ENSC 833-3: NETWORK PROTOCOLS AND PERFORMANCE
CMPT 885-3: SPECIAL TOPICS: HIGH-PERFORMANCE NETWORKS

Smart Queue Scheduling for QoS

Spring 2001
Final Report

By
Haijing Fang(hfanga@sfu.ca) & **Liu Tang(**llt@sfu.ca)

1. ABSTRACT

In this project, we try to implement a QoS router. The essential of QoS is to provide different resource for different quality of service. Our idea is using queue scheduling to allocate available resources. To realize our idea, we choose OPNET as platform to develop our router model and simulate it. We also build a network from the scratch to test our QoS router model. With several sets of parameters, we test our model and it works fine. After analyzing the collected statistic data, the following conclusions are drawn: first, queue can be used to allocate the bandwidth. Second, shorter queue leads to smaller delay bound, but larger packet loss.

2. INTRODUCTION

What is QoS:
Quality of Service (QoS) means providing consistent, predictable data delivery-service. In other words, satisfying customer application requirements. QoS is the ability of a network element (e.g. an application, host or router) to have some level of assurance that its traffic and service requirements can be satisfied.

Today's QoS Technology:
According to different end-to-end QoS capabilities, different service models should be deployed. QoS service models differ from one another in how they enable applications to send data and in the ways in which the network attempts to deliver that data. Basically, there are three kinds of service models:

- Best-Effort Service
  A single service model in which an application sends data whenever it must, in any quantity, and without requesting permission or first informing the network. For best-effort service, the network delivers data if it can, without any assurance of reliability, delay bounds, or throughput.

- Integrated Service
  A multiple service model that can accommodate multiple QoS requirements. In this model, the application requests a specific kind of service from the network before sending data. The request is made by explicit signaling. The application is expected to send data only after it gets a confirmation from the network. The network performs admission control, based on information from the application and available network resources.

- Differentiated Service
  A multiple service model that can satisfy different QoS requirements. However, unlike the integrated service model, an application using differentiated service does not explicitly signal the router before sending data. For differentiated service, the network tries to deliver a particular kind of service based on the QoS specified by each packet.

Below is the architecture of an Integrated QoS model:

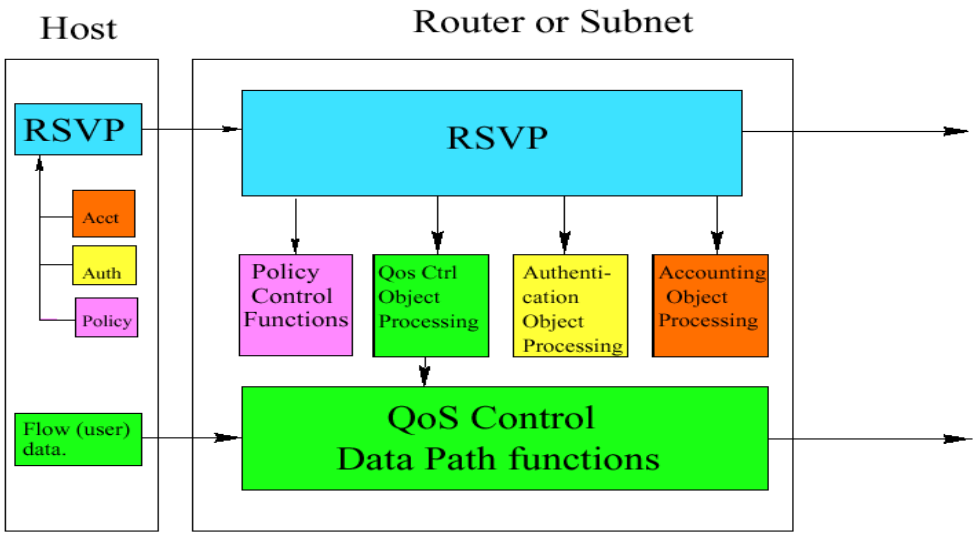

Figure 1: High level architecture for integrated services

Following is the detail diagram for QoS Control Data Path functions:

