S-SDS: A Framework for Security Deployment as Service in Software Defined Networks

Adama Coly(✉) and Maïssa Mbaye

Laboratoire D'Analyse Numérique et Informatique, Gaston Berger University, Saint-Louis, Senegal
{coly.adama,maissa.mbaye}@ugb.edu.sn

Abstract. Software Defined Networking (SDN) is an emerging networking paradigm that addresses current network design limitations. It promotes centralized control of the network by clearly separating *Control Plane* and *Data Plane*. In one hand, Security in SDN is one of the most challenging research topics. In the other hand, deployment of security as service is one of the most cutting-edge topic. In this paper, we propose a general framework for security deployment as a service in SDN networks. As a case study we proposed extension of OpenFlow protocol for IPsec VPN set. We have evaluated this proposal using a real world testbed based on Mininet and Floodlight. Preliminary results show that our proposal can enable security service without drastically degrading performance in comparison to deploy security on endpoints of communications.

Keywords: SDS · SDN · Control plane · Data plane · IPsec · OpenFlow · Security service deployment · Network Security · Floodlight · Mininet

1 Introduction

Traditional networking appears to be reaching its limits during the late of 2010s decade. Firstly, classical network design views networks as composed of specialized devices (router, switch, firewall, etc.) which have proprietary firmwares that include hard coded functions and forwarding logic. This is a big limitation to network evolution since it is almost impossible to implement a new non-standard functionality (e.g. new routing protocol or firewall extension ...) without the agreement of the device manufacturers. Deployment of new innovative network features depends on the speed with which network equipment vendors implement them in their firmwares. To overcome this limitation, network devices should be programmable. Secondly, current network devices make forwarding decision according to their local configuration set by administrator. If the device is replaced, the same tedious configuration may be done again. To overcome this

G. Bassioni et al. (Eds.): InterSol 2019, LNICST 296, pp. 92–103, 2019.
https://doi.org/10.1007/978-3-030-34863-2_9

limitation, the operating logic of network device should be separated from the forwarding function.

Software Defined Networking is an emerging networking paradigm that addresses, among others, these limitations to overcome network evolution. SDN, clearly separates *Control Plane* and *Data Plane*. *Control Plane* manages how and where to forward packets, and can be hosted by a physical server or in the cloud. *Data Plane* that manages packet forwarding based on flow tables (routing table + access control list), is implanted inside network switches [2].

This separation of Control Plane from Data Plane and centralisation in an equipement called Controller, bring new threats like Single Point of Failure and make difficult the deployment of security services on equipements. However, providing security services in SDN is one of the most challenging topics in this area [1]. It covers two aspects: security for SDN infrastructure itself and deployment of security services for the end-users using a SDN core network. Securing SDN infrastructure is a fairly well-known problem now and is addressed by several works [2–7]. However, deployment of security services provided by SDN for customers is less addressed by research community. Among problems of communications' security of end-users, SDN do not yet support efficiently security services such as confidentiality, on-demand secured tunnel establishment, etc.

In this paper, we propose a framework that enables security services (e.g. IPsec Tunnels) deployment in SDN network. Main outcomes of our work are: proposition of an architecture for security service deployment in SDN, a new extension of the OpenFlow protocol for secured tunnels management, and, finally integration of an IPsec-based tunnel mechanism in SDN as use case.

The remainder of this paper is organized as follows. The second section is focused on related works while third section presents background concepts of our proposal namely SDN architecture and IPsec Protocol. On fourth part we present our approach of security services deployment architecture and present the case of an OpenFlow's IPsec extension. The fifth part is dedicated to performance evaluation. Finally, we finish by conclusion and future works.

2 Related Works

Security in SDN covers two different aspects: security for SDN infrastructure itself and deployment of security as service in SDN.

Most of prior research efforts address SDN security threats identification and SDN security solutions [1]. As proof, there are many surveys about this topic [2–7]. SDN Security threats can be different kinds such as:

- Man-in-the middle attack because of optional use of TLS [2,3];
- DDoS attacks because of single point of failure in centralized environment around the controller [2–6];
- Lack of authentication and authorization due to no compelling authentication and authorization mechanisms for applications and more threatening in case of large number of third-party applications [2].

