Performance Analysis of Congestion Control Mechanism in Software Defined Network (SDN)

M. Z. A. Rahman[1], N. Yaakob[1*], A. Amir[1], RB Ahmad[2], S.K.Yoon[1], A.H. Abd Halim[1]

[1]School of Computer and Communication Engineering (SCCE), Universiti Malaysia Perlis (UniMAP), 02600 Arau, Perlis, Malaysia
[2] Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Kuala Terengganu, Malaysia

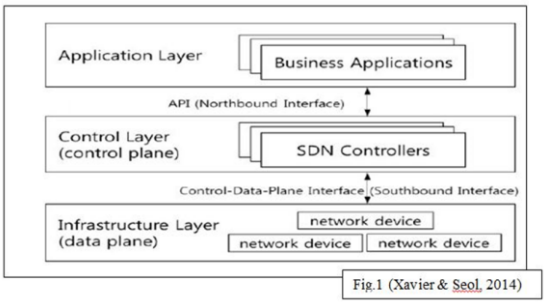
Abstract. In the near future, the traditional networks architecture will be difficult to be managed. Hence, Software Defined Network (SDN) will be an alternative in the future of programmable networks to replace the conventional network architecture. The main idea of SDN architecture is to separate the forwarding plane and control plane of network system, where network operators can program packet forwarding behaviour to improve the network performance. Congestion control is important mechanism for network traffic to improve network capability and achieve high end Quality of Service (QoS). In this paper, extensive simulation is conducted to analyse the performance of SDN by implementing Link Layer Discovery Protocol (LLDP) under congested network. The simulation was conducted on Mininet by creating four different fanout and the result was analysed based on differences of matrix performance. As a result, the packet loss and throughput reduction were observed when number of fanout in the topology was increased. By using LLDP protocol, huge reduction in packet loss rate has been achieved while maximizing percentage packet delivery ratio.

1 Introduction

Recently, the dramatic increased of Internet capacity network traffic due to the increment used of mobile devices, Internet protocol television (IPTV), server virtualization and cloud services are becoming more complicated for a traditional architecture network to handle high traffic. In traditional network, device such as router, switch and firewall has their own control plane and data plane. As a result, the network cannot handle higher user capacity because the forward plane and control plane are in the same device [1]. Thus, more dynamic and flexible network is required to easily managed the traffic.

The communication networks are growing rapidly and becoming more complex to be managed and handled. This is especially true with traditional network architecture, network systems and protocol stack which is difficult to provide adequate solutions to the contemporary networking demands. Hence, Software Defined Network (SDN) is a promising approach to computer networking that separates data plane and control plane combined with centralized controller to allow network operator to program the packet forwarding. The SDN architecture is as shown in Fig. 1

One of the major problems of congestion in network traffic is the reduction in the Quality of Service (QoS) that cause tremendous loss of several network aspects



.

Fig. 1. SDN Architecture [11].

that lead to traffic network congestion which can be detailed out as follows.

* Corresponding author: naimahyaakob@unimap.edu.my

1.1 Congestion cause high number of packet loss and lower network performance.

Congestion in SDN must be avoided because it may cause severe network performance. One of the effects of congestion is the increase in the number of packet loss. High number of packet loss causing the node to retransmit the data and this contribute to low packet delivery ratio (PDR). The time for packet to reach at destination node will also be delayed causing end-to-end delay to increase and the throughput to decrease.

Hence, the data might not be valid once it retransmit to the destination node due to high latency occurs. Therefore, avoiding congestion can reduce the afore-mentioned effects and improve the overall network performance.

1.2 Congestion reduces the QoS in SDN

The high number of users are sending the data simultaneously to the SDN controller, it will not only cause high buffer usage but also cause congestion.

This is because high redundancy in packet transmission will increase the traffic and the possibility of buffer overflow that can lead to congestion. Thus, the congestion can affect to packet loss rate increase, delay increase, low throughput, so cannot provide high QoS to users, user may experience interruption in the communication especially in the real-time applications. In addition, huge number of packets will be wasted in sending the redundant information data.

