Indonesian Journal of Electrical Engineering and Computer Science
Vol. 17, No. 3, March 2020, pp. 1157~1164
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v17.i3.pp1157-1164

Evaluation of firewall and load balance in fat-tree topology based on floodlight controller

Sarah H. Mohammed, Ammar D. Jasim
Department of of Information and communication Engineering, Al-Nahrain University, Iraq

Article Info

Article history:

Received Jul 22, 2019
Revised Sep 24, 2019
Accepted Oct 8, 2019

Keywords:

Firewall
Floodlight
Loadbalancer
Mininet
Openflow
Software defined networks

ABSTRACT

Today it has become important to reconfigure the networks in to new form to be more manageable, scalable, dynamic and programmable. The networks recently are so inflexible and failing to deal with the required changes for the Information Technology. Software Defined Networking (SDN) is a modern paradigm that focused to change the main idea of current network infrastructure (traditional network) by breaking the chain between the data forwarding and the control planes to introduce flexible programmability network. This paper makes comparison between the performance of traditional fat-tree network and SDN fat-tree network, which found that average Round Tripe Time (RTT) in SDN fat-tree topology will decrease by 8.96% than traditional fat-tree topology. Then shows the basic operation of OpenFlow protocol that can be applied on fat-tree topology by using SDN technology and how that can be effect on the performance of network and make it more flexible to enable the SDN module applications, like load balancer and firewall for optimizing the SDN network. In this paper the physical switches are replaced by software switches in a virtual network environment and display the SDN structure in GUI, also Floodlight controller is chosen to use as the network operating system for SDN network.

Corresponding Author:

Sarah Hashim Mohammed,
Department of of Information and communication Engineering,
Al-Nahrain University, Al-Jadria, Baghdad, Iraq.
Email: sara.hashem@coie-nahrain.edu.iq

1. INTRODUCTION

One of the most considerable new trends in the expansion of network technology, it's network segregate between the control and data forwarding planes, that transmit the control plane from the forwarding equipment such as switches and routers into software controller that run on a specific server, along with network applications that supported by this controller. The controller can manage this forwarding device over setup secure link with OpenFlow protocol [1].

Today the networks must be more intelligent, scalability and programmability. These reasons explicate the benefit of software defined networking (SDN) [2].

SDN technology suggested that dominance software perform through device called the SDN controller, which has a whole vision of the network structure and has the liability of making decisions. Furthermore, the controller instructions are executing by the network equipment's such as switches or routers, that are only responsible for forwarding data according to these instructions [3]. The SDN traffics can be configured and modeled depends on the requirements and the immediate needs of the network or the enabled applications [4].

The modern network architecture is SDN based network that is suggested in the last decade. In the architecture of traditional network, the network equipment, like routers and switches has both the control plane and the data forwarding plane. In SDN based networks, the essential thought is to transfer the control plane to an application layer and let the data plane in the network equipment. The network equipment is become

straightforward to configure and programed by the application layer in order to forwarding the packets according to the instructions of controller [5]. The SDN controller responsible to redirect the flow of traffic according to traffic request and depending on available paths of the network that are available for scheduling in each base station by using an OpenFlow protocol [6].

Tody the Floodlight controller is very popular for researches and academic aims [7]. Java language was used to write the Floodlight controller and it has an Apache license [1], which mean that the user has a free license and can use the software for any purposes without conditions [8]. until now the last version of Floodlight is v1.2. The Floodlight can back both OpenFlow switches the virtual and physical, and it is can deal with mixed of OpenFlow and non-OpenFlow switches [1]. The following related works focus generally on OpenFlow configuration that can be implemented using Mininet emulator tools.

An emulator tool is used to create network topology and measure numerical results that describe performance of particular network elements and functions [9]. In 2015 Kaur and Kumar [10], Singh and Ghumman designed OpenFlow and developed it over SDN POX controller-based firewall application. In this study the traffic performance can be calculated depending on the throughput and latency for the network in both cases with and without firewall. In the result they found that throughput will be decreased while the latency will be increased due to the overheads provided by the firewall.

