TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 5, October 2020, pp. 2352~2361
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i5.14851

Distributed gateway-based load balancing in software defined network

Halimah Tussyadiah[1], Ridha Muldina Negara[2], Danu Dwi Sanjoyo[3]
[1,2]Adaptive Network Laboratory, School of Electrical Engineering, Telkom University, Indonesia
[3]Cyber Physical System Laboratory, School of Electrical Engineering, Telkom University, Indonesia

Article Info

Article history:

Received Jun 6, 2019
Revised Apr 11, 2020
Accepted May 1, 2020

Keywords:

Distributed gateway system
Floodlight controller
Load balancing
Quality of service
Software-defined network

ABSTRACT

To achieve an internet with high availability and reliability, needs two or more data paths so the process for sending data can be faster. Load balancing is often plays a significant role for this technique to properly utilized every gateway in the network. This research, implemented load balancing in software defined network architecture using floodlight controller. Evaluation is done by measuring QoS (delay, bit rate, packet rate, packet success rate) while sending various traffics through the network such as UDP flow, VoIP, and DNS. Performance of load balancer is work well, because the results after load balancing is better than before. Which is the value of delay after load balancing is decreased about 30-55% compared to before load balancing, also the values of bit rate, packet rate dan packet success rate after load balancing is increased about 10-30% compared to before load balancing.

Corresponding Author:

Halimah Tussyadiah,
Adaptive Network Laboratory,
School of Electrical Engineering, Telkom University,
1 Telekomunikasi St., Bandung, West Java, 40257, Indonesia.
Email: halimahtussyadiah05@gmail.com

1. INTRODUCTION

University of California, Berkeley and Stanford University developed a concept of centralized system to manage network devices on a network architecture as known as software defined network [1]. The concept of software defined network is by separating between data plane and control plane. The forwarding plane is remains on the network devices and a control plane is in a separate entity called controller [2-5]. By using the controller network administrator is allows to configure a software defined network infrastructure without direct configuration on physical devices. Software defined network has an ability to maximize the utilization of network devices, such as load balancing, traffic engineering, and other programmable stuff [2-5]. The first concept of OpenFlow first developed at Stanford University in 2008. OpenFlow is the first standard communication protocol between control plane and data plane on the SDN network architecture. Version 1.0 of OpenFlow was released in December 2009 [6-10]. OpenFlow is currently managed by Open Networking Foundation (ONF), and user-based organization dedicated to open standards of software defined network [11, 12]. Load balancing is a technique to distribute a traffic on two or more data paths in a balanced way, thus maximize the utility of each gateway [13-15]. Floodlight controller is an enterprise-class OpenFlow controller that is apache-based and Java-based, supported by big switch networks [16-18]. Mininet is an emulator that is used to emulate a network architecture on software defined network environment [19, 20].

In order to increase the network performance, it needs two or more data path to direct the traffic from source-destination. Load balancing is used to maximize the utility of each gateways in a network architecture.

Similiar work has been done, in [21, 22] authors used pox contoller, built the network with bipartite topology with random number of L1 between 1-4 and L2 between 1-8, but the created system had a limitation of recursive function. In [22] the authors used Floodlight Controller, but actually the built-in load balancer in Floodlight controller is used for server load balancing not for gateway load balancing or network redundancy. In this paper, we chose a Floodlight Controller and implemented load balancer to the controller, with the same topology is used, as shown in Figure 1 [23] with random number of gateways connected to switches in L2 (L1) between 2-4 and number of host-connected switches (L2) between 2-8. The evaluation is done by measuring QoS (delay, bit rate, packet rate, packet success rate) while sending various traffics through the network such as UDP flow, VoIP, and DNS.

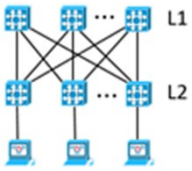

Figure 1. Bipartite topology

2. RESEARCH METHOD

We used programmable switches on mininet that support OpenFlow 1.3 as a standard communication protocol with a connected floodlight controller [24]. The IP address of the network devices in this network are stored in the controller, each switch on the network doesn't have IP address on its own. Hosts on this system can be client, servers, and other end-user devices. IP address is given by the dynamic host configuration protocol (DHCP). DHCP is run by the controller. Switches on this system are work as a relay between the host and the controller. IP Address on this system only has one subnet.

