Performance Analysis of Various SDN Controllers with Different Network Size in SDWN

Fidha Ezzaldin Aslan[1], Mohammed Basheer Al-Somaidai[2]
{fidha.enp18@student.uomosul.edu.iq[1], MohammedBasheerAbdullah@uomosul.edu.iq[2]}

Department of Electrical Engineering, Mosul University, Mosul, Iraq[1,2]

Abstract. Software-defined Wireless Networking (SDWN) is the next generation of wireless networks. It performs the same Software Defined Networking (SDN) operation by decoupling the control plane from the data plane on the network device and transfers it to the controller, to provide easy control and centralized management to the network. Since the controller represents the most important part of the software-defined network architecture. In this paper, we examine the performance of four SDN controllers: POX, RYU, Open Daylight (ODL), and Open Network Operating System (ONOS) in different sizes of the emulated wireless network in terms of the network parameters such as throughput, delay, jitter and packet loss using the Mininet WIFI platform. Tests of this experiment help to choose the appropriate controller according to various parameters and states of the wireless network.

Keywords: Controller performance, Mininet-Wi-Fi, OpenFlow, ODL, ONOS, POX, RYU, SDWN.

1 Introduction

Because of the continuous development in networks and the rapid growth of data traffic resulting from the increase in the number of connected devices and the requirements of speed performance, the traditional network architecture is not catching up with these increasing demands, Which Requires a new network architecture. This new network architecture is known as a software-defined network that addresses problems in the traditional network and provides easy control and centralized management of the network and being cost-effective.

SDN is a new network model that separates the network control plane from the Data forwarding plane in devices and transfers it to a central server called the controller, to make the network more controllable and less complicated [1]. As a result, the SDN reduces the role of the devices to only data forwarding and implementing control orders issued by the controller. The controller has a global view of the network and it is responsible for all control decisions and communicates with the forwarding elements distributed at the network level via standard interfaces [2].

SDWN, a subset of SDN, is recently launched to meet the increasing demand for wireless networking. SDWN performs the same SDN operation by switching the control plane from the data plane of the network beyond the access points to an external software control plane [3]. The access points manage the stations connected to them, according to the flow rules arranged by the controller, Having access to all information that is reported by the network devices [4], this new software-defined approach presents a cost-effective alternative for high-speed network services [5]. it enables administrators to define network behaviour in a logically central and

IMDC-IST 2021, September 07-09, Sakarya, Turkey
Copyright © 2022 EAI
DOI 10.4108/eai.7-9-2021.2314875

higher-level way through APIs provided by the controller platform that implements southbound interfaces (i.e OpenFlow protocol) to the forwarding devices to connect and set up these devices, there are several southbound interfaces but OpenFlow is the most popular one [6]. Since the controller represents the most important part of the software-defined wireless network architecture. It is necessary to evaluate the performance of different types of controllers, study their features and the extent of each effect on the network's performance to choose the appropriate controller by the network operators in line with the requirements of the service they want to provide in the network. Although there are many studies on comparing the performance of different types of controllers in wired networks, there are not enough studies on comparing their performance in wireless networks [9]. In this paper, we have studied the effect of increasing the size of the wireless network on the performance of different types of controllers POX, RYU, ODL, and ONOS and evaluated their performance based on network parameters that include delay, throughput, jitter, and packet loss. We have used the Mininet WIFI platform as an environment to emulate the SDWN and conduct the experimental tests, as well as a qualitative-evaluation has been done.

The remainder of this paper is structured as follows: Section 2 presents related works. Section3 discusses SDN platforms and OpenFlow, section 4 describes SDN Controllers, section 5 Methodology describes the steps of our experiments, section 6 analysis the simulation results and section 7 conclude the paper**.**

2 Related Works

Due to the importance of the controller in the SDN Architecture, which is attracting the attention of many researchers for being a revolutionary technology in the world of networks, many studies were conducted on the types of controllers and a comparison between their performance in wired and wireless networks. In addition to studies on the performance of wireless networks based on SDN.

