Bulletin of Electrical Engineering and Informatics
Vol. 9, No. 5, October 2020, pp. 2045~2054
ISSN: 2302-9285, DOI: 10.11591/eei.v9i5.1928

Traffic management inside software-defined data centre networking

Tariq Emad Ali[1], Ameer Hussein Morad[2], Mohammed A. Abdala[3]
[1,2]Department of Information and Communication Engineering, Al-Khwarizmi College of Engineering, University of Baghdad, Iraq
[3]Department of Medical Instruments Techniques Engineering, Al-Hussain University College, Iraq

Article Info

Article history:

Received Jun 12, 2019
Revised Sep 26, 2019
Accepted Dec 21, 2019

Keywords:

LB
ODL-CO
OVS
SDN-CO
SDN-DC

ABSTRACT

In recent years, data centre (DC) networks have improved their rapid exchanging abilities. Software-defined networking (SDN) is presented to alternate the impression of conventional networks by segregating the control plane from the SDN data plane. The SDN presented overcomes the limitations of traditional DC networks caused by the rapidly incrementing amounts of apps, websites, data storage needs, etc. Software-defined networking data centres (SDN-DC), based on the open-flow (OF) protocol, are used to achieve superior behaviour for executing traffic load-balancing (LB) jobs. The LB function divides the traffic-flow demands between the end devices to avoid links congestion. In short, SDN is proposed to manage more operative configurations, efficient enhancements and further elasticity to handle massive network schemes. In this paper the opendaylight controller (ODL-CO) with new version OF 1.4 protocol and the ant colony optimization algorithm is proposed to test the performance of the LB function using IPv6 in a SDN-DC network by studying the throughput, data transfer, bandwidth and average delay performance of the networking parameters before and after use of the LB algorithm. As a result, after applying the LB, the throughput, data transfer and bandwidth performance increased, while the average delay decreased.

Corresponding Author:

Tariq Emad Ali,
Department of Information and Communication Engineering,
Al-Khwarizmi College of Engineering,
University of Baghdad, Iraq.
Email: tariqemad@gmail.com

1. INTRODUCTION

Currently, the best professional enterprise companies, irrespective of size, consider internet networking using their own data centres (DCs) to be mandatory if they are going to compete effectively [1]. The new social and mobile media as well as cloud computing are exposing the restriction of classical networks. They are useful in virtualization and automation, but by using traditional DCs these options are constrained. Operators may spend a lot of time (hours, days or even weeks) to make a manual network configuration inside each of the networking devices individually [2, 3]. Software-defined networking data centres (SDN-DCs) support an elastic way to manage network functionality. The objective of SDN-DCs is to qualify the exchanging business necessities via a centralized control console. In SDN, the primary methodology is to design, build, and manage networks by segregating the control plane, which is the brain of the network, and forwarding the data plane [4]. This decoupling assists the DC network controls allowing them to be programmable [5]. SDN interacts with various collections of structural designs listed below:

- Software-defined mobile networking architectures (SDMN): SDMN or Full-SDMN is one of the favourable technologies that help to resolve the restrictions that are present in mobile networks. The SDMN structural design is constructed from SDN and network function virtualization (NFV), which is called also (SDN/NFV). Examples of SDMN application such as IOT and 5G [6, 7].
- Software-defined Wide Area Network architectures (SD-WAN): SD-WAN is an SDN technology used with WAN networks. It links enterprise networks such as branch sites and DC over wide geographical spaces [8, 9].
- Software-defined Local Area Network architectures (SD-LAN): SD-LAN or also called software-defined wireless Local Area Network (SDWLAN), provides wireless reachability to produce an overall-policy structural design. SDWLAN are constructed from SDN and SD-WAN concepts [10].

SDN-DC infrastructure consists of the SDN Controller (SDN-CO), which is the controller version of the SDN, and APIs (southbound and northbound). The SDN-CO provides a centralized and programmable console that can achieve zero-touch provisioning (ZTP) and other options depending on the business's requirements. SDN-DC provides a programmable console, nimbleness, and renewal that reduces capital and operating expenses (CapEx and OpEx), and centralizing management. SDN assists the enterprise company to swiftly develop new services, applications, and infrastructure that meet the exchanging business's necessities.

In 2011, R. Wang, D. Butnariu, and J. Rexford [11] proposed facilities that depend on LB to totally operate the simulated servers. Their results show that the method can adjust to modifications of the traffic sharing with the least effect on throughput. In 2014, S. Kim and H. Y. Choi [12] proposed PMIPv6 using IP tunnelling. They used the OF Proxy Mobile IPv6 (PMIPv6). Their result shows that the OF based PMIPv6 support more elastic utilization structural design.

In 2015, Y. Kyung and K. Hong [13] proposed an LB structure in addition to several controllers (CO) in SDN using IPv6. Once the capability of a CO up to the sill, the CO allows the entering packet to travel to other CO to avoid them from a block. They confirmed that their method has a lower blocking possibility and higher CO ability deployment ratio than the traditional pattern.

In 2016, J. Yu, Y. Wang and K. Pei [14] suggested a method using LB scheme to balance the traffic flow amongst CO and decrease distribution time. The outcomes of their simulation show that their method can achieve the LB traffic flow between COs with a minimum distribution time.