Solutions for these threats have been proposed such as DDoS Detection (for DDoS attacks), SE-Floodlight for lack of authentication & authorization. In the case of Man-in-the- middle attack when malicious node is between the controller and the switch, TLS can be an efficient solution. When it is between different switches, solutions based on tunnels have been proposed [3].

We also have in literature design-oriented security service based on SDN/NFV for end-users. In [8], authors use firewalling for an SDN infrastructure to secure network device from suspicious and malicious traffic. In [9] authors proposed a new architecture of security service based on network virtualization functions. Their proposition offers a security service function chain that enables ICT (Information and Communication Technology) service providers to provision a dynamic and flexible secure service on the SDN network for customers. Authors in [10] propose a solution which introduces of a third plane, the security plane, in addition to the data plane and control plane. They present an SDN security design approach, which strikes a good balance between network performance and security features. This proposed approach can prevent DDoS attacks targeting either the controller or the different hosts in the network, and how to trace back the source of the attack. Proposal in [11] use SDN based IPsec authentication to secure client application. Solution in [12] consists of an architecture that combines IDS with programmable features of SDN for detection and mitigation of malicious traffic. In [3] they used IPsec as a security application to secure a communication between two endpoints against Man-in-the-Middle attacks.

We can see that researches about security services for end-users, use externals components to acheive their goal. As far as we know, there is no proposition using only the SDN infrastructure. With our approach a security service can be deployed to secure communication of two end-users using SDN core Network. The security service is provided using an extension of the OpenFlow protocol or any other Southbound API. We will present a use of IPsec as case study for OpenFlow extension. This evaluation will permit us to determine which is beneficial between deploying IPsec, using SDN approach, between End-Points and between BGS. This will lead to see if it's relevant to apply our approach.

3 Background Concepts

3.1 Software Defined Networking Architecture

Commonly adopted SDN architecture is presented in Fig. 1. Which is composed of 3 layers: Application Layer where network applications are deployed (e.g. Routing protocols; Firewall ...); Control Layer that manages the network forwarding logic and is implemented by a node called SDN Controller; finally, Infrastructure Layer where we have physical equipment and forwarding.

Communication between two layers in SDN architecture is done by using an API: NorthBound API between Application Layer and Control Layer; SouthBound API between Control Layer and Infrastructure Layer. An example of SouthBound API and the most famous is OpenFlow [13].

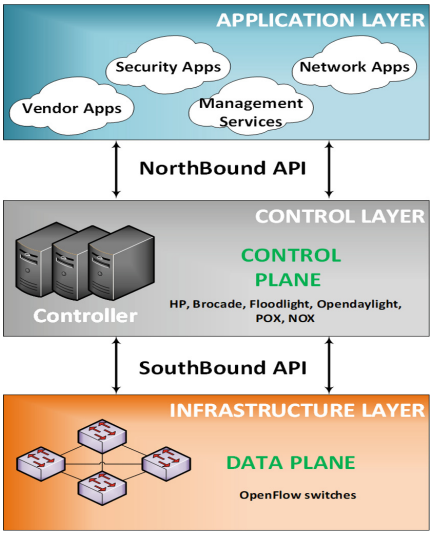

Fig. 1. Architecture SDN

SDN Security is a real challenge. Indeed, SDN should have internal security and provide security as a service for customer networks. Our main goal is to provide dynamic and extensible security service deployment architecture. With our approach a security service can be deployed to secure end-to-end communication through SDN Network in a transparent manner. We give as example the deployment of IPsec Tunnel as SDN network service. In next section will introduce the IPsec protocol.

3.2 IPsec

IPsec (Internet Protocol Security) [14] is a suite of protocols that provides security at Internet Layer of TCP/IP model. It can be used to provide a Virtual Private Network (VPN) or establish secured tunnels between two locations. The deployment of IPSec can be done between two End-Points, between two Gateways to serve different networks and between an End-Point and a Gateway. This protocol uses Internet Key Exchange (IKE) protocol for keys negotiation and management.