This research, implemented load balancing in software defined network architecture using floodlight controller. In order to run the load balancing, there are two main functions on the system, such as main system and supporting system. The main system is a system that run the main function, namely gateway load balancing, using Djikstra routing algorithm. The type of load balancing used is least-load-per-flow load balancing. Each flow is a combination of IP Address, MAC address and source-destination port [25]. Controller determine which gateway is compatible to forward matching packet from the flow. The chosen gateway is a gateway with the lowest traffic at the time. In order for the main system to run, needs support system that can run functions such as forwarding process, DHCP service, monitoring and proxy ARP.

In this paper, we tested a load balancer function that implemented on Floodlight controller with bipartite topology. The number of gateway connected to switches in L2 (L1) between 2-4 and number of host-connected switches (L2) between 2-8. Several types of traffic, including UDP flow, VoIP, and DNS is sent through the network while delay, bit rate, packet rate and packet success rate of each traffic is measured. For the evaluation we used 2 scenarios as shown in Table 1, such as scenario 1 and scenario 2.

Table 1. Evaluation scenario

	L1	L2	Number of Host Generate Background Traffic	Value of Background Traffic
Scenario 1	2	2 - 8	2;4;6;8;8;8;8	100 Mbit/s
	3	2 - 8	2;4;6;8;8;8;7	100 Mbit/s
	4	2 - 8	2;4;6;8;8;7;7	100 Mbit/s
Scenario 2	2	2 - 8	4	100 Mbit/s
	3	2 - 8	4	100 Mbit/s
	4	2 - 8	4	100 Mbit/s

3. RESULTS AND DISCUSSION

The evaluation is done by measuring delay, bit rate, packet rate and packet success rate in a bipartite topology with different combination of L1 and L2 while sending various types of traffic with two different evaluation scenarios, such as:

3.1. Scenario 1

3.1.1. UDP flows

Figure 2 shown the value of delay decreased compared to results before load balancing. The more number of switches connected to the host, the value of delay is increased due to the more number of host that

generates the background traffic in each topology. Figure 3 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 4 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 5 shown the value of packet success rate. Based on the Figure 5 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