In 2017, P. Song, Y. Liu and T. Liu [15] suggested an LB technique called flow-stealer (FS) for traffic flow in SDN-CO. The FS technique uses a small rate flow-stealing way, i.e. the idle CO segments the amount of work momentarily with the loaded COs by stealing-flow events from them. The FS technique not only can respond to the network traffic fluctuating quickly but is able to decrease the incidence of alteration immigration. The outcomes demonstrated that this technique can balance the traffic flow between the COs more efficiently, specifically at burst traffic flows.

In 2018, X. Yang and Wang [16] suggested a load balancing technique using the K-Dijkstra and HRRF algorithms, using SDN as their network architecture. Their work consists of four modules: flow-management, traffic-monitoring, dynamic LB and load calculation in the controller of SDN architecture. They used the K-Dijkstra algorithm to calculate a number of alternative paths and using the HRRF algorithm to select the optimal path amongst the alternative paths. They found by using both of these algorithms the network delay and packet loss rate were effectively reduced and the network throughput improved.

In 2018, N. Kamat [17] implemented dynamic load balancing using SDN datacentres for achieving better results and higher performance. He used the round-robin method. From tests of two parameters (transferring of packets and BW), he found that after implementing LB the transfer of packets is faster and more precise.

In 2018, T. E. Ali, A. H. Morad and M. A. Abdala [18] implemented an LB technique using OSPFv2, OF 1.3 and IPv4 to obtain superior LB on SDN-DC fat tree topology (FTT). From the results, they found that after implementing LB, network parameters such as the throughput and the average delay were improved.

In 2019, H. Sufiev, Y. Haddad, L. Barenboim and J. Soler [19] proposed a system that set CO to clusters with optimization and the maximal distance between two CO within the similar clusters. Their system increased static clustering by a multiplicative factor of 5.

Also in 2019, F. Mulya, T. W. Purboyo and R. Latuconsina [20] discussed LB simulation using an ant colony optimization (ACO) mechanism. They found the throughput using the ACO algorithm was superior to the Round-Robin algorithm; the ACO algorithm had better LB as well, with lower CPU consumption.

2. SOFTWARE-DEFINED NETWORKING FRAMEWORK CONCEPT

SDN is a technique for centralizes control by segregating the control plane from the networking devices [21]. The first protocol in SDN standards is the OF protocol, which is responsible for establishing the connection between SDN switches (i.e. open v-switch (OVS)) and SDN-CO. The OF protocol has had a series of releases as shown in Table 1 and Figure 1 shows the timeline for the OF protocol.

Table 1. OpenFlow versions

OF Version	Fearures
1.1	Multiple table, Vlan's and MPLS
1.2	OXM, Multiple Controller
1.3	Meter table and Table miss entry
1.4	Synchronizes table and bundle
1.5	Egress Table and Scheduled bundle

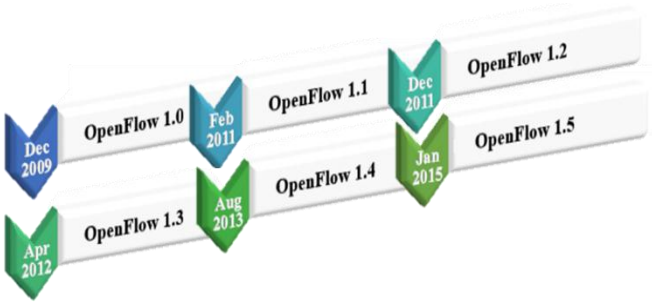


Figure 1. OpenFlow timeline

An OVS keeps up flow tables (i.e. one or more). These tables serve as packet handling. The flow tables inside the OVS can be edited (i.e. added, deleted or modified) using the OpenFlow controller (OF-CO) and receiving measurements for ports, flows and other info via the OF Protocol. Each flow table in OVS has flow entries arranged depending on the flow priority (i.e. the flows with the highest priority are set in the upper portions of the flow table while the flows with the lowest priority are set after) [22]. The entering packets are matched compared with flow entries starting from the priority at a high level. If a match is found, then the actions for that flow entry are executed. Otherwise if no match is found, the packets are either dropped or sent to the controller. The entering packets are matched versus the multiple tables in the OpenFlow switch (i.e. OVS) pipeline as shown in Figure 2.

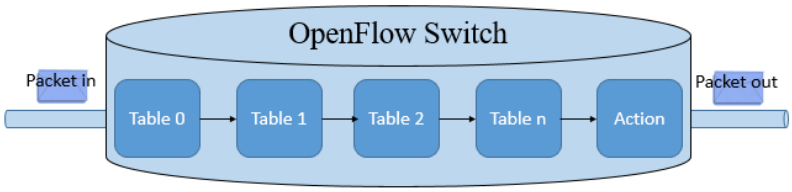


Figure 2. OpenFlow switch pipeline processing

The SDN framework is classified into the Application, Control and Infrastructure layers. Each SDN model has an SDN-CO (Control) besides APIs (south-bound APIs (SBI) and north-bound APIs (NBI)) [23]. The SDN-CO is represented as an SDN network device brain. SDN-CO allows a centralized controller overall network, assist the network engineering to operate with the underlying systems (i.e. switches-devices and routers-devices) and set the forward-plane polices (i.e. the data plane) in order to handle the network traffic. The SBIs are used to pass on the instructions to the OVS provided from SDN-CO (i.e. the connection between the controller layer and infrastructure layer), while the NBIs are used to establish a connection between the SDN-CO and the applications or business logic (i.e. the connection between the application layer and controller layer). The NBIs further assist the network engineering to programmatically shape traffic polices and deploy services. To work with OF background, any device that needs to talk with SDN-CO must support the OF protocol.

