A Comparative Analysis of Latency, Jitter and Bandwidth of IPv6 Packets Using Flow Labels in Open Flow Switch in Software Defined Network

A. A. Olabisi[1], O. D. Adeniji[2], Abeng Enangha[3]
Department of Computer Science, Faculty of Science,
University of Ibadan, Ibadan, Nigeria.
Email: [1]akinola.olabisi@gmail.com, [2]sholaniji@yahoo.com,
[3]enangha.abeng@nnpcgroup.com

ABSTRACT

Software Defined Network (SDN) architecture was developed such that the OpenFlow protocol structures communication with the control and data planes can support network devices. SDN implements network protocols that take years of testing, standardization and interoperability. OpenFlow-based SDN is currently being rolled out in a number of networking devices and software in order to deliver substantial benefits to both enterprises and carriers Network. However there are no adequate provision for quality of service (QoS) in OpenFlow using Flow Label to reduce bits required as a field to match packets in internet protocol six (IPv6). The required latency, jitter and available bandwidth of packet transfer from the host node to the destination in the IPv6 packets using flow label must be available. 20-bit flow label field in an IPv6 packet header was used to match packets from one node on the SDN network .The focus of the study is to develop IPv6 packet matching mechanism in OpenFlow Software Defined Network using Flow Label. The network was implemented using mininet-2.2.2-170321-ubuntu-14.04.4-server-i386 simulator. Analysis of Latency, Jitter and Bandwidth of IPv6 Packets Using Flow Labels in Open Flow Switch was presented in the study.

Keyword - IPv6, OpenFlow Protocol, Matching Mechanism, Latency, Jitter, Bandwidth, Flow Label

Reference Format:

Olabisi, A. A., Adeniji, O. D., and Enangha, Abeng (2019), A Comparative Analysis of Latency, Jitter and Bandwidth of IPv6 Packets using Flow Labels in Open Flow Switch in Software Defined Network,
Afr. J. MIS, Vol.1, Issue 3, pp. 30-36.

1. INTRODUCTION

Software Defined Networking (SDN) is a new paradigm in networking that offers the opportunity to re-introduce Quality of Service (QoS) control in the Internet. The centralized nature of SDN significantly reduces the complexity that is commonly associated with end-to-end QoS. The main goal behind SDN is the focus on changing rigidity of conventional network into a flexible network through

programmable infusion to networking. In this regard, Open Networking Foundation (ONF) created an infrastructure to allow physical separation of Network Control Plane from the forwarding plane.

The aim of the study is to develop IPv6 packet matching mechanism in OpenFlow Software Defined Network using Flow Label. Quality of Service is the ability of a network element to have some level of assurance that its traffic and service requirements can be satisfied. QoS reflects the performance an application may require and experience in a network. The Internet Protocol (IP), which is the underlying technology of the Internet being used today, was not designed with QoS. It was initially designed to provide Best Effort (BE) service only. However, Internet Engineering Task Force (IETF) defined two major QoS control schemes known as Integrated Service (in short form -IntServ) and Differentiated Service (in short form -DiffServ) to accommodate applications that need certain Qos.

IntServ scheme provides an end-to-end QoS solution with bandwidth reservation and admission control at each network element. The IntServ reservation system ensures that the portion of bandwidth reserved by a flow, in every link that is used by the flow. Rather than providing an end-to-end guarantee for flow, DiffServ employs per-hop behavior (PHB) with aggregation for different classes of traffic.to-end guarantee for flow; DiffServ employs per-hop behavior (PHB) with aggregation for different classes of traffic. Although DiffServ Complexity is significantly lower than IntServ. The paper is organized as follows: Section one provides an introduction to the study, Section two reviews related works, Section three is on methodology while Section four presents the result and discussion. In section five conclusion and recommendation of the study was presented.

2. LITERATURE REVIEW

IPv6 supports more levels of addressing hierarchy, a much greater number of addressable nodes, and simpler auto-configuration of addresses. Flow label field found in an IPv6 packet header provides a more efficient way for marking packet, flow classification, and flow state lookup. In relation to this, how to use this field in a specific architecture to provide QoS support remains an open issue. Due to the tight coupling between a network's control plane -where the decisions of handling traffic are made and data plane –where the actual forwarding of traffic takes place, there are various challenges related to its management and evolution as reported in (Ren S., et al, 2016). Following the SDN principle of decoupling the control and data planes were the first standard interface designed specifically for SDN in Open Flow. The standard provides high-performance, Granular traffic control across multiple vendors' Network devices (Mirchev, 2016). OpenFlow provides a standardized way of managing traffic in switches and of exchanging information (protocol) as shown in Figure 1.

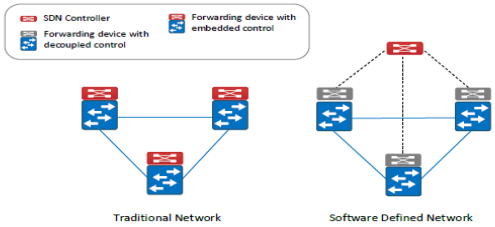

Figure 1: SDN vs. legacy Network Architecture

The end-to-end guarantees model with per-flow bandwidth reservation and admission control was proposed in (Sharma et al, 2017). The QoS and non-QoS traffic are identified by their DSCP bits. On discovering a route for a flow, two flow entries (for QoS and non-QoS flow) are installed in switches at the same time, but directed via two different queues. Bandwidth reservation for a flow is performed in its ingress switch. How the bandwidth is guaranteed in other switches in the path is not mentioned. The review in (Lu et al, 2016) was aimed at solving the scalability problem of IntServ by combining it with the aggregation of Diffserv, particularly in a cloud environment.

In the ingress switch's port (which is located inside a VM, and managed exclusively by a controller), a queue is created for all flows to reserve bandwidth. In the intermediate switches along the path, one queue per port is used to forward all QoS flows forwarded via this port. The rate of these queues is dynamic, with its initial value equal to zero. The max-rate is set for all queues, making sure that QoS flows will stay within the guaranteed rate in (Volpato F. et al, 2017). However, since only a single queue is used for

multiple QoS flows, QoS flows may contend with each other if there are QoS flows that send more than the guaranteed rate. OpenFlow1.2 which is the first version that supports IPv6 packet matching was proposed by. The first edition of OpenFlow focused on IPv4 and did not support IPv6 flow (Boley, et al, 2016). Xiaohan Liu proposed an IPv6 Virtual Network Architecture (VNET6) to support flexible services in IPv6 network. IPv6 is a critical protocol in VNET6 (Krishna, et al 2016). The VNET6 is adaptive to video service with high bandwidth and low tendency and improves quality of experiences to users. The Unified IPv6 Transition which unifies the variety of IPv6 transition mechanisms in Software Defined network was proposed by (Rowayda et al, 2018). Figure 2 present the flow table in an OpenFlow Switch.

MAC src	MAC dst	IP Src	IP Dst	TCP dport
*	10:20:.	*	*	*
*	*	*	5.6.7.8	*
*	*	*	*	25
*	*	*	192.*	*
*	*	*	*	*

Figure 2: Flow table in an OpenFlow Switch

2.1 Using Flow Label to Classify Flows of Packets

Flow Label is a new field in the IPv6 protocol proposed and defined as the length of 20 bits as shown in figure 3.

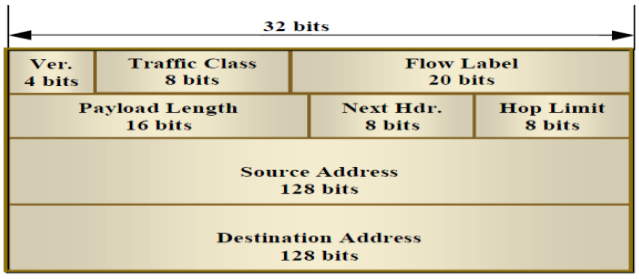

Figure 3: IPv6 Protocol Header Source: RFC 2460

Flow label can be resolved to meet the needs and rules by checking the flow label to determine which stream it belongs to. According to the forwarding rules, the routers and hosts which do not support the flow label need to set flow label field to all zeros, and the receiving packets do not modify the value of flow label. However, a unique flow must have the same attributes, including the source address, destination address, flow label as adapted in (Hesham, et al 2017). The flow label field enables an IPv6 enabled host in SDN network to label a sequence of packets for which the host requests special handling by the IPv6 routers [RFC2460].

3. METHODOLOGY

The method adopted in the study describes detailed steps required in the QoS scheme and simulation scenario of the topology of the developed scheme procedure. A further explanation on IPv6 simulation topology was equally presented.

3.1 Packet Matching Development

The packet matching approach in the developed scheme using Flow Label within IPv6 feature in IPv6 based OpenFlow consists of layer 3 to layer4 matching fields, such as IPv6 source address, IPv6 destination address, TCP/UDP source port, TCP/UDP destination port. Thus, the process based on traffic classification IPv6 flow label can be simplified to a one-step process of the developed scheme is shown below in figure 3.1.

Figure 3.1: IPv6 packets are matched against flow table in OpenFlow switcher

Layer 3 to layer 4 match fields in IPv6 network match fields based on OpenFlow. The IPv6 flow table based on OpenFlow allows the IPv6 host1 connect to an IPv6 host2, the Flow Label will be assigned according to the Packet-in packet, not the match fields of layer 3 to layer 4, such as IPv6 source address, IPv6 destination address and TCP/UDP source port, TCP/UDP destination port. The match fields of IPv6 address, which has 128 bits, will increase the size of flow table obviously. Thus the controller automatically and naturally determines the flow label number of the flow based on its location in the IPv6 network.

3.2 Developed Weighted Fair Queue Flow

Weighted fair queuing (WFQ) was adopted in the developed scheme as a method of automatically smoothing out the flow of data in packet-switched communication networks by sorting packets to minimize the average latency and prevent exaggerated discrepancies between the transmissions. The arrival time of packets was set to the depart time of the packet currently being outputted. Virtual time is updated every time a packet is dequeued, also virtual time progresses at different speeds depending on the amount of active bandwidth. The virtual time a session becomes inactive is the F of the last packet in the session, the session virtual finish time (SF). The smallest SF among active sessions and the corresponding time was calculated using the algorithm below.

enqueue(packet, i)

```
1  if not active(i)
2    activate(i)
3    active_r += r(i) -> Klein: r stands for rate
4  if queue(i) is empty
5    F(i) = SF(i) = max(F(i), V(t)) + L / weight
6  else
7    SF(i) += L / weight  -> Klein: weight = r(i). L is packet length
8  put(packet, queue(i))
```

dequeue()

```
1  i = min(active queues F(i))
2  packet = get(queue(i))
3  t += L / r
4  if active(i)
5    F(i) += Lnext / r(i)
6  for ever
7    j = min(active queues SF(j))
8    tmp_t = prev_t + (SF(j) - V(t)) * active_r / r
9    if tmp_t > t
10     V(t) += (t - prev_t) * r / active_r
11     prev_t = t
12     return packet
13   prev_t = tmp_t
14   V(t) = SF(j)
15   deactivate(j)
16   active_r -= r(j)
```

.

3.3 Network Topology of the Developed Scheme

The network topology consists of two OpenFlow switches (S1 and S2) in a topology consisting of four hosts. Switch s1 is connected to two hosts and switch s2 is connected to the remaining two hosts. Links via S1-eth0, S1-eth1, S2-eth2, S2-eth3 provide paths for H1,H2, H3, H4 respectively. Each host connected to S1 can send to or receive from a host connected to S2 for traffic generation. The traffic is bidirectional between these host pairs. The traffic used in the experiment is UDP, generated with iperf. The switches are Open vSwitch software switches installed in SDN Hub mininet simulator. The Open vSwitch used is OVS version 2.3.2, supporting OpenFlow 1.3. Any host connected to each of the switches is identified for communication in SDN network through the Ryu controller (C0). Figure 3.3 below present the network topology.

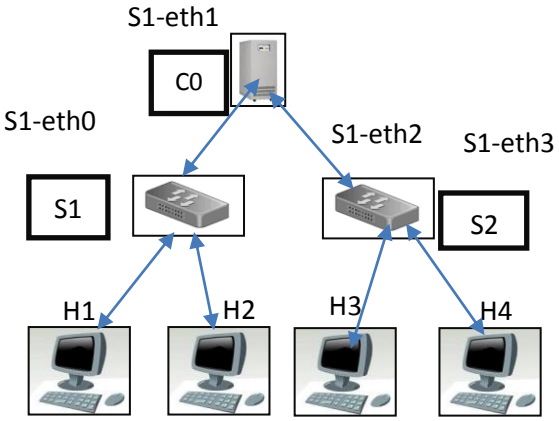

Figure 3.3: Network topology

The hardware and software specification used during the simulation modelling is shown in Table 1

Table 1: Hardware and Software Specification

Hardware and Software	**Version**
Machine: HP Dual Core	2GB RAM, 1.23GHZ
HDD	250GB HDD
Operating System	Windows 7 OS
Virtual Machine	VirtualBox-5.1.28-117968-Win_2
Mininet	mininet-2.2.2-170321-ubuntu-14.04.4-server-i386

4. RESULT AND DISCUSSION

The result during the simulation presents the communication between the remote controller and the rest of the network which was established through a Command Line Interface. The success of the connection will enable further activities on the network like sending a packet from one node to the other. The IP address shown on the CLI indicates that the network used for the research is strictly IPv6 i.e. fc::00/1 in hexadecimal notation. The figure 4.1 shows how messages are being sent from a source node to a destination node without any drop in packets.

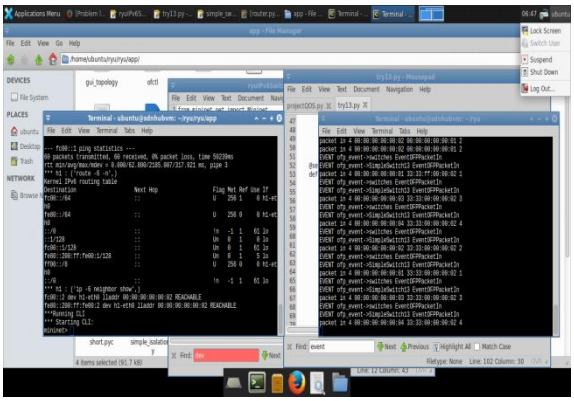

Figure 4.1: Message communication among the Switches and the Host

The developed network simulation was modelled using mininet simulator with Ryu controller. Components of the network are configured using Python programming language. The controller always shows its knowledge of the overall network like activities taking place. The ping6 command was created on the terminal and the result in Figure 4.2 was captured for analysis.

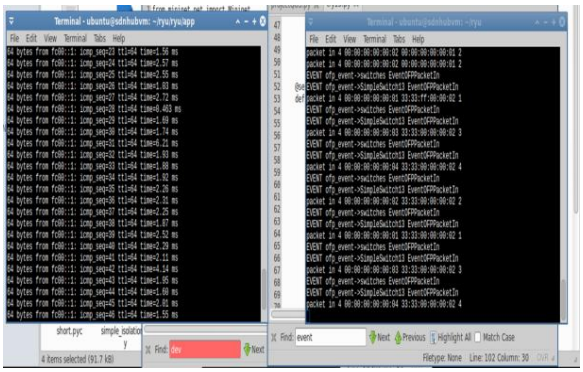

Figure 4.2: Ping6 command of IPv6 ICMP traffic message communication

The IPv6 router advertisement of packet matching during the simulation is shown in figure 4.3 below.

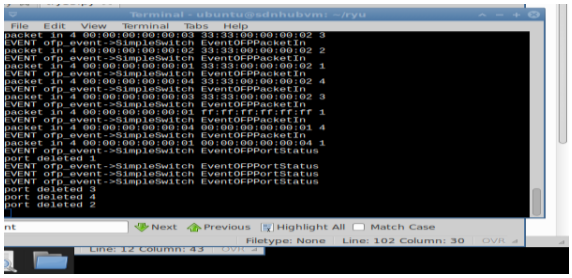

Figure 4.3:IPv6-Packet Matching during router Advertisement Messages

The developed mechanism using flow label provides the traffic for analysis. The table 4.1 shows the characteristic of the flow without flow label and with flow label.

Table 4.1: Analysis of Flow Label Mechanism and Jitter

Measurement	With Flow Label	Without Flow Label	Jitter
Minimum Time(ms)	0.458	0.47	0.012
Maximum Time(ms)	24.6	26.7	2.1
Average time(ms)	0.994	4.893	3.899
Mean Deviation(ms)	0.6	32.14	31.54

The comparison of the latency for the packets sent from host1 to host 2 during the simulation was captured. It is observed that the latency of flows of packets using the flow label is smaller than that of the other not using it. However, the time required in terms of latency to deliver services on the former, which uses the flow label bit is faster than the latter. Subsequently the bandwidth of the network is checked for comparison as well. Figure 4.4 below shows the information IPv6 routing table with the bandwidth.

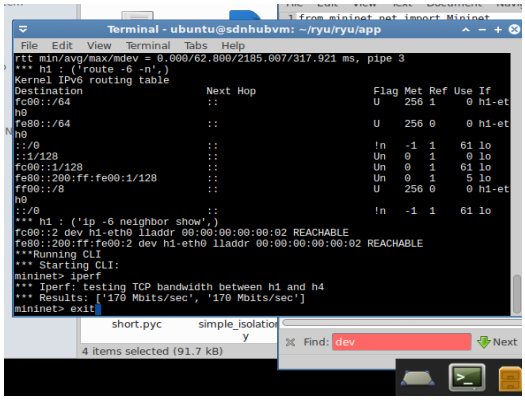

Figure 4.4: IP Routing Table Result of Message Transfer.

The result of latency comparison shows significant reduction in the time taken by flows of packets sent using the fair queue algorithm using flow label to classify flows as the first stage of QoS provisioning as shown in figure 4.5. With flow label and without flow label was evaluated.

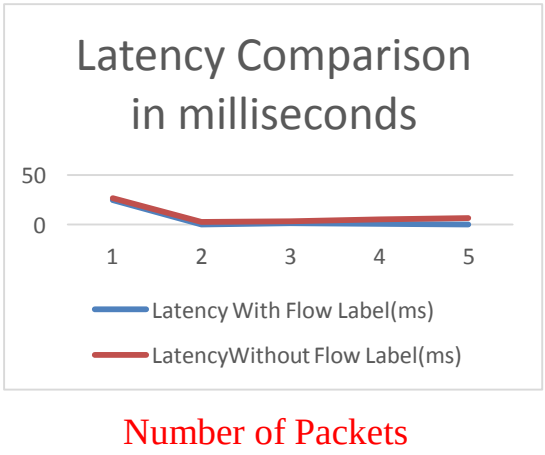

Figure 4.5: Latency Comparison With Respect to Number of Packets

Table 4.2 provides the information of result for latency of the flow label. With flow label and without flow label was evaluated as shown below.

Table 4.2: Result of Latency Evaluation

Packets	Latency With Flow Label (ms)	Latency Without Flow Label(ms)
1	24.6	26.7
2	0.546	2.91
3	1.5	3.87
4	0.784	3.16
5	0.778	3.98

Results of the implementation indicate significant improvement on the latency, jitter and bandwidth used to transfer flows of packets from one host to the other. It can be concluded that the proposed approach improves the QoS through its parameters like jitter, latency and bandwidth. The chart of the table 4.2 represented in figure 4.5 shows that during the experiment, the minimum from with flow label is 0.546ms, without flow label is 2.91ms. At minimum, the latency was 2.6%. In the same experiment, at maximum the value for with flow label was 24.6ms against 26.7ms without flow label. At maximum the percentage of latency is 7.8%. However, on average with flow label has a value of 0.994ms and without flow label a value of 4.893ms. It was gathered that 39.6% on average latency was recorded.

5. CONCLUSION

The developed scheme approach is to improve the QoS through parameters such as jitter, latency and bandwidth. The developed approach gives a better usage of the traffic with a jitter of 39% on average and 21% when the network is at the peak of the traffic usage. The bandwidth used is equally better around 10%. To guarantee that the significance and benefits stated in this research are realized, developer of network device should encode the OpenFlow switch to reduce the match field required for packets and adopt use of flow label fields to match flows of packets during the phase of their design.

REFERENCE

[1]. Mirchev, A. (2016). “Survey of concept for Qos improvement via SDN”, *Network Architectures and Services*, September 2015; doi: 10.2313/NET-2015-09-1_05

[2]. Sharma, S. (2017). “Implementing quality of service for the software defined networking enabled future internet”. In: Software *Defined Networks (EWSDN), Third European Workshop on*. IEEE, pp. 49–54.

[3]. Ren, S., Feng, Q. and Dou, W. (2017). "An End-to-End QoS Routing on Software Defined Network Based on Hierarchical Token Bucket Queuing Discipline", in DMCIT '17 *Proceedings of the 2017 International Conference on Data Mining, Communications and Information Technology DMCIT*, ACM, pp. 31.

[4]. Volpato, F., Da Silva, M. P., Gonçalves, A. L., and Dantas, M. A. R. (2017). "An Autonomic QoS management architecture for Software-Defined Networking environments", *ISCC*, pp. 418-423.

[5]. Krishna H, van Adrichem, N. L., and F. A. Kuipers, F. A. (2016). "Providing bandwidth guarantees with OpenFlow" in Symposium on Communications and Vehicular Technologies (SCVT), IEEE, pp. 1-6.

[6]. Boley, J. M; Jung, E.S. and Kettimuthu, R. (2016). "Adaptive QoS for data transfers using software-defined networking" in ANTS, IEEE, pp. 1-6.

[7]. Hesham, A.; Sardis, F.; Wong, S.; Mahmoodi, T., and Tatipamula, M. (2017), "A Simplified Network Access Control Design and Implementation for M2M Communication Using SDN", *WCNCW*, pp. 1-5.

[8]. Lu, Yiqin; Wang, Meng, and Huang, Pengsen (2017). An SDN-Based Authentication Mechanism for Securing Neighbor Discovery Protocol in IPv6 *Security and Communication Networks*, Volume 2017, https://doi.org/10.1155/2017/5838657

[9]. Ahmed, R. and Boutaba, R. (2016**).** Design considerations for managing wide area software defined networks. *IEEE Commun. Mag.*, *52*, pp. 116–123.

[10]. Rowayda, A. Sadek (2018). "An Agile Internet of Things (IoT) based Software Defined Network (SDN) Architecture". *Egyptian Computer Science Journal*, 42 (2), pp. 13–29.