Therefore, there is need to find a better way to improve packet delay ratio, throughput while reducing packet loss rate and latency.

1.3 Lack of congestion control technique in SDN.

SDN mainly deal with high total data information which cannot be afforded high congestion in network. This is because the congestion can cause high packet loss rates and may cause buffer overflow. Therefore, resolution for congestion control mechanisms need to maintain and control network congestion efficiently in SDN.

1.4 Contribution in SDN

Moreover, in this project contribution to analyse the congestion avoidance in SDN. Part of my project aim at the understanding cause network congestion in SDN. Moreover, this research aims to analyse the avoidance of congestion in SDN's network traffic. Finally, research about cause happened congestion from the sender to the SDN controller is important to the future network community for further growth.

2 Software Define Network

The authors in [1][2][3][4] defined the SDN as decoupled the control plane from data plane, difference from legacy networks which were difficult and complex to be managed since the traditional way implements protocol to allow nodes to work with other nodes to exchange information. Nowadays, the new paradigm network architecture can be performed more effective, flexible and easier to manage by using single centralized controller.

The SDN architecture is important sections to be explained because there are tightly related on each level planes. Moreover, each plane has different functions and mechanisms to work in one network system.

2.1. Application Plane

For the application layer, SDN application are programmed so that it can directly be programmable to communicate with their network demands and the behaviour network to controller via Northbound (API) [5]. This is used to communicate between the controller and services and application running over the network. The SDN application consists of one SDN application logic and one or more Northbound Interface (NBI). In the future, another layer of abstracted network control might be accommodated to provide one or more higher level of NBI. SDN's NBI is allocated between SDN application and SDN controller which fulfil the requirement and control the network behaviour.

2.2 Control Plane

In control plane, suitable and right place SDN controller located where SDN controller as centralized entity in change to translating requirement from first layer to third layer and as agent to SDN application with provide abstract of network view. In SDN controller consist of one or more Northbound Interface (NBI) agent link to NBI driver, SDN control logic and SDN Control-Data-Plane Interface (CDPI).

2.3 Data Plane

The data plane consists of the logical network devices [5]. At this level, the forwarding and data processing occurs. An SDN data path comprises a Control Data Plane Interface (CDPI) and one or more set of traffic forwarding engines and zero or more traffic processing functions. For single (physical) network element, it may consist of one or more SDN data path. Moreover, an

SDN data path can also be knowns as across multiple physical network element because logical definition is not fixed.SDN Control to CDPI is allocated between SDN controller and SDN data path, which function as controller to all forwarding operations, capability advertisement and reporting.

2.3 Open Flow

OpenFlow is one of the protocols used in Mininet which is an open source emulator for SDN [6]. It is an open standard interface that enables researchers to run experimental protocols in real SDN network. To allow researchers to handle the experiment, OpenFlow is provided with standardized features and it based on Ethernet switch but can be used in routers and wireless access point.
In a traditional switch, the data plane and control plane occur in the same device which data plane acts as the fast packet forwarding and the control plane acts as high-level decision maker. Hence, an OpenFlow decouples these two planes where data plane still works on switch, then the control plane is separated to make high level decision, normally a standard server. The communication between the control path and SDN controller via a secure channel use OpenFlow protocol.

3 CONGESTION CONTROL IN SDN

One of the methods to control congestion in SDN is implemented using Link Layer Discovery Protocol (LLDP) as shown in Fig. 2. According to the work in [7], LLDP has been proposed where SDN controller sends packets as Packet-out message during process link discovery. The packets will be sent to all switches in the network and when SDN switch receive LLDP packet, it sends the packet to all other switches directly connected to it. After other SDN switches receive an LLDP packets, they send packet to the SDN controller as Packet-in message. The main purpose for this proposed technique is to allow SDN controller to analyse which connected switches with it and can collect data transmission between SDN controller and switches.

