Scientific Journal of Informatics
Vol. 6, No. 2, Nov 2019
p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Software-Defined Networks: a Comparative Study and Quality of Services Evaluation

Herry Prasetyo Nugroho[1], Muhammad Irfan[2], Amrul Faruq[3,*]

[1,2,3]Departement of Electrical Engineering, Faculty of Engineering,
Universitas Muhammadiyah Malang
Email: *faruq@umm.ac.id

Abstract

Software-Defined Network (SDN) as architecture network that separates the control and forwarding functions, so that network operators and administrators can configure the networks in a simple and centrally between thousands of devices. This study is designed and evaluate the Quality of Services (QoS) performances between the two networks employed SDN-based architecture and without SDN-based. MinNet as a software emulator used as a data plane in the network Software Define Network. In this study, comparison of the value of the QoS on the network based on Software Defined Network and traditional network during the test run from the source node is investigated. Network testing by using traffic loads. Traffic loads are used starting from 20Mbps-100Mbps. The result is verified that the QoS analysis of the Software-Defined Network architecture performed better than conventional network architectures. The value of the latency delay on the Software Define Network range between 0,019-0,084ms, and with 0% packet loss when addressed the network traffics of 10-100Mbps.

Keywords: Software Defined Networks, Floodlight, MiniNet, Computer Networks, Quality of Services

1. INTRODUCTION

The need of technology for computer networks and its devices in a variety of industries experiencing rapid growth. It affects the need for better performance as the primary goal, the addition of network configuration becomes larger and more complex, and part of network control is getting intricated so that the network is inflexible and difficult to manage [1]. Enforcing the required policies in such a dynamic network architecture is therefore highly challenging.

Today's network architecture (traditional network) is still very complicated because each device has a different configuration (control plane) and data (forwarding plane) that is embedded in the device itself [2]. To make it even more complicated, current networks are also vertically integrated. Each device has a routing protocol that is very inflexible, inefficient, hindering innovation and evolution of the networking infrastructure. Furthermore, the configuration is done on each device. It is certainly not able to meet the operational demands with large scale networks and network devices that have different specifications [3]. This due to the control plane and forwarding plane are in one device.

SDN introduces a mechanism to improve various aspects of network management [4]. The basic concept of Software Defined Networking (SDN) is to perform the separation explicit between the control and forwarding plane. Software-defined networking is able to organize and manage up to thousands of network devices through a point of management, network monitoring both in terms of resources and connectivity, change the behavior of the networks automatically, maximizing the use of devices such as network bandwidth optimization, load balancing, traffic engineering and others associated with the programmability and scalability [5], [6]. Previously published studies have explored how SDN can provide better mechanisms for common network management and configuration tasks across a variety of problems including Internet of Things [7], cellular SDN [8], and data centre networks [9].

According to the Open Network Foundation (ONF) with the title "Software Define Networking: The New Norm For Networks" [10], SDN is defined as a network architecture that separates the functions of control and forwarding, so that network operators and administrators can configure the network in a simple and centrally between thousands of devices. In addition, the SDN controller is able to change the behaviour of the network in real-time and deploy new applications in network services in seconds. SDN architecture is divided into three layers namely the application layer, control layer, and infrastructure or data layer [11]. The application layer is an interface to manage or develop an SDN network as illustrated in Figure 1. Control layer is a centralized controller and software-based.

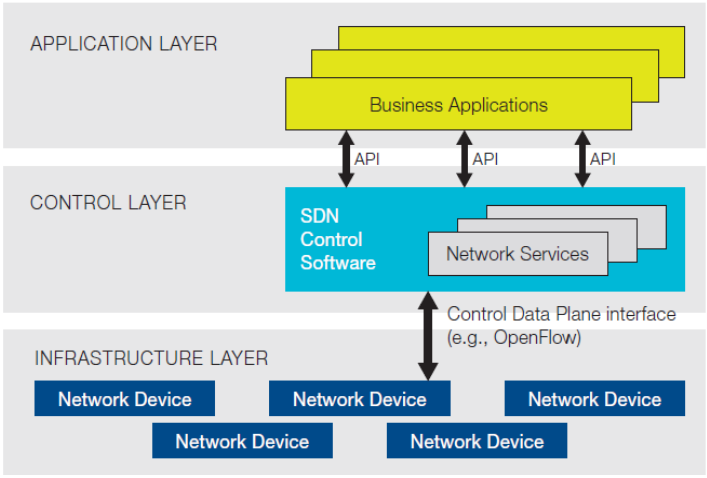


Figure 1. Default architecture of SDN defined by ONF [10]