3. SOFTWARE-DEFINED NETWORKING DATA CENTRE IMPLEMENTATION

SDN-DC is built from a number of OVS, hosts, and SDN-CO. The SDN-CO function is to make configuration polices for the OVS to forward the traffic and allow forwarding traffic between OVS and hosts that are connected to it. We used the RESTCONF API in the NBI to integrate the LB application with the ODL SDN-CO [24]. Also, the ACO algorithm used in the network topology (NT) is shown in Figure 3. Any ant represents as a packet pass on the network, and it generates randomly at any node in the NT. The main process of LB with ACO consists of four phases as listed below [25]:

- Ant Obstetrics: Let (**a**) be the total number of ants, (**s**) the total number of OVSs, and A_i (t) refers to the number of ants in Switch-i (SW_i) at time (t). Then,

$$a = \sum_{i=0}^{s} A_i(t) \quad (1)$$

- Pheromone-Initiated: The start quantity of pheromone (PH) is supposed to be near or equal to zero (i.e. TPh(t)=0), where (TPh) indicates the total PH set on along the route from SW_i to SW_j.
- Choose the next hop: Each ant has a special table using to avoid the ant to pass through the same OVS again. Furthermore, it also determines the probability (P) for the transference among two OVS depend on the remained PH on each route. So, the ant selects the next hop based on the P determines. In (2), $P_{ij}(t)$ refers to the probability of moving ant from SW_i to SW_j.

$$P_{ij} = \begin{cases} \dfrac{(TPh_{ij})^{\alpha} * (h_{ij})^{\beta}}{Cost_{ij} * (\sum_{aSet} \dfrac{(TPh_{ij})^{\alpha} * (h_{ij})^{\beta}}{Cost_{ij}})} & , \quad j \in aSet \\ 0 & , \quad else \end{cases} \quad (2)$$

The coefficients (α) and (β) refer to the proportional impact of each factor, (aSet) is a collection of hops those ants can choose and (h_{ij}) is a heuristic function defined as below:

$$h_{ij} = {}^{1}/_{L_{ij}} \quad (3)$$

Where (L_{ij}) is the link-load, $(Cost_{ij})$ is the link transmission cost, which is defined as:

$$Cost_{ij} = \left(\delta * d_{ij}\right) + \left((1-\delta) * Ploss_{ij}\right) \quad 0 < \delta < 1 \quad (4)$$

Where(d_{ij}) is the delay and $(Ploss_{ij})$ is the link pack-loss.

- Update Pheromone: Once the ant reaches the target, it will disseminate PH along the route. The PH constantly vaporizes with time. This case is called pheromone updating. Pheromone updating is defined as:

$$TPh_{ij}(t) = \left((1-\rho) * TPh_{ij}(t)\right) + \Delta TPh_{ij}(t) \quad (5)$$

Where,

$$\Delta TPh_{ij}(t) = \sum_{k=1}^{a} \Delta TPh_{ij}^{k}(t) \quad (6)$$

Here (ρ) is the ratio of PH evaporated on the route, and therefore $(1-\rho)$ refers to the part of PH remaining. Also, the ($\Delta TPh_{ij}^{k}(t)$) refer to the quantity of PH that increments by each ant on the route from SW_i to SW_j in one-cycle, taking in the consideration that the $\Delta TPh_{ij}^{k}(t)$ is initially equal to 0 for each route.

So, from the above sequence of equations, the ant can look and choose the optimal route to the next OVS depending on the probability that is shown in (2). Additionally, in this paper, FTT SDN-DC using the new version from OF 1.4 with IPv6 is implemented as shown in Figure 3. The scenario consists of ONE SDN-CO unit which is called ODL-CO (The CO lets the traffic-flow passing amongst OVS and apply the load balance). Also, it consists of TEN OVS that supports OF protocol and EIGHT hosts connected to these OVS. The ODL-CO details can be shown in Table 2. The OVS using a terminal window in the root mode for running a command [26]. In addition, all connected clients use IPV6 addresses.

