Framework for Creating Security Functions Based on Software Defined Network

Dobrin Dobrev
Technical University Sofia
Sofia, Bulgaria
e-mail: d_dobrev@tu-sofia.bg

Dimiter Avresky
IRIANC
Munich, Germany
e-mail: autonomic@irianc.com

***Abstract*—In this work, we propose a framework for security based on Virtual Security Functions, OpenFlow, Software Define Networks (SDN), Mininet, Pox Controller and Virtual Switches. By using the OpenFlow protocol in the virtualized environment of SDN, we are capable of analyzing the data streams in the network environment. An SDN controller, staying on top of the entire infrastructure, is capable of orchestrating network segment(s). By creating different virtual security functions, we have the possibility to increase network security and to avoid loops. In this paper, the process of loop elimination is achieved by automatically reconfiguring the security function by creating a spanning tree. By using the scalability of the virtual controller, we can simplify the network administration. The main benefit is that different systems like switches, firewalls, and Intrusion Detection Systems (IDS) will be replaced by the controller with Virtual Security Functions (VSF). The target is to increase availability by presenting functions that avoid loops in the network. VSF will allow to exploit the framework in order to eliminate different attacks, such as congestion driven attacks, Distributed Denial-of-Service (DDoS), Media Access Control (MAC) address spoofing, man in the middle attack and Synchronize (SYN) flood attacks. All functions can be run in parallel and we can increase the availability of the system.**

Keywords-security function; network function virtualization; virtualization, openflow; flowtable; controller.

I. INTRODUCTION

The classic model for information security defines three objectives of security [1]: confidentiality, integrity, and availability. The main goal of availability is to ensure that information and resources are available to those who need them i.e. security will be guaranteed. One of the possible ways to attack the network infrastructure is over a congested link. It is widely known that congestions can generate network loops and different types of attacks over the Internet. The loops can be generated when there is more than one path between two hosts. Attacks can be created by means of loops in the networks. Problems will appear via attacks that disrupt the regular communications. In this paper, the process of loop elimination is achieved by automatically reconfiguring the Spanning Tree Protocol (STP). We use STP in situations where we want redundant links, without loops. This process is automatic and it avoids deadlocks in our topologies. The proposed framework provides scalable solutions under the SDN [5] environment. Mininet [27] is used as an emulated environment. It is combined with OpenFlow [6].

II. RELATED WORKS

The major work in this paper is related to SDN. The novelty of our work is in the approach of targeting the spanning tree in order to ensure availability in the SDN environment. Similar research targeting availability can be found under research paper [2]. In this paper, the authors are creating different paths with specific metrics via SPT. Spanning tree has been targeted as a security issue as well in research paper [32]. In this work, by using POX controller and SPT, the authors are analyzing the network threats.

Based on our framework, different security functions can be created in order to ensure a higher level of security. In addition to loop elimination, by using this framework, the developers can create security functions to handle Distributed Denial of Service (DDoS) [4] and Intrusion Detection Systems (IDS) [31].

III. TECHNOLOGIES USED

A. *Software-defined networking layered segmentation*

SDN is comprised of multiple kinds of network technologies designed to make the network more flexible and agile [5]. It is an approach for using open protocols, such as OpenFlow [6]. The decision on how packets must be transferred is taken by the SDN Controller. With this type of technology, it is possible to change the topology without touching individual switches [7]. The delivery is possible with the network virtualization and the separation of data plane and control plane [8]. The Application layer and the Control layer [10] together can be considered as a Logical layer on top of the Infrastructure layer, which is the physical part of the communication [10]. At the Application layer, we are creating the Loop Elimination Function as a Security function to avoid loops by using a spanning tree, as shown on Figure 1.