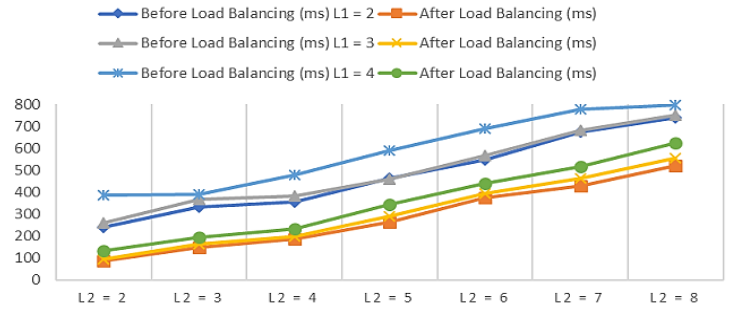

Figure 2. UDP flow delay

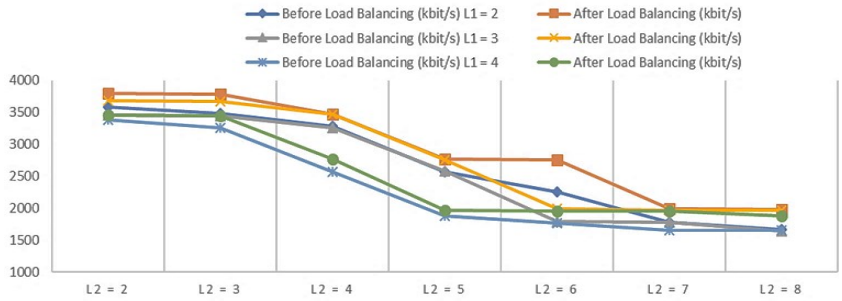

Figure 3. UDP flow bit rate

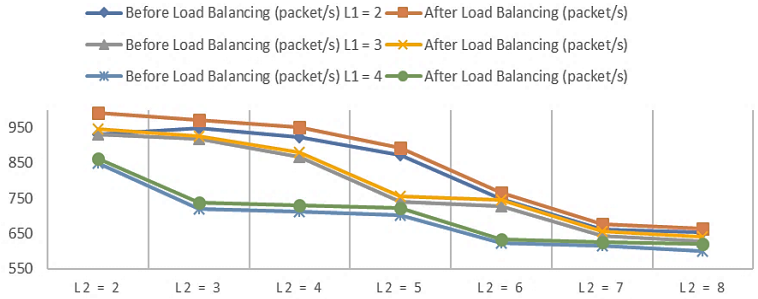

Figure 4. UDP flow packet rate

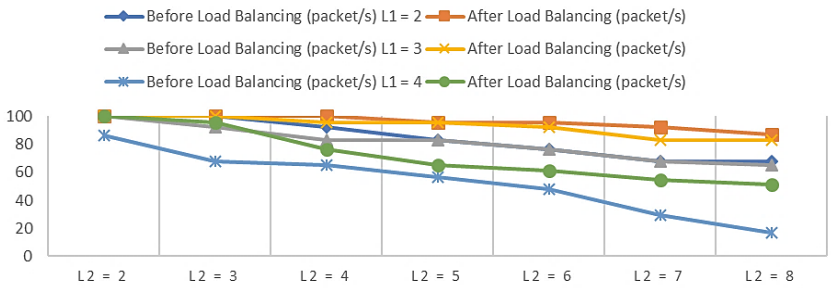

Figure 5. UDP flow packet success rate

3.1.2. VoIP

Figure 6 shown the value of delay decreased compared to results before load balancing. The more number of switches connected to the host, the value of delay is increased due to the more number of host that generates the background traffic in each topology. Figure 7 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 8 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 9 shown the value of packet success rate. Based on the Figure 9 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

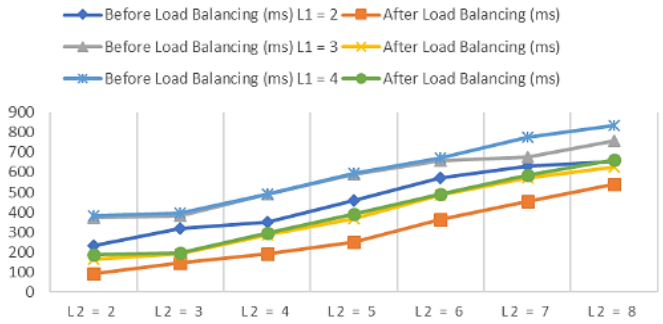

Figure 6. VoIP delay

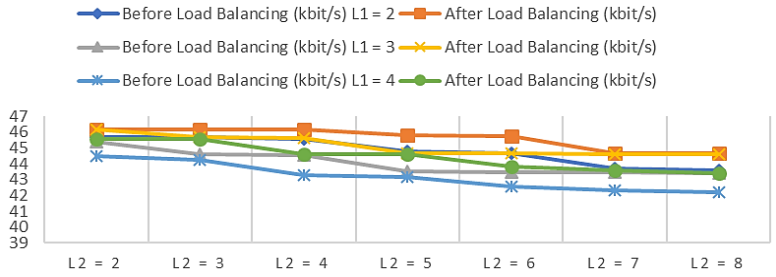

Figure 7. VoIP bit rate

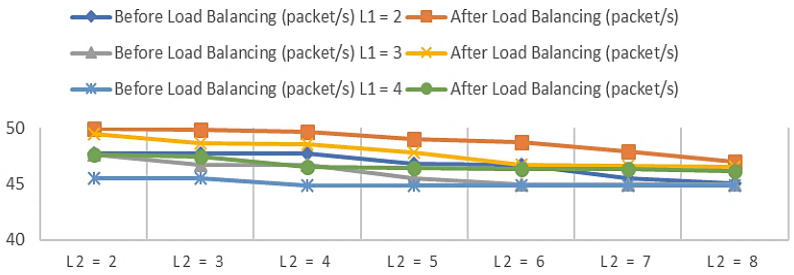

Figure 8. VoIP packet rate

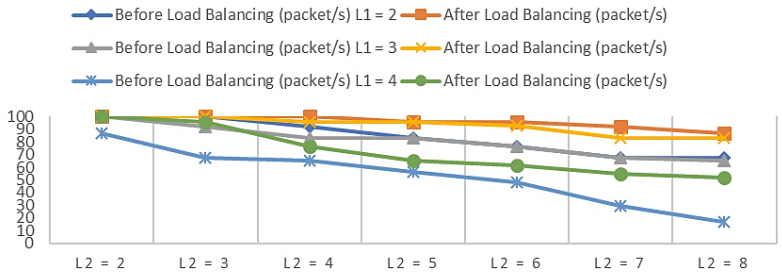

Figure 9. VoIP packet success rate

3.1.3. DNS

Figure 10 shown the value of delay decreased compared to results before load balancing. The more number of switches connected to the host, the value of delay is increased due to the more number of host that generates the background traffic in each topology. Figure 11 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 12 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 13 shown the value of packet success rate. Based on the Figure 13 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