The performance of two SDN controllers has been compared in [7], Open Daylight and Floodlight in terms of latency and loss in various network topologies and loads implemented in Mininet, wired network emulators. The authors evaluated the performance of controllers to choose the best in these topologies at different loads. They showed through the results using the confidence interval 95% that there are cases in which the ODL outperforms Floodlight in terms of delay in tree networks with light and medium loads, while floodlight showed better performance in terms of packet loss in tree networks with a heavy load, also in terms of delay in linear networks. In addition, there were cases in which the two controllers did not show any difference in performance.

I. T. Haque and N. Abu-Ghazaleh [8] presented a survey paper about the use of Software Defined Networking (SDN) in wireless networks. They evaluated four different types of wireless networks: cellular, sensor, mesh, and home networks. Moreover, a brief overview of SDN's implementations in other types of wireless networks.

The performance of the software-defined wireless network was analysed in [5], by constructing single and linear topologies using the Mininet WIFI platform with OpenFlow controller. Throughput performance decreased with the increase in the number of stations and access points and as a solution to this problem, load balancing between APs has been proposed to move stations from heavily loaded AP to other AP using the handover process.

In The paper [9], the Authors made a comparison between four SDN controllers, Ryu, Floodlight, ONOS, and ODL based on their features, in addition to evaluating their performance by measuring the delay and throughput using the bench tool with increase the load on the network. From the comparison of features, ODL was one of many features in terms of vendor support for interfaces, while from the performance evaluation comparison, ONOS showed the best throughput performance, and ryu showed the least delay performance.
S. Islam et al. [10] evaluated the performance of four SDN controllers Ryu, ONOS, Pox, and Floodlight in a wireless network (linear topology) which is simulated using Mininet WIFI in terms of network parameters such as throughput, delay, and jitter measured by iperf and tcpdump tools, to determine which one is most suitable for the wireless network. The controllers showed different performances in terms of delay and jitter, while the performance was similar in terms of throughput. These research focused on evaluating the performance of controllers in wired networks. In addition to studying the effect of increasing the network size on the throughput performance of SDWN using the default program controller, as well as evaluating the performance of the controllers in the (linear,3) topology of SDWN. The effect of increasing the network size on the performance of the different controllers in the wireless network has not been studied, which is what was focused on in our research

3 SDN Platforms and OpenFlow

3.1 Mininet

Mininet is a virtual network simulation platform that supports the rapid evolution of SDN (SDN emulator) using OpenFlow protocol. Bob Lantz and Brandon Heller developed Mininet 1.0, which was based on the original Mininet model provided by Bob Lantz. Mininet creates a virtual network using the Linux kernel with Python scripts from hosts, OpenFlow switches, and controllers in any network structure that the researcher uses on a single machine [11]. Where the network can be simulated, connected, and switched on the same computer, which helps to develop, test, and clarify the network structures related to the SDN. The software in the basic system of this virtual operation can also be transferred to a real environment.

3.2 Mininet-Wi-Fi

Minnie-Wi-Fi is an open-source software-defined wireless network emulator that was expanded from Mininet by adding virtual wireless stations and access points [12], In addition, to use Mininet nodes such as hosts, switches, and OpenFlow controllers and retaining the original SDN features [13].

3.3 OpenFlow

Network devices communicate with the controller via a southbound interface using the OpenFlow protocol, the most used protocol for the southbound interface of the SDN, which separates the data plane from the control plane. OpenFlow was initially proposed (firstly suggested) by Stanford University 2009 [14] and it was developed by the Open Networking Foundation (ONF), An organization dedicated to supporting and implementing Software

Defined Networking (SDN) and managing the OpenFlow standard. OpenFlow first appeared in the year 2009 according to OpenFlow v1.0, and updates continued until it reached the current version of OpenFlow v1.5 [15].