In 2016 Atiyah [11] focused on how to exploit the SDN centralization, openness and programmability features to enable implementing an Content-Centric Networking (CCN) prototype, by using VirtualBox and Mininet by designed CCN-Custom-SDN, CCN-Simple-Traditional Network, CCN-FatTree-SDN, and CCN-FatTree-Traditional Network, which found efficient content delivery with transmission delay lower than in traditional network about 42%, also prove SDN feasibility to implement CCN with higher throughput 21% and acceptable packet loss and jitter in comparison with traditional networks.

In 2017 Jalil [12] implemented fat-tree datacenter over traditional network and SDN then make performance comparisons between them, like throughput and delay. This network design and application was emulated using VirtualBox and Mininet. Also, they used POX controller as operating system in SDN section that used to control the network. Then they implemented layer-2 firewall application over SDN scenario and measured the network performance. In the results, the delay is decreased by 19% and throughput is increased by 9% in SDN scenario compared with traditional network.

In 2018 Ali, Morad, and Abdala [13] implemented load balancing function over fat-tree SDN Data Center Network (DCN), in order to improve network performance parameters and distributes the traffic request over a variable server. This work was performed using OpenDayLight controller as a network operating system, which is responsible to manage and configure OpenFlow switch as layer-2 switch in order to forward the network traffic through network topology.

In 2018 Wadhwani, Bajaj and Rohra [14] displayed the main idea of SDN technology and how that can solve many of the traditional network problems. Then the firewall function is enabled over Tree network topology by using the OpenDayLight controller as a network operating system. In this paper present the main idea of SDN and its effect on the network performance by implement and configure the firewall and loadbalance applications using Floodlight controller.

2. OPENFLOW PROTOCOL

OpenFlow technology a standard protocol, it is originated by the Clean Slate program at Stanford University/United States, the SDN core notion is OpenFlow technology. The interaction interface (southbound interface) between the controller plane and data plane was organized by protocol known as OpenFlow protocol [15].

The OpenFlow protocol is responsible to manage and control the communication between the data and the control planes [16]. OpenFlow is an open source protocol that program flow tables that reside in switch/router by needed instructions [17].

The OpenFlow-enabled switch includes the following three components. First is a flow table, which programmed by the SDN controller to directing the switches in order to process each packet. Second is a secure channel, which is used to provide the connection between the controller devices that was decoupled, and third part is used as a secure channel is Transport Layer Security protocol (TLS) [3]. OpenFlow is show the main idea of flow and a flow table [15]. Openflow protocol is a standard protocol that allows different vendor devices like cisco, juniper and huawei switches to be connected to the same controller [18]. Figure 1 displays the basic operation of OpenFlow protocol, in which defining new entry flow in the flow table and how to perform associated action.

