Journal of Information Systems Engineering and Management
2025, 10(30s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Quality of Service in Software Defined Network: Leveraging OpenFlow Protocol

Mrs. Medha N Kulkarni1, Dr. Jagdish W Bakal2
1Computer Engineering Department1 Pillai HOC College of Engineering and Technology Mumbai, India. mnkulkarni75@gmail.com
2Computer Engineering Department, Pillai HOC College of Engineering and Technology Mumbai, India. jwbakal@mes.ac.in

ARTICLE INFO

Received: 29 Dec 2024
Revised: 12 Feb 2025
Accepted: 27 Feb 2025

ABSTRACT

The research community has been showing an increasing amount of interest in the topic of Quality of Service (QoS) in computer networks. Now a days maximum network applications expect specific QoS assurance. In essence, Quality of Service (QoS) is essentially the capability of the network to provide the resources to satisfy the requirements of specific traffic. For instance, while streaming video there can be slight delay and when communicating data a lower packet loss rate is considered. The IETE had defined QoS architectures – IntServ and DiffServ. But both are not very much effective in supporting QoS for both end users and service providers. The Software Defined Networks (SDN) aims to increase network performance by using the facility of QoS which is available in OpenFlow. The paper briefs the QoS in conventional networks, along with a some study of OpenFlow with respect to QoS in SDN [1]. Additionally, this paper discusses the QoS abilities of SDN controllers which are commonly used.

Keywords: Software Defined Networks, Quality of Service, OpenFlow, IntServ, DiffServ, Meter

INTRODUCTION

The development of network technologies is accelerating, moving away from connectivity and toward the provision of necessary services to applications with desired quality and reliability. These apps each have their specific quality requirements and service features. For example, data transfer requires a lower packet loss rate and a video streaming application is sensitive to delays. Quality of Service (QoS) refers to the capacity of network to satisfy the demands of specific traffic. Failure to meet this criterion may result in a decrease in the Quality of Experience. So, setting up a network that is QoS-enabled becomes quite difficult and demands a lot of work to offer performance that is accessible for all kinds of traffic.

The conventional IP based networks follow "best-effort (BE)" delivery approach to deliver network services. Since the "best-effort" delivery model provides a point-to-point delivery service to get data to its destinations as quickly as possible, there are no assurances about bandwidth or latency. Reliable data delivery is through protocols like TCP is the highest assurance the network can offer. While this works well for conventional applications like FTP and Telnet, it is insufficient for VoIP, online gaming etc.

There are few QoS architectures defined by Internet Engineering Task Force (IETF) for QoS provisioning. Based on resource reservation protocol (RSVP) the Integrated Services (IntServ) model provide the QoS to end users. The typical hop-by-hop approach is used by Differentiated Services (DiffServ) paradigm. It was developed by Blake et al. (1998) and was based on the flow-aggregation. The Type of Service (ToS) field within a packet header serves to classify incoming flows based on predetermined classes. Multiprotocol Label Switching (MPLS) reduces routing table lookup complexity through its labelling method.

The traditional internet faces difficulties in allocating limited network bandwidth to network traffic efficiently while maintaining QoS guarantees for particular applications. To address Best-Effort model (traditional internet) limitations, numerous researchers use Software Defined Networks (SDN) as their solution. for better performance of networks. SDN carries a complete view of network topology for better allocation of network resources. Analysis of link utilization and the network congestion are the typical performance measurements. In the section Related Work, SDN architecture and commonly used OpenFlow protocol is discussed. Next two sections describe the QoS support in traditional IP-based network and Software Defined Networks using OpenFlow. In further section QoS support in various SDN controllers is briefed which is followed by concludsion.

RELATED WORK

A. SOFTWARE DEFINED NETWORKING

With the use of a higher-level abstraction, software-defined networking, or SDN, enables network managers to regulate network services. This is achieved by separating the system that decides traffic routing (control plane) from the underlying system that routes packets to the chosen destination (data plane) [1]. The dynamic, easy to control, cost-effective, and adaptable architecture of SDN was created to support the bandwidth-intensive applications of today.

The various features like directly programmable, agile, centrally controlled, program-configurable, built on open and vendor-neutral standards, and not reliant on network system makers like traditional networks are all offered by SDN's network architecture. The OpenFlow protocol is an essential component the SDN architecture [2]. The southbound protocol like OpenFlow is responsible for communication between SDN switches and the controller.