4 SDN Controllers

The central component of any SDN infrastructure is an SDN Controller, which has a network overview that includes all data plane devices. The controller interfaces these resources to the application plane and performs flow control actions through data plane devices based on application policies [16]. This section introduces four SDN controllers: POX, RYU, ODL, and ONOS.

4.1 POX

POX is a Python-based SDN platform. It began as an OpenFlow controller, but it can now also act as an OpenFlow switch, making it useful for networking software development in general. It currently supports OpenFlow 1.0 and the Open vSwitch/Nicira extensions. POX must be run under Linux, Mac OS X, and Windows. On some platforms, some features are not accessible. the most feature-rich operating system is Linux. [17]. POX has no official graphical user interface. Since it is written in Python; it has a weak performance comparable to that of other programming languages such as Java or C++. Furthermore, POX only supports OpenFlow version 1.0 and does not function in a distributed way, and does not support multi-threading [18].

4.2 Ryu

Ryu [19] is a multi-threaded SDN controller written in Python and based on the line went event wrapper. It supports all OpenFlow versions (1.0, 1.2, 1.3, 1.4, 1.5, and Nicira Extensions), Netconf, OF-config, and other Southbound APIs. [20] multi-threading helps Ryu to perform much better with heavy loads. However, it is still considered as a poor performance compared to newer controllers by many criteria. Ryu comes with a very simple web-based GUI which displays topology and flow details [16].

4.3 Open Daylight [21]

The Linux Foundation Sponsored In 2013 Open Daylight Project with the contribution of several companies with the project such as HP, Cisco, NEC, and IBM. The project aims to enhance the software-defined network (SDN) by offering a community-led and industry-supported system for the Open Daylight Controller, which has been called the Open Daylight Platform. It is an open-source controller and the base for many proprietary controllers, such as Ericsson's SDN Controller and Fujitsu's Virtuora, written in Java and supports many southbound protocols, including OpenFlow, Border Gateway Protocol (BGP), Open vSwitch Database Management Protocol (OVSDB), Simple Network Management Protocol (SNMP), NETCONF and Path Computation Element Communication Protocol (PCEP). ODL Supports multi-threading and distributed controllers and comes with rich documentation but these documents

are somewhat old in some features, such as The DLUX GUI and the L2switch module are still mentioned in documentation even though they were both retired [22- 23].

4.4 ONOS

Open Network Operating System (ONOS) was developed in 2014 and is a Java-based open-source SDN controller. It is part of The Linux Foundation's collaboration project, such as ODL. This controller has the properties of high performance, availability, and scalability [15]. and is the base for many controllers, such as Huawei. ONOS is one of the controllers with a broader range of SBI coverage due to its extensive SBI support for OpenFlow, P4, NETCONF, TL1, SNMP, BGP, RESTCONF, and PCEP protocols. It comes with excellent documentation and is simple to implement in the market [23]. ONOS can run in both distributed and multi-threaded mode and be created with the primary purpose of assisting in the management of large-scale, fast networks, with a focus on the ISP market segment**.** ONOS has a very useful web-based GUI [24].

5 Methodology

The performance of software-defined wireless networks is greatly affected by the number of end stations and access points that configure the network. In this section, we have created a linear topology with different numbers of access points and end stations on a Mininet WIFI environment as shown in Fig 1. Each end station in linear topology is connected with one access point which is in turn connected linearly and each access point is connected to the remote controller, four SDN controllers were used in our experimental evaluation, (ONOS, ODL, Ryu, Pox) as a remote controller We have changed the topology size every time as illustrated in Table 1, then we measured the network parameters (Throughput, Delay, Jitter, and Packet loss) for each case To evaluate the performance of the SDN controllers in the network

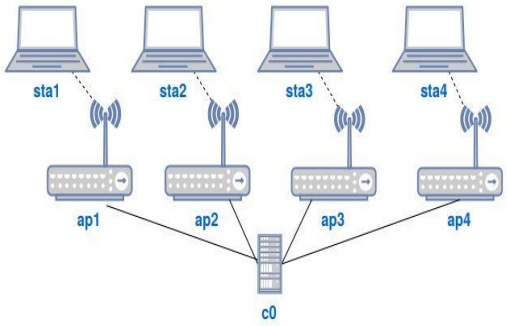

Fig. 1. Linear topology.