A very fundamental difference between SDN networks with the traditional network is the placement of control and data plane functions. In traditional network control and data plane functions are simply placed on the device itself

such as a router [12]. While on the SDN network control functions located on a piece of software into a centralized controller and function forward, in other words, data is placed on an empty device in the form of switches (forwarding device). Floodlight is the enterprise-class controller, licensed under Apache, Java-based OpenFlow controller. It is supported by a community of developers, including some engineers from Big Switch Networks. OpenFlow is an open standard maintained by the Open Networking Foundation (ONF). OpenFlow is a protocol for Controller that can modify the behaviour of network devices through "forwarding instruction set" that is well defined [13]. Floodlight is designed to work with a number of switches, routers, virtual switches, and access points that support the OpenFlow standard [14]. The detailed architecture of conventional network and SDN, as shown in Figure 2. While the floodlight controller is described in Figure 4.

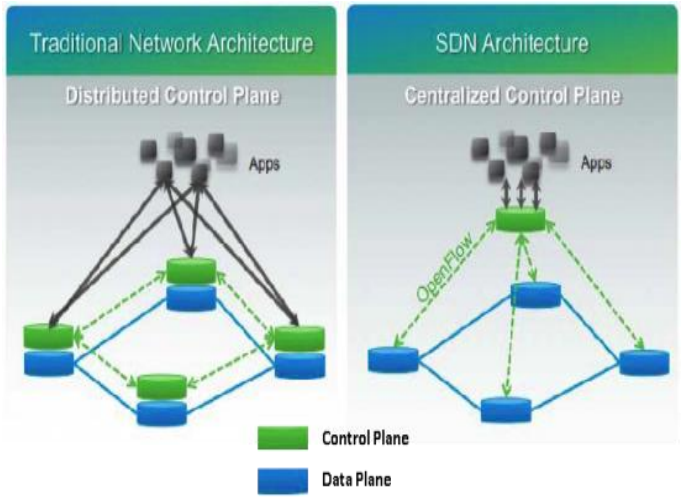


Figure 2. Differences between conventional networking and using SDN

Emulators are used in this study is MiniNet. MiniNet is a software emulator that allows to conduct experiments on a vast network and only use one engine. On MiniNet can be designed with a network topology that is cool, simply MiniNet function as an emulator on the data path to perform experiments on the network SDN. Meanwhile, to make MiniNet experiments can be done with the command "`sudo mn`" with this command by default MiniNet will emulate a network configuration consisting of one piece controller switches 1 and 2 hosts. Figure 3 shows an Emulator using MiniNet.

Therefore, in this study is tested against the performance of the SDN and non-SDN network using the same network topology in the engineering faculty of Universitas Muhammadiyah Malang (UMM). Both topologies are used in order to determine the ability of a network that can be generated by non-SDN and

network handling. This study simulates using the SDN network emulator and controller MiniNet floodlight, In this test using several parameters of QoS according to [15], including latency delay, titter, throughput, and packet loss to determine the value of the second network QoS.

