Software Defined Network based Load Balancing for Network Performance Evaluation

Omran M. A. Alssaheli, Z. Zainal Abidin, N. A. Zakaria, Z. Abal Abas
Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka, 76100, Melaka, Malaysia

***Abstract*—Load balancing distributes incoming network traffic across multiple controllers that improve the availability of the internet for users. The load balancing is responsible to maintain the internet availability to users in 24 hours by 7 days a week. However, the internet become unavailable since the load balancer is inflexibility, costly, and non-programmable for settings adjustment especially in managing the network traffic congestion. An increasing user using mobile devices and cloud facilities, the current load balancer has limitations and demands for the deployment of a Software-Defined Network (SDN). SDN decouples network control, applications, network services, and forwarding roles; hence makes the network more flexible, affordable, and programmable. Furthermore, it has been found that SDN load balancing performs intelligent action, efficient and maintains better QoS (Quality of Service) performance. This study proposes the application of SDN-based Load Balancing since it provides pre-defined servers in the server-farm that receive the arrived Internet Protocol (IP) data packet from various clients in the same number of loads and process orders for each server. Experiments have been conducted using Mininet™ and based on several scenarios (Scenario A, Scenario B, and Scenario C) of network topologies. Parameters used to evaluate the load balancing in SDN are throughput, delay, and jitter. Findings indicated that scenario A gives a high throughput, scenario B and C produce a low jitter values and scenario C produces the lowest delay. The impact of SDN brings a multi-path adaptive direction in finding the best route for a better network performance.**

***Keywords*—Load balancing; software-defined network (SDN); SDN load balancing; network performance; Mininet**

I. Introduction

The network started in the late 1990s with electronic messaging (e-mail), followed by file transfer protocol (FTP) and more network services are used such as access multimedia files (audio or video) or content distribution in TCP/IP architecture [1]. An increasing usage of video streaming of the network user has shown in recent trends. There will be 5.3 billion total users by 2023 [2]. Moreover, the enormous growth of users is due to the use of applications in cloud services, such as Internet of Things (IoT) and Data Science. These applications and network services demand a huge volume of IP traffic transmissions in a high-speed data communication infrastructure.

In the legacy network, load balancing segregates inbound network packets that coming into the network and outbound the packets across the network through multiple controllers. In a simple definition, load balancing is to ensure availability of network services and applications to users for every 24 hours in 7 days. As an analogy, load balancing performed as a traffic policemen standing in the middle of junctions and giving directions for vehicles to take turns to maneuver to avoid traffic jams or accidents.

Moreover, load balancing is responsible to balance an enormous volume of IP traffic across two or more Wide Area Network (WAN) that links without using complex routing protocols such as Border Group Protocol (BGP). The application of the balancing service produces an equilibrium network session over multiple connections in order to spread out the amount of bandwidth used by each Local Area Network (LAN) users, for example, browsing websites and accessing email.

Video streaming and gigantic data transmission from various users to another, demand another mechanism to be integrated with the load balancing in solving problems of delay, packet loss and high bandwidth utilization. Therefore, the objective of this study is to analyze the new mechanism in the load balancing and evaluate the network performance (jitter, delay and throughput) based on scenarios (A, B and C).

The following Section II describes the literature review of load balancing. Section III explains on materials and method used to perform the experiment environment of SDN using scenarios to represent the network topology. Section IV explains about results and findings of the study. Section V is the discussion that presents the challenges of doing the research. Section VI summaries the SDN load balancing and network performance evaluation based on scenarios.

II. Literature Review

LAN and WAN needs an implementation of a load balancing mechanism to competently distribute arriving packet traffic over a collection of server farm. The importance of the load balancing located at servers and routing clients requires a maximum speed, and capacity utilization across servers in order to guarantees that no server is strained that reduce the efficiency [3] of network performance.

Therefore, load balancing manages the increasing number of IP traffic in the internet for a better network performance. In WAN environment, when one of servers is failed, the load balancing device reroutes traffic [4] to available server in the farm. As the additional server is included to the farm, the load balancing instantly sends requisition to the server group for acknowledgment and provides the alternative route for the traffic packet to travel.

Nevertheless, a major issue in the load balancing is the difficulty in managing large network using only a single server [5]. A single server in the topology design produces a bottleneck [6] condition as soon as all data packets queueing at the interface in the same time to get to the destination.

In fact, the load balancing is made of hardware-based-networks controller and requires system maintenance in managing huge IP traffic. On one hand, the manufacturer has hard coded the programmable controls that is easier to install and settings at the load balancer device. On the other hand, the load balancer performs a complicated route [7]–[9], difficult maintenance process, and is time-consuming.