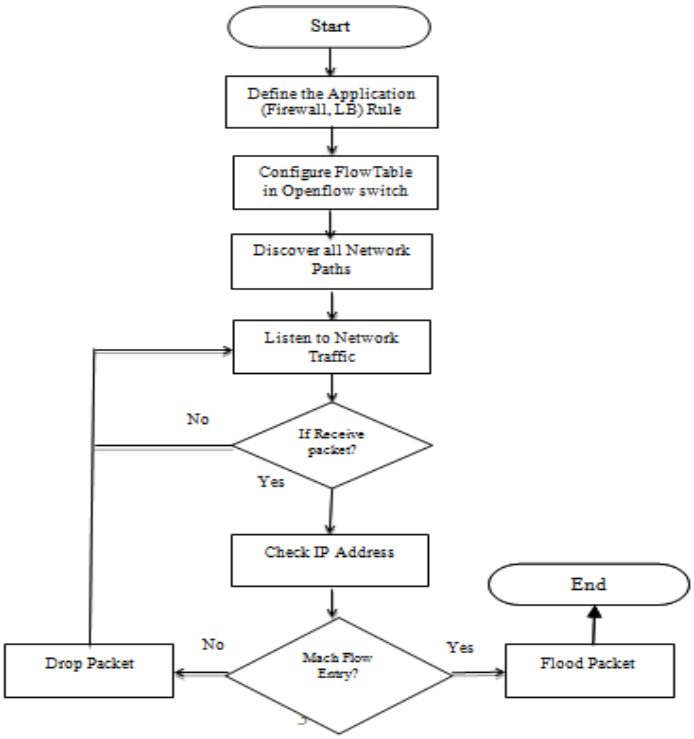

Figure 1. Flowchart for OpenFlow process

3. FAT-TREE TOPOLOGY

A Data Center Network (DCN) is making the servers and hosts easy to use which are connected by using dedicated links and switches. Current DCN can include a large number of hosts with specific delay and bandwidth requirements. There is a rising interest in academia and industry of today to integrate SDN technology with large-scale DCN. Recently, we realize that most works used specific topologies like Tree and fat-tree network topologies. It is comparatively simple to apply routing algorithms on these type of topologies [19].

Fat-Tree topology can be implemented by using multiple cheap links to balance load of network and work as back-up link when another one was failed [20]. Fat-Tree is considering one of the most appropriate topologies of DCNs for future [19].

4. DESIGN AND IIMPLEMENTATION

In this part, a fat-tree topology will be implemented using Mininet Python script and used Floodlight controller as OS of SDN network, then the SDN network structure will be display by Floodlight GUI. Fat-tree is not a static network topology, it can extend according to the value of k, in which the switches k-port in the edge tier is connected to $\frac{k}{2}$ servers form one side, and the others ports of switches are connected to the $\frac{k}{2}$ switches that allocated at the aggregation level [20]. Figure 2 and 3 are displayed the fat-tree topology which is implemented with k = 4 for both Traditional and SDN technology.

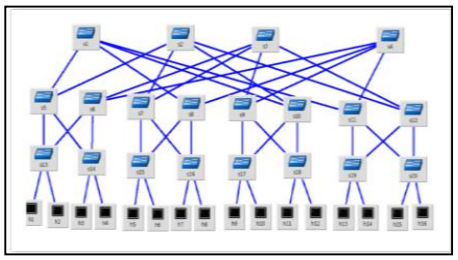

Figure 2. Traditional fat-tree topology

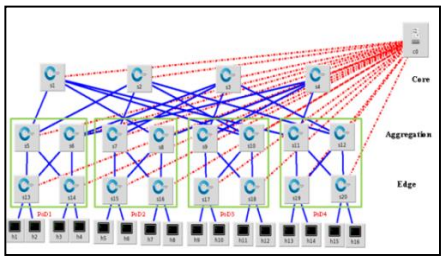

Figure 3. SDN fat-tree topology

The *ping* tool is use to checking the network connectivity and to measuring Round Trip Time (RTT) between specific network hosts. Also, the network connectivity test is can be performing by using *pingall* tool as shown in Figure 4.

```
*** Ping: testing ping reachability
h13 -> h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h16 -> h13 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h8 -> h13 h16 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h14 -> h13 h16 h8 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h9 -> h13 h16 h8 h14 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h7 -> h13 h16 h8 h14 h9 h5 h10 h4 h12 h11 h3 h1 h15 h6 h2
h5 -> h13 h16 h8 h14 h9 h7 h10 h4 h12 h11 h3 h1 h15 h6 h2
h10 -> h13 h16 h8 h14 h9 h7 h5 h4 h12 h11 h3 h1 h15 h6 h2
h4 -> h13 h16 h8 h14 h9 h7 h5 h10 h12 h11 h3 h1 h15 h6 h2
h12 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h11 h3 h1 h15 h6 h2
h11 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h3 h1 h15 h6 h2
h3 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h1 h15 h6 h2
h1 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h15 h6 h2
h15 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h6 h2
h6 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h2
h2 -> h13 h16 h8 h14 h9 h7 h5 h10 h4 h12 h11 h3 h1 h15 h6
*** Results: 0% dropped (240/240 received)
```

