

**CNT5106C Computer Networks, Fall 2009**  
**Instructor: Prof. Ahmed Helmy**  
**Homework #2**

**On Queuing, Transport Layer, Congestion Control and TCP**  
**[Date Assigned: Oct 11<sup>th</sup>, 2009. Due Date: October 19<sup>th</sup>, 2009 in class or to the TAs]**  
**Full grade points: 80, Total Points: 95, Extra points: 15 points.**

**Q1.** (4 points) Describe why an application developer might choose to run an application over UDP rather than TCP. Give examples of such applications.

An application developer may not want its application to use TCP's congestion control, which can throttle the application's sending rate at times of congestion. Often, designers of IP telephony and IP videoconference applications choose to run their applications over UDP because they want to avoid TCP's congestion control. Also, some applications do not need the reliable data transfer provided by TCP.

**Q2.** (4 points) Why is it that voice and video traffic is often sent over TCP rather than UDP in today's Internet? (Hint: not because of TCP's congestion-control mechanism)

Since most firewalls are configured to block UDP traffic, using TCP for video and voice traffic lets the traffic through the firewalls.

[more: Technically speaking, UDP would provide a better trade-off since voice/video apps (especially interactive) tolerate loss (up to ~10%) but do not tolerate delay or jitter. TCP's congestion control and reliability mechanisms lead to 100% delivery but lead to variable delays (i.e., increased jitter), and hence is technically unsuitable for voice/video delivery].

**Q3.** (4 points) Is it possible for an application to enjoy reliable data transfer even when the application runs over UDP? If so, how?

Yes. The application developer can put reliable data transfer into the application layer protocol. This would require a significant amount of work and debugging, however.

**- Queuing Theory**

Note: You may want to make use of the following equations:

- M/D/1: queuing delay  $Tq = \frac{T_s(2 - \rho)}{2(1 - \rho)}$ ;  $T_s$  is service time &  $\rho$  is link utilization

- M/D/1: average queue length or buffer occupancy  $q = \lambda.Tq = \rho + \frac{\rho^2}{2(1 - \rho)}$

- M/M/1: queuing delay  $Tq = \frac{T_s}{(1 - \rho)}$ , buffer occupancy:  $q = \frac{\rho}{(1 - \rho)}$

**Q4.** (8 points) Consider two queuing systems, serving packets with lengths that have exponential distribution, and the packet arrival process is Poisson. The first queuing

system (system I) has a single queue and a single server, and hence the packet arrival rate is  $X$ , and the server speed is  $Y$ . The second queuing system (system II) has two queues and two servers, and hence the packet arrival rate is  $X/2$ , and the server speed is  $Y/2$ . Derive a relation between the delays in each of these systems. What conclusion can you make?

**Answer: (8 points)**

We use the M/M/1 queue (because the question states Poisson arrival and exponentially distributed service time).

For the first system (I):  $T_q = T_s / (1 - \rho) = 1 / M(1 - \lambda / M) = 1 / Y(1 - X / Y)$ ,

**For the second system (II):  $T_q = 2 / Y(1 - X / Y) = 2T_q$  (of system I)**

That is, using '1' queuing system performs better than using '2' queues with half of the arrival rate and half of the output link capacity.

- Q5. i-** (4 points) What is meant by implicit congestion signaling and explicit congestion signaling? Give examples of congestion control protocols that use each type signaling.  
**ii-** (6 points) Discuss the advantages and disadvantages of each of the above schemes.

Answer: refer to the lecture notes/slides

- Q6.** (8 points) In one type of congestion control protocols the congested router sends a message (e.g., source quench) to the end points to alleviate the extent of congestion. Discuss the problems faced by such a scheme, providing potential solutions.

The router may notify sources that are not causing the congestion and are well behaved, which isn't fair. The router needs to employ a scheme to achieve fairness.

The router may have a large number of flows passing through it, and scaling is the main problem faced in such a case, where a router needs to identify how many packets are sent by each flow to determine which flow is consuming more than its fair share.

