Evaluation of QoS Provisioning over Software Defined Network using Segment Routing

Ohmmar Min Mon, Myat Thida Mon
University of Information Technology, Yangon, Myanmar
ommm@uit.edu.mm,myattmon@uit.edu.mm

Abstract

Provisioning quality of service (QoS) is a big deal to deliver different applications over the current internet. With the advancements of using multimedia applications, the necessity of Quality of Service (QoS) is increasing rapidly. As real-time applications increase, Software Defined Network (SDN) has emerged as a well-established paradigm for next generations networks. By utilizing the characteristics of SDN, this paper proposes QoS provisioning based segment routing (SR) over SDN framework to find the feasible path according to the QoS requirements. This QoS provisioning architecture includes monitoring of link states among switches and providing of flow's QoS requirements. This QoS provisioning is the use of the available bandwidth to react to the network traffic. The routing algorithm solves the problem of inefficient bandwidth. If there is no available bandwidth path, the controller will be decided depending on the flows by using the proposed algorithm. Simulation results are presented to show the effectiveness of QoS provisioning using OpenFlow/ONOS controller over SDN environment.

Keywords- Software Defined Network, QoS provisioning, Segment routing, OpenFlow

1. Introduction

Software Defined Network (SDN) has emerged as a new paradigm that can be implemented to adapt the existing network function. Providing QoS guarantee can give a strong guarantee to end hosts. With the development of the Internet, as a larger-scale networking system faces some unexpected challenges to satisfy various services request. Some applications, such as Voice over IP (VoIP), multimedia, video conferencing, HDTV etc. have been getting increasingly popular on the Internet. To meet the demand for QoS requirements, there is a Service Level Agreement (SLA) [1] between business customers and a service provider. There are many QoS parameters such as delay, bandwidth, jitter, loss probability, and cost, but the important one is bandwidth. If bandwidth for a packet flow is not enough, congestion will occur in bottleneck links, which causes severe packet drops and increases end-to-end delay.

SDN simplifies the QoS routing process and evolves rapidly. It has more advanced features while using traditional network function. SDN Controller receives the information from all switches in the network and based on the received information [9] as well as available network bandwidth information, a controller can build the network topology. Traditional network monitoring techniques such as NetFlow and sFlow support various kinds of measurement tasks. OpenFlow managed by the Open Networking Foundation (ONF) is the first popular implementation of SDN. The OpenFlow Switch performs data forwarding process based on the decision made by the Controller. Segment Routing (SR) is a new emerging traffic engineering technique and SR header contains a sequence of segments Segment Identifiers (SIDs) [12]. The segment labels are carried in the packet header and so per-flow state is maintained only at the ingress node. SR controller can take advantage of the possible segment routing by choosing segments based on the traffic. Signaling protocols are not required to accomplish resource reservation. The main challenge is how to achieve the best path for QoS flow. This paper takes full advantage of SDN's characteristic to implement QoS framework. This paper implements QoS provisioning for the available bandwidth for each application flow. The goal is to enable QoS provisioning in OpenFlow as one implementation of ONOS SDN.

The remainder of the paper is structured as follows: Section II briefly reviews the related work. Section III outlines SDN and segment routing architecture. QoS provisioning in SDN is proposed in Section IV of this paper. The performance with the evaluation experiments and test results on SDN testbed is discussed in Section V. Finally, section VI gives the conclusions and our future research.

2. Related Work

Several QoS routing algorithms have been suggested to achieve the best path using QoS aware routing algorithms. QoS problem with bandwidth and delay requirements using simulated annealing based QoS-aware routing (SAQR) algorithm to find the best fit path is

solved in [1]. However, it considered L2 legacy switches with SDN switches. The performance of the network by separating the application into bandwidth oriented and latency oriented application using routing algorithms such as Maximum Delay-Weighted Capacity Routing Algorithm (MDWCRA), Minimum Interference Routing Algorithm (MIRA) and Dynamic Online Routing Algorithm (DORA) for the multi-domain network is presented to increase network capacity in [3]. This system discussed that number of SD pairs affect BRR performance of the considered different routing algorithms.