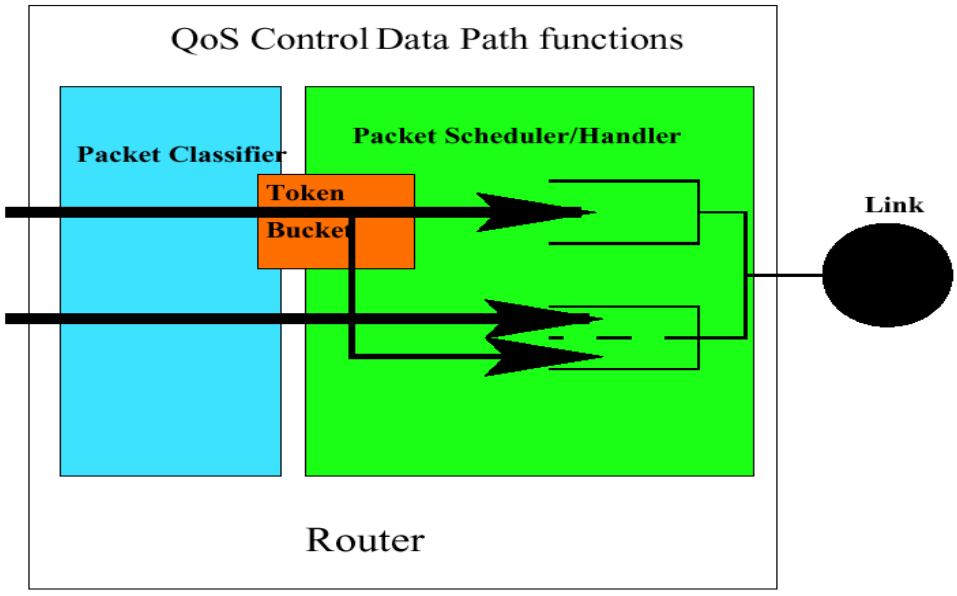

Figure 2: Diagram for QoS Control Data Path functions

So in a typical QoS router, there are three key components:
1. The admission control -- implements the decision algorithm that a router or host uses to determine whether a new flow can be granted the requested QoS without impacting earlier guarantees.

In the integrated model, it uses RSVP (Resource reSerVation Protocol) to negotiate with the host and make a QoS contract.

2. The packet scheduler -- manages the forwarding of different packet streams using a set of queues and perhaps other mechanisms like timers.

3. The classifier -- for the purpose of traffic control (and accounting), each incoming packet must be mapped into some class. All packets in the same class get the same treatment from the packet scheduler. Choice of a class may be based upon the contents of the existing packet header(s) and/or some additional classification number added to each packet.

Our Idea:
In our project, we implement an Integrated QoS model in a router. Within the router, we manage the resources by queues and let the router perform traffic control by scheduling those queues.

In the simulation, we assume that a QoS connections is established, and every incoming packet is Classified. So our job focuses on implementing a packet scheduler.

3. MAIN SECTION

Goal:
We design and implement a simple router that has capacity of QoS. To test it, we construct a network from scratch, that is, 'custom' building all models ourselves and not using 'general' available models. The reason of this choice is to have full control of the network. From the simulation, we collect statistic data and draw a conclusion.

Design:
1. Architecture:

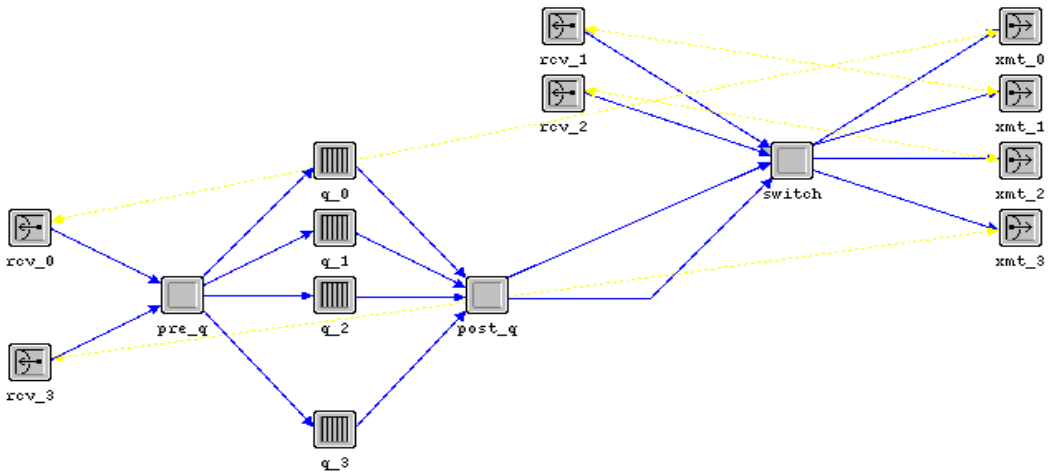

Figure 3: Architecture of Router

As presented in the figure, the router consists of three control components, a “pre_q”, a “post_q”, a “switch”, four queues “q_0/1/2/3”, and four pairs of input/output ports “rcv_0/1/2/3”, “xmt_0/1/2/3”.