Another issue related to the load balancing is the existing TCP/IP construction framework was not planned to fulfil the need of the large scale of video content distribution [10] and a high number of Internet users. To change the TCP/IP architecture involve enormous amount of expenditure. For a quick solution, a cost-effective approach to overcome the problem of TCP/IP stack is implemented using the software-defined networking (SDN) [5] since it creates another visualization layer for content processing and content distribution.

Load balancing networks are non-programmable [11] since the network administrator unable to perform new settings and the configuration is set by default that is session based load balancing. Fig. 1 shows the load balancing that uses the hardware-based-controller at the control plane for each data plane.

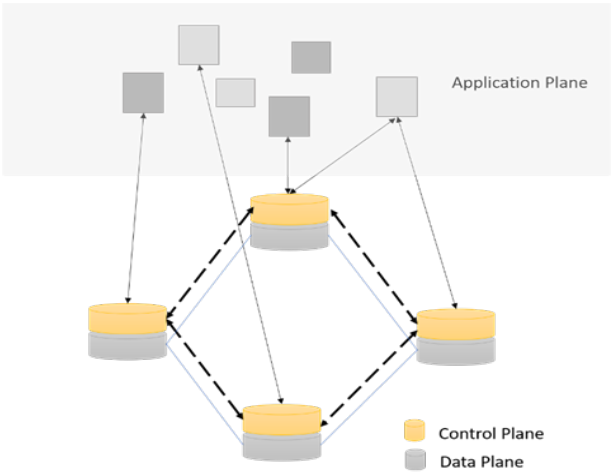

Fig. 1. Load Balancing in Network Architecture.

The configuration and setup are fixed and programmed according to functionalities and services to be provided. The control plane is situated at the data plane that increases the cost of implementation and installation of the networking infrastructure. The load balancing uses several techniques such as round robin, the least connection approach and software-defined networking load balancing.

The round robin approach is capable to forward total traffic packet by responding to DNS requests with a list of IP addresses from the nodes through poor connections and low bandwidth links. Round robin algorithm uses heterogeneous servers [12] with different link of quality, measure the load condition, and variation of security restrictions based on several scenarios of network environment. For example, a set of identical servers is allocated to provide the same services such as centralized management and control; scalable, reliable and monitoring performance. Although each server has its own IP address, it is set up to utilize the same domain name. The DNS server keeps track of all IP addresses linked with Internet domain names. When a request for an Internet domain name and its related IP address is received, the entire addresses are delivered and return in a rotating order. This architecture considers only the distribution of the incoming traffic without considering the server's side [13]. Nonetheless, round robin algorithm predicts that every server has the same capability and resource specifications such as CPU and RAM, to handle equivalent loads.

The least connection approach brings the recent server load into account. The invitation is sent to the server that has served with the lowest number of connections. Each server is assigned to a unique number. If the number of alive connections on two servers is the same, the higher weighted server gets a new request. Every server in a pool is assigned an agent, which announced to the load balancer on its latest load. This real-time data is used to determine the server that should be used to best handle requests [14], which avoid overloading a server through numbers of server connections. However, in measuring the present acquaintances, the server capacity cannot be examined.

A SDN load balancing approach produces an efficient and has a higher speed [15] of network performance. SDN load balancing occupies an important position to solve over-load traffic problem in the network. In fact, the SDN itself is an evolving field in the networking systems and is highly in demand [2]. Google, Facebook, Yahoo, and Microsoft are adopting SDN through open standards development. A SDN produces a flexible, scalable, cost-effective and adaptive features that is ideal for high-bandwidth and complex application in the content distribution. The incorporation of a few low-level features of the network application instead of hardware implementation helps the network administrators to be more effective in managing multiple servers and complex networks.

In addition, SDN allows data to be obtained based on content rather than relying on the hostname and IP address of the device. This is because video or multimedia content is allocated in the cloud platform provider and less concern on the host identification or IP address of the cloud server location. With the combination element from SDN helps load balancing to be able to access data at anywhere and anytime.

SDN is deployed in various networks [16] such as organization and campus networks, data centers and Internet Exchange Points. Moreover, SDN architecture integrates the network control and forwarding function that allow a dynamic and programmable configuration in a cloud-based network monitoring for the new generation of network management. To improve the network performance, SDN offers a simulation platform for a better performance compared to the legacy network management.

The SDN load balancing is more lightweight [17] in terms of CPU utilization and time intervals. SDN load balancing provides features of stability, reliability and scalability [18] during data transmitting and receiving from one server to another. The advantage of SDN load balancing reduces the congestion but increase the speed in data transmission.

