall versus SDN Firewall

There are a few differences between traditional and SDN based firewalls.

A traditional firewall is placed between a private network and the public Internet, and inspects packets that attempt to cross a network boundary as shown in figure 2. It assumes all insiders of the protected network are trusted, since internal traffic is not seen and cannot be filtered by the firewall. [4]

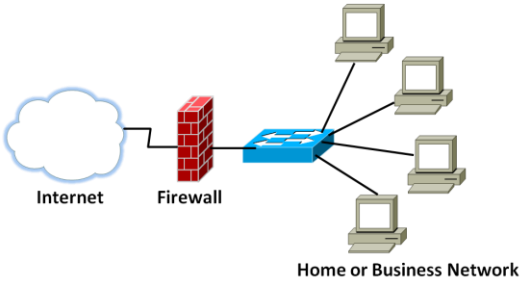

Figure 2. Traditional Firewall

SDN firewall as in figure 3 monitors all insiders as Open Flow offers a deeper level of control granularity by placing enforcement points in any entries of traffic flows in a

network. An SDN based firewall works both as a packet filter and a policy checker. The first packet goes through the controller and is filtered by the SDN firewall. The subsequent packets of the flow directly match the flow policy defined in the controller. The firewall policy is centrally defined and enforced at the controller.[4]

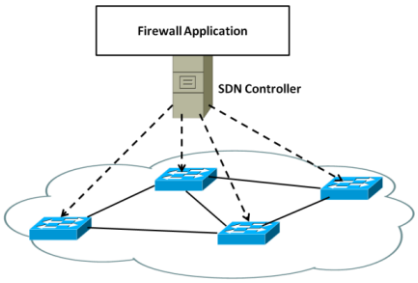

Figure 3. SDN Firewall

3.3. ONOS Controller Architecture

ONOS system is comprised of several layers as in figure 4. The pivotal core is Distributed core, which is responsible for presenting a logically centralized view of network state and logically centralized access to network control functions. Southbound interface is a high-level API through which the core interacts with the network environment using the protocol such as OpenFlow, Netconf, or others. ONOS core exposes a set of abstractions to applications and additional network services via its northbound API. [7]

Oneping application allow ping one host to another only one time. After 60 seconds, it allows the ping again.

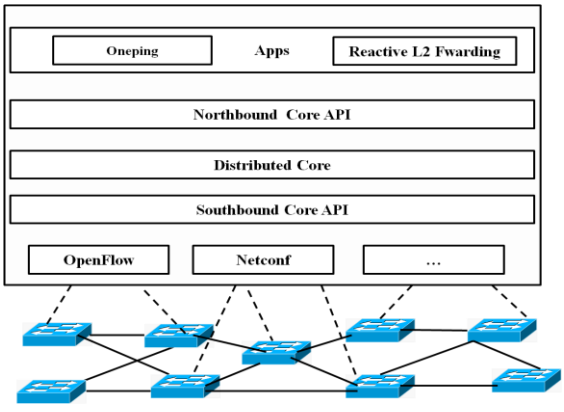

Figure 4. ONOS Controller Architecture

4. Firewall Application Process over ONOS Controller

Our firewall application add some codes to oneping application for inserting policy file and matching the incoming packets with the rule containing in the policy file. The following algorithm 1 shows the process of matching rules with the incoming packets. If the source and destination addresses of the packet are matched with the predefined rule, the packet will be allowed ping once. Otherwise, the packets will be blocked. Before allow the packet, the application checks whether the payload of packet is greater than the predefined threshold value. As the maximum allowable IP packet size is 65535 bytes including IP header (20 bytes) and ICMP header (8 bytes), the maximum allowable size of the data area of an ICMP echo request is 65507 bytes (65535 - 20 - 8 = 65507). However, the threshold value for the size of ping packet in this paper is defined as 56 bytes because it is a common ping packet size. Only if the size of the incoming ping packet is less than the threshold value, the packet will be allowed. Or else, it will be blocked assuming it may causes the PoD attack. The rule matching process is performed as usual firewall. Packets are matched with one rule after another from top to bottom. Once the matching rule is found, the process is stopped.

4.1. Simulation

The three programs were used for testing this firewall application. They are VirtualBox, Mininet and ONOS. VirtualBox is used to create the virtual network environment. Mininet is for providing virtual SDN network topology. ONOS is to control the SDN network as SDN controller. The test in this paper is specific for only Ethernet to allow or block using PING between five hosts. Thus, the prototype for testing is a single switch with five hosts over mininet as the following figure 5.

