Configurable and Adaptive QoS Management via SDN

Yi Zhang
CS538 Advanced Computer Networks
Department of Computer Science
University of Illinois at Urbana-Champaign
yzhng173@illinois.edu

Qi Wang
CS538 Advanced Computer Networks
Department of Computer Science
University of Illinois at Urbana-Champaign
qiwang11@illinois.edu

Abstract

In broadband access networks (e.g., home networks), bandwidths are limited. In the meanwhile, different applications have different Quality of Service (QoS) requirements and they all compete for the scarce bandwidth. Unfortunately, many users are not savvy enough to configure QoS parameters of the underlying network to meet their needs. In this paper, we present *QoSManager*, a configurable and adaptive QoS management framework where users can specify QoS requirements for different applications at a high level. It monitors network and dynamically installs traffic shaping rules for application flows. In QoSManager, we propose a novel algorithm that takes QoS requirements and flows information as input and outputs an assignment of flows to queues of different rate to satisfy as many QoS requirements as possible. We implement QoSManager on Ryu and Open vSwitch (OVS) and demonstrate that it can provide QoS guarantee according to users' specifications in the presence of active competing traffic.

1. Introduction

Communication networks nowadays have greatly improved the connectivity of the world. This connectivity inspires and leads to a variety of online and real-time applications, like voice over IP (VoIP), video streaming, online interactive gaming, video conferencing, etc. These applications impose diverse Quality of Service (QoS) requirements on latency, bandwidth, jitter and error-rate. On the other hand, the underlying networks can not provide unlimited bandwidth. According to [5], the average connection speed in the United States is 12.6Mbps. When users use multiple applications at the same time, these applications compete for the relatively scarce bandwidth, thus leading to degradations of overall performance.

A common approach to this problem is to prioritize applications' traffic flows so that it can effectively enforce QoS of high priority applications. To this end, many QoS mechanisms have been proposed and implemented ((**TODO: citations**)). Nonetheless, these mechanisms have not been deployed in small scale broadband access networks (e.g, home networks), for several reasons. First, many home routers have limited memory and computation resources and may not be able to process and enforce complicated QoS requirements "on-the-fly" ((**TODO: citations**)). Second, users' demands of QoS for different applications may change in different scenarios. For example, in a home network setting, a user may demand high definition (HD) quality of videos while no one is playing online video games. When other users are playing online video games, she is willing to accept standard definition (SD) quality of videos. However, many home users do not have knowledge to configure the underlying networks to meet their needs.

Recently Software Defined Networking (SDN) has emerged as a promising approach for providing flexible network programmablility. SDN proposes to decouple the control plane from the data plane, and therefore it leaves the exsiting routers and switches as simple forwarding devices. The control logic is centralized

CS538 2016 Spring, University of Illinois at UrbanaChampaign.

http://dx.doi.org/10.1145/2502323.2502331

and deployed on a server, called SDN controller, which facilitates dynamic configuration, operation and monitoring of a network. **(TODO: More contents can be added here.)**

In this paper, we present QoSManager, a configurable and adaptive QoS management framework based on SDN. In QosManger, users specify high-level QoS requirements (e.g, minimum bandwidth, recommended bandwidth and priority) for different types of traffic, and QoSManager controller will dynamically assign each flow to a queue with appropriate rate to maximize the QoS requirements satisfied under the limited link capacity. QoSManger provides an interface for modifying QoS requirements at runtime (configurable) and recompute the assignment of flows to queues when flow is added to or removed from the network (adaptive).

This paper presents several contributions. First, we take into account that the QoS requirement for an application may change in different scenarios when designing the specification language. Second, we design and implement QoSManager, a configurable and adaptive QoS management framework using SDN support. Specifically, we propose a novel algorithm to assign flows to queues with appropriate rates to effectively enforce QoS. Third, we demonstrate the effectiveness of QosManager by using Mininet [3] and D-ITG [1].

The rest of the paper proceeds as follows. Section 2 presents related work in QoS, as well as SDN-based solutions for home and broadband access networks. Section 3 describes the design of our system, as well as its implementation using Ryu and Open vSwitch. Section 4 evaluates our system in the context of competing flows. Section 5 discusses future work and concludes.

2. Related work

2.1 Traditional QoS strategies

There are two types of strategies in traditional QoS: integrated services and differentiated services. Integrated services are fine-grained and per-flow.

- Every network element has to reserve resources for each flow.
- Router has limited computational resources; hard to classify all app flows.
- Not scalable.

Differentiated services [2] are more coarse-grained.

- Rely on the 8-bit DS field in the IP header. DiffServ routers then decide on per-hop basis how to forward packets based on their class.
- It is static (because of the predefined number of classes) and lacks the ability to fine-tune the QoS of separate flows.

2.2 SDN enabled QoS approach

2.2.1 Resource Reservation

FlowQoS [8] and EuQoS [9]

2.2.2 Per-flow Routing Frameworks

For Open-QoS [7], after classification, high priority flows are placed on QoS guaranteed routes.

2.2.3 Queue Management and Packet Scheduling

2.2.4 Policy Enforcement

2.3 QoS in home netowrks

Several other approaches explore QoS in home networks. Yiakoumis et al. proposed letting users notify the ISP about their bandwidth needs for a given application; in this case, provisioning occurs in the ISPs last mile, not in the home. Georgopoulos et al. proposed an OpenFlow-assisted framework that improves users quality of experience (QoE) in home networks for multimedia flows, subject to fairness constraints. The system allocates resources to each device but does not perform per-application or per-flow QoS. Mortier et al. developed Homework, a home networking platform that provides per-flow measurement and management capabilities for home networks. Homework allows users to monitor and control per-device and per-protocol usage, but it does not provide QoS support or perform any application classification.

3. QoSManager Framework

In this section, we describe the design and implementation of QoSManager. We first present an overview of the design and then describe each component in detail.

3.1 Overview

QosManager is built on top of Ryu SDN controller. As depicted in Figure. 1, the main components in QosManager are: **(TODO: delete the arrow from traffic monitor to control module)**

- *Web Portal*: The web portal provides an interface for users to set QoS parameters for different types

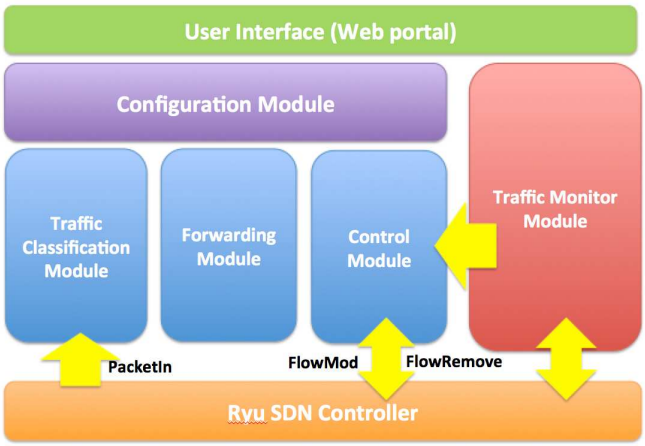

Figure 1: QoSManager architecture

of traffic. In addition, users could also view the real time statistical information about traffic flows in the web portal.

- *Configuration Module*: The QoS requirements are stored in a QoS configuration file in YAML format. Configuration module mainly handles the configuration file and passes the QoS requirements to the underlying three modules (traffic classification, forwarding and control, respectively).
- *Traffic Classification Module*: The traffic classfication module classifies the application type for each flow. In the current implentation, the classification is performed by querying a statically defined database.
- *Forwarding Module*: The forwarding module is responsible for making forwarding discussions (e.g, to which port should send the packet). Currently, we implement a simple L2 switch. The more sophisticated forwarding module are discussed in Section 5.
- *Control Module*: The control module is the core of QosManger. It dynamically computes the assignments of flows to queues with appropriate rates based on the Qos requirements and global information about the flows in the network.

When the first packet of a flow arrives at the switch, a copy of this packet is forwarded to the controller. The switch continues to perform default forwarding of the traffic flows until application identification has been performed, essentially lazily implementing QoS only once application classification is complete. The controller determines which classifier should classify the application type for the flow. Each application type is associated with a different queue, each of which is shaped according to the traffic shaping policy for the application.

3.2 QoS Parameters

In the configuration file, users only need to configure 3 parameters for a services. The following figure is part of a configuration file. The minimum rate parameter specifies the minimum bandwidth the service requires.The recommended rate parameter specifies the desired bandwidth for the service. The priority parameter specifies the users' prefenrece for the service. For example, in the figure, the minimum rate for video is 1.5M and the recommended rate for vidoe is 7M. The priority for video, which is 10, is much higher than other types of services. So our SDN controller will try to first satisfy the QoS requirements for the video services in the network.

User define configuration in YAML format

3.3 Flow classfier module

The flow classifier maintains a lookup table where the key is a flows five-tuple (e.g., source IP address, destination IP address, protocol, source port, and destination port). When a sender initiates a new flow, the switch sends a copy of the first packet of the flow to the controller. The flow classifier then checks whether the flows tuple correspond to an entry in this lookup table. The lookup then returns the type of application, such as video, VoIP, P2P, gaming, or web.

As traffic classification is not the focus of our project, in our implementation, we use static flow classifer to classfity the traffic.

It is very easy to extend our traffic classfication module to be more fine-grained and flexible. For example, we can easily integrate other approaches, such as DNS-based classifier [8], to our module. Or we can use the nmeta project [4] as our traffic classification module.

3.4 Traffic monitor module

- Flow statistics
- Port statistics

3.5 Control Module

On a high level, our control module tries to satisfy a maximum number of QoS requirements with the limited bandwidth. Each type of service has been configured by the user with a priority. We assign a weight to the service based on the actual bandwidth it will get.

For example, the weight of a service is 0.6 if the minimum rate bandwidth is achieved. The weight is 1.0 if the recommended rate bandwidth is achieved. Our algorithm allocates bandwidth to different services to get the maximum value of

$$\sum_{i=1}^{n} Weight_i * Prioriy_i$$

under the condition that

$$\sum_{i=1}^{n} Bandwidth_i <= C$$

. C is the total bandwidth of the network.

3.5.1 Dynamic queue assignment algorithm

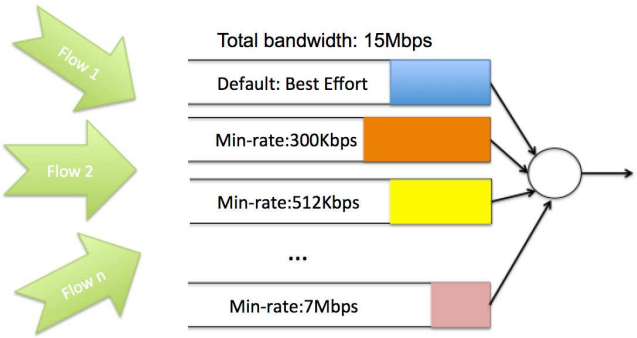

Figure 2

3.6 Web Portal Module

This module is for user configuration and traffic statistics.

3.7 Implementation

Implementation is based on Ryu SDN controller [6] and OpenVSwitch.

4. Evaluation

We now present the evaluation of our system.

4.1 Experiment Setup

Our experiments are done on a desktop PC with Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz, 16GB RAM. The network scenario is emulated using the Mininet [3] framework. It uses OS features to instantiate lightweight virtualisation of network hosts, and interconnects them with virtual switches, according to a specified topology configuration. We implemented the control application on top of Ryu [6], an open-source SDN controller.

Figure. 3 shows the experiment setup. We configuraed an Internext connection to be 15 Mbps.

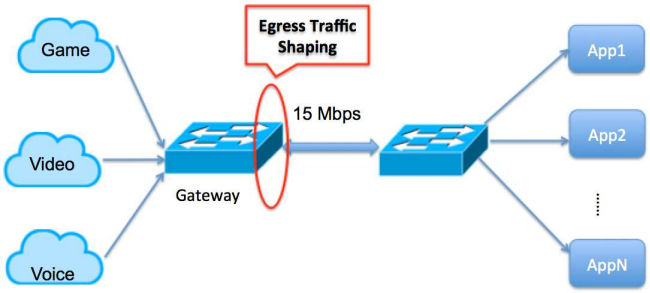

Figure 3

During the experiments, hosts request different service, such as watching a video or making a VoIP call, depending on the experiment.

We now show our system imporve the QoS of services in the face of competing traffic.

4.1.1 Scenario 1

In this scenario, one host is watching videos and the other host is downloading a file. And the priority of video is configured to be higher than file downloading. We generate traffic for the two services at the same time.

Figure illustrates the bitrates of the two services with our system, and Figure shows the bitrates of the two services withour our system.

The results show that our system allows the system to quickly converge to a higher bitrate than it otherwise would without FlowQoS enabled. This prevents bitrate oscillations and ensures the stability of the video player in terms of requested bitrates and video quality. Thus, our system improves the quality of the adaptive streaming video by both reducing bitrate oscillation and achieving a higher overall bitrate.

4.1.2 Scenario 2

The setting of this scenarios is the same as last one. But we generate traffic for file downloading first, then generate traffic for video, then we stop traffic for vidoe to see how the file downloading rate change.

4.1.3 Scenario 3

We also evaluated our system in the context of VoIP application traffic. VoIP is sensitive to delay and variation

in packet arrival times, so lower jitter is essential for good performance. We monitor the packet delay and jitter of the VoIP application using ping and iperf to monitor the RTT and the packet arrival times throughout the experiment.

5. Conclusion and Future work

For the future work, we plan to add more features to our system

- Multiple path routing
- Time-based QoS
- Different device QoS

References

[1] D-ITG, Distributed Internet Traffic Generator. http://traffic.comics.unina.it/software/ITG/.

[2] Differentiated services. https://en.wikipedia.org/wiki/Differentiated_services.

[3] Mininet: An instant virtual network on your laptop (or other pc). http://mininet.org/.

[4] Nmeta. http://mattjhayes.github.io/nmeta/.

[5] Q3 2015 state of the internet report. https://www.stateoftheinternet.com/downloads/pdfs/2015-q3-state-of-the-internet-report-infographic-americas.pdf.

[6] Ryu sdn framework. https://osrg.github.io/ryu/.

[7] H. E. Egilmez, S. T. Dane, K. T. Bagci, and A. M. Tekalp. OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks. In *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–8. IEEE, December 2012.

[8] M. Said Seddiki, Muhammad Shahbaz, Sean Donovan, Sarthak Grover, Miseon Park, Nick Feamster, and Ye-Qiong Song. Flowqos: Qos for the rest of us. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, pages 207–208, New York, NY, USA, 2014. ACM.

[9] Sachin Sharma, Dimitri Staessens, Didier Colle, D Palma, J Goncalves, R Figueiredo, D Morris, Mario Pickavet, and Piet Demeester. Implementing quality of service for the software defined networking enabled future internet. In *The European Workshop on Software Defined Networking, Proceedings*, pages 49–54. IEEE, 2014.