Table 1. Several access points and end stations of the linear topology.

No. of access points	No. of end stations
25	25
50	50
75	75
100	100

Mininet-Wi-Fi and controllers were installed and run on a Laptop with Intel core ™ i5-3320M CPU@2.60GHzx4, 8GB memory, and Ubuntu 18.04 LTS-64-bits, which is one of the more famous Linux kernel distributions as an operating system. Throughput is an actual measure of the amount of data that is successfully transferred from source to destination. Throughput has been measured using iperf (internet performance), benchmarking tool that has the functionality of client and server and it can create TCP (Transmission Control Protocol) traffic or UDP (User Datagram Protocol) traffic to calculate throughput in one or both directions between the two ends in the network. In addition, we have used Wireshark, a network performance monitoring tool to capture and analyse the traffic generated by iperf in TCP mode between the two farthest stations in the network to measure throughput and packet loss. The following equation is used to find the throughput. Throughput (b/sec) = Number of received packets × packet size / Elapsed time. (1) Delay has also been measured between the two farthest stations in the network as RTT (Round Trip Time) using the ping command for testing the connectivity between two stations (hosts) in the network. Ping sends one or more ICMP (Internet Control Message Protocol) Echo Request packets to the certain network destination IP and waits for a response**,** when the packet arrives at its destination; it sends an ICMP echo reply. Jitter has measured using the iperf command in UDP mode, between the first and the last station in the network. Jitter is the variation in delay of packet flow between two ends over time. This is often happening because of the Congestion on the network as well as path shifts. The effect of jitter is not seen in services that do not need actual time for delivery, such as data transfer, while its effect is significant in real-time services such as video and audio services.

6 Simulation Results and Analysis

This section describes the carried-out tests with iperf tool, ping command, and Wireshark tool, It can be seen that the parameters (Throughput, Delay, Packet loss, and Jitter) are directly related to increased network size as shown in the summarized results below:

6.1 Throughput

The throughput of each network is found using iperf tool at the TCP mode, considering the last station as a server and the first station as the client where through this tool the network traffic is generated and the performance of the iperf tool was analysed using Wireshark. The number of transmitted packets was about 35,320 packets, with a transmission time of 30 seconds for each

measurement. The results obtained in Fig 2, where the X-axis represents the number of stations and the Y-axis represents throughput in Mbits/sec, shows the effect of increasing the network size on the throughput of the network for all controllers used in this research

1. Linear25: there is no significant difference between the controller's performance, as the ODL controller shows the best performance (10.3 Mbits / sec), followed by the ONOS controller (9.9 Mbits / sec), then the RYU (9.2 Mbits / sec) and Pox (about 8.9 Mbits / sec)
2. Linear50: the performance of ONOS controller and ODL controller is close (9.6 Mbits / sec) and (9.3 Mbits / sec), respectively, then RYU (8.9 Mbits / sec) and Pox (8.1 Mbits / sec).
3. With the increase in network size to linear 75, the ONOS controller outperforms the rest controllers (9.2 Mbits / sec), followed by Ryu (7.1 Mbits / sec), Pox (6.3 Mbits /sec), while ODL is the least performing (5.8 Mbits / sec.)
4. Linear100, ONOS is the best performance in throughput (7.3 Mbits / sec), while ODL is the lowest (3.8 Mbits / sec) and the rest shows a comparable performance (5.2 Mbits / sec) for ryu and (5 Mbits / sec) for pox.