Algorithm 1 Matching Firewall Rules in Policy File

1: **Input**: Policy File line (srcMac, dstMac) , Incoming Packet src , dst, actualpayload
2: **Output**: Ping Result : allow (oneping) or block(banping)
3: Set hit = false, threshold = 56
4: Read line in Policy File
5: **while** line != null || hit == false **do**
6: Set Array Sx = Split line
7: Set srcMac = S[1], dstMac = S[2]
8: **if** src == srcMac && dst == dstMac **then**
9: **if** actualpayload > threshold **then**
10: Return banping
11: **else**
12: Return oneping
13: hit = true
14: **else** Read line in Policy File
15: **end if**
16: **end while**
17: **if** line == null **then**
18: Return banping
19: **end if**
20: End

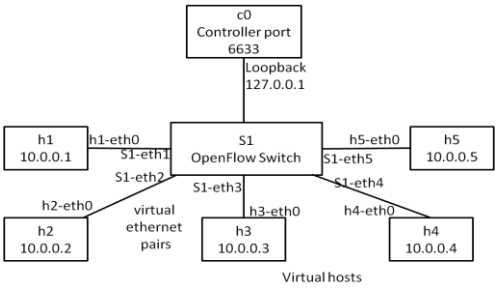

Figure 5. SDN Network Topology for Testing

The sample rules containing in the firewall policy file are:

1, 00:00:00:00:00:01, 00:00:00:00:00:03
2, 00:00:00:00:00:03, 00:00:00:00:00:01
3, 00:00:00:00:00:02, 00:00:00:00:00:04
4, 00:00:00:00:00:04, 00:00:00:00:00:02
5, 00:00:00:00:00:03, 00:00:00:00:00:05
6, 00:00:00:00:00:05, 00:00:00:00:00:03
7, 00:00:00:00:00:04, 00:00:00:00:00:05
8, 00:00:00:00:00:05, 00:00:00:00:00:04
9, 00:00:00:00:00:01, 00:00:00:00:00:05
10, 00:00:00:00:00:05, 00:00:00:00:00:01

4.2. Simulation Results

Since the firewall application is assumed to allow the matched PING packets with the rules in the policy file, all the remaining packets are blocked by the firewall.

Figure 6 describes blocking PING packets between h1 and h2 and its log messages as the rule for h1 to h2 is not included in the policy file. And figure 7 also expresses allowing PING packet from h2 to h4 with 50 percent packet loss. Even though the packet matched with one of the rules in the policy file, oneping application permit only one of two pings request. Moreover, this figure also shows the log messages for each PING packet. While the existing oneping application allows any ping, the firewall application blocks some packet according to the rules in the policy file. As a result, the firewall application can filter the untrusted PING traffic from attackers.

Then again, the comparison results of normal ping and abnormal ping are shown in figure 8 and 9 respectively that are especially for proving the PoD attack. Figure 10 shows the ping with oversize of common ping packet and its log messages.

mininet> **h1 ping -c 2 h2**
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
--- 10.0.0.2 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1024ms

2016-11-23 03:38:24,946 | INFO | ew I/O worker #2 | Firewall | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | 00:00:00:00:00:01 to 00:00:00:00:00:02 is blocked

Figure 6. Results of Blocking PING Packet between h1 and h2 of Firewall Application

```
mininet> h2 ping -c 2 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=41.1 ms
--- 10.0.0.4 ping statistics ---
2 packets transmitted, 1 received, 50% packet loss, time 1001ms
rtt min/avg/max/mdev = 41.129/41.129/41.129/0.000 ms

2016-11-23 03:38:00,398 | INFO  | ew I/O worker #2 | Firewall                  | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | Thank you, Vasili. One ping from 00:00:00:00:00:02 to 00:00:00:00:00:04 received by of:0000000000000001
2016-11-23 03:38:00,398 | INFO  | ew I/O worker #2 | Firewall                  | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | 00:00:00:00:00:02 to 00:00:00:00:00:04 is allowed once
```

Figure 7.Results of Allowing PING Packet between h2 and h4 of Firewall Application

As the firewall often has a tradeoff between security and performance, our firewall application also has a bit tradeoff when comparing with oneping application. Figure 9 shows the performance comparison of the two applications. We take the average time of ping packets from some source hosts to destination hosts that are allowed in the policy file. We used the same topology, and pair of source host and destination host for both applications in order to get the comparison of ping results from them. Although our firewall application improves security by blocking denied packets according to the policy file, the performance degradation by the application is not more than 11 percent of oneping application. In addition, the time complexity and space complexity for additional firewall matching rule algorithm is only *O(n)* and *O(n)* respectively and n is the number of lines in the policy file. Thus, we consider the developed firewall application is reasonable for security oriented organizations.

```
mininet> h1 ping -c 1 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3:icmp_seq=1 ttl=64 time=168 ms

--- 10.0.0.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 168.058/168.058/168.058/0.000 ms

2016-12-16 00:54:39,304 | INFO  | ew I/O worker #3 | Firewall | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | Actual load:84
2016-12-16 00:54:39,304 | INFO  | ew I/O worker #3 | Firewall | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | Threshold:56
2016-12-16 00:54:39,304 | INFO  | ew I/O worker #3 | Firewall | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | Thank you, Vasili. One ping from 00:00:00:00:00:01 to 00:00:00:00:00:03 received by of:0000000000000001
2016-12-16 00:54:39,305 | INFO  | ew I/O worker #3 | Firewall | 175 - org.onosproject.onos-app-oneping - 1.7.0.SNAPSHOT | 00:00:00:00:00:01 to 00:00:00:00:00:03 is allowed once
```

Figure 8.Results of Allowing PING Packet between h1 and h3 with Normal Packet Size in Firewall Application

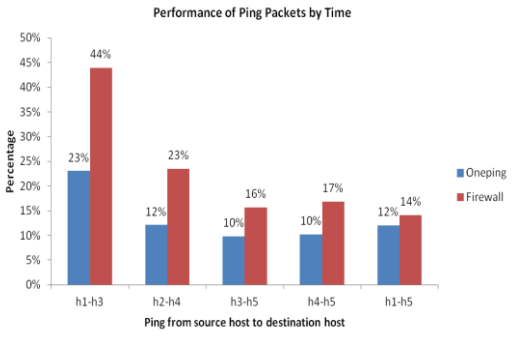

Figure 9. Results of Performance Comparison between Oneping and Firewall

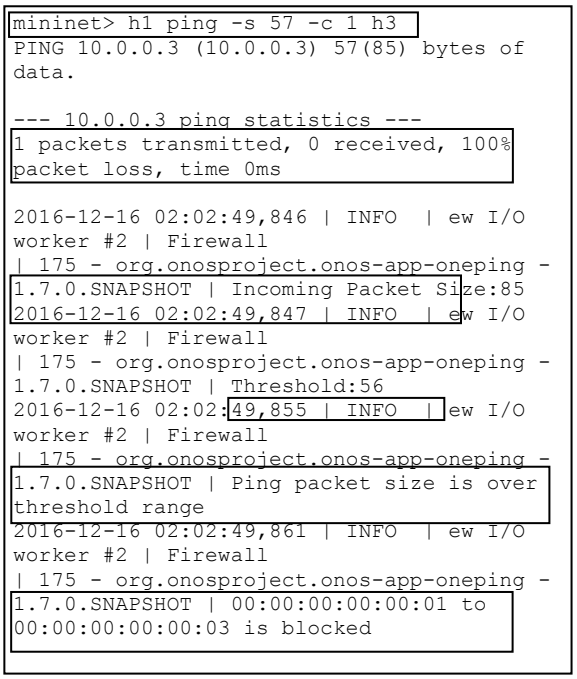

Figure 10. Results of Blocking PING Packet between h1 and h3 with Abnormal Packet Size in Firewall Application

8. Conclusion

Firewall is an excellent security mechanism not only in conventional network but also in advanced network, SDN. It protect networks from intruders, and it can establish a relatively secure barrier between a system and the external environment. ONOS controller is developed to be able to control in various platform of SDN networks from data center, or campus network to service provider networks. Thus, essential network security applications are required to develop for the controller. Therefore, firewall application over SDN controller is developed in this paper. Since the application is based on the existing `oneping` application, it is only layer 2 packet filtering or stateless firewall for ICMP protocol. It will be extended to be stateful firewall application which can filter not only ICMP but also IP, TCP, and UDP protocols.

References

[1] A. Arins, "Firewall as a service in SDN OpenFlow network", Information, *Electronic and Electrical Engineering (AIEEE)*, IEEE 3rd Workshop on Advances, 2015 Nov 13, pp. 1-5.

[2] *Attack Detection and Defense Mechanisms*, Revision 2, Juniper Networks, Inc.,1194 North Mathilda Avenue Sunnyvale, California 94089, USA, 2012- December .

[3] A. Wool, "Packet filtering and stateful firewalls", Handbook of Information Security,3, 2006, pp. 526-536.

[4] "Implementing software-defined network (SDN) based firewall", http://opensourceforu.com/2016/07/implementing-a-software-defined-network-sdn-based-firewall, 2016-July-5.

[5] J. Jeong, J. Seo, G. Cho, H. Kim, JS. Park, "A Framework for Security Services based on Software-Defined Networking", *Advanced Information Networking and Applications Workshops (WAINA*), IEEE 29th International Conference, 2015- Mar-24, pp. 150-153.

[6] M. Suh, SH. Park, B. Lee, S. Yang, "Building firewall over the software-defined network controller", 16th International Conference on Advanced Communication Technology, 2014-Feb-16, pp. 744-748.

[7] T. Vachuska, "Overview of ONOS Architecture", https://wiki.onosproject.org/display/ONOS/Overview+of+ONOS+architecture, 2015- May -14.

[8] S. Hogg, "SDN Security Attack Vectors and SDN Hardening [opinion]", http://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html, 2014, Oct 28.

[9] S. Morzhov, I. Alekseev,M. Nikitinskiy, "Firewall application for Floodlight SDN controller", *International Siberian Conference on Control and Communications (SIBCON)*, 2016-May-12, pp. 1-5.

[10] S. Shin, L. Xu, S. Hong, G. Gu, "Enhancing Network Security through Software Defined Networking (SDN)", *Computer Communication and Networks (ICCCN)*, 25th International Conference, 2016- Aug, pp. 1-9.