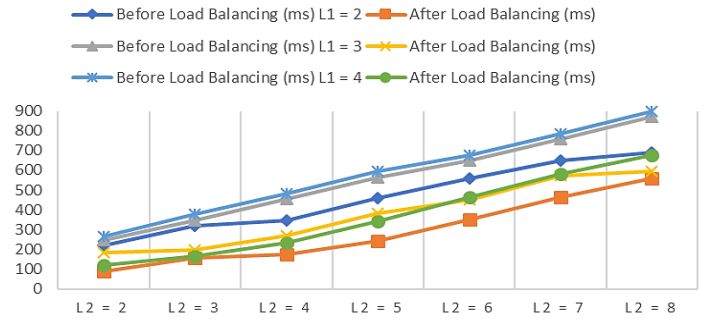

Figure 10. DNS delay

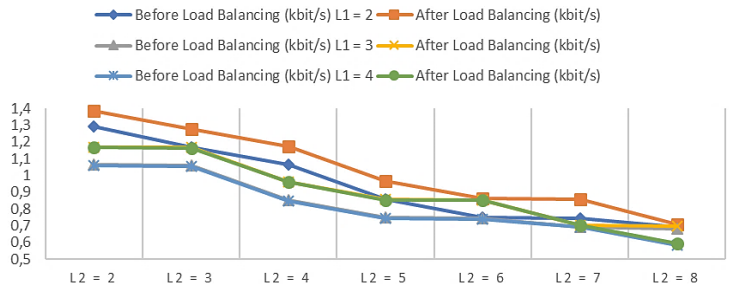

Figure 11. DNS bit rate

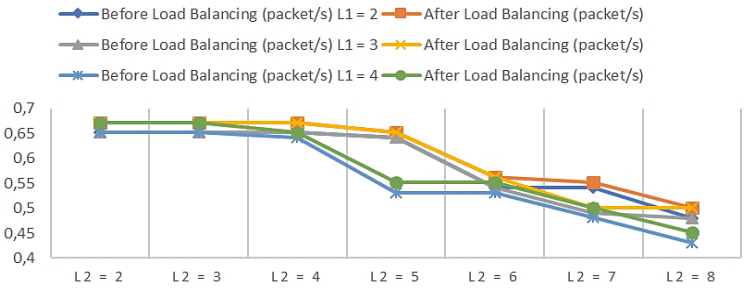

Figure 12. DNS packet rate

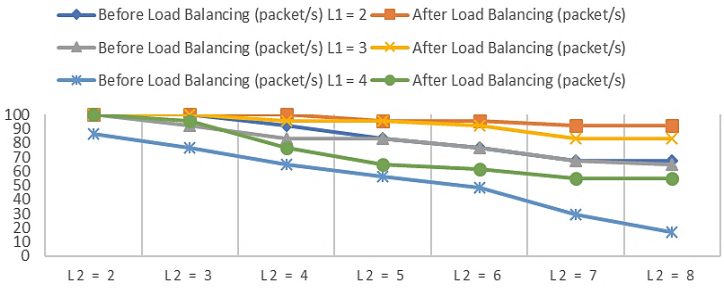

Figure 13. DNS packet success rate

From all results above for scenario 1, we can conclude that Load balancer performance is work well and can handle all types of traffics. The results after load balancing is better than before load balancing. But, adding the number of gateways, doesn't increased the performance of the network. Since the value of delay at L1 = 2 is better than L1 = 3 and L1 = 4.

3.2. Scenario 2

3.2.1. UDP flows

Figure 14 shown the value of delay decreased compared to results before load balancing. Although the number of host that generates background traffic is same, the more number of switches-connected to host increased the value of delay. It cause using the Djikstra routing algorithm, it takes time to find the path and calculate the link cost of each path. Figure 15 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 16 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 17 shown the value of packet success rate. Based on the Figure 17 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