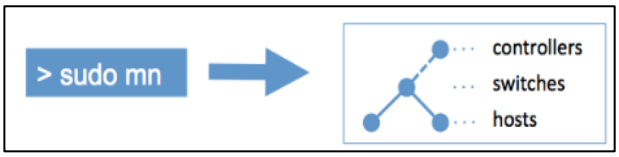


Figure 3. Emulator MiniNet

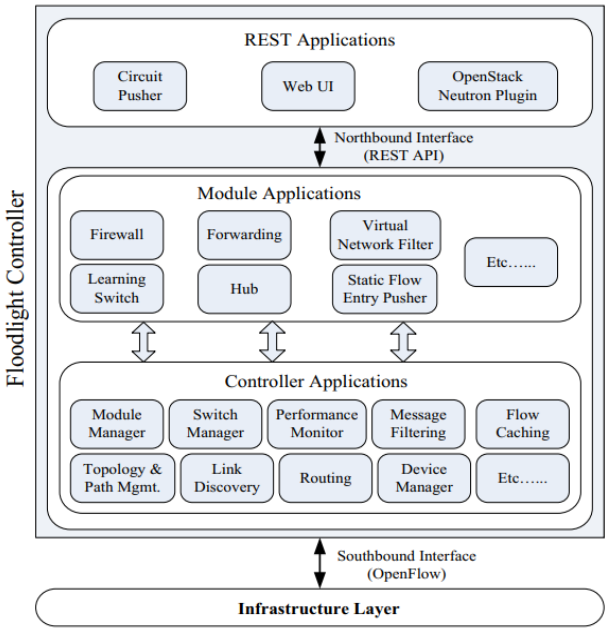


Figure 4. Internal Floodlight Controller in SDN according to [16]

2. METHODS

At this stage, the literature review is investigated in the form of journals, e-books, scientific articles, as well as the source of Internet sites related to the concept of Software Defined Network (SDN), the concept OpenFlow, the testing parameters performance network (QoS), and MiniNet configuration, and Floodlight controller [2], [16]. While the mentioned figure are detailed topology used in this study. Figure 5 and 7 respectively shows a network design without SDN and with SDN.

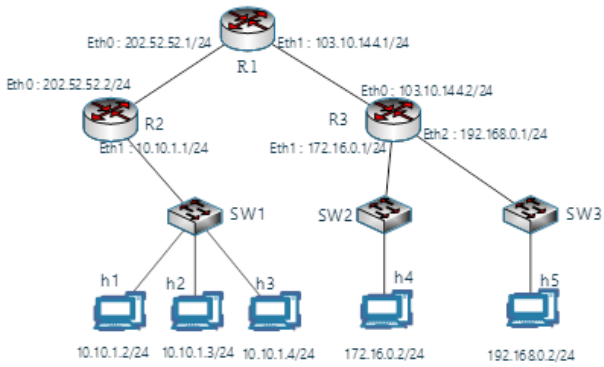


Figure 5. Network Topology without SDN

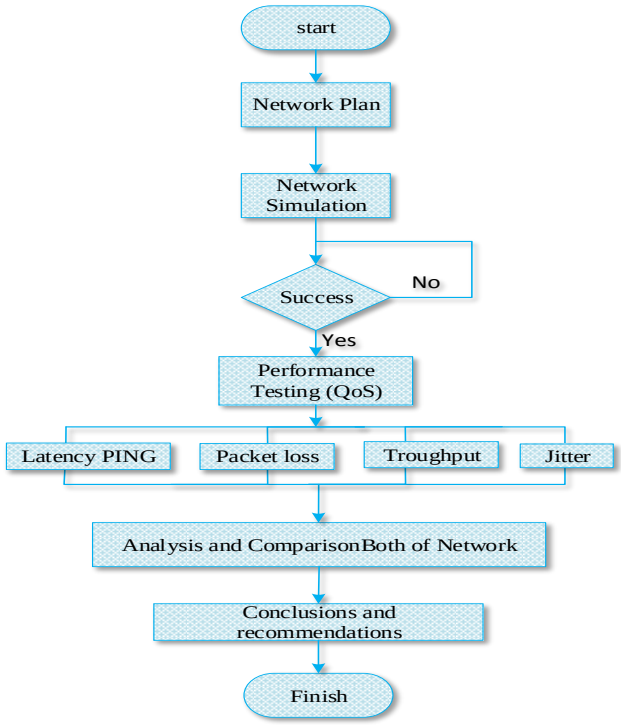


Figure 6. Flowchart System Design Process

System design related to network performance testing on the network architecture, both using SDN and without SDN. The steps taken in this study with references from the literatures [13], [16] as can be seen in flowchart Figure 6. To build a computer network design in Faculty of Engineering UMM-based on

Software Define Network, there are some of the main constituent components, namely: (a) Control Plane, a server which served as the control center of the network Software Define Network. (b) Data Plane, a server that served as a client on a network emulator for Software Define Network. And (c) Software GNS3, as a conventional network emulator to a comparison of network-based Software Define Network. Overall system design in this study as illustrated in Figure 6.