QoS routing methods depending on application requirements and link cost values to measure the maximum bandwidth and delay between the proposed algorithm and traditional shortest path algorithm using Dijkstra algorithm are described in [4]-[5].

Available bandwidth is an important dynamic characteristic of a network path. Here [6] used the passive method to measure available bandwidth for any time. They discussed the bandwidth measurement overhead due to the passive way and [7] solved the problem of the lack of timestamp using OpenFlow. In our approach, we used the results obtained in different network configurations. The adaptive video is video streaming and DASH [8] is expected to be the future standard for adaptive video transfer. It discussed to obtain the appropriate path for video flows depending on the segment. It also considers the available bandwidth and bitrate of the segment.

R.Kumar [10] proposed mechanisms that provide end-to-end delays for critical traffic in real time systems using COTS SDN switches. This system shows that increasing the number of flows slightly decreases end-to-end delays. And [11] presented SDN/OpenFlow control framework that provides bandwidth guarantees for priority flows and implemented the experiments that proved its benefits in comparison with best-effort shortest path routing and IntServ. Our approach used the results for the available bandwidth of QoS flows using segment routing.

3. Software Defined Network and Segment Routing

In the SDN architecture, network architecture and the network intelligence is separated from the data plane. Forwarding is handled as flows. The controller has a logically centralized view of the flow, removing the requirement to carry such administrative information in packets. SDN scene has experienced significant growth in the number of projects and is investigated for various network functionalities such as security, quality of service etc. The most used standard is OpenFlow as shown in Figure 1. From a scale and simplicity perspective, Segment Routing is especially powerful in the era of SDN with application requirements programming the network behavior. SDN controller intelligence is used to map the optimal path onto segments. Segment Routing enables to use non-shortest paths by specifying alternative routes. Packets are forwarded through the shortest path from the source to the first segment, then to the second segment and so on.

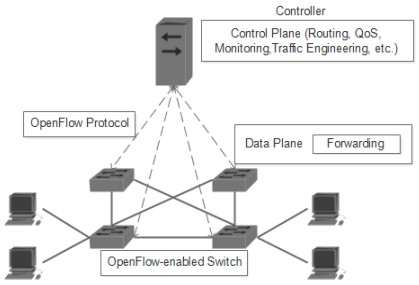

Figure 1. Software Defined Network Infrastructure

There are three actions that are performed on segments by SR-capable nodes. They are associated with operations performed on MPLS labels in MPLS networks. Segment Routing operations are: (a). PUSH – a segment is pushed on the top of segment stack (b) NEXT – an active segment is completed and it is removed from the stack (c). CONTINUE – active segment is not completed yet and it remains active. In Segment Routing network, it is enough to have an IGP protocol and once Segment Routing is configured, IGP will take labels and redistribute them within the domain.

In this paper, the switches update their forwarding tables according to the instructions taken from the controller. The switch informs the controller about the requested flow. The controller selects a path considering the requested bandwidth. After selecting the path, the controller sends flow information to the switches along the selected path using OpenFlow protocol. To determine the path, the controller needs to calculate the available bandwidth of the paths. For this purpose, the controller queries the switches periodically via sending OFPC_PORT_STATS messages which are defined in OpenFlow protocol to obtain information about available bandwidth on the links. When a traffic flow has to be routed along the shortest path to its destination, a segment list including only one label can be used (i.e., the SID of the destination node).

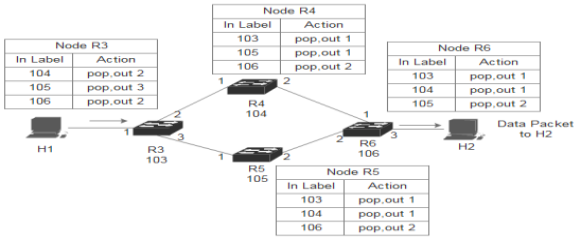

Figure 2. Example of Segment Routing

For the traffic from H1 to H2, if the controller selects the SR path P {R3, R4, R6}, a possible segment list used for P is SL= {106}. The packets are then forwarded along P without modifying the segment list to node R4 where the label 106 is popped and the packet is forwarded to node R6.