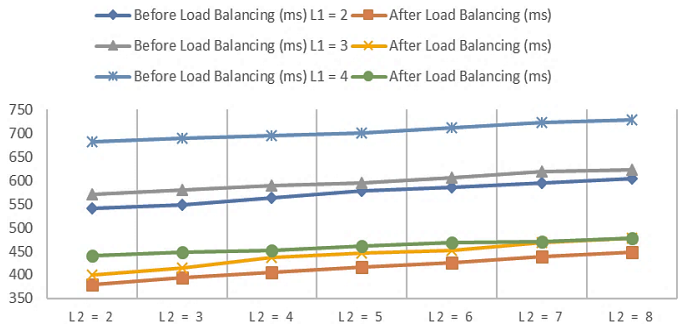

Figure 14. UDP flow delay

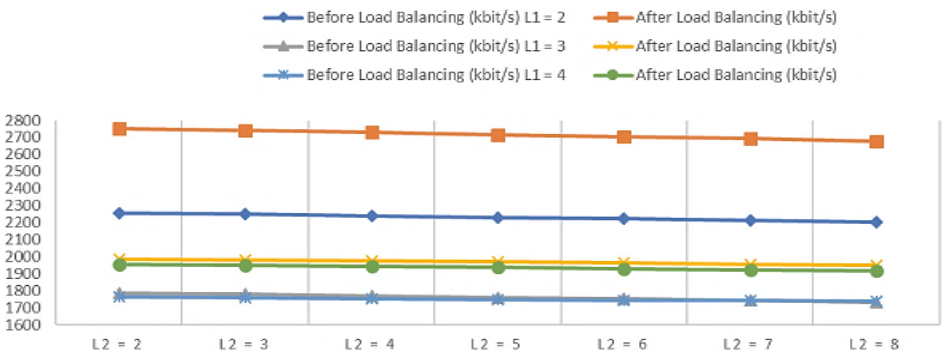

Figure 15. UDP flow bit rate

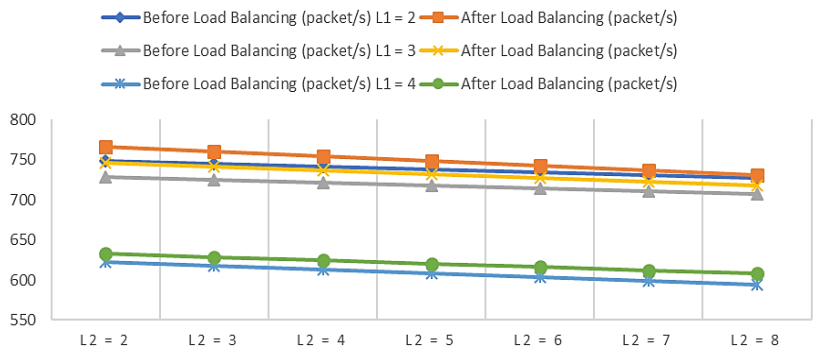

Figure 16. UDP flow packet rate

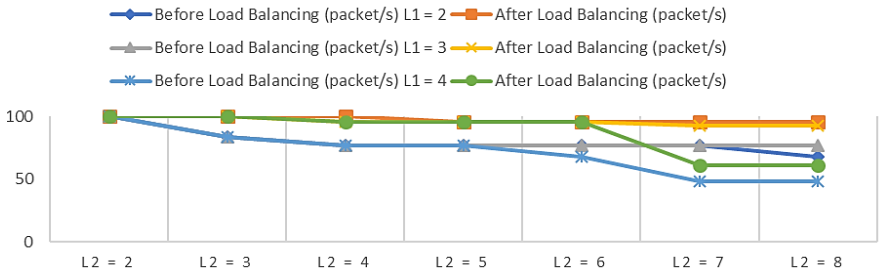

Figure 17. UDP flow packet success rate

3.2.2. VoIP

Figure 18 shown the value of delay decreased compared to results before load balancing. Although the number of host that generates background traffic is same, the more number of switches-connected to host increased the value of delay. It cause using the Djikstra routing algorithm, it takes time to find the path and calculate the link cost of each path. Figure 19 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 20 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 21 shown the value of packet success rate. Based on the Figure 21 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