In SDN, load balancing performs as an "aware-routing" protocol, which is an essential element that aids availability and scalability, resulting in the shortest possible application response time. Millions of individuals are linked to the internet, resulting in increased web traffic, network congestion, and packet losses, thus, the use of load balancing strategies improves network efficiency. Moreover, the SDN load balancing brings benefits in terms of:

A. Enhance end-to-end Network Quality-of-service (QoS)

SDN load balancing enhances the overall networking system's efficiency and QoS [19],[20]. In terms of latency, reaction time, and network performance, QoS provides a better user experience.

B. Optimize Resource Utilization

Resource usage is critical, and it must be optimized for maximum efficiency. Bandwidth, processor, connection, and memory utilization [21] are all network resources that must be exploited.

C. Decrease Transmission Latency

The term "transmission latency" [22] states time taken for a switch to send information. Latency is affected by numerous aspects of switch performance, including congestion and data packet size. The switch load state is represented by congestion in the link, and SDN accumulates the data packets transferred within a transmission rate and session. Latency is a network performance characteristic that must be less than 100ms in order to maintain a good data delivery.

D. Minimize Response Time

Response time means time intermission of a server demands and transmit information are achieved [23]. Thus, the load balancing algorithm uses a distributed SDN network to minimize the response time.

E. Avoiding Bottlenecks

Network congestion creates bottleneck at the load balancer [24]. To avoid bottleneck, the SDN load balancing is configured to avoid the switch or controller to get overloaded. Configuration options that are optimized reduce resource use while increasing efficiency, scalability, and response time. The network performance is more effective, there is failover prevention and reduce bottlenecks.

F. Maximize the throughput

The SDN load balancing maximize throughput [25] during data transmission. A high throughput is vital for a good network performance since how much data could be delivered within a conversation, which traffic packets are distributed evenly to many nodes and in various types of platforms. The size of traffic packets is delivered in the same kilobytes over a period of time and from one node to another.

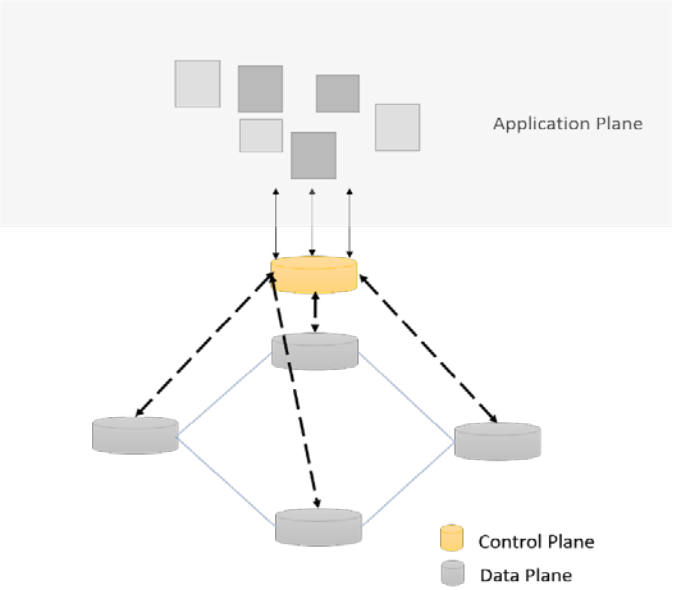

Fig. 2. SDN Load Balancing in Network Architecture.

Fig. 2 illustrates the SDN load balancing in the network architecture. The use of controller has been minimized and cost of installation is reduced. The data plane and the control plane are separated at a different location and space. The purpose is to allow users who use a huge multimedia data to access information from various locations regardless the operating system platform and the network architecture in a small expenditure.

The performance of SDN load balancing is based on parameters, which are throughput, delay and jitter.

Throughput provides information on the successful data packet travelling from one start point to the end point in a provided period. Packet arrival at the destination node in a fast speed is the key indicator for the best service performance in a network. Using throughput, the problem of packet loss is investigated to find the root cause of the problem. Load balancing maximizes throughput by decreasing response time intervals and decreasing data traffic jams.

Based on Equation 1, Mathematical calculation for the throughput can be calculated:

$$\text{Throughput, } \tilde{y}(N) = c\left(1 - \frac{1}{1+\left(\frac{1+\beta}{1-\beta}\right)N}\right) \quad (1)$$

Delay is the time taken for a message or data packets to travel from the source of destination, arrive at the destination node and get back to the source of destination. This is called as the round-trip time (RTT) of the network. Based on Equation 2, Delay is calculated:

$$Pr(E = x)$$
$$\begin{Bmatrix} \sum_{i=0}^{\infty} f_i(a), \sum_{i=0}^{\infty} f_i(b), & x = 0 \\ \sum_{i=0}^{\infty} f_i(a), f_{2x+i}(b) + \sum_{i=0}^{\infty} f_i(b), f_{2x+i}(a), & x > 0. \end{Bmatrix} \quad (2)$$

Jitter is defined as a variation in the delay of data packets flowing through the network from one node to another node. The jitter situation occurred when data packets do not reach

the destination node in the same arrangement as they were sent. The inconsistency of delay values during data transmission determines the quality of network services provided to Internet users. An outcome of the jitter is a higher a jitter value, a higher change of the delay and packet loss happened in the network. Based on Equation 3. Jitter is calculated:

$$\text{Jitter, } \sigma_c = lim_{n=\infty} \sqrt{\frac{1}{N}} \sum_{n=1}^{N} (\Delta T_m)^2 \quad (3)$$

Thus, in this paper, the implementation of SDN load balancing with multiple servers is proposed for load balancing since it helps to prevent the bottleneck problem and reduces the data packet congestion in the network.