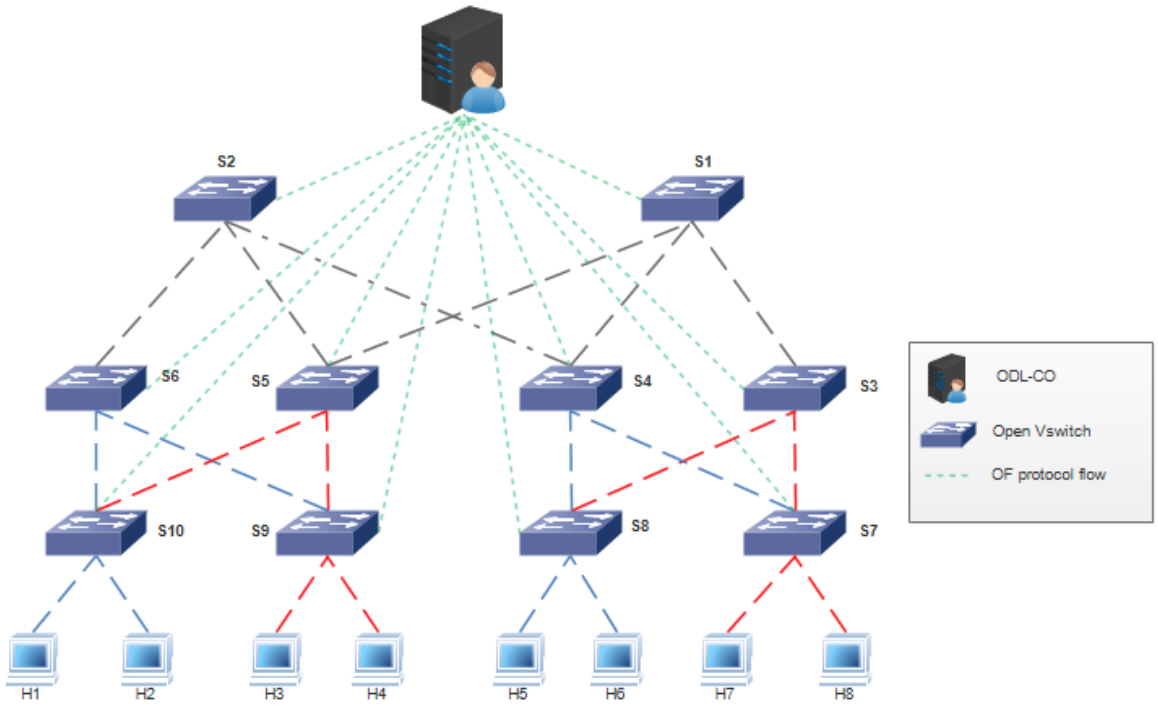


Figure 3. FTT SDN-DC

Table 2. SDN-DC Specifications

Parameter	Controller
IP address	2001:db8:3c4d:15::1/64
Port Number	6633
CLI	Enable
OF version	1.4
Procotol	TCP
Swicth	OVS

4. IMPLEMENTATION OUTCOMES

The implemented LB method with minimum latency in SDN-DC was tested by comparing the results of networking parameters before and after implementing the LB. In this implementation, the ACO algorithm is used to discover the optimal path between nodes in a network topology. Network routing protocols are a common application used for the shortest path algorithm, such as intermediate system to intermediate system (IS-IS) and open shortest path first (OSPFV3), which it supports for IPv6. Also, in this implementation, the ODL-CO is set by default to forward the traffic flow to all OVS switches ports. So, rules have to be pushed from the ODL-CO to achieve appropriate LB yield inside the DC.

Firstly, the ACO algorithm discovers the routes with the optimal-route to destination in a graph depending on the conduct of ants looking for a route between their colony and a food location [27], then the optimal-path choice for traffic-flow. After that, a static flow entry is pushed to all OVSs using the best route that is selected (e.g. ingress and egress port, the source and destination for both IPv6 and MAC address). The main function of REST APIs that are used in our implementation is to gather the networking devices and operative information of the fat tree topology.

When executing the LB function between all hosts at the same time, the shortest best route for all is selected by ODL-CO depending on the path congestion and shortest path. Our NT is emulated using a MiniNet Network Emulator [28], which manages a group of virtual clients, OVSs, internal CO and links on a single Linux kernel. Additionally, it provides many custom topologies creating and emulating some connection parameters like packet-loss, speed-link, throughput, transferring data and delay. We used a tool for traffic control (TC) inside the MiniNet called **qdisc,** which uses a portion of the Ubuntu kernel's TC module to determine the linkage properties like bandwidth and delay. Also, we used another tool inside the MiniNet called **iperf3** [29], which is a traffic generation (TG) tool between clients using TCP or UDP to determine the performance of network parameters such as the throughput and transferring data. Therefore, in

order to investigate and determine the effect of LB using ODL-CO with ACO on SDN-DC using the TC and TG tools, we compared before and after the LB deployment on several networking parameters such as:

- Throughput in (bps), which is indicating to the actual number of bits that drifts over a network linking in a given period of time. Throughput permanently is less or equal than the bandwidth but can never be more than the bandwidth.
- Transferring data (goodput), which is the real quantity of data in bytes that is transported in a given period of time and reached or received at their destination.
- Bandwidth, which represents the maximum number of bits that be able to stream over a network linking in a given period of time.
- Average Delay is the one-way period it takes for traffic to egress from the source and reach the destination. The delay is affected by latency. The latency is the period between data entering a device like a switch or a router and leaving it, for example, a switch may add 1 ms of latency to the delay.

Figure 4 shows the throughput between hosts before and after LB using the fat tree topology SDN-DC.

