Project assignment 3

The Goal of the project: build a TCP like congestion control using UDP.

Deadline: May 4nd 11:55pm, 2013

Task 1: Implement ARQ and random delay

From project 2, you should have implemented a 2-hop network with QoS which looks like figure 1.

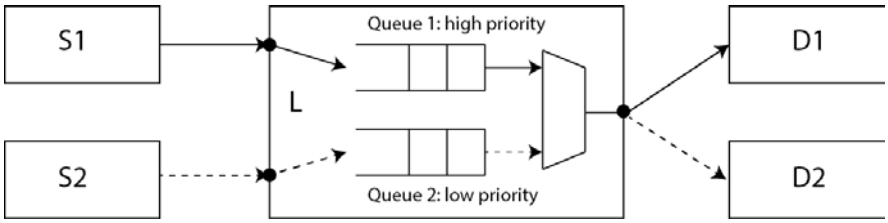

Figure 1: QoS network in project 2

Now you need to implement an additional variable delay on low priority flow (S2, D2). Every packet arriving at D2 should experience a random delay before being received by D2. This delay is to be uniformly drawn from [0, b] ms, where b alternates between 5 and 15 every 5 seconds, i.e., b is 5 for first 5 seconds, and 15 for next 5 seconds and so on.

Let the service rate of the router **L** be 10 ms/pkt, and let the sending rate of **S1** and **S2** to be **2R** and **R** respectively. Remember the rates are defined as average inter-packet time, and they follow exponential distribution. Choose R the same as in project 2 task 2.2. Recall from last project, that in case the router L is busy serving high priority queue 1, queue 2 can build up and packets are dropped due to buffer overflow. Therefore we need an error-control for low priority flow to ensure flow (S2, D2) has no packet loss.

Select and implement a window-based error-control mechanism.

For simplicity, let's assume that the feedback path goes directly from D2 to S2. The additional variable delay can be implemented in the D2 process. Every packet arriving at D2 will experience a random delay uniformly drawn from [0, b] ms. It should look like figure 2.It should look like figure 2.

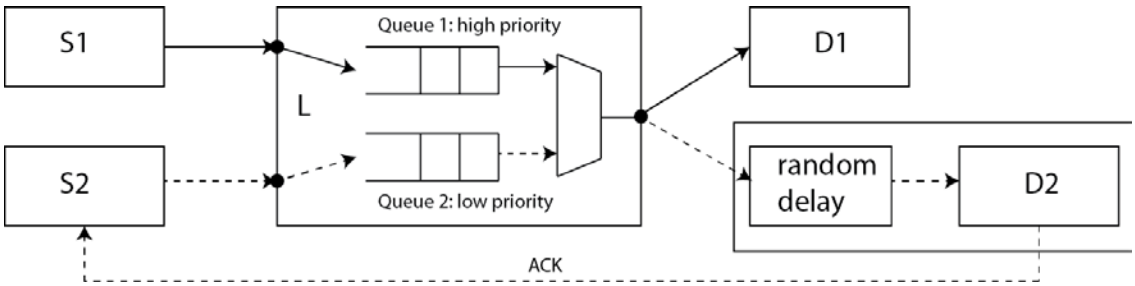

Figure 2: Adding error control and random delay for low priority flow (S2, D2)

One problem we face is: how long should the source wait for an ACK from the destination? i.e., how long should we set the timeout timer at the source? In theory we should use the Round-Trip Time (RTT) for data-ACK pair, however, due to random delay in the network, RTT can never be accurately measured. As a matter of fact, if one data packet is dropped and hence an ACK is never returned, the RTT could be infinity! Therefore, typically RTT is estimated by using adaptive exponential averaging from multiple packets. Please read section 7.4.4 in the book "Communication Networks: A concise introduction" by Jean Walrand and Syham Parekh (available online).

1.1. **Explain how you designed the error-control method to ensure that all packets from S2 reach D2.**

1.2. **Define efficiency = (number of packet received / number of attempts by transmitter) * 100%. Plot efficiency vs window size for low priority queue, and comment.**

1.3. **Plot timeout value used by your algorithm vs time for 60 seconds, and comment.**

1.4. **Plot the load of the router over 60 seconds for the two queues respectively, and comment.**

Task 2: Congestion Control

We want to you to implement a TCP-like congestion control using UDP for flow (S2, D2). TCP congestion algorithm is AIMD – additive increase multiplicative decrease. The source S2 increases its window size additively as long as it receives ACKs. When ACKs are missing, S2 divides its window size by 2 and resume increase again. For this project, make S1 send at alternating rate as follows: for first 5 second, S1 sends at rate R = 10 ms/pkt; for the next 5 second, S1 stops sending; then S1 sends at R = 10 ms/pkt and alternates again. The router L can store up to B = 64 packets equally split for both queues, and the service rate is 5 ms/pkt.

Design and implement the AIMD congestion control. The following experiments are to be done

(a) Within the University (both laptops)

(b) at least one for the laptops over an DSL line or form a Hot-spot

(c) in case of rather weak connectivity at this laptop

For <u>each case</u>

2.1. **Plot the window size of S2 decided by your AIMD algorithm as a function of time for 60 seconds, and comment.**

2.2. **Plot efficiency over time for 60 seconds, and comment.**

2.3. **Plot the load of the two queues for 60 seconds with congestion control. Compare with that without congestion control, and comment.**

Task 3: Observe with Wireshark

Wireshark is an open-source software for network analysis. It is a very good tool to help understand the structure of different networking protocols. You can download and install the Wireshark from http://www.wireshark.org/. You may also need to install WinPcap in order to make Wireshark work on your computer. You can find Wireshark user's guide here http://www.wireshark.org/docs/wsug_html_chunked/, and Wireshark tutorial here https://blog.wireshark.org/2012/10/wireshark-tutorial-series/.

Run your program and capture the real packets in the network with Wireshark. Note you may capture other packets in the air which are intended for other programs or even other computers. Try filter out packets from this project. **Please show screen print for your answer.**

3.1. What is the IP address of your computer?

3.2. Within the IP packet header, what is the value in the upper layer protocol field?

3.3. How many bytes are in the IP header? How many bytes are in the payload of the IP datagram?

3.4. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.

3.5. Select one UDP packet from your trace. From this packet, determine the individual fields of the header, their length in bytes and report their content In the format: Field – length – content

3.6. Examine a pair of UDP packets in which the first packet is sent by your host and the second packet is a reply to the first packet. Look at the headers of the SAME packets on both the laptops involved in the transmission. Describe the relationship between the port numbers in the two packets.

(a) at the University network

(b) in case one of the laptops is connected over DSL /hot spot

Comment the differences.

POLICIES AND GUIDELINES

- Discussion on the interpretation of the assignment, the reasonable assumption, background information, references or programming language itself is highly encouraged across teams and is not regarded as a violation of Academic Honest and Integrity Policy.
- Fill in the gap yourself. You are free to choose anything not explicitly specified in this assignment with reasonable justification. Remember that there can be several ways to achieve the same goal.
- Timer control does not have to be precise. Keep it simple.
- You are flexible in all design choices not explicitly specified in this assignment. All such choices must be documented in the project report.

Submission

For every task, each team submits a project report in pdf format and an electronically archived source code (in zip, tar.gz, or tgz) to bspace. The report should be named **EE122project3report-(last name)-(last-name).pdf.** The archive file should be named **EE122project3-(last name)-(last-name).{extension}** . In your project report, you must describe your project design and answer the questions in the project description with appropriate discussions and assumptions. You are free to discuss any other observation that you find relevant.