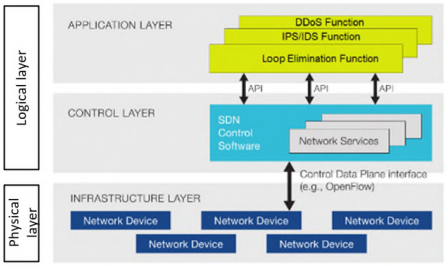

Figure 1. SDN Layers and Security function

 ISBN: 978-1-61208-729-0

The Infrastructure layer is responsible for data transferring. SDN Virtual Switches (*Vswitch*) are used in the SDN environment. *Vswitches* are supporting protocols such as NetFlow, sFlow and OpenFlow. They are utilised to connect the Infrastruture layer with the upper Control Layer via Control Data Plane Interface, as shown in Figure 1. The Control Layer is the middle layer, where network services are located. This layer maintains a global view of the network and provides hardware abstractions to SDN applications. The implementation of the SDN control plane can follow a centralized, hierarchical, or decentralized design [10]. This is a logical entity that receives instructions or requirements from the SDN Application layer and relays them to the Infrastructure layer. There are a lot of different options for applications on this level, such as: Intrustion Prevention System (IPS), Distributed Denial of Service functions protecting SYN Flood attaks, ping of death, amplification attack, and many more. Other services can be added as well, such as networking management, analytics, or business applications. SDN gives the flexibility to every programmer and administrator to choose the best path for their services by configuring the SDN controller with a custom policy. This way, they are able to prioritize, deprioritize, or even block specific types of packets with a very granular level of control [12]. Also, it can be used in a cloud computing, multi-tenant architecture, because it gives the traffic the possibility to be managed in more efficient and flexible manner [13]. Using this model, the owner of the data has the control of the data. Several research ideas based on SDN/OpenFlow have been proposed in [14], since the publication of DDoS protection for SDN [15], which is a key component in realizing the SDN concept for decentralized protection. We will focus on the security protection realized by VSF. Open Virtual Network (OVN) [3] is an example of an open source product using OpenFlow. According to OVN, some community main challenges are: Congested network link between physical switches; Poorly performing virtual switches; Overloaded servers and Distributed Denial of Service (DDoS) attacks. OVN are using their proprietary model [3] for congestion elimination. The difference is that we are using a spanning tree and *Mininet* environment for avoiding loops. According to the research in [32], STP is used for threat analyses of network. In our research, we are focusing more on the aspect of availability assured after the elimination of loops.

B. OpenFlow as a technology

OpenFlow is a communications protocol that gives access to the forwarding plane of a switch or router over the network. This is the protocol used to connect the Infrastructure layer and the Control layer. With OpenFlow, packet-moving decisions are centralized. Having the control separated from the forwarding allows for a more sophisticated management. This type of protocol allows a more effective use of the network resources compared to the traditional networks [16]. OpenFlow is layered on top of the Transmission Control Protocol (TCP). The Datapath of an OpenFlow switch is determined via Flow Tables. A Flow Table matches incoming packets to a particular flow and specifies the functions that are to be performed on the packets. Flow Tables entrees can be divided in three sections: rules, actions, and states. Packets are assigned actions based on the rules that are matched to the Flow Table [17]. Those actions can be: to forward the packet, to send it to the controller, to send it to a normal processing pipeline, or to even perform NAT by changing the source IP address. The controller will manage all communications and will have visibility on all types of protocols. On the lower level, the controller has the knowledge of switch ports, source and destination IPs, of Media Access Control (MAC) addresses and of Layer 4 ports.

C. SDN security functions

SDN is a new paradigm and technology that is currently widely developed. After the development of OpenFlow, the separation of data plane and control plane is now possible, although at the same time there is still an issue with the security. This area needs to be developed further in order to ensure a proper level of protection. OpenFlow provides critical information for any connection. It can be pointed out that, by using a Flow Table, every connection can be analyzed by the controller and, based on this analysis, traffic can be allowed or rejected. In order to ensure better security, a lot of security functions are under development currently. There are theoretical solutions for firewalls, IPS/IDS, stateful firewalls; however, currently, there is no solid solution for switch fails redundancy. [2] discusses that, when a switch fails, its immediate downstream switch(es) will take time to recalculate paths and to assure availability. A downstream switch is protected if it has a neighbor whose path to the controller is not affected by this failure. By rerouting its traffic to this neighbor, the protected switch will bypass the failure. Nowadays, enhancing the security of SDN, improving autonomy and ensuring privacy of the data can be improved significantly. This could be achieved by creating virtual security functions with a different scope. In order to implement additional level of security in SDN, it has to be applied on the Application layer. To ensure a sufficient level of security according to ISO 27001[20] in a traditional environment, different hardware devices need to be used. With SDN, additional functions can be developed, such as Congestion elimination, Denial of Service (DoS), Distributed Denial of Service (DDoS), Intrusion Prevention System (IPS), Intrusion Detection System (IDS), Slower attack on Level 7, or layer 2 attacks using MAC address spoofing techniques. The main functions under development are firewalls [21]. They are using OpenTables in order to assure Datapath and to provide static control on incoming and outgoing traffic. After the development, Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) [22] have been added as functionalities on SDN firewall. Snort IDS [23] is an example of already developed IDS with predefined signatures. Applying/Using the SDN model allows those virtual security functions to operate on the Application layer and VSF are deployed over a virtualized environment.