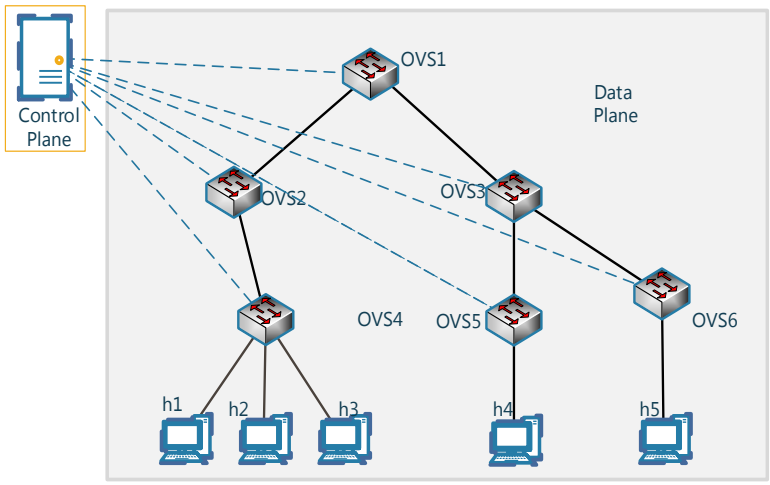


Figure 7. Network Design Based SDN

3. RESULT AND DISCUSSION

Performance test has been conducted on the network using SDN and without SDN. The test is done by providing regular traffic loads range from 20Mbps - 100Mbps network, and adding nodes. QoS parameters used for testing include Latency (*delay*), Jitter, throughput, and Packet Loss. Applications used to capture data using Wireshark software, Wireshark application can record packages that run on the network and recognize the many protocols and their*port protocol* on the network.

3.1. Scenario #1

At this stage will be tested without the burden of network traffic, testing is done using PING between hosts on both networks and the time of network testing with normal network traffic, no additional load on the network traffic values obtained as shown in Table 1.

3.2. Scenario #2

At this stage of testing will be done by using the network traffic load. Traffic load used ranging from 20Mbps - 100Mbps. The following results for his performance test are provided in Figure 8.

Table 1. Network Test Results Without expenses

Network Model	Average Latency (ms)	Jitter (ms)	Throughput (KBps)	Packet Loss (%)
Non-SDN	1,638	0.127	1,638	0
SDN	0,084	0,012	0,072	0

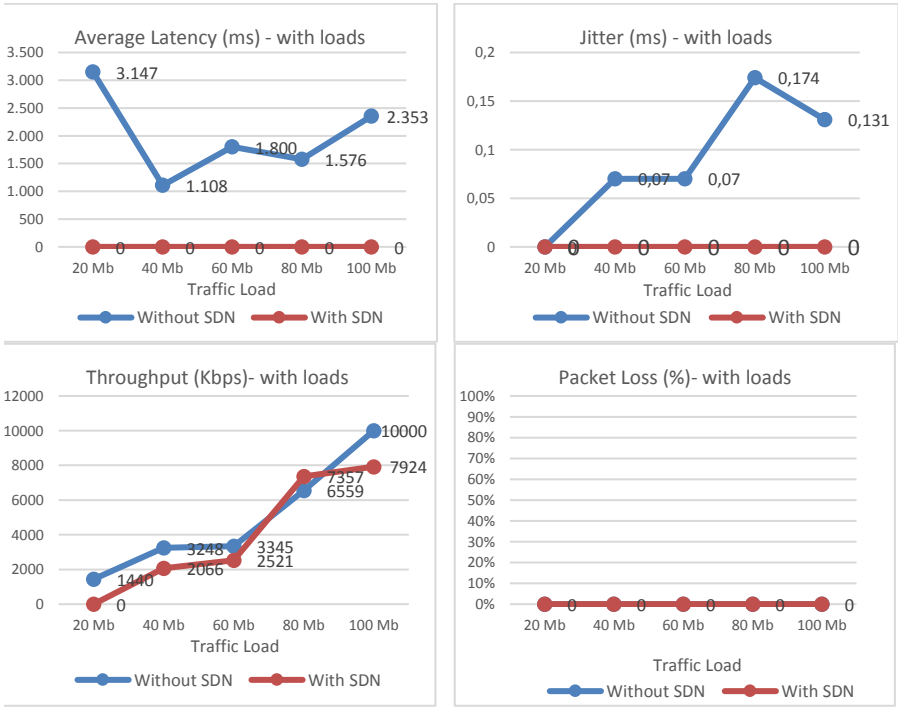


Figure 8. The performance evaluation result of the emulated network architecture with loads employed. Including Latency, Jitter, Throughput, and packet loss