III. Materials and Method

The method of network configuration in this study is implemented according to simulation-based-experiments using Mininet software [26] in a simple LAN [27] environment. Multiple servers are configured since a single server creates congestion in the network.

The performance measurement and monitoring response are analyzed through different scenarios of load balancing using network controller software tool. There are 3 scenarios implemented, which are scenario A, scenario B, and scenario C. Every scenario consists of SDN Controller, switches, and numbers of clients' connection, which is important for the evaluation. The servers act as a server pool connected to the SDN switch controller and as a data packet reaches at the SDN switch controller, the next selected server appears in the list of all servers on the network system. As a result, every server in the database handles orders with the same number of loads.

A. Scenario A

In this scenario, the topology is based on a simple design, which consists of four clients and two servers. The controller was created using an OpenFlow controller. Fig. 3 illustrates the topology consists of a switch controller, two servers and four hosts.

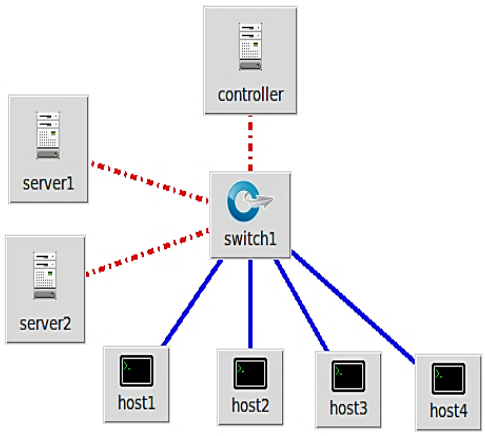

Fig. 3. Topology of Scenario A.

The load is utilizing all of the CPU assets on both web servers. Assume the load on those servers gets to be overwhelming and the Internet location execution diminishes drastically, fair basically includes a third server to the cluster giving extra assets. The arrange activity would at that point be disseminated over three servers as restricted to two.

B. Scenario B

The second scenario consists of four clients, and there are four server pools connected to the switch controller, which is the OpenFlow controller. Each host is located at a dedicated server, which helps to increase the outputs performance. The network performance is measured and monitored based on the delay, throughput and jitter values. Expected outcome of this experiment is to produce a lower delay, better throughput and lower jitter. Fig. 4 illustrates the scenario B experiment setup.

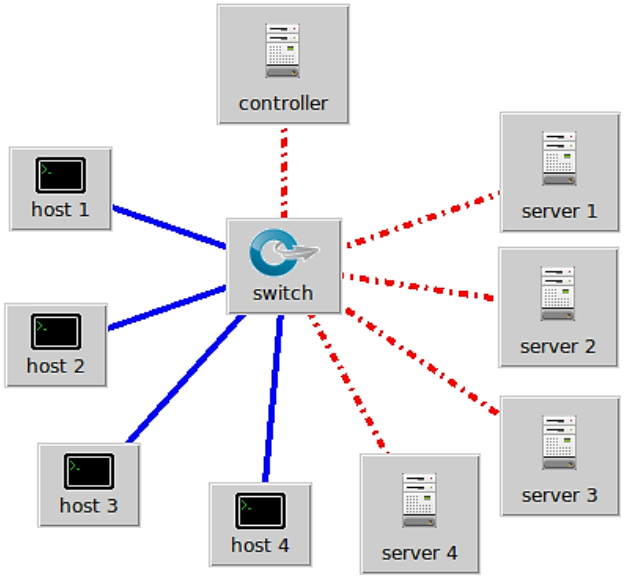

Fig. 4. Topology of Scenario B.

C. Scenario C

The third topology is scenario C consists of one switch controller connected to the four server pools, which support eight clients. The design is similar to the previous two topologies, but there is an additional number of clients. The load balancers lag in these design topologies where the response time and latency increase the load balancers. The SDN controller and the servers remain the same as in Scenario B, and the number of hosts is doubled into 8 hosts. Fig. 5 illustrates Network C.