 ISBN: 978-1-61208-729-0

IV. PROPOSAL

The area we are working on now will be situated at the Application layer. Security functions for each type of attacks can be implemented at this layer. Problems such as compatibility are eliminated because the entire environment is virtualized. Virtualized Network Functions (VNF) is an emerging paradigm that builds on cloud computing and the virtualization technology to eliminate the drawbacks of traditional physical network infrastructure. Different types of SDN applications can be created as a security function. Using Network Functions Virtualization Infrastructure (NFVI) allows all hardware elements to be located on a physical layer and all services to be deployed on a virtualized environment, as shown on Figure 2.

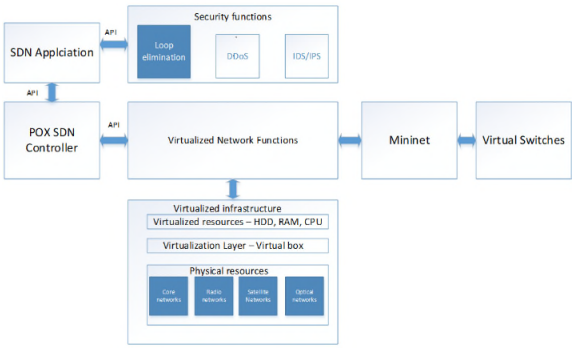

Figure 2. Virtual framwork with SDN properties

VNF is used for building blocks. They are performing the same services as the traditional firewalls or IDS/IPS, but on a virtualized environment. Loop elimination is the function that we focus on in our research.

V. SDN ENVIRONMENT

Our development environment is entirely virtual, as we are using a Virtual box as hypervisor. Oracle VM VirtualBox is a free and open-source hypervisor for x64bit computers currently being developed by the Oracle Corporation [25]. We have installed on this hypervisor one Linux with Ubuntu 14.04.4 and a customized version on Linux for Mininet 2.2.2. We used the Ubuntu Virtual machine to install the POX Controller [26] and we used Mininet to emulate the SDN environment. In this environment, one SDN switch and two emulated hosts are utilized [27]. In order to be connected to the host environment, we are using Putty [28], a terminal emulator, and Xming [29] X11 display server.

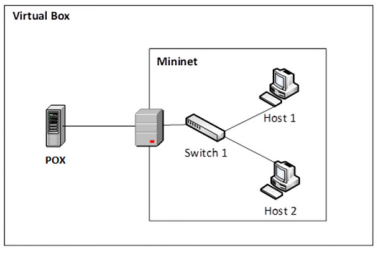

Figure 3. Virtual framwork with SDN properties

We are creating different virtual security functions for mitigating several types of attacks by using a virtualized environment. Examples of security functions are Loop Elimination, DDoS and IDS/IPS, as shown on Figure 3. In this paper, we are concentrating mainly on the Loop Elimination security function by using spanning tree.

VI. EXPERIMENTAL RESULTS

A. *Delay analyses in OpenFlow infrastructure topology*

By using the Mininet environment, different types of topologies have been created. We can use those types of graphs in order to describe our framework`s environment and to test our concepts and models. The presented framework creates different types of topologies and ensures communications between all hosts. One example of topology is shown in Figure 4.