After testing the QoS parameter values obtained on the network, the testing is done by generating traffic on the network. Generating traffic aimed to see how the response is happening on the network at a given load, and see the effects on QoS parameters. From the results, the using SDN shows that Packet Loss does not happen at a given load of 20-100Mb. In the networks non-SDN has the value of higher QoS parameters, whereas the SDN network has QoS parameter values are lower, which means SDN network is better than without SDN.

Latency: Figure 9 shows that the average delay on the simulated SDN network is 0,084ms when the network has not given the traffic load. And shows that the latency network non-SDN is 1,638ms. While at the given traffic load into the

network at 100M, the latency value on the network SDN turned into 0,022ms and on the non-SDN network becomes 2,353ms. Value of delay network on both network non-SDN and using SDN are still meet G.1010 QoS standardization ITU.T [15] which is minimum is the 60s. The increase in delay is affected by the increase in traffic on the network, network conditions will be a queue full longer so that the delay in delivery package becomes higher. The value of the network delay and non-SDN SDN very different. In non-SDN network, each node on the network, the more time required for packet transmission between hosts. In contrast with the SDN network delay value generated is very small despite the extra traffic on the network, because the SDN has the control center set up a centralized network performance using the OpenFlow protocol.

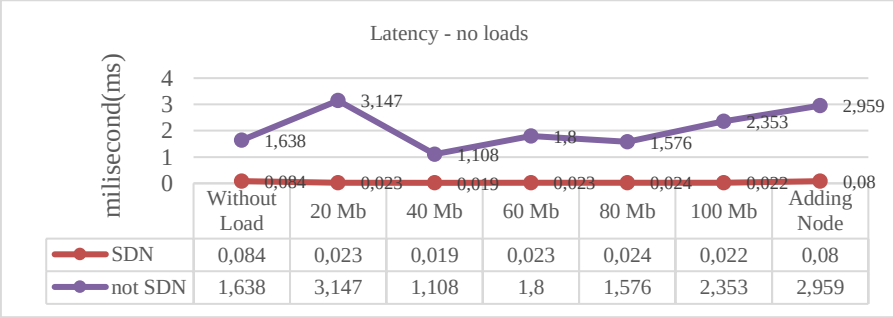
Jitter: Figure 10 shows that the value of Jitter on the simulated network without loads, the SDN network is 0,012ms and the non-SDN network is 0,127ms. When given traffic load 100M on the network, Jitter value rose to 0,050ms on SDN network and the network of non-SDN rise up to 0,131ms, the increasing value of Jitter on the network SDN looks more stable, whereas the non-SDN network Jitter value is not stable. The value of Jitter network on non-SDN and SDN are still meet G.1010 QoS standardization ITU.T [15] of <60s. Increase in value depends on the length of queue Jitter in the network, the density of traffic on the network can cause data collisions (congestion). In non-SDN network Jitter generated value higher than the SDN for the non-SDN network does not have management traffic through the network. Jitter value obtained using a network analyzer `Wireshark` software and `Iperf` or by the formula in Equation 1.

$$Jitter = \frac{Total\ delay\ variations}{Total\ packet\ inbound - 1} \qquad (1)$$

Throughput: Figure 11 shows the value of the network throughput for SDN is 0,072KBps and 0,168KBps for non-SDN respectively when the network has not given traffic load. When given the traffic load on the network of 100M, the throughput in SDN rise up to 11000KBps and the non-SDN network rise up to 10000KBps. The increase in throughput resulting from the granting of traffic on the network, the second increase in throughput on the different network because many factors, throughput usually always associated with bandwidth in actual conditions. More bandwidth is fixed while throughput is dynamic depending on the condition of the existing network traffic. Throughput as in Equation 2 can be searched using a network analyzer Wireshark software support.