Figure 4. Check the connectivity of network

Two different numbers of packets, 500 and 1000 packets had been used to check the performance of the traditional and SDN fat-tree networks. The ping command was selected to executing on the hosts (h4, h6, h9, h13) with the IP address of destination (10.0.0.10), which is the IP address of the (h1). In the terminal of these hosts the following command is executed and the results show in Figures 5 and 6 for both traditional and SDN fat-tree topology:

```
# ping -c 500 10.0.0.1
# ping -c 1000 10.0.0.1
```

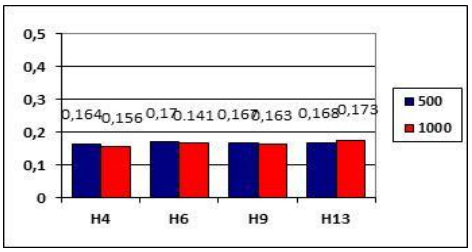

Figure 5. The measured RRT for 500and 1000 packets based Traditional fat-tree topology

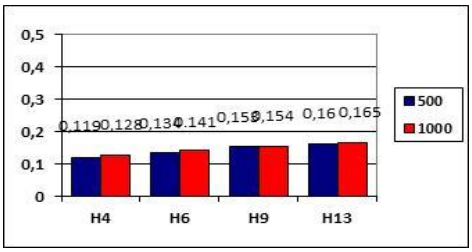

Figure 6. The measured RRT for 500and 1000 packets based SDN fat-tree topology

The above two graphs show that delay of SDN based fat-tree network topology will be increased simultaneously with number of transmitted packets over the network and it had lower values compared with the traditional fat-tree network topology, so the network performance improved by using SDN technology. The Floodlight controller provides ability to display the network topology through GUI as shown in Figure 7.

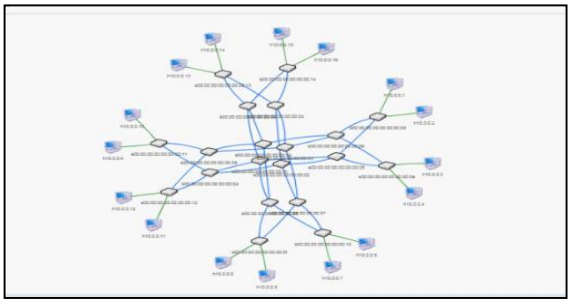

Figure 7. Network topology as shown in floodlight GUI

4.1. Firewall Implementation

A firewall can be defined as a network security system that detects and dominates to the all traffics on the network based on required rules [21]. A firewall is protecting the network from unauthorized access, the main process of the firewall is to check each in/out packet passing through the network then decides whether to accept/deny the packet depending firewall defined rules [22].

The main idea to enable the firewall on the SDN network is to provide a central security that run above the control plane. The administrators are permitted by SDN to determine dynamic policies in order to lead to a cost-effective balance between user's suitability and network protection. For this reason, the cost of security operation in the SDN network will reduces compared to the traditional network [23].

An SDN idea is to make the flexibility is possible to control and dominates network devices. It enables to apply firewall rules not only for devices in the edge tier but also on the whole switching in the network. Next applying firewall rules to the network devices which produces an unauthorized traffic that has a positive effect on whole performance of network. In this case the number of useless packets will be reducing from traveling over the network [21].

Firewall is enabled within the SDN fat-tree topology without adding any rules to prevent all packets from passed thought network, as show in Figure 8. Then adding firewall rules to allow the packets from passing thought selective hosts, according to the adding rules H3 and H7 allow the packets to passing thought it as show in Figure 9. The following commends will activating the firewall in all host of network in which all packets will be dropped, and then enable firewall rules.