For all scenarios, the OpenFlow controller is configured using the IP address of 10.0.1.1 and connected to the switch through servers with IP addresses of 10.0.0.1 and 10.0.0.2. The port number of 6633 is used for listening the network packet. The data is captured and tabulated as shown in Table I, Table II and Table III at Section IV.

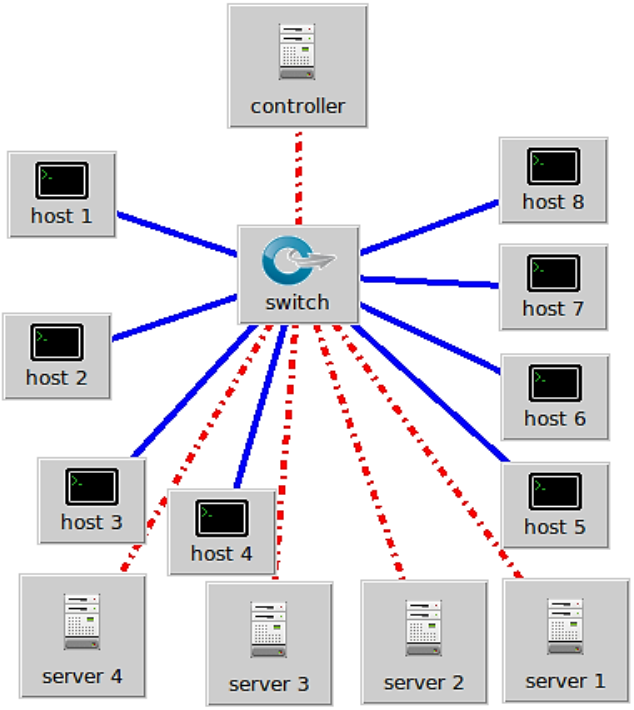

Fig. 5. Topology of Scenario C.

IV. Results and Findings

The network performance is analyzed according to metrics, namely throughput, delay and jitter for every scenario. Comparison is done using simulation and outcomes are evaluated in scenarios based on the number of clients and servers. The result shows that the SDN load balancing is measured in traffic packet of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic.

A. Throughput

SDN load balancing used a pox controller load balancing connected to number of hosts. Table I illustrates the result of throughput based on scenarios obtained from the experiment conducted.

TABLE I. Throughput for Every Second

Number of Request	Scenario		
	A	*B*	*C*
10	41,826	2016	1253
20	16,349	1324	4302
30	17,578	2255	5649
40	33,072	1901	4339
50	52,576	2235	6355
60	55,906	4857	4544
70	72,538	3991	4348
80	66,886	4768	5580
90	69,595	2990	6082
100	76,580	3839	5591

Based on the number of traffic packet requested and transmitted by the server, Scenario A shows larger amount of throughput as the number of traffic request increased. Meanwhile, the Scenario B and Scenario C illustrates an almost same range of values of throughput. Fig. 6 illustrates the throughput values in a form of graph for Scenario A, Scenario B and Scenario C for a better visualization and comparison.

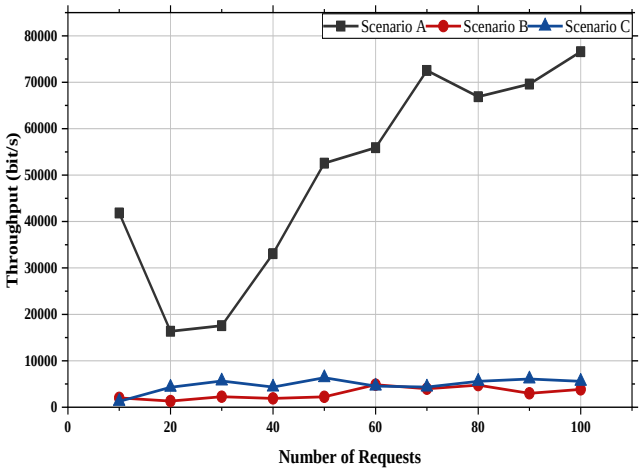

Fig. 6. Throughput Performance based on Scenarios.

Scenario A has given the highest throughput among other scenarios. Even though Scenario A only has two servers but with the SDN based switch controller helps the data transmission to be in effectively and efficiently delivered. Scenario B and Scenario C demonstrate a low throughput value. This is because data packets lost during transmission will cause poor or slow network performance, while poor performance indicates problems such as packet loss. Using performance to measure network speed is beneficial for troubleshooting because it can root out the exact cause of network slowdowns and alert administrators to packet loss-related issues.

B. Delay

The purpose of the SDN load balancing method is to reduce network lag and improve link load balancing by optimizing route calculation and multipath scheduling. Table II provides the delay values for Scenario A, Scenario B, and Scenario C.