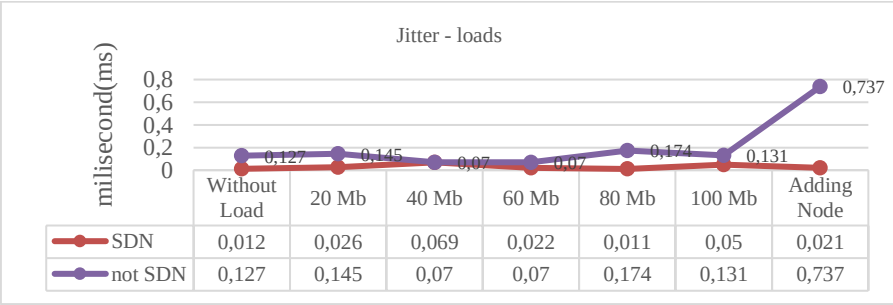
$$Throughput = \frac{Total\ packets\ send}{Time\ of\ data\ send} \qquad (2)$$

Packet Loss: Figure 12 shows the value of packet loss on the same network (0%) or can be seen does not happen Packet Loss on both networks. Packet Loss will occur when the network is given a load exceeds the limit that can be passed on the network. ValuePacket Loss on both network using SND and non-SDN are still meetG.1010 QoS standardization ITU.T [15] Packet Loss is allowed at 0%. packet loss due to an increase in the amount of data that is passed from the initial node to the destination node. Due to his numerous amount of data that is passed there will be a delay in the delivery of the package and allows breakdowns in sending data because the network is not able to regulate the traffic that goes in it so that there is congestion.



	Without Load	20 Mb	40 Mb	60 Mb	80 Mb	100 Mb	Adding Node
SDN	0,084	0,023	0,019	0,023	0,024	0,022	0,08
not SDN	1,638	3,147	1,108	1,8	1,576	2,353	2,959

Figure 9. Overall Latency performance test result of the emulated network architecture



	Without Load	20 Mb	40 Mb	60 Mb	80 Mb	100 Mb	Adding Node
SDN	0,012	0,026	0,069	0,022	0,011	0,05	0,021
not SDN	0,127	0,145	0,07	0,07	0,174	0,131	0,737

Figure 10. Overall Jitter performance test result of the emulated network architecture

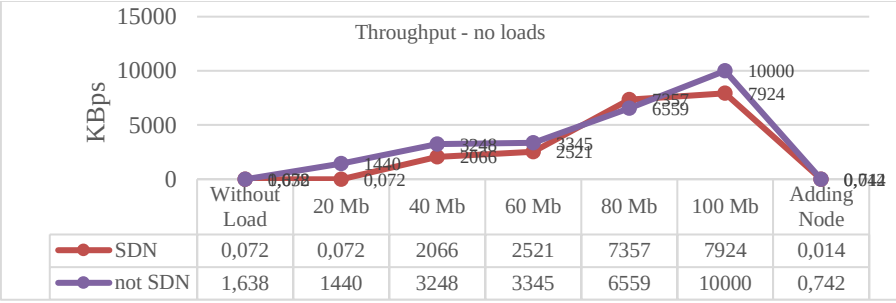


Figure 11. Overall Throughput performance test result of the emulated network architecture

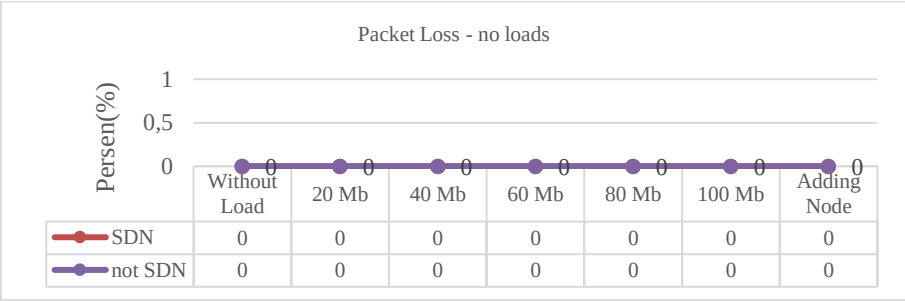


Figure 12. Overall Packet loss performance test result of the emulated network architecture