Software Defined Networking (SDN), created a new architecture that centralizes network control and reduces network management complexity by breaking vertical integration. Thus, multiple network devices are controlled by software in one place. In reality, SDN has provided the networks with all-encompassing intelligent control in the control plane with the controllers like Ryu, Open Daylight, Floodlight etc. The network managers can easily monitor, secure, and optimize the network resources with SDN applications. The flexibility is the key feature that distinguishes SDN from traditional networks is its flexibility. The key to its success is keeping the data plane and control plane apart.

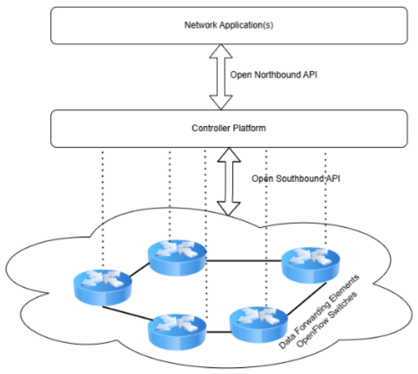

Figure 1 SDN Architecture

The SDN architecture is depicted in Figure 1. The application layer, control layer, and infrastructure layer (data layer) are the three layers that make up SDN architecture, as shown in the Figure 1. The common network applications or features that businesses utilize are found in the application layer. These applications include firewalls, load balancing, and intrusion detection systems. Traditional networks employ specialized modules for load balancing or firewalling. In SDN, the centralized controller that oversees data plane management takes the place of the discrete equipment.

Southbound and Northbound APIs are used for communicating through the three layers of SDN. The brain of SDN is the controller. It resides in the control plane of SDN. The controller is in charge of network flow and policy management. The data plane (infrastructure layer) of the network contains the actual switches. Applications communicate with the controller via the Northbound interface. The controller and switches communicate with each other using the Southbound interface. One example of a southbound protocol is the OpenFlow protocol. The SDN controller connects the network devices, check the current status of network and implement forwarding instructions to the control messages defined by the OpenFlow. OpenFlow offers thorough and adaptable traffic control, additionally.

B. OPENFLOW

The most widely used southbound protocol, OpenFlow, was established in 2008 at Stanford University and is currently run by the Open Networking Foundation (ONF) [4]. Through the installation of forwarding decisions on the network devices, this communication protocol enables the SDN controller to regulate the data plane. Using this protocol, the controller can add, remove, and alter flow table entries in the switch. Through the secure channel, the controller communicates with the OpenFlow switch. One of the interfaces is used to link OpenFlow switch to the controller is the secure channel. The controller uses this interface to control and manage the switch. The standardized OpenFlow protocol is needed to send each message through the secure channel.

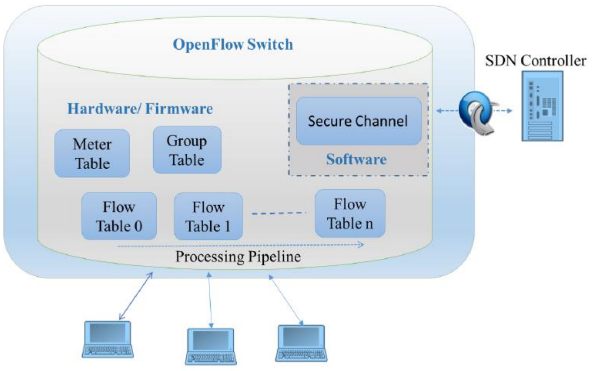

Figure 2 OpenFlow Architecure [2]

As illustrated in Figure 2, An OpenFlow switch is a software switch that has an OpenFlow connection to an external controller, one or more pipelined flow tables, and a group table that handles packet lookups and forwarding [3]. The flow table functions as lookup table with match fields and actions and with pipelined processing. Traffic flows are contained in pipelined flow tables. A flow that matches the initial flow table may be sent to a port or another flow table.

A sequence of packets sent from a particular source to a particular unicast, multicast or anycast, destination that is labelled as a traffic flow [4]. The 5-tuple - destination address, source address, protocol, destination port, and source port, is usually the foundation of flow classifiers. The main advantage of flow-based routing is that it does away with the necessity of performing per-packet lookups to the routing database. The first packet in a flow can have its route looked up, and each subsequent packet in the series can subsequently undergo the same transformation.