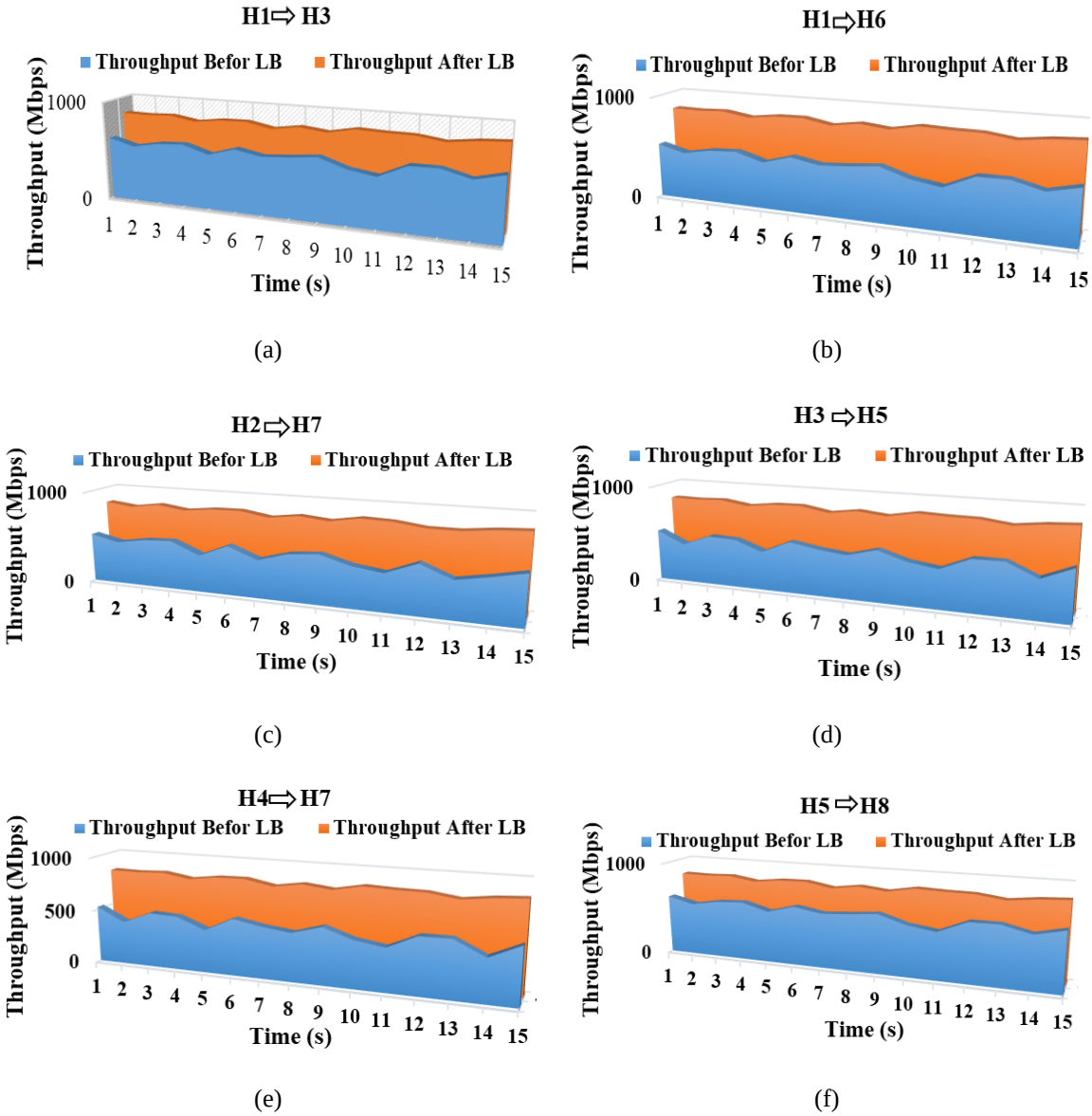


Figure 4. Throughput between hosts using TCP Traffic

Figure 5 shows transferring data between hosts before and after LB using the fat tree topology SDN-DC. Figures 6 and 7 show the bandwidth and the average delay between hosts before and after LB using the fat tree topology SDN-DC, respectively. The results show the throughput, data transfer and bandwidth increasing while the delay decreasing when applying the LB algorithm. These results are appropriate, since the traffic flow is random inside the SDN-DC network before the LB, which causes congestion in one of the network topology links, while the traffic flow distribution is approximately 97% uniform when applying the LB algorithm that resulting in a very small probability of congestion accruing.