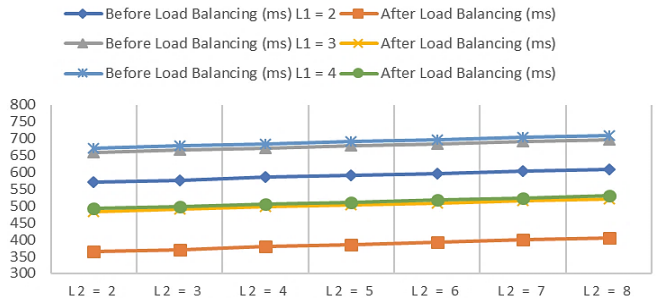

Figure 18. VoIP delay

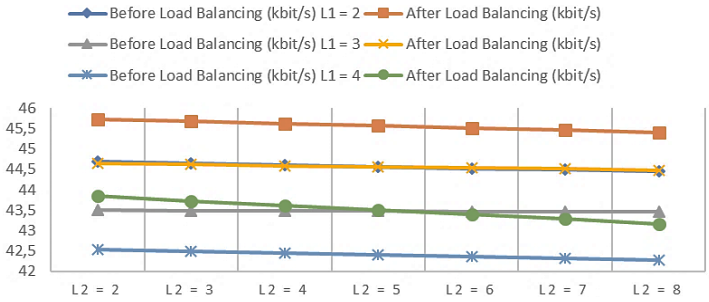

Figure 19. VoIP bit rate

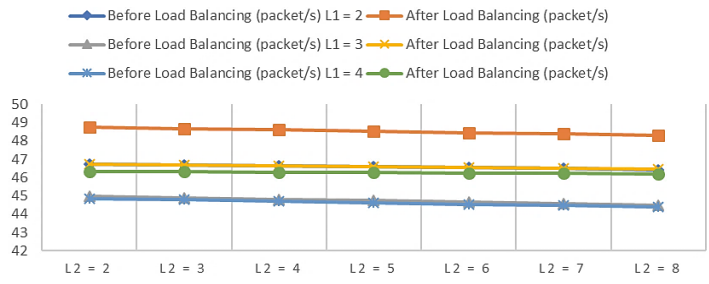

Figure 20. VoIP packet rate

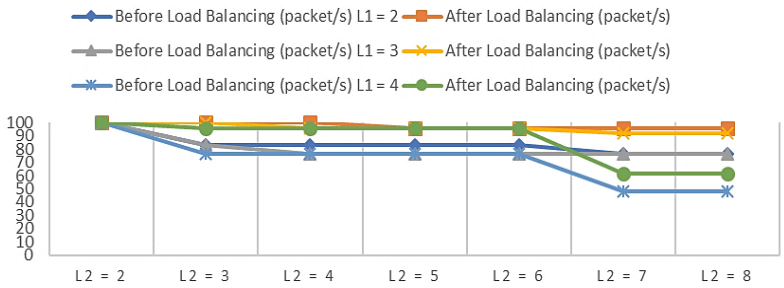

Figure 21. VoIP packet success rate

3.2.3. DNS

Figure 22 shown the value of delay decreased compared to results before load balancing. Although the number of host that generates background traffic is same, the more number of switches-connected to host increased the value of delay. It cause using the Djikstra routing algorithm, it takes time to find the path and calculate the link cost of each path. Figure 23 shown the value of bit rate. The decreased value of delay after load balancing can increased the value of bit rate. Figure 24 shown the value of packet rate. The decreased value of delay after load balancing can increased the value of packet rate. Figure 25 shown the value of packet success rate. Based on the Figure 25 there are obtained a packet loss but the load balancer can reduce the value of packet loss.

From all results from scenario 2, we can conclude that load balancer performance is work well and can handle all types of traffics. The results after load balancing is better than before load balancing. Although the number of the host that generates background traffic is same, the more number of switches connected to the host increased the value of delay because using the djikstra algorithm it takes time to find the path and calculate the link cost of each path. But, adding the number of gateways, doesn't increased the performance of the network. Since the value of delay at L1 = 2 is better than L1 = 3 and L1 = 4.