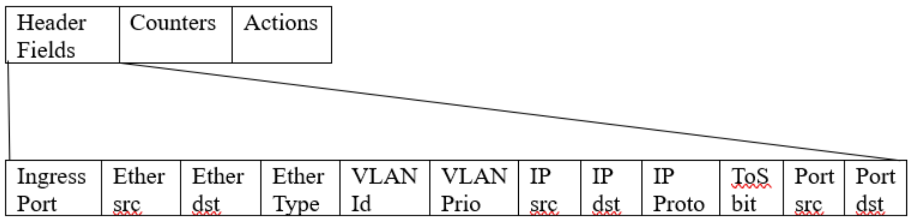

Figure 3 Flow Entry

OpenFlow in the data plane, manages packet forwarding for the OpenFlow controller [3]. Packet forwarding and lookups are managed by the flow tables. A flow entry, as illustrated in Figure 3, is composed of header fields (such as source and destination IP addresses and ports) to identify each flow uniquely, counters to record the number of times a flow entry is used successfully, cookies for annotation by the SDN controller, timeouts to regulate the amount of time a flow entry is kept in the flow table, and priority to assist the switch in selecting among several matches (if any).

Finally, the policy for properly matched packets is determined by certain actions. The switch begins searching for a match when a packet reaches it, and the action - such as forwarding the packet on a particular port is determined by the matched flow entry. Additionally, there is a unique flow entry known as a table-miss flow entry that matches every packet and has a priority of zero. All mismatched flows are caught in this entry. This setting could instruct the device to send or drop unknown packets to the controller. The controller has two options: discard the packet or put a new flow entry on the switches for the flow. This is as if-then-else L3 router. [5].

B.1 OPENFLOW FLOW TABLE

The flow table is a lookup table with match fields and actions and is processed like a pipeline. This flow entry will contain an action set which can either output the frame to a specific port, apply actions, or send the frame to another table. In the event of a table miss the frame is dropped by the switch. When there is no matching rule in the table to match the frame, a table miss happens. Each Flow Table contains the following fields like Matching Fields, Priority, Counters, Instructions, Timeouts and Cookies [3]. Recall that a flow router consists of a lookup table and an action set; this is the lookup table that matches on various fields in the packet header.

B. 2 OPENFLOW MESSAGES TYPES

The OpenFlow communicates using messages: controller-to-switch, asynchronous, and symmetric. The controller uses the controller-to-switch messages to query information from, transmit packets to, or configure the switch. Normally, the controller initiates the controller-to-switch message with or without being required to send a response from the switch. On the other hand, asynchronous messages are sent without solicitation from the controller. Examples of these include packet-in messages, flow-removed, port-status, and packet-out messages. Symmetric messages require a response from the receiving party. Examples of these are hello, error, echo, and experimenter messages. [3] OpenFlow defines a specification that one can use to talk to OpenFlow switches, and this technology will be utilized throughout this thesis to provide the mechanism to insert flows into these switches.

QOS PROVISIONING IN TRADITIONAL NETWORK

QoS provisioning over the network is essential to ensure high-quality performance for different applications. IETE suggest two major approaches for supporting QoS into IP based network:

1. Integrated Services (IntServ) - works with Resource ReSerVation Protocol (RSVP) [6]. It is one of the approaches to provide per-flow end-to-end QoS guarantee by allocating network bandwidth resources to guarantee QoS for a specific flow (e.g., a video streaming session). According to an application's QoS request, the network resources are allocated and subject to bandwidth management policy.
2. Differentiated Services (DiffServ) [7]: DiffServ was created to offer a basic and coarse method for aggregating and classifying network traffic flow. All flows are categorized by DiffServ into a small number of classes, and each class is given a unique "per-hop behaviour." For this, it uses six bits from the Type-of-Service (TOS) field in the IP header to specify per-hop actions. The clients then design a specific traffic profile and pay the network provider for it. The edge routers mark client packets. The 6-bit per-hop behaviour code from the IP header will allow the core router to determine what to do when the packets reach it.

Some of these QoS parameters may be time depending but may not be mutually independent. There are a number of different QoS protocols and algorithms to accommodate the need for these different types of QoS:

Multi-Protocol Labeling Switching (MPLS) [8]: MPLS is data forwarding technology that provides bandwidth management for flow aggregates via network routing control according to labels in packet headers.

Subnet Bandwidth Management (SBM) [9]: This signaling scheme that provides prioritization and categorization data link layer (L2) on IEEE 802 networks.

QOS SUPPORT IN OPENFLOW

One of the most common used southbound protocols is OpenFlow. In addition to major and minor changes from their predecessors, each OpenFlow version introduced a few new features. Table 1 lists the QoS-related features and changes made to the various OpenFlow specifications.

Table 1 – QoS related features in different versions of OpenFlow

OpenFlow	Specific Features
1.0	Enqueue action, minimum rate property for queues and new header fields
1.1	More control over VLAN and MPLS
1.2	Maximum rate property for queues and controller query queues from switches

1.3	Introduction of meter table, rate-limiting and monitoring features
1.4	Several monitoring features introduced
1.5	Replacing meter action to meter instruction

Depending on its ports, an OpenFlow-enabled switch may have one or many queues on its ports. OpenFlow has the capability to gather data about switch queues. OpenFlow lacks functionality to manage queue creation and modification tasks. The queue configuration can be established through the use of OF-CONFIG protocol [10] and OVSDB.

A. OVSDB

The two southbound protocols which help to control forwarding device operations beyond forwarding decisions are OF-Config and OVSDB. The OVSDB protocol governs switch tunneling processes as well as switch port status while managing both queue configurations and Quality of Service (QoS) control management [14]. The OVSDB protocol contains multiple tables including flow tables and port tables as well as NetFlow tables. and the extra tables handle both QoS and queue functions.

After configuring the switch's QoS and queues the OpenFlow set queue action directs flows to a specified queue. This operation will forward the flows that satisfy the matching criteria to the designated queue. The queues' minimum and maximum rates control the total flow rate at the egress when multiple flows are passing through the switch at once. The following characteristics of queues' minimum rate are stated in the OpenFlow specification [3].

The following characteristics of queues stated in the OpenFlow specification [3].

min rate - A queue's guaranteed minimum data rate. Depending on each queue minimum rate, the capacity is distributed proportionately. The switch will prioritize the queue to reach the specified minimum rate after the min rate has been set.

max rate - A queue's maximum permitted data rate. The switch will either delay packets or drop them to meet the maximum rate if the actual flow rate exceeds the specified queue's maximum rate.

The Linux-based OpenvSwitch implements queues using the Linux Kernels Traffic Control (TC) [13] application. By default, it supports FIFO, HTB, HFSC, and CoDel to enhance network quality and resource utilization.

B. METER TABLES

In OpenFlow 1.3, a new feature called meter table was introduced. While queues regulate the egress rate, meter tables [11] monitor the flows' ingress rate. Meters are more closely related to network flows. Per-flow meters are defined by meter and all meter entries found in a meter table. OpenFlow can use Per-flow meters to implement QoS techniques like rate-limiting. To implement sophisticated QoS frameworks such as DiffServ, queues per-port can be combined with meters. Before being forwarded, every flow that is connected to a meter must go through the meter and meter bands. Each flow's rate is measured by the meter and with the aid of Meter Bands, operations based on rates can be imposed.

The following elements make up a meter table entry:

Meter Identifier: The flows use this 32-bit unsigned integer identifier to uniquely identify which meter entry they are associated with.

Meter Band: It calculates each incoming attached flow's rate. Based on the measured flow rate, the meter band carries out the operations and stores the instructions. The instructions for processing the related packets that specify what to do when a flow reaches a predetermined rate are contained in each meter band. When the flow rate exceeds the meter's set rate, the meter band takes action [11].

Counters: It is a simple counter that is updated every time a packet is processed by a meter. It is primarily used for statistical analysis.

Although a meter can have more than one meter band defined, only one meter band can be used each time a packet goes through the meter. Drop and dscp remark are the two band types that specify how a packet should be handled. Traffic that surpasses the specified rate is affected by bands. Packets exceeding the band rate are dropped by the drop

49 band. A rate-limiting band can be defined with it. In contrast, the DSCP remark band is used to raise the IP header's DSCP field's drop precedence. DiffServ [15] can be implemented with it.

QOS IN SDN CONTROLLER

Queue configuration is beyond OpenFlow's capabilities. OF-CONFIG and OVSDB (OpenvSwitch Database Management Protocol) [12] protocols, which are implemented in OVS switches, manage the queue configuration. Software Defined Networks use a variety of SDN controllers, including Ryu, OpenDayLight, and ONOS. However, none of the controllers offer a uniform approach to queue management. Numerous proprietary and commercial SDN controllers from various vendors are available, each with a range of functionalities for supporting QoS. Additionally, there are a few open-source and cooperative projects that are actively being developed with assistance from the industry and research community. Table 2 lists a few of the collaborative, open-source, and operational SDN controllers along with information on how well they handle the QoS.

Table 2 – QoS features supported in various controllers

Controller	**Features and Modules**
ONOS	OpenFlow Metering (limited QoS Support)
FloodLight	QoS Module, QueuePusher
OpenDayLight	OVSDB API
Ryu	OVSDB API

CONCLUSION AND DISCUSSION

To address the issues of traditional networks efficiently, SDN was introduced with new architecture. The flow level programmability of SDN helps OpenFlow to provide the QoS requirements of the applications. With help of programmability, SDN and OpenFlow, most commonly used southbound protocol networks are becoming more intelligent and controllable. Network administrators can manage networks effectively. OpenFlow supports two features, the queue and meter for QoS in the SDN environment. A queue or queues are configured at egress port of OpenFlow switch to handle the packets. The queue management is handled outside of the OpenFlow. It is typically handled by OVSDB protocol.

The bandwidth is main resource of any network. With the increasing Internet traffic bandwidth demand is the common issue faced by all networks. One of the solutions is to increase the bandwidth but that is not cost effective for internet service provides and customers using it. At the same time, the applications like cloud services and multimedia streaming require strictly some resources as a requirement of Quality of Service. This requirement varies according to service type, price. QoS provisioning mechanism of a network depends on the user’s requirement, availability of resources. Providing high QoS in existing network architectures is a on-going open issue in the networking area.

REFRENCES

[1] Nwe Thazin, Khine Moe Nwe, “Quality of Service (QoS)-Based Network Resource Allocation in Software Defined Networking (SDN)”, International Journal of Sciences: Basic and Applied Research Journals (IJSBAR), ISSN: 2307-4531 [Online], January 2020.

[2] Diego Kreutz, Fernando M. V. Ramos, Paulo Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig, “Software-Defined Networking: A Comprehensive Survey”

[3] S. J. Vaughan-Nichols, “OpenFlow: The next generation of the network ?,” IEEE Computer 44 (8) (2011) 13-15. URL http://dblp.unitrier.de/db/jo-urnals/ computer /computer44.html#Vaughan-Nichols11

[4] “Open Networking Foundation,” accessed: 2016-05-25. [Online]. Available: https://www.opennetworking.org/

[5] O. N. Foundation, “OpenFlow-open networking foundation” [Online]. Available: https://www.opennetworking.org/sdn-resources/openflow (accessed August 23, 2016).

[6] R. Braden, D. Clark, and S. Shenker. “Integrated Services in the Internet Architecture: an Overview. RFC 1633 (Informational).” Internet Engineering Task Force, 1994. url: http://www.ietf.org/rfc/rfc1633.txt.

[7] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. “An Architecture for Differentiated Services. RFC 2475 (Informational).” Updated by RFC 3260. Internet Engineering Task Force, 1998. url: http://www.ietf.org/rfc/rfc2475.txt.

[8] E. Rosen, A. Viswanathan, R. Callon, RFC 3031: Multiprotocol Label Switching Architecture (2001). URL www.ietf.org/rfc/rfc3031.txt

[9] Y. Yiakoumis, K.K.Yap, S. Katti , G. Parulkar, and N. McKeown, “Slicing home networks,” in Proceedings of the 2nd ACM SIGCOMM workshop on Home networks (HomeNets '11). ACM, New York, NY, USA, pp. 1-6, 2011.

[10] H.E. Egilmez, “Distributed QoS Architectures for Multimedia Streaming over Software Defined Networks,” Multimedia, IEEE Transactions on, vol.16, no.6, pp.1597-1609, Oct. 2014.

[11] “OpenFlow Switch Specification”. [Online]. Available: https://www.open-networking.org/wp-content/uploads/.../ OpenFlow-spec-v1.3.1.pdf. (accessed August 1, 2015).

[12] B. Pfa_, B. Davie, RFC 7047: The Open vSwitch Database Management Protocol (2013). URL www.ietf.org/rfc/rfc7047.txt

[13] D. G. Balan and D. A. Potorac, “Linux HTB queuing discipline implementations, IEEE First International Conference on Networked Digital Technologies, Ostrava, Czech Republic, pp. 122-126, Jul. 2009.

[14] “Open vSwitch”. [Online]. Available: http://openvswitch.org/support/

[15] Krishna, H., van Adrichem, N., & Kuipers, F. (2016). “Providing Bandwidth Guarantees with OpenFlow” In 2016 Symposium on Communications and Vehicular Technologies (SCVT) (pp. 1-6). IEEE. https://doi.org/10.1109/SCVT.2016.7797664