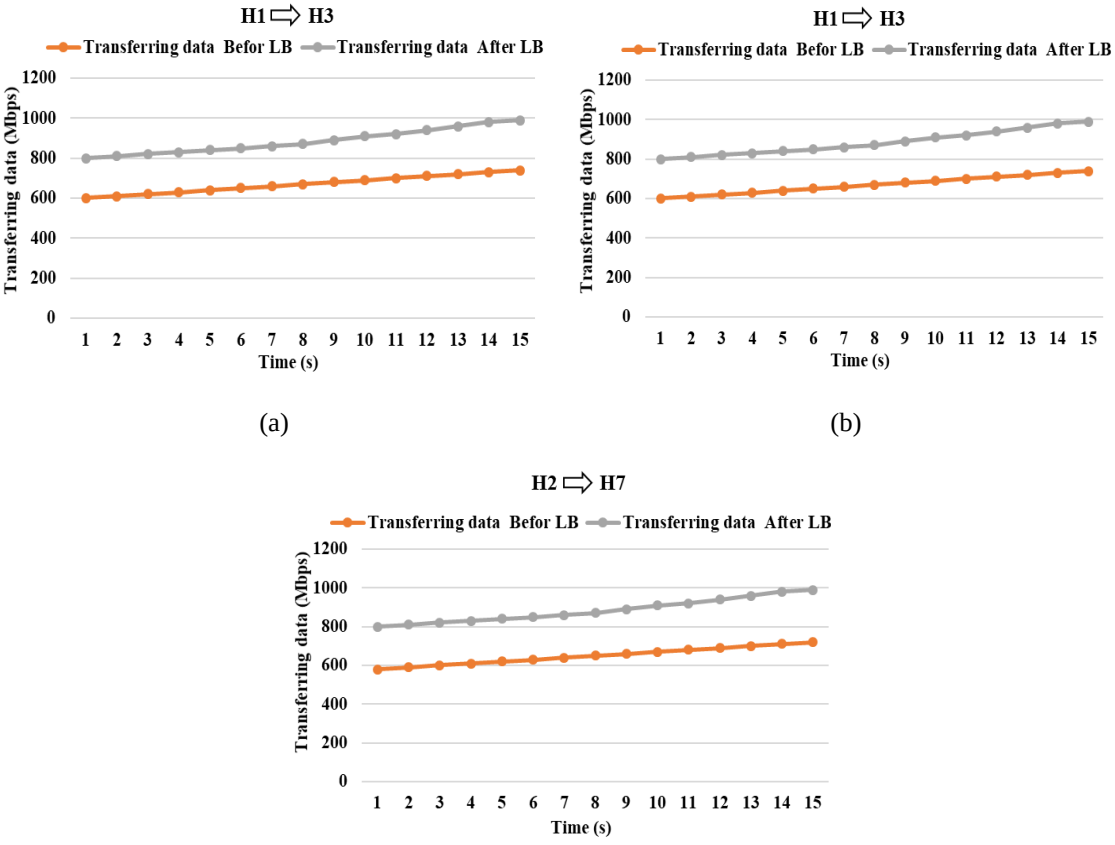


Figure 5. Transferring data traffic between hosts

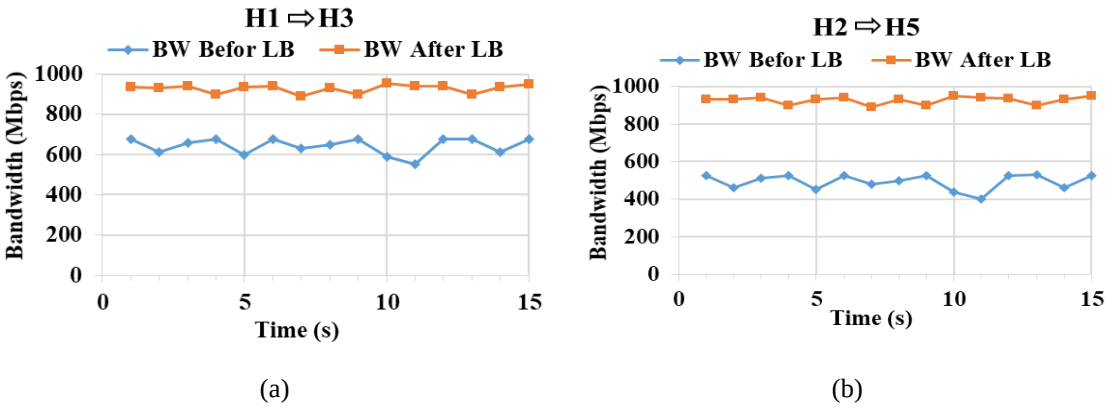
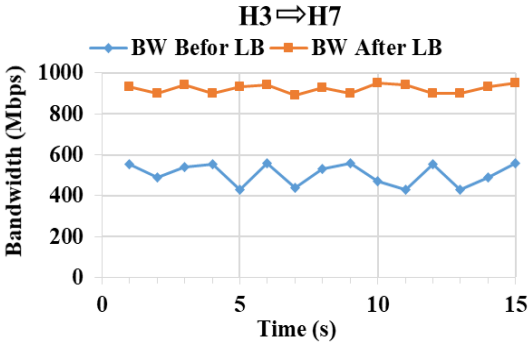
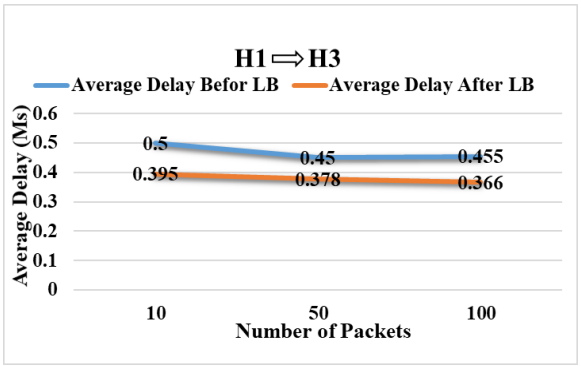