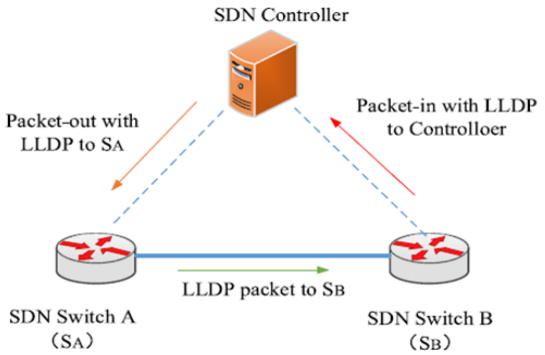


Fig.2. LLDP for SDN topology [7]

As mention before, the first action from controller is sending out the packet as Packet-out message with LLDP packet to the first switch. At the beginning, the second switches will first broadcast its LLDP packet as Packet-out message to find another switch neighbours.

For the first-time operation, the LLDP packet from controller will send at shortest path way to switches. Compared with the OpenFlow (OF) method, OF will send the high number of packets to enable all packets to discovery all switches in the SDN topology.

Hence, the packet will randomly travel to all switches and after that all the packets will identify the switch. Packets will flow through all switches before arriving at the controller because by using OF protocol, there is no forwarding rule in the switches. Let say in the experiment that has a large topology, the OF needs take longer time to arrive at the controller. At the neighbours of switch, LLDP packet will enable FlowTable to send the LLDP packet directly to controller.

3.1 Link Layer Discovery Protocol (LLDP)

In this technique, Packet-out-messages will be sent to all other switches and from the switch, LLDP packets will be sent as Packet-in-message to controller.

This is done because there is no corresponding forwarding rule in the switch's FlowTable. As being discussed in previous section, OpenFlow is an as open source protocol to enable communication between controller and data plane. The OpenFlow protocol takes longer to travel from one switch to another switch before the packet is sent back to the controller. Since it might take longer time to travel, the possibility for the data to be lost is high due to high chances of collision and dropped because of congestion. In this experiment, the method is known as link discovery technique based on LLDP protocol.

The method performs by using LLDP protocol will take the shortest time for packet travel in the link before the packet is retransmitted to the controller and it can analyse which switch are connected directly to each other. This approach is one of the efforts to reduce packet collision and packet drop. The following sections explain about the concept of LLDP.

In the process of implementing LLDP protocol in SDN topology, there are several objectives need to be satisfied. First, LLDP protocol should be able to reduce the number of packet loss during the transmission. This is because each packet loss causing retransmission of the data and every retransmission process will waste the time for a controller to analyse the entire network. Second aims, is to minimize the number of transmissions. This is because by reducing the number of transmission packet will reduce the possibility packet collision with other packet and at the same time percentage of packet loss will be reduce.

3.2 Flowchart of LLDP

From the Fig.2(a), the first packet of LLDP was started at SDN Controller. Then, when first packet LLDP move to first switch, the LLDP packet was tagged as Packet-out message. The Packet-out message will checking another available switch (as S^n). If another switch is available, the packet with LLDP from available switch will sending out Packet-out message to another switch neighbours.

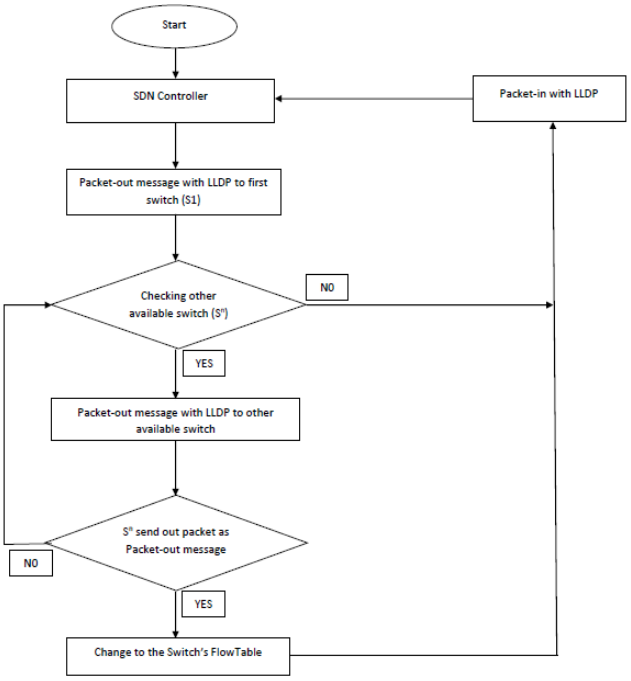


Fig.2(a). Flowchart of LLDP.