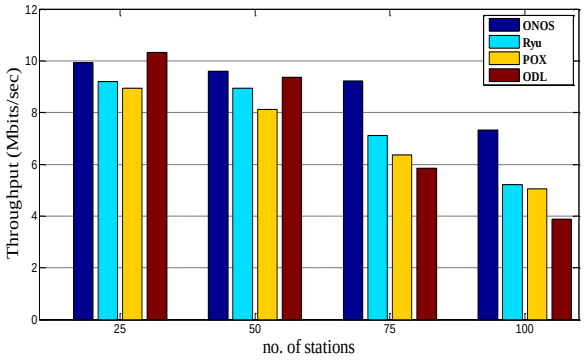

Fig. 2. Throughput comparison

6.2 Delay

As indicated in the previous part, RTT delay was calculated by creating a connection via ping command for 10 packets between the first station and the last station in the network. In Fig 3, the X-axis shows the number of stations in the network and the y-axis represents the RTT delay in msec
In all of the different network sizes, RYU and ODL are the least delayed and outperform POX and ONOS.

1. Linear25: ODL is least delayed (1.1 msec) then RYU (1.2 msec), pox (msec7.4), and ONOS (8.9 msec).
2. Linear50: ODL (3.5 msec), RYU (3.7 msec), pox (62.5 msec), onos (71.4 msec)
3. Linear75: With increased network size, RYU is the least delayed (6.1 msec), as ODL performance decreases (12.8 msec), pox (7.4 msec), ONOS (8.9 msec).

4. Linear100: RYU (17.5 msec), ODL (23.1 msec), pox (164.2 msec), ONOS (300.6 msec).

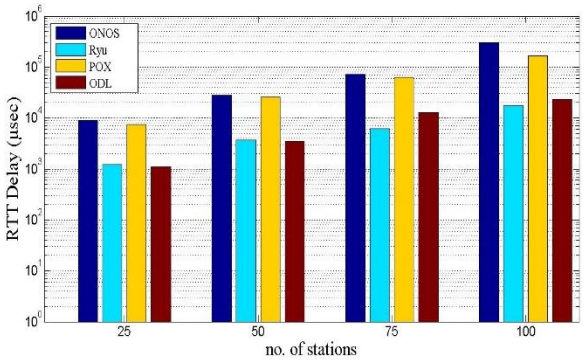

Fig. 3. RTT Delay comparison

6.3 packet loss

Fig 4 represents the relationship between the increase in the network size and the loss of the packets within 30 seconds of sending and receiving the packets and capturing them with the Wireshark tool (about 35,000 packets) where the X-axis represents the number of stations, and the Y-axis represents the number of lost packets for each controller.

1. Linear25: at linear25 ONOS controller is like RYU in packet loss, about (16) packets and (15) packets respectively. ODL and Pox also shows similar packet loss results,(21) packets and (20) packets respectively
2. Linear50: the four controllers show almost the same performance of packet loss, ONOS (53) packets, RYU (49) packets, POX (50) packets, ODL (55) packets.
3. Linear75: ODL shows worse in terms of packet loss (220) packets while Ryu shows less packet loss (81) packets, POX and ONOS show almost similar performance (122) packets and (114) packets respectively.
4. Linear100: ODL shows the worst performance of packet loss (321) packets, while RYU shows the best packet loss performance among them (124) packets. ONOS controller shows about (177) packets and pox (140) packet lost.

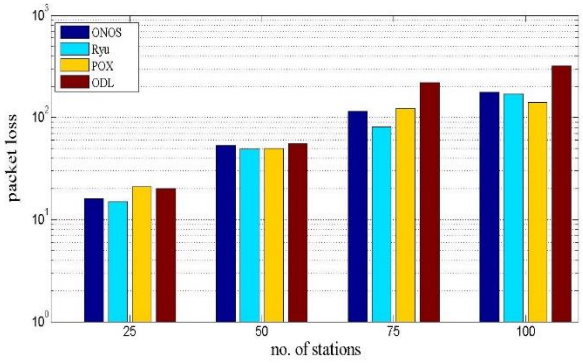

Fig. 4. Packet loss comparison

6.4 Jitter

In Fig 5, the X-axis represents the number of stations, and the Y-axis represents the average jitter in msec. The jitter was measured for 30 seconds, and the average value has been taken. We noticed the following:

1. Linear25: the four controllers show a converging performance, ODL is the least jitter (0.035 msec), ONOS (0.047 msec), POX (0.048 msec), and RYU (0.05 msec)
2. Linear50: the worst performer is RYU (4.575 msec) followed by POX (4.395 msec) and ODL (4.214 msec) and the best performance is ONOS (4.199 msec)
3. Linear75: onos performs worse compared to other controllers (5.716 msec), followed by RYU (5.479 msec), pox (5.301 msec), then ODL (5.254 msec)
4. Linear100: the performance of ONOS is the worst (7.03 msec), then RYU (6.724 msec), pox (6.695 msec), and ODL (6.348 msec).

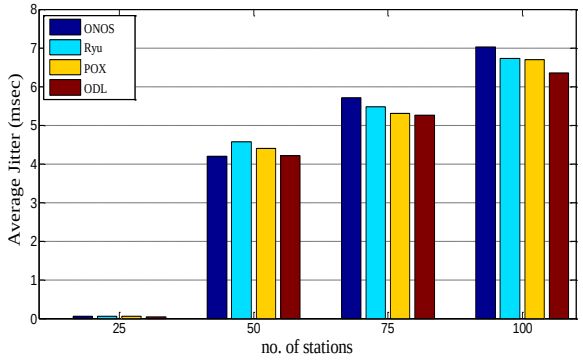

Fig. 5. Average jitter comparison

7 Conclusion and Future work

SDWN is a new structure for wireless networks by separating the control plane from the data plane and centralizing the network control through the controller to overcome the complexity in the traditional network architecture due to the rapid growth in data traffic because of the increasing demand for a fast-performing and low-cost network. In this paper, a performance evaluation of four SDN controllers has been done. The performance evaluation results showed that ONOS performs well with the increase in the network sizes in terms of throughput and packet loss. In terms of delay, ODL and Ryu showed the best delay results under different network sizes. Jitter results showed that ODL is the least Jitter results. The future work will focus on evaluating the performance of different SDN controllers in a custom wireless network topology using mobile devices with a handover mechanism between access points when one of the access points is overloaded.

References

[1] Agarwal S, Kodialam M, Lakshman TV. Traffic engineering in software defined networks. In2013 Proceedings IEEE INFOCOM; 2013 Apr 14; Turin, Italy. (pp. 2211-2219). IEEE.

[2] Kim H, Feamster N. Improving network management with software defined networking. IEEE Communications Magazine. 2013 Feb 14;51(2):114-9.

[3] Jany MH, Islam N, Khondoker R, Habib MA. Performance analysis of OpenFlow based software defined wired and wireless network. In:2017 20th International Conference of Computer and Information Technology (ICCIT); 2017 Dec 22; Dhaka, Bangladesh. (pp. 1-6). IEEE.

[4] Yang M, Li Y, Jin D, Zeng L, Wu X, Vasilakos AV. Software-defined and virtualized future mobile and wireless networks: A survey. Mobile Networks and Applications. 2015 Feb;20(1):4-18.

[5] Assegie S, Nair P. Performance analysis of emulated software defined wireless network. Indonesian Journal of Electrical Engineering and Computer Science (IJEECS). 2019 Oct;16(1):311-8.

[6] Fontes RD, Mahfoudi M, Dabbous W, Turletti T, Rothenberg C. How far can we go? towards realistic software-defined wireless networking experiments. The Computer Journal. 2017 Oct 1;60(10):1458-71.