2. Work Flow:

Our design is using queues to allocate bandwidth. Here, we have four queues, each of which has an equal opportunity to occupy the output channel. Thus each queue has one forth of bandwidth. How many queues a transmission occupies then how much bandwidth it has. We divide these queues into two classes, queue_class_1 and queue_class_2, for two types of services. Queue_class_1 has three queues, and queue_class_2 has the rest one. Class_1 is for the high priority service that requires less latency, while the class_2 is for the low priority service that doesn’t care much about latency.

Assumption:

- We have two kinds of QoS, class_1 and class_2.
- Every packet is pre-defined a QoS, either class_1 or class_2.
- A packet is classified when it comes into the router.

The assumptions are implemented in our experiment like this: first, we specify the QoS of traffic by its source. Second, we use a fixed connection pattern. That means we always connect class_1 traffic to “rcv_0” of the router and class_2 to “rcv_3”. Thus, from the router’s point of view, every packet coming in from the “rcv_0” is supposed to get QoS of class_1, while every packet from the “rcv_3” get QoS of class_2.

The “pre_q” picks up packets from input-stream. If a packet is from “rcv_0”, it will be put into one of queues of queue_class_1. The queue is selected from the queue_class_1 in a Round-Robin scheme. Otherwise, the packet will be put into “queue_3”.

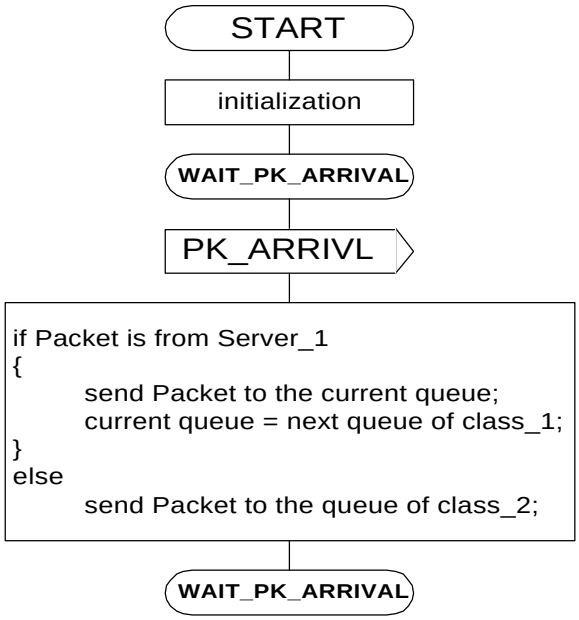

Figure 4. flow chart for “pre_q”

The “post_q” checks the four queues in a Round-Robin scheme. If a queue is not empty, it picks up a packet and forwards that packet to the “switch”.

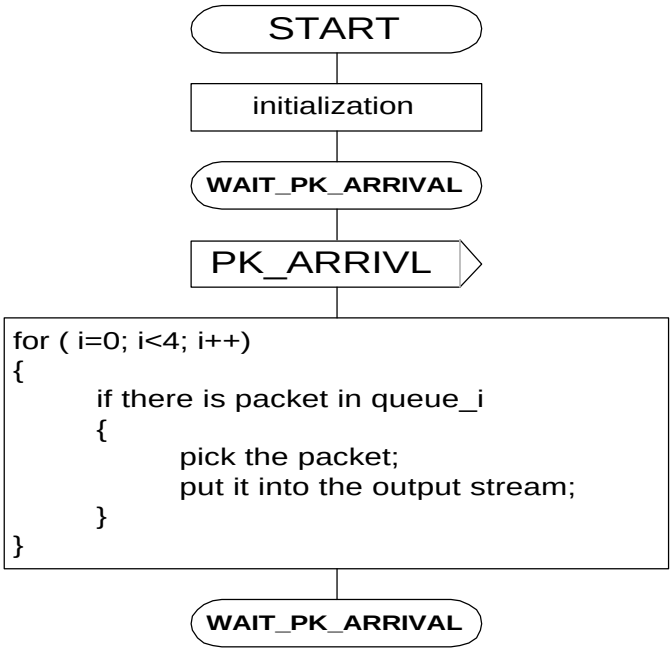

Figure 5. flow chart for “post_q”

The “switch” retrieves the destination address of each incoming packet, and outputs the packet to corresponding output-port.