IPsec has two sub-protocols: The Encapsulation Security Payload (ESP) [15] and the Authentication Header (AH) [16]. A "Security Association" (SA) is a one-way (inbound or outbound) agreement between two communicating peers that specifies the IPsec protections to be provided to their communications. This includes the specific security protections, cryptographic algorithms, and secret keys to be applied, as well as specific types of traffic to be protected [17]. IPsec also uses two databases: Security Association Database (SAD) and Security Policy Database (SPD). SAD is a database for SA repository of different peers

while SPD is a database that expresses the security protections to be provided to different types of traffic [17].

4 IPsec Tunneling as SDN Security Service

4.1 General Architecture

Our proposal aims to provide secure IPsec Tunnel between two legacy networks using bump in the wire configuration (BITW). With this architecture, endpoints of communication can communicate securely by delegating security to SDN network. The general architecture is illustrated on Fig. 2. In this architecture non SDN networks are linked by a SDN core network. SDN core network is composed of Border Gateway Switches (BGS) and Core Internal Switches (CIS). BGS are ingress and egress of secured communications. They can establish on demand tunnels with instructions from controller. SDN controller communicates with BGS using extended Southbound API messages to deploy the service in Data Plane. Service is deployed only in BGS in data path. In the two cases, all switches are ready to activate security service if they receive corresponding instructions from Controller via Southbound API. An extension of OpenFlow southbound API is used for these instructions.

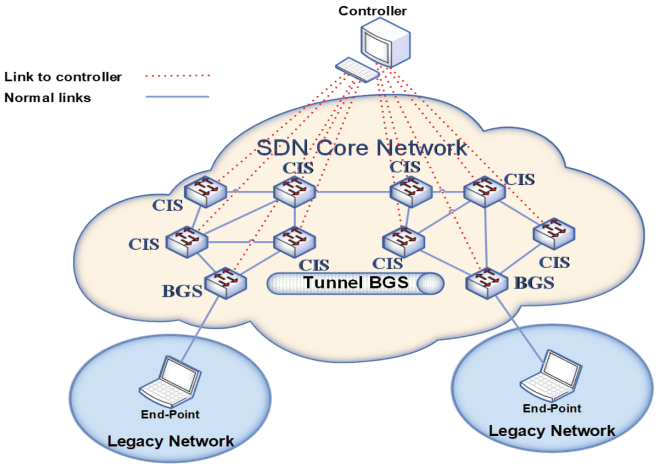

Fig. 2. Architecture of SDN based IPsec

When a client subscribes to an SLA (Service Level Agreement) including a security service, controller is configured according to this contract. If a flow matches to corresponding client traffic, controller deploys the service by sending a "SecTrans" message with security policies to the Border Gateways Switches involved in this communication. The remaining transaction is managed by South-Bound Extension. As a use case that illustrates how our proposal operates, we

deploy secure tunneling service based IPsec as SDN Security Service. To achieve this, we extended OpenFlow protocol for IPsec tunnel establishment and we define an extended structure for Flow Tables inside Switches.

4.2 Secure Communication Service Deployment

Our first proposal is an extension of OpenFlow standard protocol, by adding "SecTrans" message. It is a controller-to-switch message which is used to allow the IPsec tunnel to be used between involved switches.

Our second proposal is to add a new column on the Flow Table named"IPsec" which can be set to "Yes" to allow the use of IPsec between two BGS or "No" if we don't. The new structure of our new Flow Table is represented by the Fig. 3. With these proposals, the controller is responsible for generating and transmitting IKE credentials. It is also responsible for the control and application of IPsec SPDs. It therefore has a centralized view of the network and security policies. The IKE implemented in the network resource runs to create the IPsec