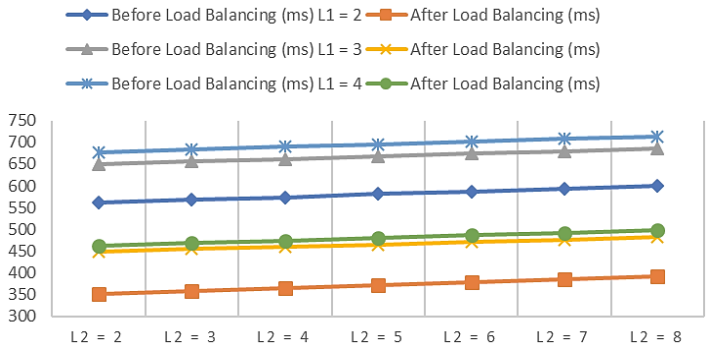

Figure 22. DNS delay

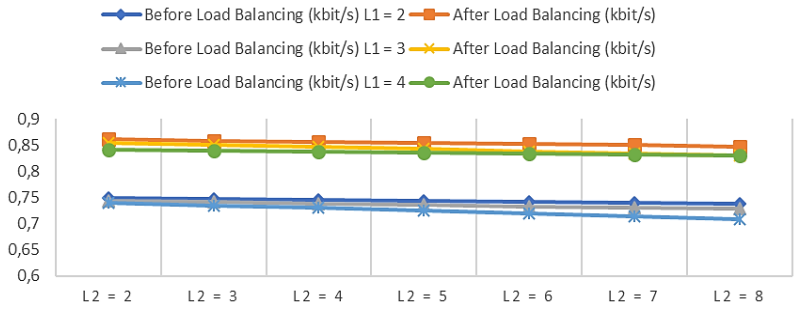

Figure 23. DNS bit rate

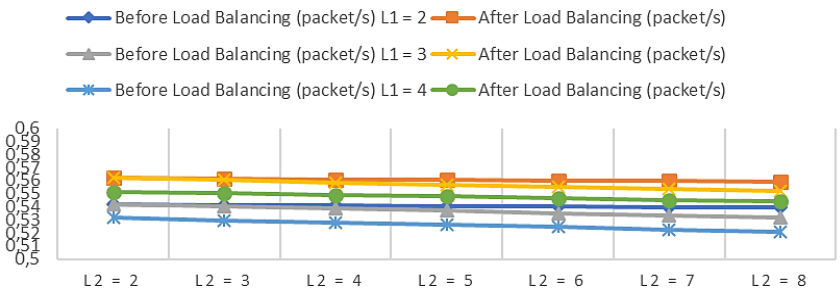

Figure 24. DNS packet rate

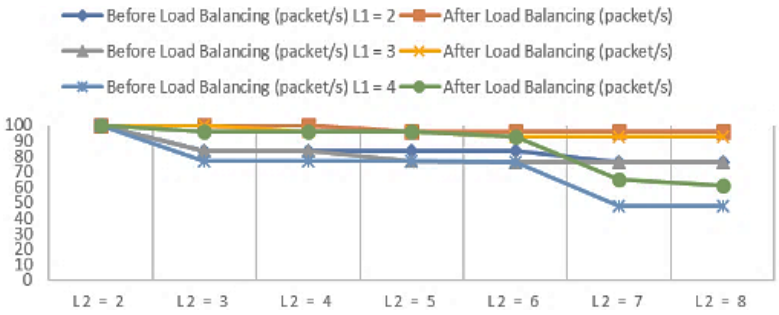

Figure 25. DNS packet success rate

4. CONCLUSION

The purpose of this study is to understand the effects of load balancing on SDN using a floodlight controller. Separating control fields from data fields on SDN creates different architectural settings so that our experiment is to get the best settings. In this paper, we perform two load balancing functions, namely the main system and the support system. The main system is a system that performs the main function, namely Gateway Load Balancing using the Djikstra routing algorithm. The type of load balancing used is load balancing with a minimum load. This study was designed to measure the effects of bipartite topology with two scenarios. The first scenario is adding the number of gateways and the second scenario adding the number of hosts that generate background traffic. We evaluate that increasing the number of gateways to Qos performance on UDP, VoIP and DNS services is not large. We, therefore, conclude that load balancing with SDN can be applied to the maximum even though the number of gateways is added as the number of hosts passes through the network. The results of this analysis revealed that SDN is ready to be applied to large networks but still must pay attention to the QoS requirements of realtime services such as VoIP because the number of gateways makes a slight increase in delay. Further work needs to be done to determine whether there are different effects when we use other algorithms and different topologies using multiple gateways.

REFERENCES

[1] Ramdhani M. F., Naning S., Dirgantara B., "Multipath routing with load balancing and admission control in Software-Defined Networking (SDN)," *2016 4th International Conference on Information and Communication Technology (ICoICT)*, 2016. doi: https://doi.org/10.1109/ICoICT.2015.72314.

[2] H. T. Zaw, A. H. Maw, "Traffic management with elephant flow detection in software defined networks (SDN)," *International Journal of Electrical and Computer Engineering,* vol. 9, no. 4, pp. 3203-3211, August 2019. doi: 10.11591/ijece.v9i4.pp3203-3211.

[3] Madhusanka Liyanage; Andrei Gurtov; Mika Ylianttila, "Software Defined Networking Concepts," in Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture, John Wiley & Sons, Ltd., pp. 21-44, 2015.

[4] Jain R., Paul S. "Network Virtualization and Software Defined Networking for Cloud Computing: A Survey," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24-31, November 2013.

[5] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14-76, Jan 2015.

[6] Open Networking Foundation. (2009). "OpenFlow Switch Specification 1.0.0 (Wire Protocol 0x01)," Current, 0, 1-36, 2013. https://www.opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.0.0.pdf

[7] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner. "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer*