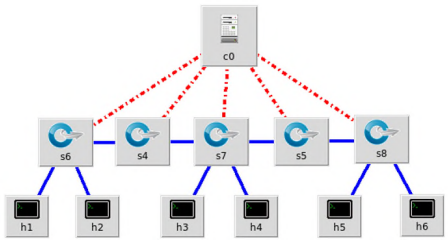

Figure 4. OpenFlow infrastructure topology

During this deployment, we used POX as controller and we were able to orchestrate the entire communication. Based on the virtualized environment, we established connections between each host orchestrated by an SDN controller and by using OpenFlow as a communication protocol, console output, as shown in Figure 5.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h4 h5 h6 h2 h3
h4 -> h1 h5 h6 h2 h3
h5 -> h1 h4 h6 h2 h3
h6 -> h1 h4 h5 h2 h3
h2 -> h1 h4 h5 h6 h3
h3 -> h1 h4 h5 h6 h2
*** Results: 0% dropped (30/30 received)
```

Figure 5. Host reachability by using Mininet and OpenFlow

We use ICMP as a protocol for demonstration purposes. Based on the results, we were able to analyse and to calculate different communication delays. It can be easily seen that the first packet needs more time to arrive at the destination than the following packets and we have zero percent packet lost. This is represented in Table I.

TABLE I. PING DISTRIBUTION

	Table with times in milliseconds					
	h1	***h2***	***h3***	***h4***	***h5***	***h6***
h1 first ping	x	17.30	24.10	37.81	76.30	13.70
h1 second ping	x	0.36	0.78	0.63	1.00	2.58
h2 first ping	14.50	x	11.00	47.50	70.60	61.41
h2 second ping	0.46	x	0.63	0.65	0.98	1.35

 ISBN: 978-1-61208-729-0

h3 first ping	55.10	34.00	x	27.10	52.30	16.87
h3 second ping	0.64	0.87	x	0.36	0.73	0.64
h4 first ping	39.10	18.45	10.45	x	42.40	8.56
h4 second ping	0.69	0.88	0.35	x	0.83	0.75
h5 first ping	11.10	13.01	22.07	7.01	x	14.53
h5 second ping	1.03	0.97	0.74	0.64	x	0.36
h6 first ping	27.10	17.50	29.20	27.61	32.10	x
h6 second ping	0.90	1.20	0.64	0.66	0.36	x

Table I presents and analyses different ping time values during the communication between hosts. The ping time is the reaction time during connections between all hosts. The values represent how much time it takes to get a response after a request has been sent. Ping is measured in milliseconds (ms). In our case, the first packet always takes more time to be delivered then the second. After that, the communication stays stable and ping time becomes smaller. The first packet takes so much time because the controller must take the decision how to treat this communication. Comparing to the traditional model, in SDN environment the controller needs to recalculate each path and to put it in the tree. Once it has passed through the initialization phase, it has the knowledge how to reach the destination. After the first ping, information is available in the Flow Table, such as source IP addresses and MAC addresses. Looking at the results obtained from the testing scenario, we can conclude that delays can be caused also by the distance and the number of hops to reach the destination. Ping time increases due to the number of hops needed for packets to be delivered. Figure 6 shows a graphical representation of the results from Table I. The ping times are located on the ordinate axis in milliseconds. All sets of pings are presented on abscissa axis with different colors.

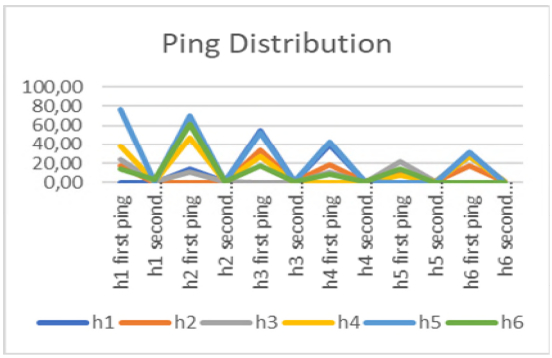

Figure 6. Ping distribution

A major conclusion for this topology, based on all tests that we have performed, is that the first ping takes more time than the following pings, and the delay depends on the number of hops to the destination. As a result, it could be concluded that this type of topology is not scalable due to the increased ping time. This reflects directly on the communication between hosts.

B. *Loop Elimination security function*

We are presenting in this paper the model using a loop free environment by applying Loop Elimination security function based on spanning tree. The spanning tree function is used as preconfigured virtual security function by the POX controller [10]. With the VSF, our system allows to build a loop free environment. Based on this approach, we are increasing the security by assuring availability and eliminating the possibility of loops in the network. Spanning tree will be used for preventing Storm attacks [4]. It is worth noting that data storm is excessive transmission of broadcast traffic in a network. By creating the spanning tree, we are blocking the port that can reduplicate the traffic and we are creating a safe route to each destination. For our model, we will use the topology in Figure 7. i.e., POX controller (c0), four switches (sN) with redundant links and four hosts (hN) for testing purposes. We are applying ICMP as a protocol for testing purposes, but the same results can be achieved with TCP internet-based protocol. Hosts are representing servers, as shown in the figure.

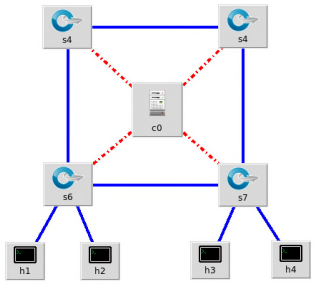

Figure 7. OpenFlow infrastructure

On this topology, we observe that the initial communication will take more time to select the best path, as it needs to eliminate the loops in the topology. Comparing the time in Section A (Table I) to the time using spanning tree, there is significant delay of the delivery. This delay comes from the need to recalculate the path by the Security Function. On the other hand, if a standard exchange protocol is used on the same topology, without a spanning tree, the virtual function will create a loop and hosts will not able to communicate with each other. In the presented console output in Figure 8, it is visible that none of the hosts are reachable.