4. Quality of Service Provisioning

Quality of Service is an area of ongoing research and has been increased the interests in the research community. In today Internet, there are two main categories of QoS techniques: approaches proposed before SDN and SDN enabled approaches. When considering QoS approaches before SDN, the Internet Engineering Task Force (IETF) categories QoS into two major architecture: Integrated services (IntServ) and Differentiated Services (DiffServ). IntServ, per-flow design, specifies the elements to guarantee QoS and keep the network status information on every router. DiffServ, a class-based architecture, operates by classifying the traffic into classes with per-hop behavior. However, it cannot provide enough resources to guarantee QoS of different flows.

SDN enabled approaches to tackle all of the problems of the traditional network. SDN is a relatively new practice, and the cost of new technology is high. SDN controller can specify policies without the need to reconfigure low-level settings at each of the forwarding devices. The set of policies and the different flow classes are unrestricted. The rules can be defined per flow and the controller has the task to apply them properly to the different network elements. There are some QoS requirements of applications such as Bandwidth, hop-count, delay and jitter. In this system, the route selection of the flows is done by considering the paths from the controller to the switches. When a user needs a certain bandwidth, it sends a bandwidth request packet to the controller. Request packet contains information how much bandwidth it needs as a Packet-in message.

The SDN controller determines the segmented routed path in the network. When a new packet arrives at an OpenFlow switch, the switch will first check the packet header against all the preserved rules. If there is a match, then the switch will execute the matched rule action, otherwise, the network controller will be asked on how to deal with the incoming packet via receiving a *packet-in* request from the particular switch. Then, the controller will process the switch's request and respond by installing the proper rules through the *flow-mod* message.

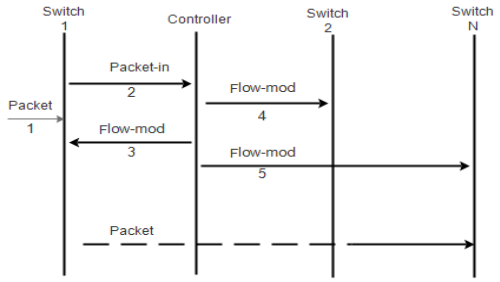

Figure 3. Interactions between Controller and OpenFlow Switch

Table 1. Notation Lists

Notation	Description
G(V,E)	The directed graph representation of the topology with vertex and edge
u_E	Utilization of link in the topology graph
C_E	The capacity of link
bw_u	The link usage bandwidth
bw_P	The available bandwidth of each path
$P_{S->D}$	The set of all available paths from S to D

The interactions between SDN controller and OpenFlow switch is as shown in Figure 3. This system uses this information to build up the network *G*(*V*, *E*) as shown in Figure 4, where the vertex *V* corresponds to the switches and the edge *E* corresponds to the links as shown in Table 1.

In this case, 'C' is the capacity of link, '*u*' is the link bandwidth utilization and bw_u is the bandwidth usage by monitoring the traffic flow of the link. For each link *E*, the available bandwidth resource of link E is $C_E - bw_u$. We define the available bandwidth of each path in Equation. (1):

$$bw_P = \min_{1 \le i \le n}(C_E - bw_u) \qquad (1)$$

Here, we have to get the path between source and destination in the network where the available bandwidth is the largest. This can be calculated through the following Equation. (2):

$$bw_a = \max_{P \in P_{S->D}} \min_{1 \le i \le n}(C_E - bw_u) \qquad (2)$$

Modified Dijkstra algorithm is used to get the path with largest available bandwidth. The cost of the path C_p is measured by the minimum bandwidth cost to obtain the best path according to Equation (3).

$$C_P = \sum_{i=1}^{n-1} C(bw_u, C_E) \quad (3)$$

where the cost of the path C_p is the sum of the capacities of the link.