Communication Review, vol. 38, no. 2, pp. 69–74, March 2008. doi: https://doi.org/10.1145/1355734.1355746.
[8] Sivaramakrishnan S. R., Jelena Mikovic, Pravein G. Kannan, Chan Mun Choon, Keith Sklower, "Enabling SDN Experimentation in Network Testbeds," *Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec '17)*, pp. 7-12, 2017. doi: https://doi.org/10.1145/3040992.3040996.
[9] Foukas, Xenofon, Mahesh K. Marina, and Kimon Kontovasilis. "Software defined networking concepts." Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture. Wiley, pp. 21-44, 2015. http://homepages.inf.ed.ac.uk/mmarina/papers/sdn-chapter.pdf
[10] Dabkiewicz, S., Van Der Pol R. and Van Malenstein, G., "OpenFlow network virtualization with FlowVisor," System and Network Engineering, University of Amsterdam's Research Project, vol. ll, pp. 5-6, 2013. [Online]. Available: https://delaat.net/rp/2012-2013/p28/report.pdf.
[11] M. Alsaeedi, M. M. Mohamad, A. A. Al-Roubaiey, "Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey," *IEEE Access*, vol. 7, pp. 107346-107379, 2019.
[12] R. G. Barba, M. Criollo, N. Aimacaña, C. Manosalvas and C. Silva-Cardenas, "QoS Policies to Improve Performance in Academic Campus and SDN Networks," *2018 IEEE 10th Latin-American Conference on Communications (LATINCOM)*, Guadalajara, pp. 1-6, 2018.
[13] Ejaz S. K., "Analysis of the trade-off between performance and energy consumption of existing load balancing algorithms," Technische Universität Dresden Department of Computer Science Chair for Computer Networks, Dresden, 1 November 2011, [online]. Available : https://www.rn.inf.tu-dresden.de/uploads/Studentische_Arbeiten/Belegarbeit_Ejaz_Syed%20Kewaan.pdf
[14] Jiang J., Yahya W., Ananta M. T., "Load Balancing https://www.rn.inf.tu dresden.de/uploads/Studentische_Arbeiten/Belegarbeit_Ejaz_Syed%20Kewaan.pdf and Multicasting Using the Extended Dijkstra's Algorithm in Software Defined Networking," *Frontiers in Artificial Intelligence and Applications*, vol. 274, pp. 2123-2132, 2015.
[15] Katta Rishabh, Kartik Angadi, Karthik Chegu, D. S. Harikrishna, S. Ramya, "Analysis of Load Balancing Algorithm in Software Defined Networking," *2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*, pp. 1-4, 2017.
[16] Wael Hosny Fouad Aly, "Controller Adaptive Load Balancing for SDN Networks," *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, Zagreb, Croatia, pp. 514-519, 2019.
[17] M. M. Soto Cordova, G. Chavez Hidalgo, R. Ñiquen Ortega, "Approach of Performance Analysis for Controllers of Software Defined Networking," *2019 Congreso Internacional de Innovación y Tendencias en Ingenieria (CONIITI)*, BOGOTA, Colombia, pp. 1-6, 2019.
[18] B. A. M. Haidi, T. W. Au, S. H. Shah Newaz, "Single Cluster Load Balancing Using SDN: Performance Comparison between Floodlight and POX," 2019 IEEE 19th International Conference on Communication Technology (ICCT), Xi'an, China, pp. 1332-1336, 2019.
[19] Lantz B., Heller B., McKeown N., "A network in a laptop: rapid prototyping for software-defined networks," *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, HotNets 2010, Monterey, CA, USA, 2010.
[20] Keti F., Askar S., "Emulation of Software Defined Networks Using Mininet in Different Simulation Environments," *2015 6th International Conference on Intelligent Systems, Modelling and Simulation*, pp. 205-210, 2015. https://doi.org/10.1109/ISMS.2015.46.
[21] Naqvi H. A., Hertiana S. N., Negara R. M., "Enabling multipath routing for unicast traffic in Ethernet network," *2015 3rd International Conference on Information and Communication Technology (ICoICT 2015)*, pp. 245-250, 2015. https://doi.org/10.1109/ICoICT.2015.7231430.
[22] R. Tulloh, H. Tussyadiah, R. W. Hutabri, and R. M. Negara, "Load Distribution Analysis On Bipartite Topology Using Floodlight Controller", *Journal of Theoretical and Applied Information Technology,* vol. 96, no. 5, pp. 1238-1252, March 2018.
[23] A. K. Arahunashi, S. Neethu, H. V. Ravish Aradhya, "Performance Analysis of Various SDN Controllers in Mininet Emulator," *2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, Bangalore, India, pp. 752-756, 2019.
[24] ONF, The Architecture of Software-Defined Networks, 2013. Available: https://www.opennetworking.org/wp-content/uploads/2013/05/7-26%20SDN%20Arch%20Glossy.pdf.
[25] V. Deeban Chakravarthy, B. Amutha, "Path based load balancing for data center networks using SDN," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. 3279~3285, August 2019. doi: 10.11591/ijece.v9i4.pp3279-3285.