[7] Rowshanrad S, Abdi V, Keshtgari M. Performance evaluation of SDN controllers: Floodlight and OpenDaylight. IIUM Engineering Journal. 2016 Nov 30;17(2):47-57

[8] Haque IT, Abu-Ghazaleh N. Wireless software defined networking: A survey and taxonomy. IEEE Communications Surveys & Tutorials. 2016 May 19;18(4):2713-37.

[9] Mamushiane L, Lysko A, Dlamini S. A comparative evaluation of the performance of popular SDN controllers. In:2018 Wireless Days (WD); 2018 Apr 3; Dubai, United Arab Emirates. (pp. 54-59) . IEEE.

[10] Islam S, Khan MA, Shorno ST, Sarker S, Siddik MA. Performance Evaluation of SDN Controllers in Wireless Network. In:2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT); 2019 May 3; Dhaka, Bangladesh. (pp. 1-5). IEEE.

[11] Al-Somaidai MB, Yahya EB. Survey of software components to emulate OpenFlow protocol as an SDN implementation. American Journal of Software Engineering and Applications. 2014 Dec 16;3(6):74-82.

[12] Fontes RR, Afzal S, Brito SH, Santos MA, Rothenberg CE. Mininet-WiFi: Emulating software-defined wireless networks. In:2015 11th International Conference on Network and Service Management (CNSM); 2015 Nov 9; Barcelona, Spain. (pp. 384-389). IEEE.

[13] Fontes RR, Rothenberg C. Wireless Network Emulation with Wireless Network Emulation with Mininet-WiFi. Columbia (USA): Christian Esteve Rothenberg; 2019. p. 14

[14] Braun W, Menth M. Software-defined networking using OpenFlow: Protocols, applications and architectural design choices. Future Internet. 2014 Jun;6(2):302-36.

[15] "Open Networking Foundation." https://opennetworking.org/ (accessed Mar. 11, 2021).

[16] Zopellaro Soares AA, Lucas Vieira J, Quincozes SE, Ferreira VC, Uchôa LM, Lopes Y, Passos D, Fernandes NC, Monteiro Moraes I, Muchaluat-Saade D, Albuquerque C. SDN-based teleprotection and control power systems: A study of available controllers and their suitability. International Journal of Network Management. 2021 May;31(3):e2112.
[17] "Installing POX — POX Manual Current documentation." https://noxrepo.github.io/pox-doc/html/ (accessed Mar. 11, 2021).
[18] Stancu AL, Halunga S, Vulpe A, Suciu G, Fratu O, Popovici EC. A comparison between several Software Defined Networking controllers. In2015 12th international conference on telecommunication in modern satellite, cable and broadcasting services (TELSIKS); 2015 Oct 14; Nis, Serbia. (pp. 223-226). IEEE.
[19] "GitHub - faucetsdn/ryu: Ryu component-based software defined networking framework." https://github.com/faucetsdn/ryu (accessed Mar. 11, 2021).
[20] Ahmad S, Mir AH. Scalability, Consistency, Reliability and Security in SDN Controllers: A Survey of Diverse SDN Controllers. Journal of Network and Systems Management. 2021 Jan;29(1):1-59.
[21] "Home - OpenDaylight." https://www.opendaylight.org/ (accessed Mar. 11, 2021).
[22] Eftimie A, Borcoci E. SDN controller implementation using OpenDayLight: experiments. In2020 13th International Conference on Communications (COMM) 2020 Jun 18; Bucharest, Romania. (pp. 477-481). IEEE.
[23] Neto FJ, Miguel CJ, de Jesus AC, Sampaio PN. SDN Controllers-A Comparative approach to Market Trends. In9th International Workshop on ADVANCEs in ICT Infrastructures and Services (ADVANCE 2021) 2021 Feb 2.
[24] "ONOS - ONOS - Wiki." https://wiki.onosproject.org/display/ONOS/ONOS (accessed Mar. 11, 2021).

Acknowledgments. The authors wish to acknowledge mosul university for providing the facilities needed for this paper.