Fig. 3. New structure of our new Flow Table

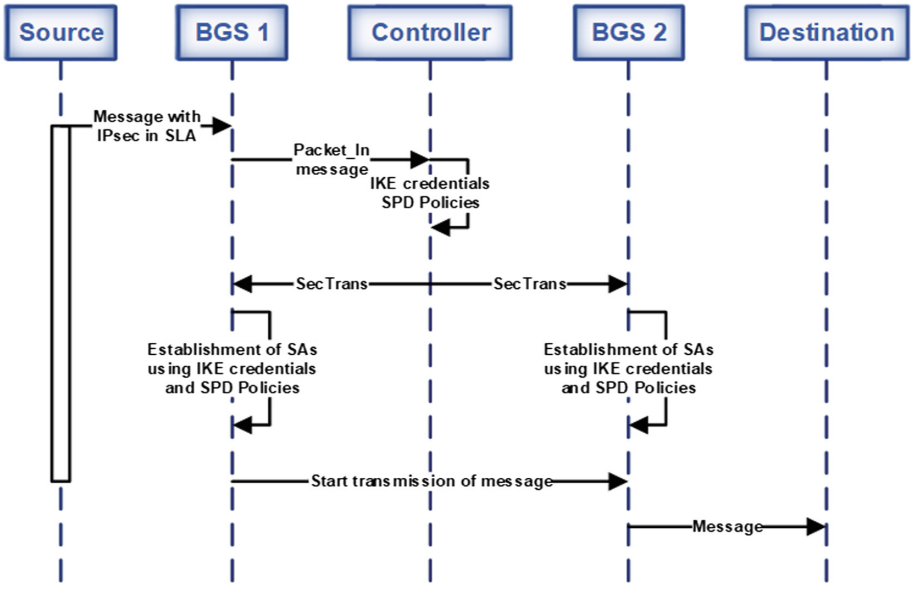

Fig. 4. Flow chart of IPsec service deployment

security associations by using these policies and credentials. Figure 4 illustrates a typical communication in SDN including deployment of IPsec tunnel.

If a source end-point sends a traffic to another client using IPsec tunnel, the following procedure will take place:

1. The Border Gateway Switch connected to the source will send a"Packet_In" message (an OpenFlow message) to the controller to ask what to do with these packets.
2. If there is matching with a service subscriber's traffic, the controller will generate IKE credentials and SPD Policies and sends them to the two switches (ingress and egress) involved BGS in addition to the SecTrans message to allow transmission in IPsec tunnel. The message will add/modify a flow with "yes" on IPsec field.
3. The border routers use the IKE credentials and SPD Policies to establish SAs before starting transmission.

When this procedure is executed, all the messages between these two endpoints are transmitted using the tunnel. In fact security is bumped in the wire in this architecture. Main benefit of this is security is deployed on BGS and used on demand by endpoint clients.

5 Implementation and Performance Evaluation

5.1 Testbed Description

We evaluated our proposal with a testbed based on commonly used environment to implement SDN, namely Mininet [18] to simulate Openflow switches and Floodlight [19] as our SDN Controller. Indeed, there are many others controllers that can be used but we used Floodlight for our first step of evaluation. Our goal is to evaluate overhead of our proposal in comparison to implement end-to-end tunnels created and maintained by edges of communications.

Table 1. Testbed systems configurations.

Hosts	Operating system	Components (software)	CPU (Core/GHz)	RAM
Controller	Debian 8.4	Floodlight master	4/3.2	4 GB
SDN BGS A	Debian 8.4	Mininet , Racoon, IPsec-tools	4/3.2	4 GB
SDN BGS B	Ubuntu 18.04	Mininet , Racoon, IPsec-tools	4/2.4	4 GB
End-Point Node A	Ubuntu 18.04	Racoon, IPsec-tools	8/4.0	8 GB
End-Point Node B	Ubuntu 18.04	Racoon, IPsec-tools	4/3.2	8 GB