```
mininet> pingall
*** Ping: testing ping reachability
h2 -> X X X
h3 -> X X X
h4 -> X X X
h1 -> X X X
*** Results: 100% dropped (0/12 received)
```

Figure 8. Host reachability problems

Consequently, the communication system will have a deadlock and all communications will be blocked; also, the packets will be dropped. On the controller (c0), we can register a huge amount of traffic, which overloads the

controller, as presented in Figure 9. The same packet is transmitted many times on different ports and creates a loop.

```
[openflow.of_01          ] [00-00-00-00-00-03 4] connected
[forwarding.l2_learning ] Same port for packet from 92:1a:dc:16:d6:3d
on 00-00-00-00-00-01.2.  Drop.
[forwarding.l2_learning ] Same port for packet from 92:1a:dc:16:d6:3d
on 00-00-00-00-00-04.1.  Drop.
```

Figure 9. Controller deadlock

In order to avoid loops that are blocking the traffic, we enable the Loop Elimination function. In Figure 10, we observe that by applying the security function with spanning tree, hosts can communicate with each other and avoid loops. We can see that the communications between hosts are stable and there are no dropped packets.

```
mininet> pingall
*** Ping: testing ping reachability
h2 -> h3 h4 h1
h3 -> h2 h4 h1
h4 -> h2 h3 h1
h1 -> h2 h3 h4
*** Results: 0% dropped (12/12 received)
```

Figure 10. Host reachability

The spanning tree is realized by algorithm for link detection. The controller constructs a spanning tree and then disables flooding on switch ports that are not part of the tree. This is a different process to classic spanning tree with root bridge election, presented below on Figure 11.

```
[openflow.discovery     ] link detected: 00-00-00-00-00-04.2
[openflow.spanning_tree ] Spanning tree updated
[openflow.spanning_tree ] 7 ports changed
[openflow.spanning_tree ] Spanning tree updated
```

Figure 11. Spanning tree update

Based on the Loop Elimination security function, each host is creating a table with addresses for communication between all reachable hosts. The Address Resolution Protocol (ARP) table, shown in Figure 12, presents results and provides information for MAC and IP addresses. This information can be obtained from a conventional infrastructure. Based on ARP, it is possible to discover the link layer address, such as a MAC address, which is associated with a given network layer address. This will help us to identify proper communication between hosts. For communications, the controller needs information for the hardware (MAC) address and the IP address.

```
mininet> h1 arp
Address                  HWtype  HWaddress
10.0.0.4                 ether   26:d9:4e:e9:e0:43
10.0.0.2                 ether   b2:0c:bb:90:2a:19
10.0.0.3                 ether   8a:e1:27:94:12:b6
```

Figure 12. Arp table

For clarification of our results, we present an output from the POX controller. The console output represents a Flow Table as shown in Figure 13. Using the *tcpdump* command, it is possible to observe useful information about the transferred packets. The description of the contents of packets on a network interface is defined with the *tcpdump* command.