Figure 6. Bandwidth between hosts

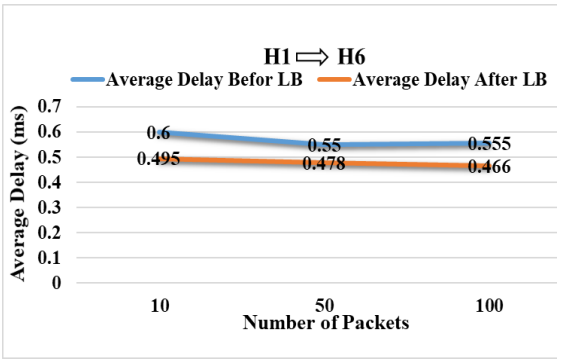


(c)

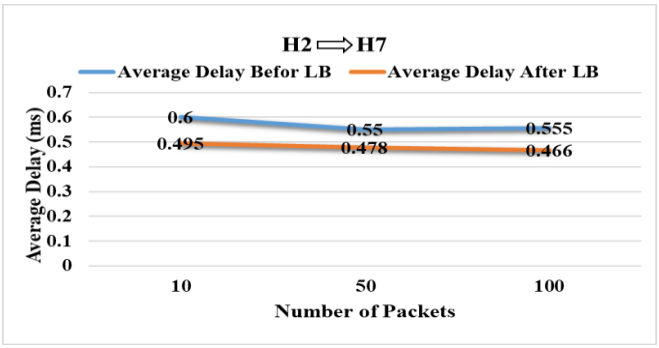
Figure 6. Bandwidth between hosts *(continue)*



(a)



(b)



(c)

Figure 7. The average delay between hosts

5. CONCLUSION

SDN is a suitable technology for development to meet current network requirements. SDN does not need to deploy different networking equipment or hand-setting all network equipment; SDN network technology just requires a network device that supports OF protocol features to allow the SDN-CO to deal with the SDN network devices. In this paper, the LB method is developed and applied using OF 1.4, an ant colony optimization algorithm and IPv6 with traffic generated 1Gbps to ensure better traffic flow distribution inside the SDN-DC using fat tree network topology. The implementation results collected show that after applying the LB algorithm inside the SDN-DC, the measured network parameters (throughput, data transfer, bandwidth and average delay) are enhanced. At the same time, when we compare these results with literature

survey mentioned above, we can note that the networking parameter performance is enhanced with LB by using OF 1.4 and the ACO algorithm with IPv6 rather than OF 1.3 and the round-robin algorithm with IPv4. In addition, we note that the networking parameter performance was enhanced with LB by using OF 1.4 and ACO algorithm with IPv6 rather than OF 1.0 and ACO algorithm with IPv4 as well. Also, the performance of the networking parameters like throughput, delay, etc. using ODL-CO was better than when using a POX controller. So, we can conclude that the OF 1.4 and ACO algorithm using ODL-CO is more suitable for the LB method.

REFERENCES

[1] D. J. Kadhim and W. K. Hussain, "Design and Implementation of A Proposal Network Firewall," *Al-Khwarizmi Engineering Journal*, vol.2, no.1, pp 52-69, 2006.

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

[3] H. Kim, T. Benson, A. Akella, and N. Feamster, "The Evolution of Network Configuration: A Tale of Two Campuses," *Proceedings of the 2011 ACM SIGCOMM conference on Internet Measurement Conference*, pp. 499–514, 2011.

[4] T. E. Ali, M. A. Abdala, and A. H. Morad, "SDN Implementation in Data Center Network," *Journal of Communications*, vol. 14, no. 3, pp. 223-228, March 2019.

[5] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 2, pp. 87-98, April 2014.

[6] D. Sinh, L. Le, B. P. Lin and L. Tung, "SDN/NFV—A New Approach of Deploying Network Infrastructure for IoT," *2018 27th Wireless and Optical Communication Conference (WOCC)*, pp. 1-5, 2018.

[7] A. A. Z. Ibrahim and F. Hashim, "An architecture of 5G Based on SDN NV Wireless Network," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 14, no. 2, pp. 725-734 , May 2019.

[8] K. Golani, K. Goswami, K. Bhatt and Y. Park, "Fault Tolerant Traffic Engineering in Software-defined WAN," *2018 IEEE Symposium on Computers and Communications (ISCC)*, pp. 01205-01210, 2018.

[9] S. Gordeychik, D. Kolegov, and A. Nikolaev, "SD-WAN Internet Census," *ArXiv*, Aug 2018.

[10] M. Chan, C. Chen, J. Huang, T. Kuo, L. Yen and C. Tseng, "OpenNet: A simulator for software-defined wireless local area network," *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, Istanbul, pp. 3332-3336, 2014.

[11] R. Wang, D. Butnariu, and J. Rexford, "OpenFlow-Based Server Load Balancing Gone Wild," *Hot-ICE'11: Proceedings of the 11th USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, 2011.

[12] S. Kim, H. Choi, P. Park, S. Min and Y. Han, "OpenFlow-based Proxy mobile IPv6 over software defined network (SDN)," *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, pp. 119-125, 2014.

[13] Y. Kyung, Kiwon Hong, T. M. Nguyen, Sungho Park and J. Park, "A load distribution scheme over multiple controllers for scalable SDN," *2015 Seventh International Conference on Ubiquitous and Future Networks*, Sapporo, pp. 808-810, 2015.