The following commends are used to configure firewall rules according the network requirements:

Curl -XPOST-d'{'sw-mac': '00: 00: 00: 00: 00: 00:0a'}http://<Contoller-Port-No>/wm/firewall-rules /json

Curl –XPOST-d{'src-host-ip':'10.0.0.03/32','dst host-ip':'10.0.0.07/32'.'dlltype':'ARP/ICMP'}' http://<Contoller-Port-No>:8080/wm/firewall-app/ add-rules/json

Curl -X POST -d '{'src-ip': '10.0.0.07/32','dst-ip': '10.0.0.xx/32','dl-type':'dlltype':'ARP/ICMP'}'http://< Contoller-Port-No>/wm/firewall-app/add-rules/json

Figure 8. Enable firewall in all hosts

Figure 9. Enable firewall with adding rules

4.2. Load Balancer Implementation

Load balancing is one of significant application that had been developed and evaluated by SDN community. Load Balancing (LB) basic idea is process workloads and distributed it between servers by distributing the traffic across multiple paths in order to optimize the efficiency of the network [24]. LB is important for many reasons, first the availability, it has a considerable impact, in case if a single server fails down, the LB will be responsible to redirect the network traffic to any other available servers and to maintain the performance of network as required. According to that the website won't go down even if one server fails because it has a good load balancing method [25]. The LB is enabled through some commends as show next and according to this commend H1 and H2 are considers as servers and all other hosts as clients.

Curl –XPOSTd"{'id':'1','name':'vip1','protocol':'icmp ','ip-address':'10.0.0.100','port-no':"80"} 'http://<Contoller-Port-No>/quantum/v1.0/vips/

Curl -XPOST-d{'id':'1','name':'pool1','protocol': 'icmp','vip-id':'1'}'http://<Contoller-PortNo>/quantum /v1.0/pools/

Curl -XPOST-d{'id':'1','ip-address':'server1-ip-address','port-no':'80','pool-id':'1'}' http://<Contoller-Port-No>/quantum/v1.0/members/

In this section h1 and h2 are chosen to configure as server in order to balancing the traffics in the network. Bandwidth measurements were performed between the two hosts (h1 and h9) in which the effect of having load balancing is shown in following in Table 1.

Table 1. Bandwidth between h1 and h9

Without load balancing (Mbit/sec)	With load balancing (Mbit/sec)
94.4	93.7
94.6	95.3
95.5	96.9

The averages of the measured values are 94.8 Mbit/s for the fat-tree network without the load balancing module and 95.3 Mbit/s when applying the load balancing module. The Floodlight controller has excellent throughput in manage and control bandwidth by having a large number of modules that working together in order to perform specific action to improve the network efficiency. Also, it has ability to configure the network applications through the terminal or through its Web API.

5. RESULT AND DISCUSSION

The virtualization tool Mininet is used for the implementation and simulation of the expensive network, it's includes basic interfaces that used to create network topology. The most important thing in SDN based study is choosing the controller, and the Floodlight controller was chosen for this paper, because the Floodlight controller given the developers the ability to easily adjust software and develop applications. Also, it is included a representational state transfer application program interfaces (REST APIs) which provide simple a programming interface with the product.

In the SDN network the firewall rules can be added according to the needed of the users. And according to the added rules, in this work the packets will permitted to follow between h3 and h7 only and these rules can be modifying as needed. During the load balancer execution, it will identify the best path between hosts, and then pushed this path to switches. The load balancer was use to avoids the congestion could be happen between the hosts during the transition of packets, and according to Figure 10 it is obvious that apply SDN load balancer in Floodlight controller is efficient to manage the traffic in the network.