If no another switch neighbours receive the LLDP packet, the received switch will sending out the packet as Packet-in with LLDP to the SDN Controller.

4 Methodology

4.1 Simulation Tools

For this research, the emulator tools using for simulating real network devices is Mininet. It was installed in open sources operating system which is Ubuntu 14.04 LTS 64-bit. Mininet is a popular open source network emulator that is capable to combine all components for testbed in SDN. This emulator was created in Phyton language and provide a Phyton application program interface for user customize any level of topology. Moreover, Mininet can virtualize a single operating system. Its hosts can be operated in standard Linux operating system and Linux can support ODL controller.

The installation of Mininet require open source operating system so that Mininet is compatible in Linux. Since it runs a real Linux kernel and network stack, it can be compatible to ready hardware devices with less changes of command line interface (CLI)

4.1.1 OpenDayLight.

OpenDayLight (ODL) of Linux Foundation is open source software platform for SDN and NVF (Network Function Virtualization). Many IT companies such as Cisco, IBM and Dell are participating in ODL project to improve the quality of SDN and NFV. ODL supports OpenFlow, OpFlex and many different network devices control protocol and it also has the advantage with respect to portability since it uses standardized model.

4.1.2 Mininet.

Mininet provides the low-cost testbed for SDN, Authors in [8] using Mininet as SDN testing platform. This is because Mininet allows to develop network of virtual host, SDN controller and switches in very huge scale of topology that up to thousand nodes and it can be perform the test easily. Moreover, Mininet also allows creating the emulator as real-world network scenario and it can emulate SDN controller and OpenFlow devices by using open source operating system such as (Linux Ubuntu server version 14.04. An author in [9], creates the SDN controller using Phyton programming language, hence some C programming language cannot be used to custom the topologies. Authors in [10], was stated that the Mininet is a low cost SDN testbed that can be used for educational and research purpose.

4.2 Simulation Environment

In this experiment, the default command is used for simulation with three different fanout on ODL and virtual network system environment. In table 1 shows the different number of switch for each number of fanout.

Fig. 3. Number of fanout.

Number of fanout	Controller	Switch	Host	Link
1	1	4	1	4
2	1	15	16	30
3	1	40	81	120
4	1	85	256	340

5 Result and Discussion

There are several performance metrics that are being analysed to measure the performance of the proposed method. Those performance metrics are:

1. Packet Loss Rate
2. Packet Delivery Ratio
3. Throughput

For this experiment, only one variable parameter has been tested which is the number of fanout in SDN topology. The first part will present the result of SDN with varying number of packet, while the second part will explain the results types of topology.

The graph shown in the following subsections prove the occurrence of congestion in SDN network before using LLDP protocol. Since we are using Mininet emulator, the topology is already created for basic concept of SDN which consist of controller, switches and many hosts. To implement the first experiment to get variety number of packets, the type of topology was developed in same pattern but difference number of fanout.

The first parameter that we have measured in order to study the effectiveness of LLDP method is packet loss rate. The following section shows the result obtained through the simulation in Mininet.

5.1 Packet Loss Rate.

Referring to Fig. 4, the topology of SDN architecture without LLDP protocol gives highest rate of packet loss. This shows that, congestion is possible to occur in a typical SDN architecture.