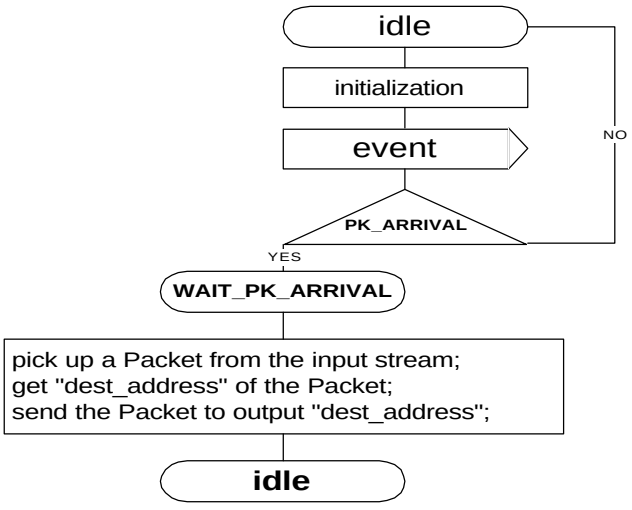

Figure 6. flow chart for “switch”

Simulation:

1. Tool:
We use OPNET as the development and simulation platform for its friendly GUI and flexibility.

2. Scenarios:
We have two scenarios. Every configuration except those for the router is the same for both scenarios. One is for QoS enabled router, the other is for QoS disabled.

3. Network Topology:

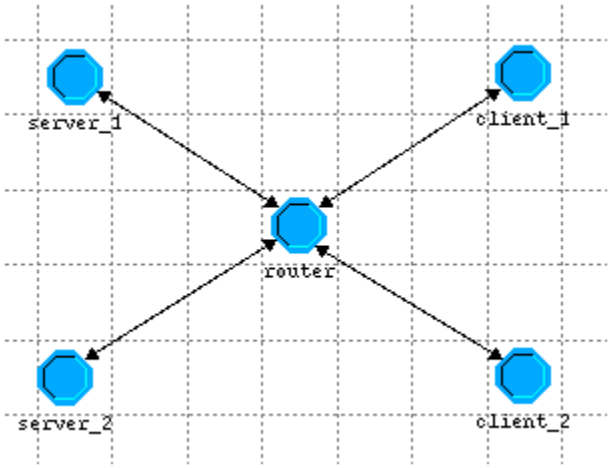

Figure 7. network topology

The network is constructed with two server nodes, two client nodes and a router node.

4. Packet Format:
The packet format is defined as below:
- 8 bits for destination address
- 8 bits for source address
- 8 bits for packet identification
- Payload of packet is padded by the system automatically as configuration
- The source always generates packets with destination address = 1 or 2 for this simulation.

5. Router:
For scenario 1:
We disable the QoS functions of the router.
For scenario 2:
We enable the QoS functions of the router.

6. Server Node:

The structure of the server node is shown by the figure.

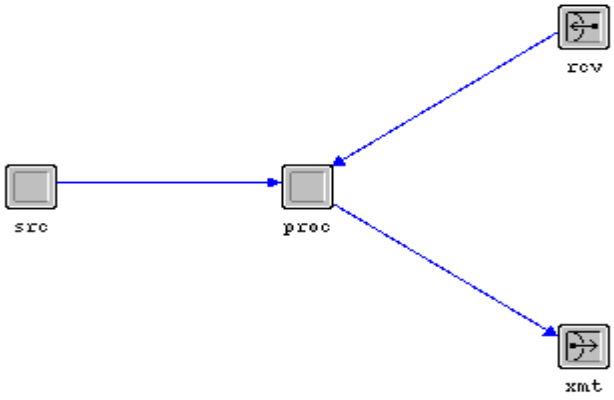

Figure 8. Architecture of server node.

The “src” is a simple source, which generates packets with our defined format and sends them to “proc”.
The “proc” has two functions. First, it acts as an on-off filter making a burst output traffic. Second, it sets the source address of each packet with the identification of the server and puts them into the output stream.