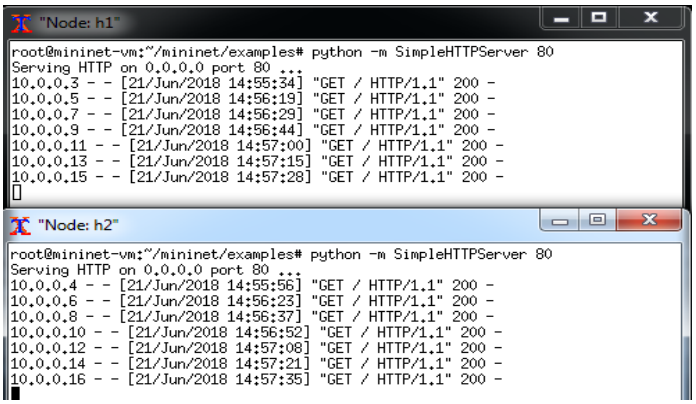

Figure 10. Enable load balancer

6. CONCLUSION

The main idea of this work is to show that SDN technology made the network configuration easier to control and manage through OpenFlow protocol and to improve network performance by enable central controller in order to configure the network on the contrary to traditional network. This central controller can enable the module applications in the network without configure each switch individually. The reason of chosen the fat-tree network topology, which it has complex structure and it possible to implementing this work in another network structure. At the end of work the Floodlight controller is so flexible controller, which is easy to apply and use to the real network, also can configure to achieve needed features.

The SDN technology makes it possible to enable many different applications, like firewall and loadbalance over datacenter without the need for special hardware device, but this application can configure and implement on the network controller through software code. Configure the firewall application in SDN which enable a centralized view of a network can functionally survey on overall network traffics to detect internal policy infringement. While configure the LB used SDN technology permits the administrator to write their own algorithms as network needed.

The average RTT in the SDN based fat-tree network topology is reduces about 9% with SDN than traditional topology for the same network. The main reason of this reduction in addition to SDN effect is that SDN Floodlight controller supports the multithread, which means that Floodlight controller can process more than one function at the same time.

REFERENCES

[1] S. Morzhov, I. Alekseev and Mikhail Nikitinskiy, "Firewall application for Floodlight SDN controller", *SIBCON, IEEE*, 2016.

[2] V. Moorthy, K. Sharma and R. Sachdeva, "Firewall and Load Balancer Implementation in Software Defined Networking", *International Journal of Pure and Applied Mathematics*, Vol. 115 No. 7, 2017.

[3] D. Kreutz, F. Ramos, P. Verissimo, C. Esteve, S. Azodolmolky and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey", *IEEE*, Vol. 103, No. 1, pp. 14-76, 2015.

[4] Ibrahim and F. Hashim, "An architecture of 5G based on SDN NV Wireless Network", *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, Vol. 14, No. 2, pp. 725-734, 2019.

[5] H. Akcay and D. Yiltas-Kaplan, "Web-Based User Interface for the Floodlight SDN Controller", *Int. J. Advanced Networking and Applications*, vo. 08, No. 05, pp. 3175-3180, 2017.

[6] Z. SHU, J. WAN, J. LIN, S. WANG, D. LI, S. RHO and C. YANG, "Traffic Engineering in Software-Defined Networking: Measurement and Management", *IEEE Access*, Vol. 4, pp. 3246-3256, 2016.

[7] L. Chen, M. Qiu and J. Xiong, "An SDN-Based Fabric for Flexible Data-Center Networks ", *Computer Society, IEEE*, pp. 121-126, 2015.

[8] E. Hahn, "An Overview of Open-Source Software Licenses and the Value of Open-Source Software to Public Health Initiatives", *JHUAPL*, Vo. 32, No. 4, 2014.

[9] R. Mohammed and Y. Ueno, "An FPGA-based Network Firewall with Expandable Rule Description", *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, Vol. 10, No. 3, pp. 1310-1318, 2018.

[10] K. Kaur, K. Kumar, J. Singh and N. Ghumman, *"Programmable Firewall Using Software Defined Networking"*, 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, pp. 2125-2129, 2015.