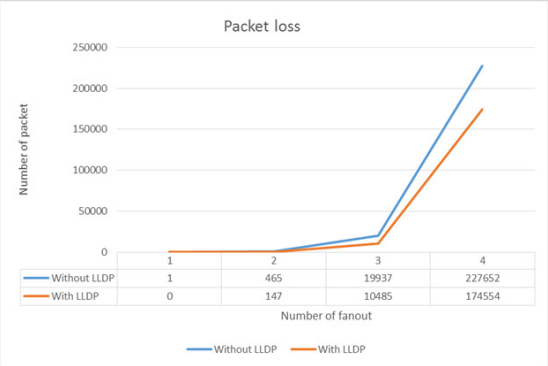


Fig. 4. Graph shows the rate of packet loss for different number of fanout

However, as the LLDP protocol was applied in the topology, the rate of packet loss is reduced. This implies that the congestion has been successfully controlled. From the figure, the both methods show increasing pattern of packet loss rate which reflects that as the number of fanout increases, the number of packet loss also increases. However, when the LLDP protocol is implemented, the number of packet loss is reduced.

This is due to less packet collision occurs when the transmission of packet to other host. To make sure that the number of packet loss is really decreasing when using LLDP, the percentage of the packet loss is calculated and compared. Fig. 5 shows the comparison of packet loss rates in both cases (with and without LLDP protocol) where the pattern for LLDP is decreasing. The congestion is successfully controlled using LLDP since the percentage of packet loss is less than 70% which is the threshold value [6] for the large SDN topology that can be accepted. Basically, the pattern of the packet loss should be decreasing as the number of fanout is lower.

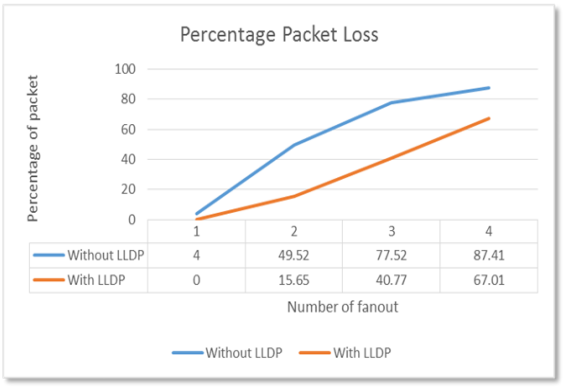


Fig. 5. Graph shows the percentage of packet loss with and without LLDP protocol.

This is because the decreasing of number of nodes in an SDN topology will reduces the number of packets, hence minimize the probability of having collision with other packets while at the same time reduces the congestion. Less congestion means less number of packet are is lost or dropped.

From Fig. 4, the number of packet loss at four number of fanout is very high which reaches 227652 packets especially when there is no LLDP protocol applied. This is because congestion in SDN is highly likely to occur in the increasing number of switches is 85. However, as the number of fanout decreases, the number of packet loss is also reduced. This shows that number of fanout is one, perform 0% number of packet loss when the LLDP protocol was implemented. Moreover, the graph shows decreasing pattern of packet loss value in varying number of fanout with the reduction number when using LLDP protocol. This implies that the LLDP protocol improves the performance of by successfully reduces the collision between packets in the network and thus avoid congestion from happening.

5.2 Packet Delivery Ratio (PDR).

Packet Delivery Ratio is illustrated in Figure 6, which shows the percentage of PDR in both cases (with and without LLDP). As can be seen in the figure, LLDP (which is represented by the red line) achieves 100% percentage of PDR when the number of fanout is zero. This is because no packet loss or packet drop occurs in SDN after LLDP protocol was implemented into the topology. On the other hand, the collected data without the implementation of LLDP protocol shows 92.30% percentage of PDR due to the process of link discovery

which takes long period of time because the switch now is not controlled by the controller, thus the packets will travel to each switch before the packet arrive at controller.

However, as the number of fanout increases from 2 to 4, the percentage of PDR is decreasing slightly due to the increasing number of switches in the SDN topology. The implementation of LLDR protocol into SDN topology really increases the percentage of PDR when the number of fanout is two, where the comparison shows that SDN with LLDP is achieve 42.5% PDR.