```
mininet> dpctl dump-flows
*** s3 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=311258.123s, table=0, n_packets=0, n_bytes=0, idle_age=655
34, hard_age=65534, priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLL
ER:65535
 cookie=0x0, duration=311258.149s, table=0, n_packets=101571, n_bytes=4164411, i
dle_age=0, hard_age=65534, priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88c
c actions=CONTROLLER:65535
```

Figure 13. OpenFlow table

During our experiments, an OpenFlow table in Figure 13 has been built and we observed that the entire communication passes over a virtualized environment. There is a link between an OpenFlow table and a Loop security function for selecting the best path. In case of a faulty link or a loop, a new path will be selected by the Loop elimination function. Looking in to the stream from the controller, by using OpenFlow as a protocol, we can analyze different data from the Flow Table, as shown in Figure 13. Based on this output analyses, we can obtain the set of MAC addresses, IP addresses, protocols and Ports. Currently, we are testing with ICMP packet, but we could also utilize TCP or UDP. By detecting a loop, we can also add additional levels of security by assuring availability for all hosts in the segment. However, observing not only IP addresses, but MAC addresses as well, allows for more flexibility. Looking at the example with the STP [2], a specific role for each port can be assigned for avoiding man-in-the middle attack, i.e., where the attacker will try to present himself as a root bridge. In this case, the port that can create a loop is blocked and there are no possibilities for deadlock. This attack targets the actual data that flows between endpoints, compromising the confidentiality and the integrity of the data. For instance, it gets banking credentials or other sensitive information. Based on the information that we can get from the flow table, we can create different security functions and apply them directly on the controller. OpenFlow provides flexibility using only virtualized environment to orchestrate the entire traffic without need of adding other devices. IDS, IPS, stateful firewall, Web proxy can be replaced by virtual security functions that are deployed at the Application layer of one controller, as presented in the framework diagram (see Figure 2).

VII. CONCLUSION AND FUTURE WORK

Using this framework, we have proved that SDN can be used on virtual environments. This will give us the possibility to apply the solution in complex environments. By creating spanning tree functions, we have assured availability on our network segment and storm attack has been eliminated. During our experimental results, we tried implementation with different topologies. During those tests, we had many failures on connections creating dead locks and scrutinizing the system. By applying this framework, the following security problems can be addressed: DDoS protection; geolocation protection and ping of death protection. They are examples that can be used as Virtual Security Function(s). All these functions can be created as future work, based on the proposed framework. Each security function will have its own assignment; therefore, many functions can be run in parallel in order to assure better

 ISBN: 978-1-61208-729-0

throughput. The utilization of VSF gives the possibility to every programmer to assign the needed level of security. This will allow the user to avoid using different high-level vendor solutions. All those functionalities will be taken by the controller (c0 in Figure 7). In addition, different types of controllers can be used for performance analyses, such as Floodlight or OpenDaylight. Web server and other type of protocol can be added in the research. As future work, the proposed framework based on SPT security function and SDN will allow us to solve very challenging problems on current the network, such as botnet detection and several types of DDoS identification. It allows to find a solution for avoiding complex attacks such as Mirai botnet and Github attack.

As a conclusion, in the paper, we identified that one of the current problems in SDN is availability. Based on the proposed framework, using POX controller, we are controlling the dataflow by creating different security functions. The Loop Elimination security function is using a spanning tree as protocol and is creating single security function for availability. This framework can not only work in real-time, it is also capable of eliminating loops. This is achievable based on the controller possibility to orchestrate the entire network segment. Performance analysis has been performed on different topologies. The main goal of this framework was to eliminate loops as a possible way of protecting from network attacks. In the future, different security functions can be created, such as DDoS elimination, IPS and many others.

ACKNOWLEDGMENT

The authors acknowledge support from the project UNITe BG05M2OP001-1.001-0004/28.02.2018 (2018-2023).

