COMPUTERS & SECURITY 53 (2015) 79–108

Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

Security of Software Defined Networks: A survey

Izzat Alsmadi[*], *Dianxiang Xu*

Department of Computer Science, Boise State University, 1910 University Drive, Boise, ID 83725, USA

ARTICLE INFO

Article history:
Received 21 January 2015
Received in revised form
19 March 2015
Accepted 21 May 2015
Available online 4 June 2015

Keywords:
Software defined networking
Security
Software Defined Security
Networking
Network security

ABSTRACT

Software Defined Networking (SDN) has emerged as a new network architecture for dealing with network dynamics through software-enabled control. While SDN is promoting many new network applications, security has become an important concern. This paper provides an extensive survey on SDN security. We discuss the security threats to SDN according to their effects, i.e., Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, and Elevation of Privilege. We also review a wide range of SDN security controls, such as firewalls, IDS/IPS, access control, auditing, and policy management. We describe several pathways of how SDN is evolving.

1. Introduction

Today Internet-based systems, such as cloud services and social networks, change their network requirements (e.g., bandwidth demand, topology, and routing information) dynamically. Hardwired networks, however, have very limited ability to cope up with such frequent changes. To address this issue, Software Defined Networking (SDN) has emerged as a new network architecture that allows for more flexibility through software-enabled network control. The basic idea is to separate control plane from data plane into a program, called controller, for dynamic orchestration of network components.

While SDN is enabling new network applications, security has become an important concern as security is not yet a built-in feature in the SDN architecture. Research has shown that various security attacks can be conducted against SDN through different network components. As SDN relies on software, code vulnerabilities also have an important impact on SDN security. Moreover, SDN offers abundant opportunities for implementing security controls as SDN controller applications. Such software solutions can enable more flexible security controls in dynamic and virtualized network environments. They provide a practical means for software-defined security control.

In this paper, we conduct an extensive survey on SDN security. We study the security threats to SDN according to their effects, i.e., Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service (DoS), and Elevation of Privilege (STRIDE). This classification of security threats, known as STRIDE (Howard and LeBlanc, 2001), has been widely applied to threat modeling of computer, software, and network systems. We also review a wide range of SDN security control applications, such as firewalls, Intrusion Detection/Protection

* *Corresponding author*. Tel.: +1 2089726299.
E-mail address: izzatalsmadi@boisestate.edu (I. Alsmadi).
http://dx.doi.org/10.1016/j.cose.2015.05.006

System (IDS/IPS), access control, auditing, and policy management. In addition, we discuss several open issues and research topics that worth further investigation.

The rest of the paper is organized as follows. To facilitate discussions on SDN security, Section 2 briefly introduces the architecture of SDN. Section 3 reviews SDN security threats and countermeasures according to the STRIDE classification. Section 4 focuses on SDN security controls. Section 5 concludes this paper.

2. SDN architecture

SDN aims at providing open, centralized, decoupled, programmable, flow-based, and dynamic network switching mechanisms.

- **Open**: Traditional networking components such as switches and routers are vendor specific. They provide limited ability for users to experiment their own networking protocols on live networks with real traffic. With SDN, developers can develop middle-boxes that interact with the controller and network switches. Many controller platforms are open source, such as OpenDayLight, Floodlight, Ryu, and Beacon.
- **Centralized**: The control of different switches is co-allocated in one logical place, i.e. the controller. In design terms, this is about splitting "the what" from "the how". Such architecture is capable of handling very dynamic network situations. For example, network traffic based on dynamically changing usage demands may require switches to suddenly join or leave a particular virtual network.
- **Decoupled:** Network functionalities include tasks related to two in-cohesive components: control and data. Splitting data from control improves overall reusability and maintainability of network systems. Policies are decoupled from switches' rules. User level security policies should be expressive and close to users' language and terms, whereas network level information (i.e. Flow or firewall rules) should be simple and close to network attributes. Furthermore, in SDN, virtual or logical network is decoupled from the physical network.
- **Programmable:** Controller can be accessed and programmed by user level applications or middle-boxes. Such programmability is considered a major characteristic of SDN. Developers can modify open source controller modules. Programmability in SDN can be extended far more than just writing applications or modifying controller functionality. It can offer network administrators the ability to write policies and monitor OpenFlow networks.
- **Flow-based management**: SDN shifts networks from IP-based to flow-based management and control. While flow-level control is technically possible in traditional networks, routing protocols make decisions based on IP addresses. SDN is a flow-based architecture, where forwarding decisions in switches are made according to flows. Records or rules in switches and firewalls are per flow. This will impact many applications that depend on network traffic. For example, typical firewall rules deny or permit packets based on source or destination IP, MAC addresses or ports. Future firewall rules may become more dynamic and be updated frequently based on real time traffic.
- **Dynamic**: A major advantage of software over hardware is that it can accommodate frequent changes for more dynamics and flexibility. Configuration or reconfiguration of hardware is labor intensive. Software can be programmed to respond to activities and make decisions dynamically. This is extremely important to those applications with highly dynamic bandwidth demand, such as cloud computing, dynamic datacenters, smart devices, and social networks.

Fig. 1 shows an overall architecture of SDN, with the consideration of the most recent technological advances. We will use this architecture as a reference throughout the paper.

SDN architecture can be divided largely into controller core or internal modules and external modules. The core modules can be seen in the central or inner rectangle of Fig. 1 including: Network manager, APIs, Network Operating System (NOS), internal services and drivers. A bare bone controller should at least contain those modules that represent main functionalities. External modules of SDN controller can be divided logically into 4 parts interacting with the controller from the 4 different directions:

- Southbound section: This can be considered as the most currently popular direction of interaction between the controller and its switches. The OpenFlow protocol is a southbound interface to the controller that represents the connecting bridge between the controller and forwarding elements such as switches. Having a non-vendor-specific protocol is important to allow all vendors join this open architecture.
  Recent improvements on SDN architecture proposed a service abstraction layer (SAL) or a hypervisor in this section to enable controller and protocols to evolve without impacting each other. Open source FlowVisor can be considered as an instance of SALs.
- Northbound section: All types of applications (also called middle-boxes) that want to interact with the controller and underlying network or traffic can be typically designed in this section. There are some proposals of a standard protocol or interface similar to OpenFlow. REST API can be considered a significant achievement toward that goal although it does not represent a standard and a secure method to communicate with the controller. In addition, there are no clear agreements regarding handshaking methods between the northbound applications and the controller, how to manage communicative permissions or authorization, or how to handle decisions' conflicts. SDN-specific policy programming languages such as Pyretic and Frenetic also communicate with the controller through northbound section. It is highly desirable that all the security applications such as firewall, access control, IDS/IPS use a common API for interactions with the controller.

From security perspectives, many concerns are raised that enabling applications to interact with controller that may have special privileges can cause several security risks.