Based on Table II, Scenario B indicates a highest delay even though a dedicated server has been provided for each node. Another finding shows that Scenario C shows a higher delay than Scenario A but changed after the number of requests increased at value of 20. After a peak at number of requests of 30, Scenario A keeps showing a constant delay. Fig. 7 presents the delay in a form of representative graph.

Based on results from Table II, Scenario C shows a continuous delay in time taken to transmit the packet. Scenario B shows imbalance condition of delay as the network is monitored to be slow in performance even though a dedicated server is provided to each node, which the concept of single server is not applicable using SDN controller. As a result,

Scenario C performs a slow in speed of data transmission in the network performance, Scenario B consists of the highest delay due to type of data packets travels depending on the application that the user used. Scenario A shows unstable condition at the beginning but then the delay is increasing as the number of packets are requested by clients.

TABLE II. DELAY FOR EVERY SECOND

Number of Request	Scenario		
	A	*B*	*C*
10	10,160	21,383	31,273
20	12,738	57,435	18,223
30	71,322	51,088	20,816
40	49,251	83,701	35,595
50	41,975	88,362	30,841
60	44,300	47,294	52,105
70	41,278	69,933	58,059
80	51,581	69,052	57,458
90	56,749	125,331	58,777
100	60,214	104,335	72,069

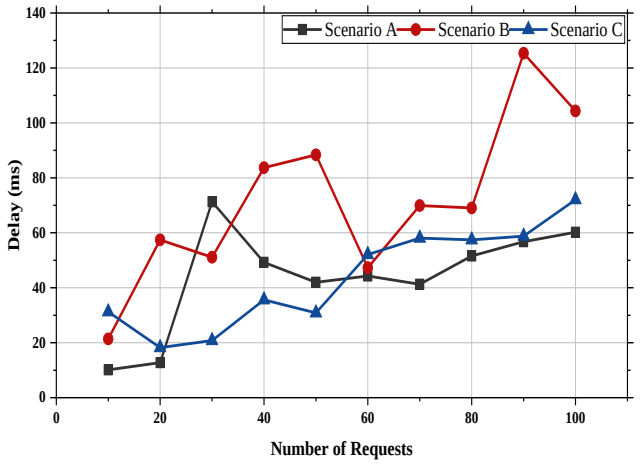

Fig. 7. Delay Performance based on Scenarios.

C. Jitter

Using SDN load balancing, jitter is measured to find the variation of delay in arriving packets to destinations. The variation of jitter represents the pattern or behavior when there is a route changed or congestion.

Table III shows the value of jitter obtained from the Scenario A, Scenario B and Scenario C. Based on the jitter values, Scenario B and Scenario C produce almost the same pattern of jitter since both scenarios using four servers with different number of clients. Finding shows that the jitter produces the similar variation of delay of the server that is designed with or without a dedicated server to a particular client.

Fig. 8 shows Scenario A has given an inconsistency condition compared to Scenario B and Scenario C. Scenario A demonstrates that the topology A built a large data packet queue that causes a huge delay and bursts of jitter. Finding found that Scenario A has a higher jitter than Scenario B and Scenario C because it processes the packet request more to transmit to the destination. Thus, it disrupts in the idle condition of transmitting data packets. The jitter is represented in a variance in time delay in milliseconds for data packets over a network.

Scenario B and Scenario C gives a reliable and scalable network performance. The jitter condition is noticeable as the graph pattern shows an almost constant variation form. For example, a user using Cisco WebEx or Microsoft Teams for online meeting has been several times disconnected. Not only that, user cannot hear the voice of another user or the voice quality has been distorted. Therefore, the longer the data packet arrives, the greater the negative impact of jitter on video and audio quality.

TABLE III. JITTER FOR EVERY SECOND

Number of Request	Scenario		
	A	*B*	*C*
10	1.127	0.998	1.000
20	2.237	0.986	1.006
30	0.001	1.008	1.008
40	0.007	1.013	1.002
50	5.481	1.001	1.016
60	6.852	1.021	1.001
70	0.032	1.004	1.034
80	4.440	1.019	1.009
90	2.307	1.011	1.017
100	1.006	1.007	1.029

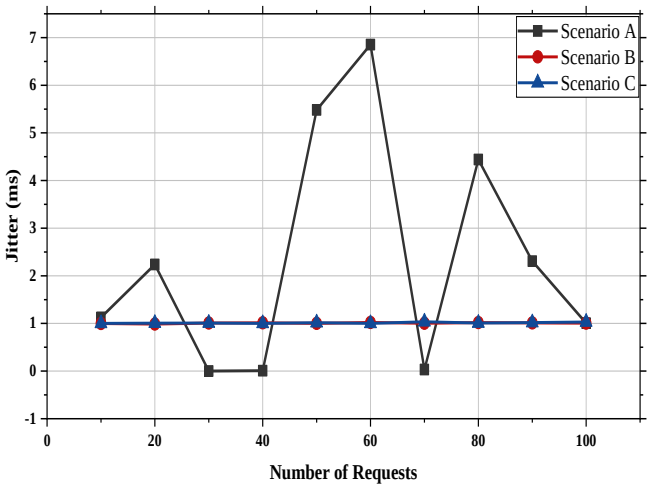

Fig. 8. Jitter Performance based on Scenarios.