The routing algorithm is committed to find the best path for specific QoS requirements. A feasible bandwidth is the one that can provide sufficient resource to satisfy all QoS requirements of the flow. The algorithm is divided into two steps. The first step is to find the feasible bandwidth which can assure flow's QoS requirements, while the second step is to find a best-effort flow when feasible bandwidth doesn't exist. If feasible bandwidth exists, one of the paths will be selected to transmit the flow. To explain QoS routing Algorithm, consider a Mininet testbed for the network with four nodes shown in Figure 4. In the testbed, source node is S1 and destination node is S2. When the flows entered the network, for the case of higher bandwidth application flow S1-S3-S2, S1 would push a SR header with segment list *{*101,103,102*}*, and forward it to S2. Best-effort flow should be steered over the shortest path, which is S1 to S2. S1 would place the segment list *{*101,102*}* in the SR header, and forward to S2. Flow Classification is defined by Table 2.

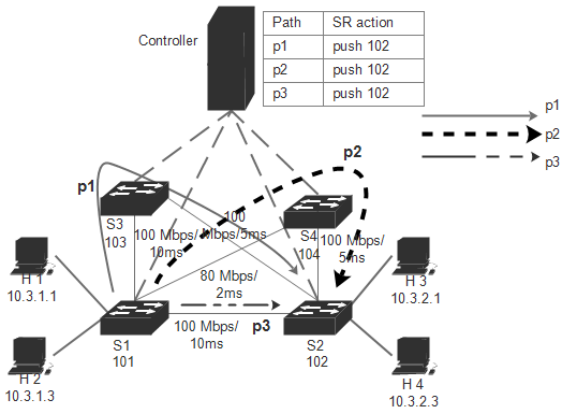

Figure 4. Test set up with Mininet

Table 2. Characteristics of Flows

Flow Type	Flow's QoS requirements	Flow
Minimum bandwidth flow (srp 1)	50 Mbps, < 10ms	H1-H3
Higher bandwidth flow (srp 2)	100 Mbps, < 20ms	H1-H3
Best-Effort flow (srp 3)	>100Mbps, -	H2-H4

```
//Create LeafSpine Topology//
Input: Switch, Host;
Initialize: S=0, H=0, N=4;
For all i=1 to N do
        Si =addSwitch (Si);
        Hi =addSwitch (Hi , IPi);
        Li = createLink (Si);
Topo T = S,H,L;
Return T;
```

Figure 5. Algorithm of SR Topology

Flow type is obtained after flow classification as the basis to specify the QoS requirements. In this paper, it distinguishes the traffic to guarantee the bandwidth to three types. The first type is one to provide minimum bandwidth flow (srp 1) such as VoIP. The second type is set to use higher bandwidth flow (srp 2) such as video conferencing. The remaining type is set to best-effort flow (srp 3). The experiment is employed with Mininet testbed to evaluate QoS routing. We used the pseudo code to emulate the Mininet testbed of the network as shown in Figure 5. This section includes the setup and verification of SR test application, utilizing Mininet and the ONOS controller.

The pseudo code for our algorithm is summarized in Figure 6. Let G(V, E) be the network graph where the node set *V* corresponds to the switches and the edge set *E* corresponds to the links. Then srp defines segment routing (SR) path, bw_f defines the feasible bandwidth, bw_a represents available bandwidth and d_f is the feasible delay.

```
Initialize# Finding Available Bandwidth
Input: Topology: G(V,E)
      Bandwidth Threshold bwmax; bwf ;
      Delay df ;
      n= number of available paths,

Initialize: bwa=0;
BEGIN
if (bwf ≥ bwmax)
      then bwa = bwf
        if (bwa ≤ 50 && df  < 10 )
            then
                Add bwa to srp 1
                goto FINISH:
        else if (50< bwa ≤100 && df  < 20)
                Add bwa to srp 2
                    goto FINISH:
           else Add bwa to srp 3
end if
FINISH
for all n ∈ N do
    If bwn< bwa || bwn< bwmax then bwn =0
endfor
```

Figure 6. Algorithm of QoS Routing

5. Experimental Results

We show the simulation results performed on different configurations to emphasize the capabilities and the performance of the QoS on a virtual machine (VM) with the configuration using Leaf-Spine architecture. This Leaf-Spine architecture is designed to provide very scalable throughput across thousands to hundreds of thousands of ports.