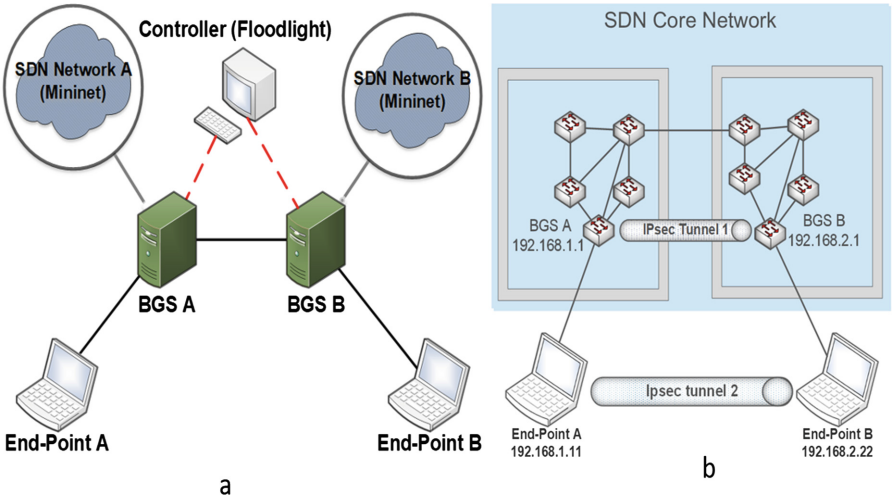

Fig. 5. a. Physical topology of testbed. **b.** Logical topology of testbed

We build a topology with five physical computers/server interconnected as illustrated in Fig. 5a and with configurations as in Table 1.

We use Mininet to run an SDN core networks by combining two networks located on different servers. We also use Floodlight as the SDN controller. Table 1 summarizes node system configurations.

On each BGS of each network is connected two End-Points illustrated as computer laptop in Fig. 5b.

For this testbed, we have two scenarios : One with an IPsec tunnel between End-Points ; and a second one with an IPsec tunnel between Border Gateway Switches (BGS). The first scenario corresponds to a normal situation in which SDN network does not play any role in IPsec Tunnel management. The second scenario corresponds to our proposal with establishment of tunnels between BGS. Comparison of these two will enable us to evaluate the overhead of our proposal.

SAs keys are managed by the Racoon daemon [20] on the different entities (see Table 1). Table 2 is a summary of IPsec SA configuration in both scenarios.

5.2 Performance Results and Analysis

We want to evaluate the impact of our proposal in the Quality of Service of the Network. We use standard QoS parameters for each scenario: Data transmission delay; Throughput; Jitters; CPU load for cryptography overhead.

Using iperf tool [21], we evaluate delay for three cases : the case of an SDN Network Without IPsec; the case of an SDN Network with an IPsec Tunnel between End-Points and finally in the case of an SDN Network with an IPsec Tunnel between Border Gateway Switches (BGS). For this evaluation, iperf will send packets in high speed then we can evaluate difference between differents cases.

Table 2. IPsec configuration.

Gateways	Flow	Source IP	Destination IP	SAs encapsulation	Secured network address
BGS A	Out	192.168.1.1	192.168.2.1	AH, ESP	172.100.1.0/24
	In	192.168.2.1	192.168.1.1	AH, ESP	
BGS B	Out	192.168.2.1	192.168.1.1	AH, ESP	172.100.2.0/24
	In	192.168.1.1	192.168.2.1	AH, ESP	
End-Point A	Out	192.168.1.11	192.168.2.22	AH, ESP	172.100.1.0/24
	In	192.168.2.22	192.168.1.11	AH, ESP	
End-Point B	Out	192.168.2.22	192.168.1.11	AH, ESP	172.100.2.0/24
	In	192.168.1.11	192.168.2.22	AH, ESP	

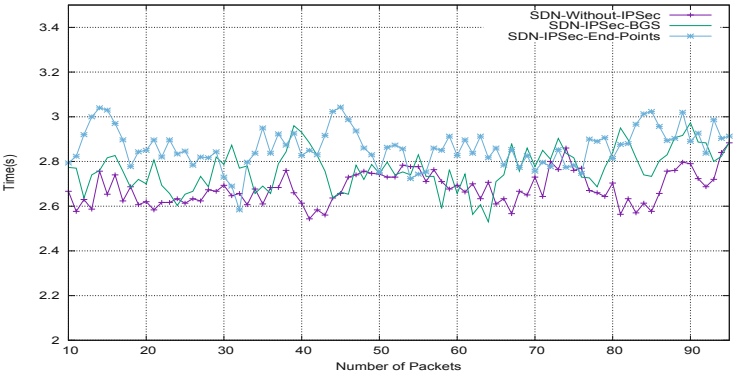

Fig. 6. Performance of SDN security service in term of Delay

Figure 6 illustrates the data transmission delay for 100 packets in related cases. We can notive that the delay is more important when the IPsec tunnel is implemented between End-Points than between Border Gateway Switches. This result can be an argument that have a security bumped in the wire as we proposed does not affect as much performance of the network in term of delay.