REFERENCES

[1] N. El Moussaid, A. Toumanari and M. El Azhari, “Security analysis as software-defined security for SDN environment”, 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, Spain, 8-11 May 2017

[2] Z. Yang and K. Yeung – “ILP formulation for controller tree design in SDN”, 2017 IEEE 18th International Conference on High Performance Switching and Routing

[3] Open Virtual Network (OVN) – Sflow - Telemetry, analytics, and control with sFlow standard - September 21, 2015

[4] V. Reddy and D. Sreenivasulu – “Software Defined Networking with DDoS Attacks in Cloud Computing”, International Journal of Innovative Technologies.

[5] R.S. Braga, E. Mota end A. Passito „Lightweight DDoS flooding attack detection using NOX/OpenFlow“, Proceedings of the 35th Annual IEEE Conference on Local Computer Networks, in: LCN, 2010.

[6] M. Canini, D. Venzano, P. Peresíni, D. Kostic and J. Rexford, “ANICE way to test OpenFlow applications”, USENIX Symposium on Networked Systems Design and Implementation, 2012

[7] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P.Sharma, S. Banerjee and N. McKeown “Elastictree: Saving energy in data centernet works”, Proceedings of the 7th USENIX Symposium on Networked Systems Designand Implementation, NSDI, 2010.

[8] A. Nayak, A. Reimers, N. Feamster and R. Clark, “Resonance: dynamic access control for enterprise networks”, Proceeding sof WREN, 2009.

[9] T. Muciaccia and V. Passaro, “Performance characterization of a novel DWDM all-optical SDN-like metro-access network” , 2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)

[10] SDN Architecture Overview - Stanford OpenFlow Deployment https://openflow.stanford.edu/display/ONL.html (accessed July, 2019)

[11] R. Miller, “The OSI Model: An Overview”, SANS Institute InfoSec Reading Room.

[12] R. Sherwood, G. Gibb, K.K. Yap and G. Appenzeller, “Can the production network be the test bed”, Proceeding sof USENIX Operating System Designand Implementation, OSDI, 2010.

[13] S. Shin and G. Gu, “Network security monitoring using OpenFlow in dynamic cloud networks”, 7th Work-shopon Secure Network Protocols (NPSec’12), collocated with IEEE ICNP’12, 2012.

[14] S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu and M. Tyson, “Modular composable security services for software defined networks”, in: 20th Annual Network and Distributed System Security Symposium (NDSS’13), 2013.

[15] C. Yoona, T. Parka, S. Leea, H. Kanga, S. Shina and Z. Zhang – “Enabling security functions with SDN”, A feasibility study - ELSEVIER

[16] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J.Rexford, S.Shenker and J.Turner, “Enabling innovation in campus networks”, SIGCOMM Comput. Commun. Rev. 38(2008)69–74.

[17] A. da Silva, C. Machado, R. Bisol, L. Granville, A. Schaeffer-Filho “Identification and Selection of Flow Features for Accurate Traffic Classification”, 2015 IEEE 14th International Symposium on Network Computing and Applications

[18] D. Hyun, J. Kim, D. Hong and J. Jeong, “SDN-based network security functions for effective DDoS attack mitigation” - Information and Communication Technology Convergence (ICTC) 2017

[19] N. Bawany, J. A. Shamsi, “DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions” - Computer engineering and computer science

[20] ISO 27001- https://www.iso.org/standard/73906.html (accessed July, 2019)

[21] J. Deng et al., “An NFV/SDN Combination Framework for Provisioning and Managing Virtual Firewalls.”, IEEE Conference on NFV and SDN, 2015.

[22] C. Munecas and A. Tirados, “Intrusion Detection Systems In Sdn-Based Self-Healing Pmu Networks”

[23] Snort IDS https://github.com/pratiklotia/SDN-Intrusion-Prevention-System-Honeypot

[24] S. Mustafiz, F. Palma, M. Toeroe and F. Khendek, “A Network Service Design and Deployment Process for NFV Systems” - 2016 IEEE 15th Symposium on Network Computing and Applications

[25] Virtual box installation https://www.virtualbox.org/wiki/Downloads (accessed July, 2019)

[26] S. Kaur, J. Singh and N. S. Ghumman, “Network Programmability Using POX Controller”, Computer Networks 147, October 2018

[27] K. Kaur, J. Singh and N. Ghumman, “Mininet as Software Defined Networking Testing Platform” – 2014

[28] PuTTY, [Online] www.putty.org (accessed July, 2019)

[29] Xming, [Online] https://sourceforge.net/projects/xming/ (accessed July, 2019)

[30] K. Atwal, A. Guleria and M. Bassiouni, “A Scalable Peer-to-Peer Control Plane Architecture for Software Defined Networks” - 2016 IEEE 15th Symposium on Network Computing and Applications

[31] S. Seeber, L. Stiemert, G. Rodosek, “Towards an SDN-enabled IDS environment” 2015 IEEE Conference on Communications and Network Security (CNS)

[32] A. Rehman, F. Siddiqui, J. Khan and M. Saeed, “Spanning Tree Protocol for Preventing Loops and Saving Energy in Software Defined Networks Along with Its Vulnerability and Threat Analyses.”, Advances in Intelligent Systems and Computing, vol 857.

 ISBN: 978-1-61208-729-0