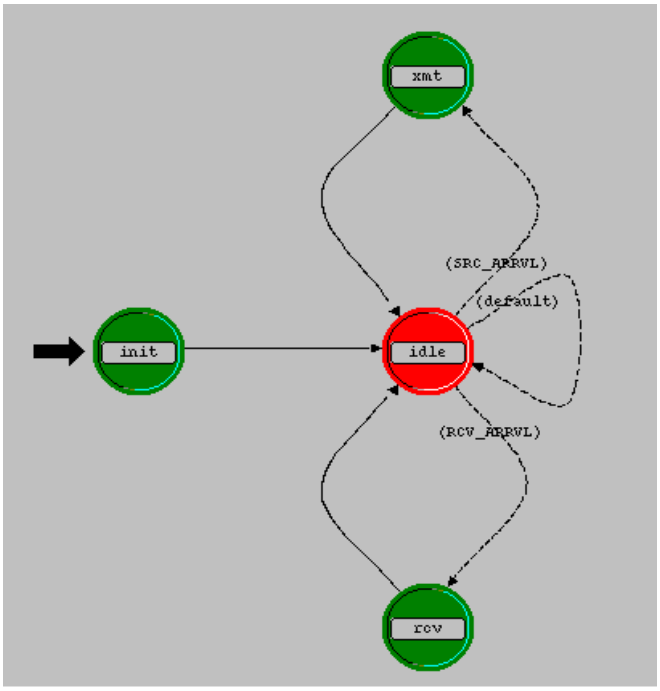

Figure 9. FSM for “proc” of server node.

The “rcv” and “xmt” ports are interfaces between the node’s internal input/output streams and the link.

7. Client Node

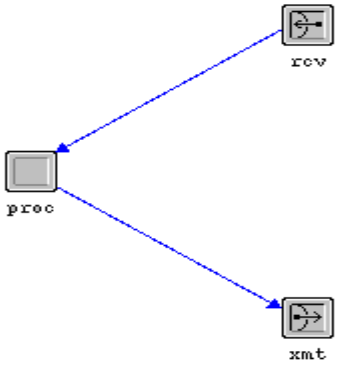

Figure 10. Architecture of client node

The structure of the client node is like this:
The “proc” picks up packets from the input stream. And retrieves each packet’s source address and identifies its class of QoS. Then, it counts the number of received packets, calculates the delay time of packets, and writes collected data into global statistic handles. Finally, it destroys packets to release memory.

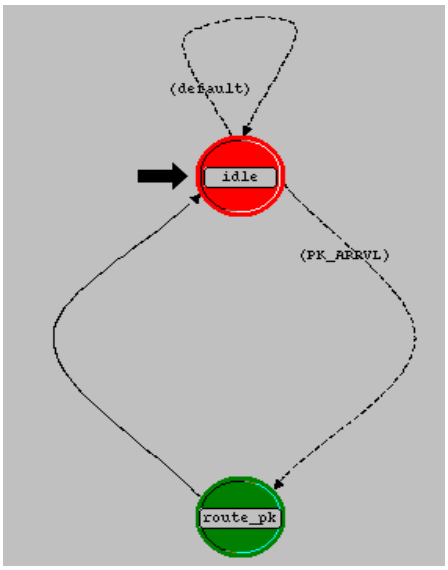

Figure 11. FSM for “proc” of client node

8. Link:
The link model used is as follows:

- All links are point to point duplex links.
- All link rates are 100 Mbps.
- All links support only packets in the format defined earlier.

By using high-speed link, we ensure that all the delays only occur in the router queues.

Results:

Parameters for network components:

- Packet Inter-arrival Time: Exponential (0.001), which means 1000 pks per second.
- Packet Size: Exponential(100)
- Queue service rate & capacity:
  - Queue 0~2: service rate: 24,000bps, capacity: 10 pks
  - Queue 3: service rate: 24,000bps, capacity: 500 pks
- Simulation duration: 1 minutes.
- Values per statistic: 100.
- Seed: 431.

Scenario 1 (enable QoS):

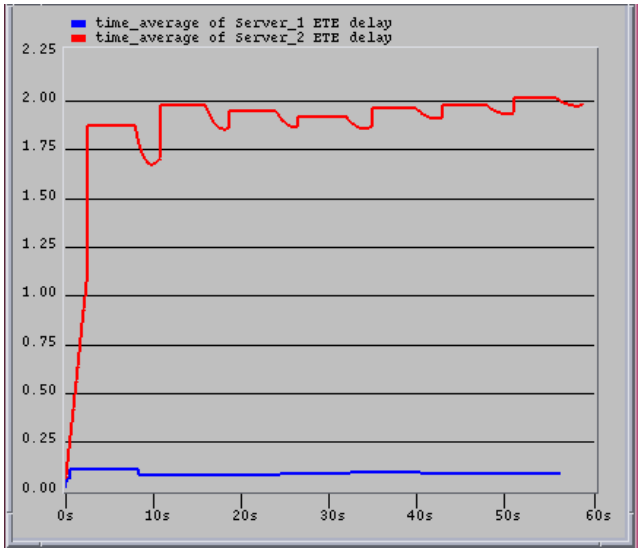

Figure 12. ETE (End To End) delay

Note:
The blue curve is the time_average ETE delay of the data transmission with high priority QoS (class_1).
The red curve is the time_average ETE delay of the data transmission with low priority QoS (class_2).