[11] M. Atiyah, "*Development of Information Centric Network Over Software Defined Network Emulation*", M.Sc. Thesis, AL-Naharin University, Baghdad, Iraq, 2016.

[12] A. Jalil, *"Performance Evaluation of Software Defined Networks"*, M.Sc. Thesis, AL-Naharin University, Baghdad, Iraq, December 2017.

[13] T. Ali, A. Morad, and M. Abdala, "Load Balance in Data Center SDN Networks", *International Journal of Electrical and Computer Engineering (IJECE)*, Vol. 8, No. 5, pp. 3086-3092, 2018.

[14] V. Wadhwani, Y. Bajaj and Y. Rohra, "SDN Based Firewall", *IOSR Journal of Engineering (IOSRJEN)*, Vol. 08, No. 6, pp.24-30, 2018.

[15] R. Peng, and Lei Ding,"Research on Virtual Network Load Balancing based on OpenFlow", *AIP*, pp. 020014-1–020014, 2017.

[16] V. Moorthy, K. Sharma and R. Sachdeva, "Firewall and Load Balancer Implementation in Software Defined Networking", *International Journal of Pure and Applied Mathematics*, Vo. 115 No. 7, pp. 9-15. 2017.

[17] J. Fernández, L. Villalba and T. Kim, "Software Defined Networks in Wireless Sensor Architectures", *Entropy*, Vol. 20, Issue 4, pp. 1-23, 2018.

[18] T. Assegie, "A review on software defined network security risks and challenges ", *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, Vol.17, No. 6, 2019.

[19] T. Wang, H. Chen, G. Cheng, and Y. Lu, "SDNManager: A Safeguard Architecture for SDN DoS Attacks Based on Bandwidth Prediction", Hindawi, Article ID 7545079, pp. 16, 2018.

[20] Y. Liu, J. Muppala, M. Veeraraghavan, Dong Lin, and Mounir Hamdi, "Data Center Networks Topologies, Architectures and Fault-Tolerance Characteristics", *Springer*, 2013.

[21] K. Thimmaraju, L. Schiff and S. Schmid, "Outsmarting Network Security with SDN Teleportation", *IEEE European Symposium on Security and Privacy*, Paris, France, 2018.

[22] S. Assegie, P. Nair, "Performance analysis of emulated software defined wireless network", *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, Vol. 16, No. 1, pp. 311-318, 2019.

[23] S. Ali, V. Sivaraman, A. Radford and S. Jha, "A Survey of Securing Networks using Software Defined Networking" *IEEE Transactions on Reliability*, Vol. 64, No. 3, PP. 1086-1097, 2015.

[24] V. Deeban and B. Amutha, "Path based load balancing for data center networks using SDN", *International Journal of Electrical & Computer Engineering (IJEECS)*, Vol. 9 Issue 4, pp. 3279-3285, 2019.

[25] M. Qilin, and S. WeiKang, *"A Load Balancing Method Based on SDN"*, Seventh International Conference on Measuring Technology and Mechatronics Automation, pp. 18-21, IEEE, 2015.

BIOGRAPHIES OF AUTHORS

Sarah h. Mohammed was born in Baghdad/Iraq in 1989. She recived the B.S degree from College off Information Engineering at Al-Nahrain University, Iraq 2011 and have MSc. Degree form college of Information and communication Engineering at Al-Nahrain University, Iraq 2019. Currently, she is working as project manager at FastIraq Telecommunication company.

Ass. Prof. Dr. Ammar D. Jasim was born in Iraq/ Iraq 1978. He received the B.S degree from College of Engineering at Al-Nahrain University, Iraq 1999, has the MSc. Degree from the college of Engineering at Al-Nahrain University and, has the Phd. Degree from the college of Engineering at Al-Nahrain University. His researchs interests include internet of things, computer networks and Protocols, Machine Learning and Software Define Network.
Email: ammar@@coie-nahrain.edu.iq