- Q7.** (10 points) Based on your understanding of how congestion occurs in the network and the phases of congestion, discuss how TCP is attempting to control network congestion. (use a graph to illustrate)

Answer: refer to the graph in the lecture notes/slides: TCP attempts to remain around the top of the curve (in phase II) to maximize the network utilization and throughput. In slow start, TCP starts at the bottom of the curve and quickly (using slow start, opening the window exponentially) climbs up the curve, causing the network to go into congestion, lose packets, and then it backs off. If the congestion is severe then the backoff is aggressive and TCP goes to  $\text{CongWin} = 1$  and climbs the curve again from the bottom, until it reaches half of the curve quickly (using slow start) then it switches to a slow climb using linear increase in the congestion avoidance phase. If the congestion is not severe and TCP is able to recover using fast retransmit fast recovery, then TCP goes to half of the curve and climbs slowly until it hits another congestion, ... so on.

**Q8. i.** (6 points) In TCP slow start mechanism, what is the equation used to increase CongWin and how does the window grow according to this equation? How does the equation change for the congestion avoidance phase and how does the window grow then?

**ii.** (8 points) In the details of the fast retransmit-fast recovery mechanism the window grows according to the equation used in slow start (before the cumulative ACK is received). Does that mean that TCP can send as many segments as it would have in slow start (for the same window size)? Why or why not?

In slow start:  $\text{CongWin} = \text{CongWin} + 1$  for every ACK received

This leads to an exponential growth of the congestion window size.

In congestion avoidance:  $\text{CongWin} = \text{CongWin} + 1$  for every RTT

This leads to a linear increase of the congestion window size.

In fast retransmit-fast recovery, although the equation used is:

$\text{CongWin} = \text{CongWin} + 1$  for every ACK received

The Acks received in this case are duplicate acks, that do not advance the beginning of the window (since there is a lost segment, the beginning of the window is ‘frozen’).

Hence, in general TCP will not be able to send as many segments as it does in the slow start.

**Q9. i.** (4 points) Why do we say that TCP uses an AIMD mechanism?

**ii.** (4 points) Why does TCP use AIMD?

**iii.** (4 points) What happens if TCP uses MIMD?

The increase of CongWin in TCP in steady state uses linear (additive increase, or AI) where CongWin is increased by 1 with every RTT, hence the AI term.

When TCP senses the loss of a packet, using duplicate Acks, it cuts down the CongWin by half [and this occurs everytime 3 dup acks are received], hence the multiplicative decrease (MD) term.

**Q10.** (6 points) If you use TCP to transfer a big file, you are likely to get a fair share of the network bandwidth (i.e., similar to that given to other long TCP connections).

Describe a way in which you can get more than your fair share. Reason about how much more bandwidth you can get.

By using parallel TCP sessions, each sending a chunk of the file. If every session gets R/M rate (where R is the capacity of the bottleneck link and M is the number of flows crossing that link), then by using N parallel flows instead of one, an end host can get  $NR/(N+M)$ .

**Q11.** (6 points) In ATM ABR congestion control the equation to increase the rate is given by:

$Rate_{new} = Rate_{old} - Rate_{old} * RDF$ , where  $RDF$  is the rate decrease factor,

- a. discuss how fast/slow does the sender respond to congestion for the various value of  $RDF$ .
- b. If the equation was changed to  $Rate_{new} = Rate_{old} * alpha$ , do you think the response will be better or worse and why.

Answer to be posted separately.

**Q12.** (9 points) Argue for or against this statement (reason using examples as necessary): “Packets are lost only when network failures occur (e.g., a link goes down). But when the network heals (e.g., the failed link comes back up again), packets do not get lost.”

When the network fails (e.g., a link goes down), a number of packets that cross that link may be lost.

When the network heals, packets/flows may cross relatively shorter paths to get to the destination. Shorter paths have a delay-bandwidth product less than longer paths, and hence can hold fewer bytes in the pipe (i.e., fewer segments can be in flight). Having a TCP connection with a high CongWin go over a shorter path may also cause packet loss, since the shorter paths will get congested, and buffers may overflow causing multiple (sometimes severe) packet losses.