We created a Mininet testbed containing four nodes. The evaluation was performed on the Mininet testbed depicted in Figure 4. The script in this system is based on Python to generate flows. The Mininet testbed is comprised of 4 virtual switches interconnected with an SDN controller, 4 virtual hosts and virtual Ethernet links interconnecting the switches.

The SDN controller is an Open Network Operating System (ONOS) controller using OpenFlow configured to communicate with the data plane as shown in Table 3. The ONOS SPRING-OPEN project VM image has a 64-bit Ubuntu 16.04 installed as the guest OS. These are the minimum requirements to run the environment. The PC has Microsoft Windows 8.1 OS. The system was run with Core(TM) 1.6 GHz CPU and 4 GB of RAM. After collecting key performance parameters from both traditional network and SDN, this system model the data for a graphical representation.

In this system, the controller includes a routing policy based on a maximum bandwidth threshold 30Mbps between the switches. For this experiment, a Mininet testbed shown in Figure 4 is used. The requested services cannot provide if the request exceeds the threshold level of the link bandwidth. Srp 2 has a higher priority than srp 1 and we change the congestion level by injecting srp 3 into the network. The srp 1 and srp 2 having the higher level than srp 3 will have a higher priority queue to acquire sufficient bandwidth resource.

Hosts h1 and h2 send QoS traffic to h3 and h4 with the guaranteed rate of 30 Mbps. The actual rate sent by h1 is 100 Mbps and 50 Mbps respectively. Srp 3 rate between hosts is >100 Mbps. All flows are generated with iperf. The received throughput is observed from iperf's statistics.

Table 3. Parameters

Parameter	Values
Number of switches	4
Bandwidth threshold	30 Mbps
Delay sensitive threshold	20s, 10s
SDN controller	ONOS
Simulation tool	Mininet 2.2.1

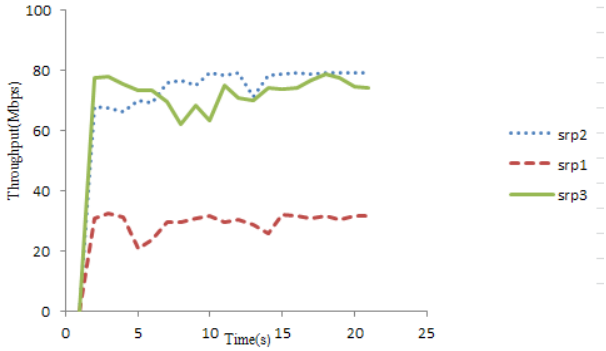

Figure 7. Throughput

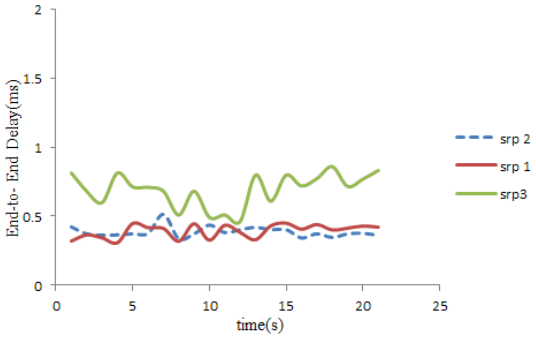

Figure 8. End-to-End Delay Variation

Figure 7 shows the time-varying throughput for srp 1, srp 2 and srp 3. We implement QoS control scheme at 15 s and detect the difference of throughput. Srp 1 uses the path S1-S4-S2 which is near 32 Mbps and Srp2 uses the path S1-S3-S2 which is 80 Mbps. Srp 1 and srp 2 are acceptable throughputs than srp 3 because srp 1 and srp 2 arrive at some peak rate at 20s. Congestion occurs when srp3 is at 10s and it is not available to guaranteed QoS. The routing algorithm has not acquired the proper route for srp 3 because of the constraint of delay.