V. Discussion

The outcome of this study showed that the application of SDN controller in load balancing brings a better scheduling mechanism, regardless of the number of clients requesting the packet to reach the destination. In addition, the number of servers deployed does not directly affect the overall network performance, especially in the case of network congestion and delays. Metrics are used to measure the SDN controller based load balancing in the network are throughput, delay and jitter.

According to the findings of this study, a SDN load balancer requires evidence regarding on the network's throughput. A start node delivers traffic to an end node to measure traffic on the path, and each packet is time-stamped. The packets are routed across the network's switches without bias decision, and the receiver determines the entire end-to-end delay of a packet based on the time-stamp provided. Similarly, to path measurement, each SDN switch stamps each packet with a time-stamp, which the receiver uses to calculate the transmission delay for each link in the path.

Another significant finding is that the excessive delay is caused by the SDN switch's relatively poor CPU throughput. A delay of 30 milliseconds or more, on the other hand, produce network distortion. The jitter must be less than 30 milliseconds for the video transmission to perform properly. Higher receiving jitter slows the network performance; produce a packet loss and audio quality concerns. The SDN controller behaves as expected, for instance, the response time tends to reduce as we add additional servers. When we compare a server farm with two servers to one with three servers, the response time is cut in half, but the result remains nearly constant as the number of servers increases.

VI. Conclusion

Load balancing is crucial for ensuring the availability of network to users. Nonetheless, a high demand of Internet services from users creates an adoption of load balancing. The state-of-the-art indicates that SDN load balancing is highly recommended due to features such as lightweight, stability, reliability and scalability. SDN load balancing increases user interaction by enabling the administrator to monitor the condition of the servers, check the status of the balancer through log files, and set to disable mode for the balancing features at the specific faulty servers.

The SDN controller has been analyzed in the load balancing method for network performance evaluation, which is based on three simulated scenarios. Scenarios A, B, and C each used a number of servers and multiple clients connected to the servers to produce a different outcome, which is based on the delay, jitter and throughput. The network administrator is able to modify the SDN controller settings in a less expensive and more user-friendly way for a better network performance.

Further investigation on algorithms based on the field of artificial intelligence are needed to reduce the complexity of implementing SDN load balancing in enabling autonomous scheduling and intelligent routing functions.

Acknowledgment

Thank you Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka.

References

[1] T. Guesmi, A. Kalghoum, B.M. Alshammari, H. Alsaif, A. Alzamil, “Leveraging Software-Defined Networking Approach for Future Information-Centric Networking Enhancement,” Symmetry, vol. 13, no. 441, pp. 1-19, 2021.

[2] “Cisco Annual Internet Report (2018–2023) White Paper,” 2020. [Online]. Available: www.cisco.com. [Accessed September 2, 2021].

[3] G. Singh and K. Kaur, “An improved weighted least connection scheduling algorithm for load balancing in web cluster systems,” International. Research. Journal. Engineering. Technology, vol. 5, pp. 1950–1955, 2018.

[4] C.-T. Yang, S.-T. Chen, J.-C. Liu, Y.-W. Su, D. Puthal, and R. Ranjan, “A predictive load balancing technique for software defined networked cloud services,” Computing, vol. 101, no. 3, pp. 211–235, 2019.

[5] T. Semong, T. Maupong, S. Anokye, K. Kehulakae, S. Dimakatso, G. Boipelo and S. Sarefo, “Intelligent load balancing techniques in software defined networks: A Survey,” Electronics, vol. 9 no.7, pp. 1-24, 2020.

[6] D. Shen, W. Yan, Y. Peng, Y. Fu and Q. Deng, “Congestion Control and Traffic Scheduling for Collaborative Crowdsourcing in SDN Enabled Mobile Wireless Networks,” Wireless Communications and Mobile Computing, article no. 9821946, pp. 1-12, 2018.

[7] Q. Youssef, M. Yassine and A. Haqiq, “Secure Software Defined Networks Controller Storage using Intel Software Guard Extensions,” International Journal of Advanced Computer Science and Applications, vol. 11, no. 10, pp. 475-481, 2020.

[8] H. Polat and O. Polat, “An Intelligent Software Defined Networking Controller Component to Detect and Mitigate Denial of Service Attacks,” Journal of Information and Communication Technology, vol. 20, no. 1, pp. 57-81, 2021.