Figures 7, 8 and 9 represent respectively the variation of the average throughput, the jitter and CPU Load during the transmission of data in the cases as tests in Fig. 6.

We can see that the throughput when IPsec is located between BGS is more important than between End-Points. We can relate this result to the previous.

Figure 9 shows CPU variation during data transmission through the IPsec tunnel, for the Border Gateway Switch of the mininet network A (when the tunnel is implemented at the BGS) and for the End-Point A (when the tunnel is implemented at the End-Points).

We can see that the data transfer over the IPsec tunnel doesn't really affect the CPU Load of End-Point A. However, in the BGS, there is a variation of at most 4% of CPU LOAD. This is because the BGS does not have a pretty powerful CPU.

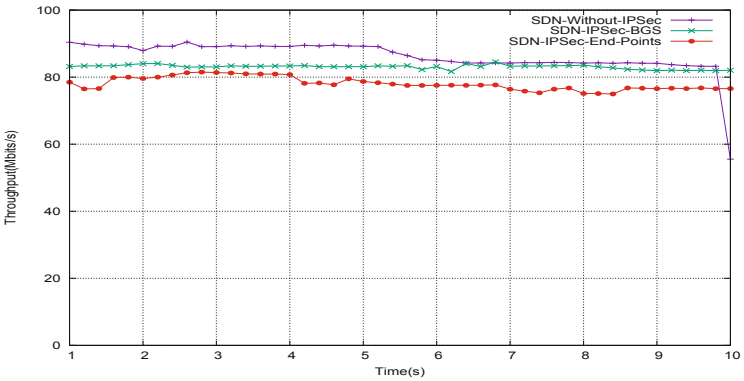

Fig. 7. Performance of SDN security service in term of Throughput

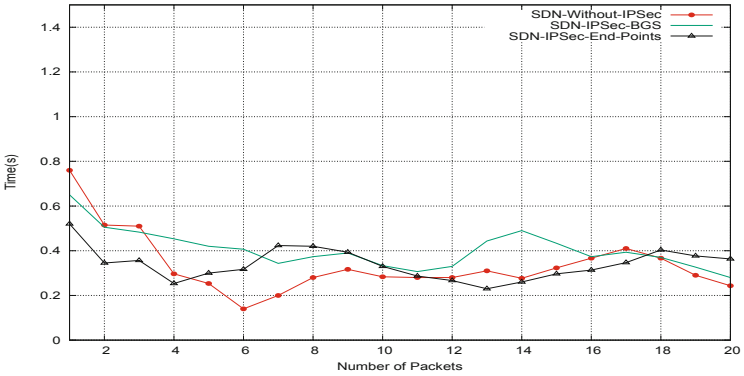

Fig. 8. Performance of SDN security service in term of jitters

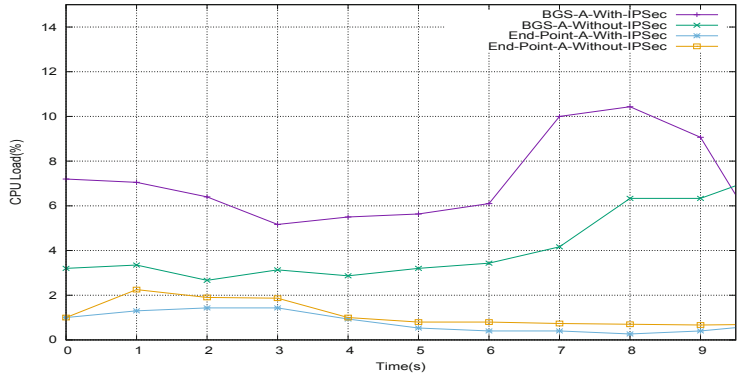

Fig. 9. Performance of SDN security service in term of CPU Load

In summary based on these results we can say that implementation of an IPsec tunnel at the BGS for the data transmission through a SDN core network does not significatively affect network performance as much in comparison to implement it at the end-Points.

Nevertheless the limits of our proposition is that we only have 2 End-Points connected to BGS. Results may be different if we have more than one End-Point on each BGS. In addition, BGS requires power of calculus in the case of they provide confidentiality.