With our queuing scheduling, the delay is less for high priority QoS than that for low priority QoS.

Below is the delay within queues:

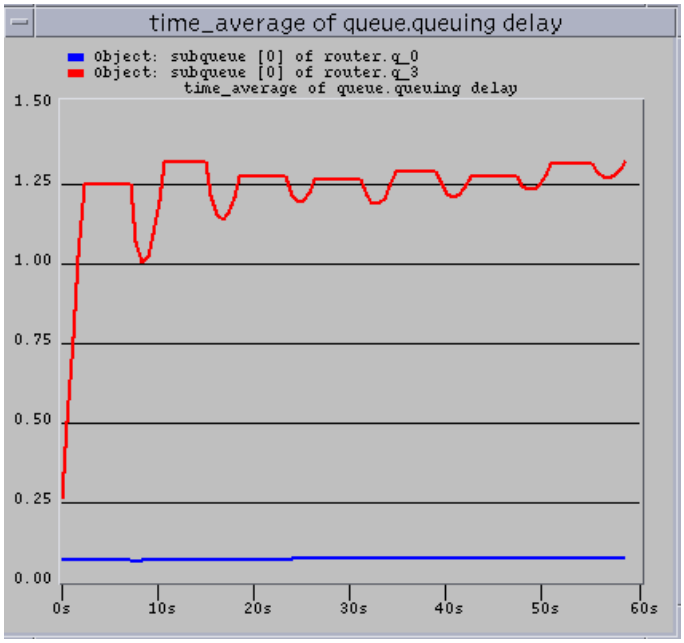

Figure 13. Delay within queue

Note:
The red curve is the time_average delay within q_3 (low priority queue-class).
The blue curve is the time_average delay within q_0 (high priority queue-class).
Because q_0 has three other peer queues, while q_3 is the only queue for its queue-class, the capacity of queue_class_1 is almost the three times of that of queue_class_2. Packets scheduled to queue_class_1 will be handled faster than those scheduled to queue_class_2. Thus delay within q_0 is much less than that for q_3.

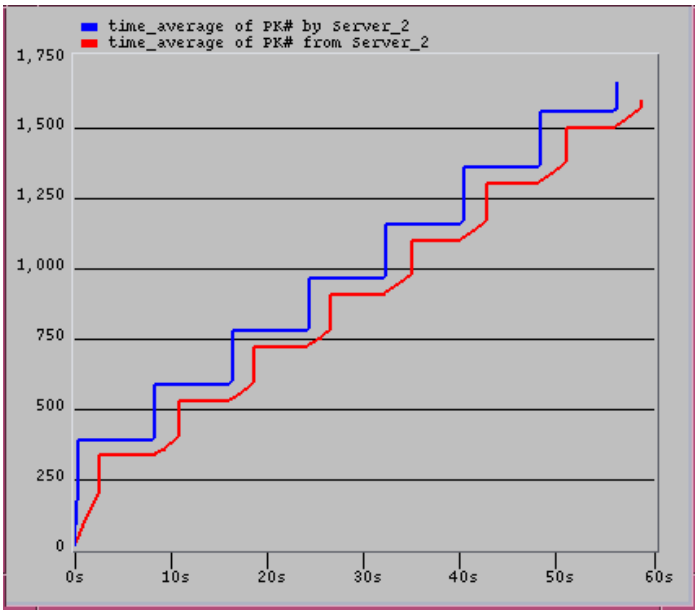

Figure 14. ETE loss for Server_2 (with low priority QoS)

Note:
The blue curve is the number of packets generated by the Server_2.
The red curve is the number of those packets received by the client nodes.
These two numbers are almost equal. So the packet loss for low priority QoS data transmission is almost 0.