[14] J. Yu, Y. Wang, K. Pei, S. Zhang and J. Li, "A load balancing mechanism for multiple SDN controllers based on load informing strategy," *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Kanazawa, pp. 1-4, 2016.

[15] P. Song, Y. Liu, and T. Liu, "Flow Stealer: Lightweight Load Balancing by Stealing Flows in Distributed SDN Controllers," *Science China Information Sciences*, vol. 60, no. 3, March 2017.

[16] X. Yang and L. Wang, "SDN Load Balancing Method Based on K-Dijkstra," *International Journal of Performability Engineering*, vol. 14, no. 4, pp. 709-716, 2018.

[17] Kavana H M, Kavya V B, Madhura B, Neha Kamat, "Load Balancing using SDN Methodology," *International Journal of Engineering Research & Technology*, vol. 7, no. 05, pp. 206-208, May 2018.

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

[19] H. Sufiev, Y. Haddad, L. Barenboim, and J. Soler, "Dynamic SDN Controller Load Balancing," *Future Internet*, vol. 11, no. 3, 2019.

[20] F. Mulya, T. W. Purboyo, and R. Latuconsina, "Experimental Study of Load Balancing on A Software-Defined Network Using Ant-Colony Optimization," *ARPN Journal of Engineering and Applied Sciences*, vol. 14, no. 17, pp. 3032-3037, September 2019.

[21] V. Shukla, "Introduction to Software Defined Networking-Openflow & VxLAN," CreateSpace Independent Publishing Platform, California, US, 2013.

[22] B. Oh, S. Vural, N. Wang, and R. Tafazolli, "Priority-Based Flow Control for Dynamic and Reliable Flow Management in SDN," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1720-1732, Dec. 2018.

[23] S. Schriegel, T. Kobzan and J. Jasperneite, "Investigation on a distributed SDN control plane architecture for heterogeneous time sensitive networks," *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, Imperia, pp. 1-10, 2018.

[24] J. Medved, R. Varga, A. Tkacik and K. Gray, "OpenDaylight: Towards a Model-Driven SDN Controller architecture," *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, Sydney, NSW, pp. 1-6, 2014.
[25] S. Sathyanarayana and M. Moh, "Joint route-server load balancing in software defined networks using ant colony optimization," *2016 International Conference on High Performance Computing & Simulation (HPCS)*, Innsbruck, pp. 156-163, 2016.
[26] S. Ganesh and S. Ranjani, "Dynamic Load Balancing using Software Defined Networks," *International Conference on Current Trends in Advanced Computing*, pp. 11-14, 2015.
[27] L. W. Santoso, A. Setiawan, and A. K. Prajogo " Performance Analysis of Dijkstra, A* and Ant Algorithm for Finding Optimal Path Case Study: Surabaya City Map", *Proceedings of 2nd Makassar International Conference on Electrical Engineering and Informatics (MICEEI)*, pp. 303-310, 2010 .
[28] Lantz B., Heller B., McKeown, N., "A Network in A Laptop: Rapid Prototyping for Software-Defined Networks," Hotnets-IX: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, no. 19, pp. 1-6, 2010.
[29] Tirumala, A., *et al.*, iPerf: the TCP/UDP Bandwidth Measurement Tool. [Online]. Avalable: https:// iperf.fr/

BIOGRAPHIES OF AUTHORS

Tariq Emad Ali has a B.Sc. and M.Sc. in Electronics and Communication Engineering, College of Engineering, Baghdad University. He is an Assistant lecturer at the University of Baghdad, Al-Khwarizmi College of Engineering, Information and Communication Engineering Department. He has 6 published scientific & technical papers including IEEE explorer. Mr. Tariq Emad has 12 years of academic & practical's and consulting experience in Networking & Communication. He currently teaches & conducts research programs in the areas of software computer networks, Network protocols, Network management , Network automation, soft computing, Artificial intelligence, Ad-Hoc networks, Wireless Sensor Networks, Routing Protocols, Security of VANETs, Smart Antenna in MANETs , WiMAX Networks, SDN Networks, SD-WAN , IoT, IOE, IOV, VM's and others.

Ameer H. Morad is an Associate Professor in the Department of Information and Communication, Al Khwarizmi College of Eng., University of Baghdad. Presently he is involved in research work in the areas like Image Processing, Computer Vision, Visual Cryptography and Network security

Mohammed A. Abdala was Born in Baghdad, IRAQ, 5/12/1962. Doctor of Philosophy (Ph.D.), Electronic Engineering-Transconductance & Noise Analysis in GaAs MESFETs. Lancaster University, UK. Oct. 1991. Master of Science (M.Sc.), Semiconductor Devices, Lancaster University, UK. Oct 1988. Bachelor of Science (B.Sc.) (Hons.), Electrical & Electronic Engineering. Grade: Distinction, University of Technology, Baghdad, Iraq. June 1984. He is now the head of the Medical Instruments Engineering Department at Al-Hussain University College, Iraq. He has more than 30 years of academic & consulting experience in Networking, Microelectronics, Solid State Electronics, Software Engineering, Pattern Recognition, and Verification. He published more than 45 scientific & technical papers. He currently teaches & conducts research programs in the areas of image processing, Genetic and PSO Optimization Algorithms, pattern recognition & verification, VHDL implementation.