6 Conclusion and Future Works

In this paper we have proposed a SDN architecture that enables deployment of IPsec tunnelling as SDN service. To achieve this we proposed an extension of Openflow protocol by adding a message and a security field on the flow table. Our solution moves complexity of IPsec to the controller. Customers can then subscribe to this service and transparently use it.

We evaluated performance of our proposal using a testbed based on mininet and floodlight. First results show that processing power available in Controller and use of cryptographic algorithm are crucial in forwarding performance. The uses of much powerful entities acting as BGS and Controller will be required to better supports this proposition.

Fistly, we plan to implement our approach in an environment with a powerful server, to act as a controller and physical Openflow compatible switches to build our core SDN. Secondly we will evaluate scalability of our proposal.

References

1. Bakhshi, T.: State of the art and recent research advances in software defined networking. Wirel. Commun. Mob. Comput. **2017**, 35 (2017). Article ID 7191647
2. Ahmad, I., Namal, S., Ylianttila, M., Gurtov, A.: Security in software defined networks: a survey. In: IEEE Communications Surveys and Tutorials, vol. 17, no. 4, pp. 2317–2346 (Fourthquarter 2015)
3. Ertaul, L., Venkatachalam, K.: Security of software defined networks (SDN). In: International Conference on Wireless Networks, Las Vegas, Nevada, USA, 17–20 July 2017 (2017)
4. Feghali, A., Kilany, R., Chamoun, M.: SDN security problems and solutions analysis. In: 2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS), Paris, pp. 1–5 (2015)
5. Patil, V., Patil, C., Awale, R.N.: Security challenges in software defined network and their solutions. In: 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Delhi, India, pp. 1–5 (2017)
6. Dargahi, T., Caponi, A., Ambrosin, M., Bianchi, G., Conti, M.: A Survey on the Security of Stateful SDN Data Planes. IEEE Communications Surveys and Tutorials **19**(3), 1701–1725 (2017)

7. Shin, S., Xu, L., Hong, S., Gu, G.: Enhancing network security through software dened networking (SDN). In: 2016 25th International Conference on Computer Communication and Networks (ICCCN), Waikoloa, HI, pp. 1–9 (2016)
8. Satasiya, D., Raviya, R., Kumar, H.: Enhanced SDN security using firewall in a distributed scenario. In: 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, pp. 588–592 (2016)
9. Chou, L.D., Tseng, C.W., Huang, Y.K., Chen, K.C., Ou, T.F., Yen, C.K.: A security service on-demand architecture in SDN. In: 2016 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, pp. 287–291 (2016)
10. Hussein, A., Elhajj, I.H., Chehab, A., Kayssi, A.: SDN security plane an architecture for resilient security services. In: 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), Berlin, pp. 54–59 (2016)
11. Li, Y., Mao, J.: SDN-based access authentication and automatic configuration for IPsec. In: 2015 4th International Conference on Computer Science and Network Technology (ICCSNT), Harbin, pp. 996–999 (2015)
12. Monshizadeh, M., Khatri, V., Kantola, R.: Detection as a service: an SDN application. In: 2017 19th International Conference on Advanced Communication Technology (ICACT), Bongpyeong, pp. 285–290 (2017)
13. Software-Defined Networking (SDN) Definition. https://www.opennetworking.org/sdn-definition. Accessed 13 Jan 2018
14. Seo, K., Seo, K.: Security architecture for the internet protocol. RFC 4301 (Standard), Obsoletes 2401, December 2005
15. Seo, K.: IP Encapsulating Security Payload (ESP). RFC 4303 (Standard), Obsoletes 2406, December 2005
16. Seo, K.: IP Authentication Header. RFC 4302 (Standard), Obsoletes 2402, December 2005
17. Frankel, S., Krishnan, S.: IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap. RFC 6071 (Informational), Obsoletes 2411, February 2011
18. Official website of Mininet. http://mininet.org. Accessed 11 Dec 2017
19. Official website of Floodlight. http://www.projectfloodlight.org/floodlight/. Accessed 2 Dec 2017
20. Official website of Racoon. https://packages.debian.org/fr/sid/racoon. Accessed 14 Mar 2019
21. Official website of Iperf. https://iperf.fr/. Accessed 13 Jan 2018