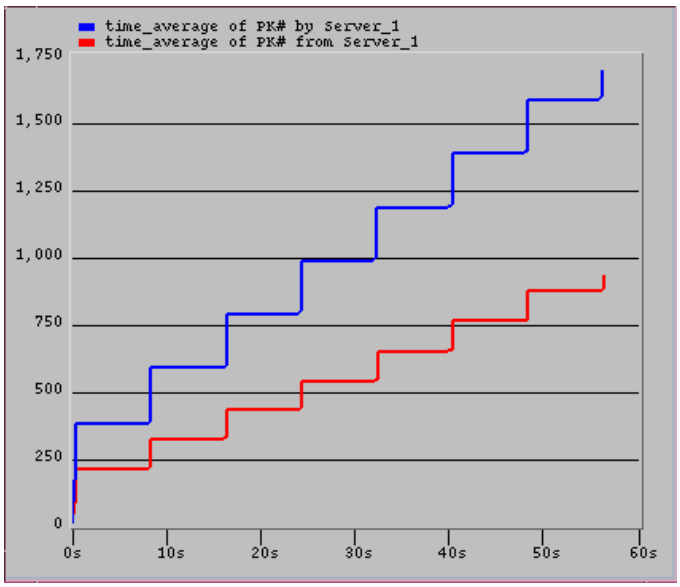

Figure 15. ETE loss for Server_1 (with high priority QoS)

Note:
The blue curve is the number of packets generated by the Server_1.
The red curve is the number of those packets received by the client nodes.
The packet loss for high priority QoS data transmission is pretty large, much more than that for low priority QoS transmission.

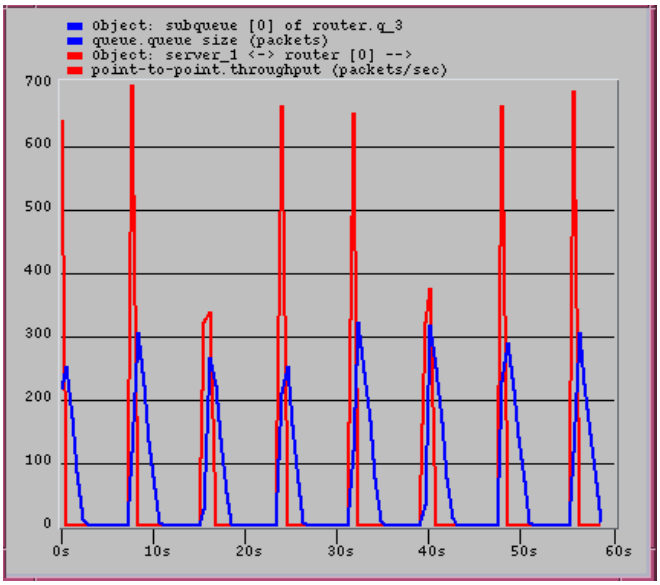

Figure 16. Bursty-source node's throughput and q_3's size

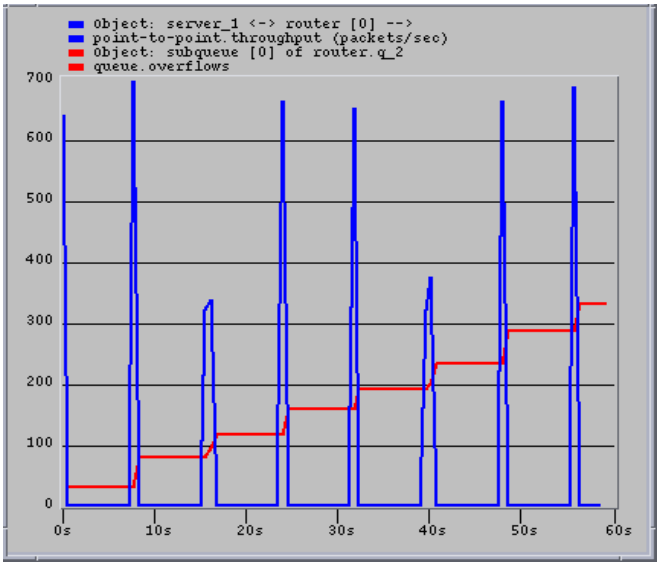

Figure 17. Bursty-source node's throughput and q_0's overflow

Note:
The higher curves are the throughput curves, and the lower ones are the queue size curves. With larger size, q_3 buffers burst data and guarantees zero packet loss. While with much smaller size, q_0 has overflow and drops packets for burst data.

Scenario 2 (disable QoS):