The test result of delay variation is shown in Figure 8. The flows have suffered from huge variation at 10s. The srp 1 and srp 2 experienced at desired seconds. Srp 3 increases significantly at 10s due to congestion. The delay variation of srp 1 and srp 2 has decreased in 20s than srp3.

6. Conclusion

QoS provisioning is an important approach for several new architectures. This paper has implemented a solution to provide available bandwidth for QoS provisioning across SDN/OpenFlow network. This paper proposes QoS provisioning based segment routing over SDN environment to guarantee bandwidth efficiently for each application flow. In order to provide the QoS of higher bandwidth flow, segment routing is used to satisfy the

requirement of service. The available bandwidth via experiments is implemented over SDN using Mininet testbed and ONOS controller. Based on the simulation results, the performance of our proposed QoS provisioning can improve the QoS requirements since the network throughput is stable compared with the throughput results in Best-Effort flows. In our future work, we plan to conduct extensive experiments with more complex network topologies to have more traffic by using OpenFlow Protocol.

7. References

[1] L.Chienhung, W.Kuochen and D.Guocin, "A QoS-aware routing in SDN hybrid networks". FNC 2017.

[2] S.Tomovic, I.Radusinovic and N.Prasad, "Performance comparison of QoS routing algorithms applicable to large-scale SDN networks". In EUROCON 2015-International Conference on Computer as a Tool (EUROCON), September 2015, pp. 1-6.

[3] H.Cho, J.Park, J.M.Gil, Y.S.Jeong, and J.H.Park, "An Optimal Path Computation Architecture for the Cloud-Network on Software-Defined Networking". Sustainability, 7(5), May 2015, pp. 5413-5430.

[4] U.Pongsakorn, I.Kohei, U.Putchong, D.Susumu and A.Hirotake, " Designing of SDN-Assisted Bandwidth and Latency Aware Route Allocation". (HPC), July 2014, pp. 1-7.

[5] T.T.Nguyen and D.S.Kim, "Accumulative-load aware routing in software-defined networks". In Industrial Informatics (INDIN), IEEE 13th International Conference, July 2015, pp. 516-520.

[6] P.Megyesi, A.Botta, G.Aceto, A.Pescapè and S.Molnár, "Available bandwidth measurement in software defined networks". In Proceedings of the 31st Annual ACM Symposium on Applied Computing, April 2016, pp. 651-657.

[7] P.Megyesi, A.Botta, G.Aceto, A.Pescapé and S.Molnár, "Challenges and solution for measuring avaiclable bandwidth in software defined networks". Computer Communications, 99, February 2017, pp. 48-61.

[8] C.Cetinkaya, E,Karayer, M.Sayit and C.Hellge, "SDN for segment based flow routing of DASH". In Consumer Electronics–Berlin (ICCE-Berlin), 2014 IEEE Fourth International Conference, September 2014, pp. 74-77.

[9] D.J.Hamad, K.G.Yalda and I.T.Okumus, "Getting traffic statistics from network devices in an SDN environment using OpenFlow". ITaS, 2015 Sep, pp. 951-956.

[10] R.Kumar, M.Hasan, S.Padhy, K.Evchenko and R.B,Bobba, "End-to-End Network Delay Guarantees for Real-Time Systems using SDN", 2017.

[11] S.Tomovic, I.Radusinovic and N.Prasad, "SDN control framework for QoS provisioning", 22nd Telecommunications forum TELFOR 2014, Serbia, Belgrade, November 2014.

[12] C. Filsfils, N.K. Nainar, C. Pignataro, J.C. Cardona, and P. Francois, "The segment routing architecture", In Global Communications Conference (GLOBECOM), December 2015, pp. 1-6.