Information Sciences 327 (2016) 288–299

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Leveraging software-defined networking for security policy enforcement

Jiaqiang Liu [a], Yong Li [a,*], Huandong Wang [a], Depeng Jin [a], Li Su [a], Lieguang Zeng [a], Thanos Vasilakos [b]

[a] Department of Electronic Engineering, Tsinghua University, Beijing, China
[b] Department of Computer Science, Kuwait University, Safat, Kuwait

ARTICLE INFO

Article history:
Received 9 June 2014
Revised 21 July 2015
Accepted 15 August 2015
Available online 21 August 2015

Keywords:
Software defined network
Security
Network update

ABSTRACT

Network operators employ a variety of security policies for protecting the data and services. However, deploying these policies in traditional network is complicated and security vulnerable due to the distributed network control and lack of standard control protocol. Software-defined network provides an ideal paradigm to address these challenges by separating control plane and data plane, and exploiting the logically centralized control. In this paper, we focus on taking the advantage of software-defined networking for security policies enforcement. We propose a two layer OpenFlow switch topology designed to implement security policies, which considers the limitation of flow table size in a single switch, the complexity of configuring security policies to these switches, and load balance among these switches. Furthermore, we introduce a safe way to update the configuration of these switches one by one for better load balance when traffic distribution changes. Specifically, we model the update process as a path in a graph, in which each node represents a security policy satisfied configuration, and each edge represents a single step of safely update. Based on this model, we design a heuristic algorithm to find an optimal update path in real time. Simulations of the update scheme show that our proposed algorithm is effective and robust under an extensive range of conditions.

1. Introduction

With the rapid development of information technology, we have now entered a networked world, in which we access information, communicate with others and run business, do all of them on the Internet. Security then becomes an important concern for these network systems, e.g., how to prevent malicious attacks and guarantee that the data are only accessible by authorized users. Security is especially important for enterprise and data center [35–37], because they usually host a variety of services and reposit a large volume of sensitive data, e.g., user profiles, company confidential data, which could belong to multiple tenants in cloud environment [6]. If not well protected, the malicious users may get access to these sensitive data or services by exploiting system vulnerabilities like weakness of access control and using some fingerprinting techniques like port scanning, IP spoofing, etc. [42]. To achieve the protection, the operators need to draw up and deploy fine-grained policies like access control, rate-limiting, and communication isolation in the network [39,40].

* Corresponding author. Tel.: +86 13910285294.
E-mail address: liyong07@tsinghua.edu.cn (Y. Li).

http://dx.doi.org/10.1016/j.ins.2015.08.019
0020-0255/

Table 1
The example of security policies (IDS: intrusion detection system).

The target traffic	The action
srcIp = 123.4.1.*, dstPort = 22 or 443	Drop
srcIp = 115.6.2.*	Pass
dstIp = 177.8.9.6	Forward to IDS

In traditional networks, middleboxes such as firewalls and Deep Packet Inspection (DPI) are exploited to carry out security policies [2]. There are two approaches to deploy them, i.e., placing them on network paths between end-points [1] or placing them off network paths by connecting them to middle switches [9]. However, both of them are inflexible. In the first approach, inserting new middleboxes is hard due to the restriction of network topology [32]; while in the second approach, installing and updating network configuration are complicated and insecure since the operators cannot directly control packet forwarding [33]. Besides, managing and maintaining these middleboxes are expensive. For example, even a small network with tens of middleboxes requires a management team of 6–25 personnel [43].

The emergence of software-defined network (SDN) offers a great opportunity to address the above issues [22,23]. In the SDN, data plane and control plane are isolated. The operator can program the control plane to directly control packet forwarding in the data plane through standard interfaces, e.g., OpenFlow [14]. This ability enables operators to adopt applications in the control plane to flexibly and automatically configure switches to forward packets to different middleboxes, which significantly simplifies the management and minimizes the misconfiguration [41]. Besides, since OpenFlow supports more actions than forward, such as drop, modify packets header and queue [15], the operators can directly use OpenFlow switches to implement security policies, e.g., installing flow entries on them to directly drop invalid packets defined by the security policies. Furthermore, the controller has a global view of the network [7,11], therefore it is easier to guarantee the consistency and completeness of security policy enforcement. In addition, SDN-based approach reduces the requirement of human based maintenance and hence also saves the operational cost.

Given above benefits, many companies, such as Google [8,16] and VMware [30], have used SDN in their data centers. It is also broadly recognized that SDN will be widely deployed in data centers in the near future [12,17]. Therefore, in this paper, we focus on SDN based security policy enforcement. We assume that the underlying switches are OpenFlow enabled, and there is a logically centralized controller, since the control plane in SDN is expected to be logically centralized. In implementation, the controller can use multiple physical servers [11] for scalability and use FlowVisor [27] for network virtualization. Applications running on the controller generate and install flow entries on OpenFlow switches to customize packet processing.

We first show that most security policies can be transferred into flow entries and deployed on OpenFlow switches. Particularly, since the total number of required flow entries may be millions [39,40] and the flow table size of a single switch is usually limited [38], flow entries need to be deployed on multiple switches. We propose a scalable and efficient architecture with two layer OpenFlow switches for security policy enforcement. In the first layer, packets are classified into different classes and forwarded to different switches in the second layer, according to security policy constraints and load balance considerations. In the second layer, the concrete security policies of different classes are deployed on specific switches. The operator can easily scale the system by adding more switches, and efficiently use the provided processing capability by adjusting the switches assigned to each traffic class.

After that, we show that update the proposed two layer OpenFlow switches may generate security holes. But fortunately, with centralized control, the update in SDN is able to be customized and carefully planned to avoid security vulnerabilities. We introduce a graph model to characterize the configuration update process, and propose a heuristic algorithm to find a safe update sequence with no security holes and has minimum cost. We evaluate the algorithm with experimental simulation, and the results show that it is effective when the traffic load is moderate and robust to different number of traffic classes and filter switches.

The remainder of this paper is organized as follows. In Section 2 we present the proposed SDN-based approach for security policies deployment. We show why configuration update would generate security holes through a simple example in Section 3. Then in Section 4, we introduce a safe update scheme by mathematically modeling the update process. After that, we evaluate the safe update scheme by experiment simulation in Section 5. We detail the related work in Section 6 and finally conclude the paper in Section 7.

2. Security policies deployment

2.1. Motivation

Generally, each security policy consists two parts, the target traffic and the corresponding action. The target traffic can be determined by a set of packet header fields, such as IP address, port and protocol. The action includes drop, pass, forward to a middlebox, etc. We show an example of security policies for one application in Table 1. In this example, users with IP addresses 123.4.1.* are untrusted users and therefore their SSH (dstPort = 22) and HTTPS (dstPort = 443) traffic should be dropped. On the other hand, users with IP addresses 115.6.2.* are authorized and all their traffic can pass through. Besides, the server with IP address 177.8.9.6 hosts important service or sensitive data and therefore packets forwarded to it have to pass the IDS at first. In