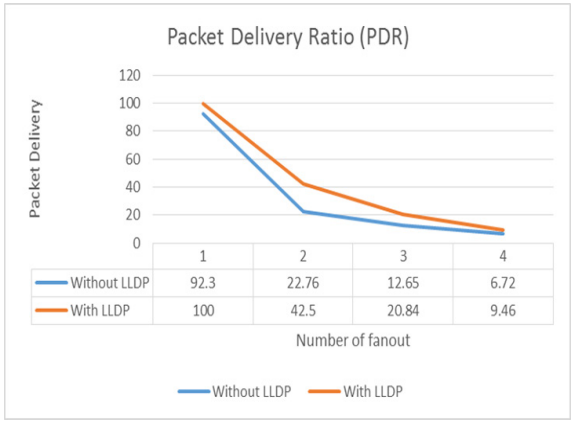


Fig. 6. Red line shows percentage of packet delivery ratio in the presence of LLDP protocol and blue line shows the percentage of packet delivery ratio without LLDP.

This is because LLDP packet will sent back to the controller as Packet-In message, since there is no corresponding forwarding rule in the switch's FlowTable, while without LLDP is 22.76%. Moreover, the pattern is decreasing as the number of node increase when the number of fanout is four. It reflects higher number of switches and the increase in the period link discovery can decrease the PDR.

5.3 Throughput.

Throughput is the average or mean of the packet that successfully send to the destination through a communication link over a period (second). From Figure 7, both graphs show the pattern of decreasing for the throughput. For the original method without the LLDP protocol, the throughput value is in the range of 20 to 100. While for SDN with the LLDP protocol, the throughput is in the range of 50 to 150.

According to the figure 7, the throughput reading shows the decreasing pattern as the number of fanout increase because when the number of nodes increase, more transmission occurs in the topology and it can increase the packet collision. Then number of packets to be dropped or lost will also be higher. Hence, the number of packet successfully sent per time slot will be decreased. The results obtained is then compared with theoretical value and we can conclude that the results comply with the theory. This is due to the more transmission occurs in the network in the sense that the switch that is implemented with LLDP will help sending the Packet-In message to the controller and controller can analyse which switches are connected directly to each other. On the other hand, the switch that is not be implemented with LLDP need takes long period time travel during process of link discovery. As a conclusion, the lower number of nodes to transmit the packet, the higher network throughput

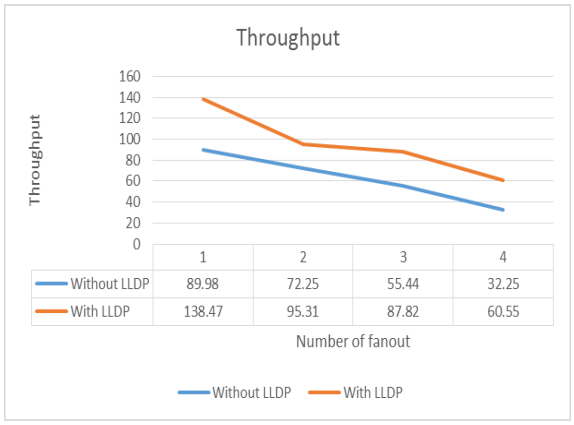


Fig. 7. Red line shows throughput for implement LLDP protocol and blue line shows the throughput without LLDP.

.

6 Conclusion

Software Defined Network is rapidly growing that has changed the current network topology. It not only handles hundreds switch but it need handle more than thousands of switches to able a billion devices and user in the future. Therefore, to make very big topology, the high possibilities the congestion in the SDN networking can happened. Thus, controller is one of the most important part in develop SDN topology because it used to manage and monitoring entire network from any congestion and traffic load.

SDN controller normally used to collect important information regarding the occurrence of congestion in network link that caused by the larger number of data sends more than the capability and it can make the higher the possibility for congestion to occur. The effect of congestion can be seen by the degradation in the overall quality and the increase in the packet loss rates leading to packet drops, buffer delay and end-to-end delay. As SDN have very limited resource, the congestion network issue is critical to be solved. Concerning on this issue, researchers was introduced Link Layer Discovery Protocol (LLDP) to control congestion in the SDN topology. The experiment already conducted to see the effect of LLDP towards the SDN topology.

7 Future Work

Here are some suggestions that can be considered to improve future projects. This research has focused on the method to avoid congestion and the technique used to deal with the subsequent issues that could affect performance of SDN and its underlying platforms. Even though it has achieved desirable performances, possible future studies could be a platform to further improve our proposed solution for using high speed transmission rate in Mininet emulator. Based on experiment, it is more specified only to several topologies and considered as small topology compared if the SDN applied in real network topology that needs huge and capability switches to connect with each other. Therefore, an extension of our work including for high speed transmission rate and buffer limiter should be taken care for the next research investigations.

Acknowledgment

The research reported in this paper is supported by Research Acculturation Grant Scheme (RAGS) [Grant number: 9018-00082]. The author would also like to express gratitude to the Malaysia Ministry of Higher Education (MOHE) and University Malaysia Perlis for the facilities provided

References

1. H. Kim, J. Kim, and Y.-B. Ko, “Developing a cost-effective OpenFlow testbed for small-scale Software Defined Networking,” 16th Int. Conf. Adv. Commun. Technol., pp. 758–761, 2014.
2. S. Song, J. Lee, K. Son, H. Jung, and J. Lee, “A congestion avoidance algorithm in SDN environment,” Int. Conf. Inf. Netw., vol. 2016-March, pp. 420–423, 2016.
3. Y. Huang, M. Lee, X. Huang, and C. Hsu, “Minimizing Flow Initialization Latency in Software Defined Networks,” pp. 303–308, 2015.
4. R. Horvath, D. Nedbal, and M. Stieninger, “A Literature Review on Challenges and Effects of Software Defined Networking,” Procedia Comput. Sci., vol. 64, pp. 552–561, 2015.
5. Open Networking Foundation, “SDN Architecture Overview,” Onf, pp. 1–5, 2013
6. P. M. Mohan, D. M. Divakaran, and M. Gurusamy, “Performance study of TCP flows with QoS-supported OpenFlow in data center networks,” IEEE Int. Conf. Networks, ICON, pp. 1–6, 2013.
7. Z. Shu et al., “Traffic Engineering in Software-Defined Networking : Measurement and Management,” vol. 3536, no. c, 2016.
8. D. Kumar and M. Sood, “Software Defined Networks (S.D.N): Experimentation with Mininet Topologies,” Indian J. Sci. Technol., vol. 9, no. 32, 2016.
9. C. Decusatis, A. Carranza, and J. Delgado-caceres, “Modeling Software Defined Networks using Mininet,” Proc. 2nd Int. Conf. Comput. Inf. Sci. Technol. Ottawa, Canada –, no. 133, pp. 1–6, 2016.
10. J. L. M. N. Weerawardhana, N. J. A. W. Chandimal, and A. Bandaranayake, “SDN Testbed for Undergraduate Education,” vol. 2015, 2015.
11. H. F. Xavier and S. Seol, “A Comparative Study on Control Models of Software-Defined Networking (SDN),” vol. 7, no. 32, pp. 1747–1753, 2014.
12. M. Chen, Y. Qian, S. Mao, W. Tang, and X. Yang, ” Software-defined mobile networks security,” Mobile Netw. Appl., doi : 10.1007/s11036-015-0665-5.
13. F. Pakzad, M. Portmann, W. L. Tan, and J.Indulska, ‘‘Efficient topology discovery in software defined networks,” in *Proc. 8th Int. Conf. Signal Process. Commun. Syst. (ICSPCS)*, Dec.2014, pp. 1-8.
14. S. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, "PayLess : A low cost network monitoring framework for software defined networks,” in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, May 2014, pp. 1-9.
15. S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking,” in *Proc. Int. Symp. Recent Adv.* Intrusion Detection, Menlo Park, CA, USA, Sep. 2011, pp. 161-180.