[9] H. Babbar, S. Rani, D. Gupta, H.M. Aljahdali, A. Singh and F. Al-Turjman, "Load Balancing Algorithm on the Immense Scale of Internet of Things in SDN for Smart Cities,” Sustainability, vol. 13, no. 9587, pp. 1-16, 2021.

[10] N. Fotiou, “Information-Centric Networking (ICN),” Future Internet, vol. 12, no. 35, pp. 1-2, 2020.

[11] I. Alam, K. Sharif, F. Li, Z. Latif, M.M. Karim, S. Biswas, B. Nour and Y. Wang, “A Survey of Network Virtualization Techniques for Internet of Things using SDN and NFV,” ACM Computing Surveys, vol. 53, no. 2, Article 35, pp. 1-40, 2020.

[12] M. Alotaibi and A. Nayak, “Linking handover delay to load balancing in SDN-based heterogeneous networks,” Computer Communications, vol. 173, pp. 170-182, 2021.

[13] O. Hohlfeld, J. Kempf, M. Reisslein, S. Schmid, and N. Shah, “Scalability Issues and Solutions for Software Defined Networks,” IEEE Journal on Selected Areas in Communications, vol. 36, no. 12, pp. 2595-2602, December 2018.

[14] B. Alankar, G. Sharma, H. Kaur, R. Valverde and V. Chang, “Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud,” Sensors, vol. 20, no. 7342, pp. 1-26, 2020.

[15] M. Hasan Al Bowarab, N. A. Zakaria, Z. Zainal Abidin, “Load Balancing Algorithms in Software Defined Network,” International Journal of Technology and Engineering, vol. 7, no. 6S5, pp. 686-693, 2019.

[16] D. O'Briain, D. Denieffe, D. Okello and Y. Kavanagh, “Enabling models of Internet eXchange Points for developing contexts,” Development Engineering, vol. 5, pp. 1-8, 2020.

[17] H. Mokhtar, X. Di, Y. Zhou, A. Hassan, Z. May and S. Musa, “Multiple-level threshold load balancing in distributed SDN controllers,” Computer Networks, vol. 198, pp. 108369, 2021. https://doi.org/10.1016/j.comnet.2021.108369.

[18] S. Liang, W. Jiang, F. Zhao and F. Zhao. "Load Balancing Algorithm of Controller Based on SDN Architecture under Machine Learning," Journal of Systems Science and Information, vol. 8, no. 6, pp. 578-588, 2020. https://doi.org/10.21078/JSSI-2020-578-11.

[19] K. Tolga Bagci and A. Murat Tekalp "SDN-enabled distributed open exchange: Dynamic QoS-path optimization in multi-operator services," Computer Networks, vol. 162, pp. 1–10, 2019. https://doi.org/10.1016/j.comnet.2019.07.001.

[20] S. Khan, F.K. Hussain and O.K. Hussain, "Guaranteeing end-to-end QoS provisioning in SOA based SDN architecture: A survey and Open Issues," Future Generation Computer Systems, vol. 119, pp. 176–187, 2021.

[21] H. Xue, K.T. Kim and H.Y. Youn, "Dynamic Load Balancing of Software-Defined Networking Based on Genetic-Ant Colony Optimization," sensors, vol. 19, no. 2, pp. 1-17, 2019.

[22] M.R. Belgaum, S. Musa, M.M. Alam, M.M. Suud, "A Systematic Review of Load Balancing Techniques in Software-Defined Networking," IEEE Access, vol. 8, pp. 98612-98636, 2020.

[23] S. Jamali, A. Badirzadeh, M.S. Siapoush, "On the use of the genetic programming for balanced load distribution in software-defined networks," Digital Communications and Networks, vol. 5, no. 4, pp. 288–296, 2019.

[24] X. Shi, Y. Li, H. Xie, T. Yang, L. Zhang, P. Liu, H. Zhang, Z. Liang, "An OpenFlow-Based load balancing strategy in SDN," Computers Materials and Continua, vol. 61, no. 3, pp. 385–398, 2019.

[25] K. Ramya, M. Sayeekumar, G. Karthik, "Software defined networking based solution in load balancing for media transfer in overlay network," Journal of Computational and Theoretical Nanoscience, vol. 17, no. 1, pp. 43–47, 2020.

[26] S. Prabakaran and R. Ramar, "Software Defined Network: Load Balancing Algorithm Design and Analysis," The International Arab Journal of Information Technology, vol. 18, no. 3, pp. 312-318, May 2021.

[27] O. M. A. Alssaheli, Z. Zainal Abidin, N.A. Zakaria, "Mininet Network Emulator: A Review," International Journal of Computer Science and Network Security, vol. 19, no 9, pp. 147-155, 2019.