4. CONCLUSION

From the testing that has been done to design and network management Faculty of Engineering, University Muhammadiyah of Malang-based on Software Define Network can be concluded that: (1) SDN-based network has a better performance value than the conventional network. (2) Tested latency delay and Jitter, delay on the network SDN smaller than non-SDN network and still meet the QoS standards G.1010 ITU.T allowable delay value minimum of the 60s. (3) Results of testing the packet loss by the addition of background traffic than 20mb-100mb, both network packet loss does not occur. This is in accordance with the Quality of Services (QoS) standards set by ITU.T G.1010 for packet loss value of 0%. (4) The value of the network throughput on non-SDN higher than the mean value SDN network throughput on non-SDN network is still better and (5) At SDN, networks are centralized control to manage network traffic.

5. ACKNOWLEDGEMENT

The authors wish to thank the Department of Electrical Engineering, Faculty of Engineering, Universitas Muhammadiyah Malang for supporting this study.

6. REFERENCES

[1] K. Ahmed, J. O. Blech, M. A. Gregory, and H. W. Schmidt. (2018). Software defined networks in industrial automation. *J. Sens. Actuator Networks*, *7*(3).

[2] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. (2015). Software-defined networking: A comprehensive survey. *Proc. IEEE*, *103*(1), 14–76.

[3] S. Singh and R. K. Jha. (2017). A Survey on Software Defined Networking: Architecture for Next Generation Network. *J. Netw. Syst. Manag.*, *25*(2), 321–374.

[4] T. Truong, Q. Fu, and C. Lorier. (2016). FlowMap: Improving network management with SDN. *Proc. NOMS 2016 - 2016 IEEE/IFIP Netw. Oper. Manag. Symp.*(Noms, pp. 821–824).

[5] A. El-hassany and L. Vanbever. (2016). SDNRacer : Concurrency Analysis for Software-Defined Networks. in *ACM SIGPLAN conference on Programming Language Design and Implementation, PLDI*, 402–415.

[6] H. Kim and N. Feamster. (2013). Improving network management with software defined networking. *IEEE Commun. Mag.*, *51*(2), 114–119.

[7] M. Afrin and R. Mahmud. (2017). Software Defined Network-based Scalable Resource Discovery for Internet of Things. *EAI Endorsed Trans. Scalable Inf. Syst.*, *4*(14) 1–6.

[8] A. Bradai, K. Singh, T. Ahmed, and T. Rasheed. (2015). Cellular Software Defined Networking: a Framework. *IEEE Commun. Mag. — Commun. Stand. Suppl.*(June, 1–8).

[9] A. Duque-torres, F. Amezquita-su, O. Mauricio, C. Rendon, A. Ord, and W. Y. Campo. (2019). An Approach Based on Knowledge-Defined Networking for Identifying Heavy-Hitter Flows in Data Center Networks. *Appl. Sci.*, *9*(4808), 1–19.

[10] ONF. (2012). Software-Defined Networking: The New Norm for Networks. *Open Netw. Found.*(April), 1–12.

[11] Z. Yao and Z. Yan. (2016). Security in software-defined-networking: A survey. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 10066 LNCS 4, 319–322.

[12] B. A. Linuwih, A. Virgono, and B. Irawan. (2016). Design and Analysis Software Defined Networking For LAN Network: Application and Analysis Routing Method Path Calculating Using Dijkstra's Algorithm. *e-Proceeding Eng.*, *3*(1), 749–756.

[13] F. D. S. Sumadi and D. R. Chandranegara. (2018). Controller Based Proxy for Handling NDP in OpenFlow Network. *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, *4*(1).

[14] J. H. Porras Duque, D. O. Ducuara Beltrán, and G. A. Puerto Leguizamón. (2018). On the features of Software Defined Networking for the QoS provision in data networks. *Inge Cuc*, *14*(2), 106–115.

[15] International Telecommunication Union. (2001). ITU-T Recommendation G. 1010: End-user multimedia QoS categories (Quality of service and performance). *Int. Telecommun. Union*, 1010).

[16] I. Z. Bholebawa and U. D. Dalal. (2018). Performance analysis of SDN/openflow controllers: POX versus floodlight. *Wirel. Pers. Commun.*, *98*(2), 1679–1699.