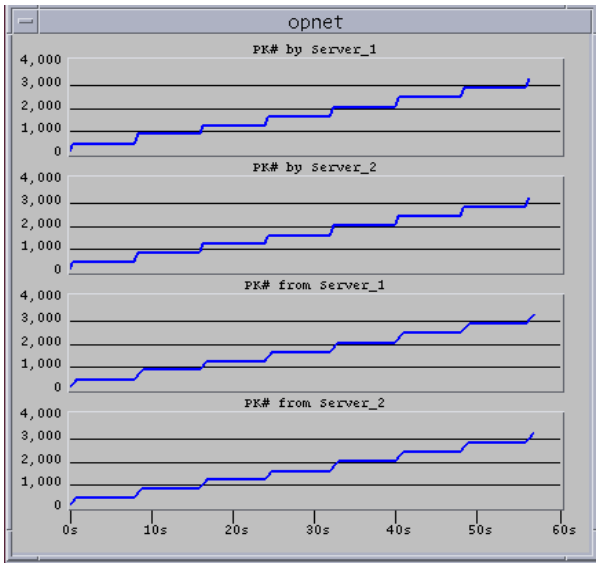

Figure 18. ETE packet loss for both servers

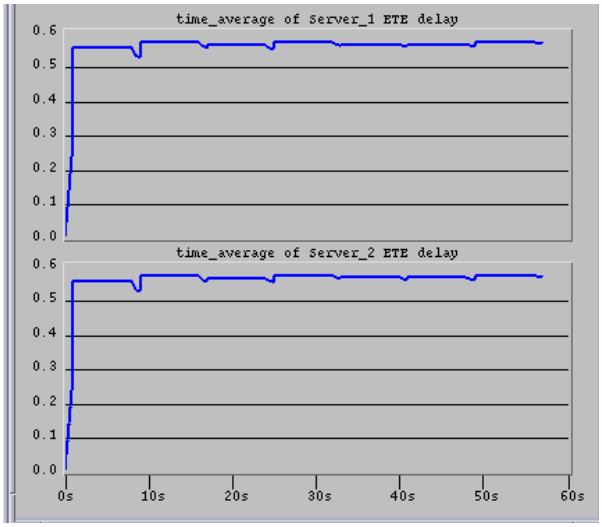

Figure 19. ETE delay for both servers

Without QoS capacity, the router treats traffics from both servers the same way. So both transmissions have the same packet loss ratio and delay.

In a word:

- Service of high priority has smaller delay than Service of low priority.
- Service of high priority has bigger packet loss than Service of low priority.

4. DISCUSSION AND CONCLUSIONS

Problems encountered:

The project was finished without any major problems. But our aborted first try did consume much of our time. Originally, we planed to implement our design in an ATM network. But lack of experience of transferring parameters from one layer to the other prevented our try, although we spent quite a lot time on it. However, this try gave us a chance to be familiar with the OPNET and helped to make design for our final network model.

Besides the fact that our network model had to be changed mid way through the semester. Other challenges were:

- Steep learning curve in the first half of the semester.
- Mistakes when building the network.
- Various errors due to inconsistent configurations.
- Debugging the codes in an unfamiliar environment.

Conclusions:

- The tradeoff between queuing delays and probability of packet loss and its dependence on buffer sizes was observed.
- Increasing the buffer size definitely decreases probability of loss.
- Increasing buffer size also increases queuing delays.
- There is always an optimal value of buffer size, depending on the requirements on delays and loss probabilities. Simulation is sometimes the only way to reach such optimum values.

Through the effort, we gained more than reasonable expertise in the use of OPNET in modeling and simulating customized networks. The final network was built from scratch, and simulating it has given us a lot of confidence in using OPNET.

At the same time, reading and studying lots of tutorials and paper related to QoS enriches our knowledge and deepens our understanding of it.

Overall, all the efforts were a very fruitful experience and rewarded.

Future work:

Naïve queuing algorithm and simple network model are used in the project. Many improvements can still be made if we have more time:

- More advanced and complicated queue algorithms could be implemented to get better results, such as Class_base Queuing, Random Early Detection, and so on.

- To implement QoS in real world, admission control is also very important.
- Another direction of improvement is to use more measurements to do comparison.

5. REFERENCES

J. Walrand and P. Varaiya, *High-performance Communication Networks, Second edition*, Morgan Kaufmann, 2000.

Yuxing Tian, "**A Survey on Connection Admission Control in ATM Networks",** DePaul University

E. Knightly and H. Zhang, "**Connection Admission Control for RED-VBR, a Renegotiation-Based Service**," in *Proceedings of the 4th International IFIP Workshop on Quality of Service*, Paris, France, March 1996.

http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html

http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcintro.htm *

http://qos.ittc.ukans.edu/

http://www.qosforum.com/

http://citeseer.nj.nec.com/kelly00distributed.html

http://www.cisco.com/documentation/ccm/v24/cm_location.htm

http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121newft/121limit/121x/121xi/121xi_3/dt3trsvp.htm

http://iwander.csl.uiuc.edu/wireless/presentations/wireless-mpfq/ppframe.htm

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/qos.htm

http://www.wiley.com/compbooks/ferguson/

APPENDIX:

Codes & Comments: