- First read the introduction to Part 2 available in the course home page.
- Answers can be written in English or Finnish.
- Answers are to be returned into the box under the notice board in G-wing, 2nd floor.
- **Deadline for returning second part is Monday 27.2.2006 at 20:00**
  - Reports returned after the deadline cannot receive higher grade than **pass-1**

Project Work, Part 2

There are a few main elements of TCP which have been developed over the years:

- Calculation of `TimeOut`s
- Introduction of congestion avoidance
- Slow start
- Fast retransmissions

After doing the part 2 of the group work, you should have an idea why most of the above mechanisms were added to the original TCP and what were the circumstances that lead to the development of such mechanisms. In this simulation experiment, we will try to relive these moments that lead to the current TCP algorithms.

1. **Simulate with the different congestion control options the congestion window of the TCP source.** [worth 10 points]

   Different congestion control possibilities

   i) Use RFC793edu TCP agent with additive increase and multiplicative decrease, but send full window at beginning
   - set additiveinc true, others (exponinc, slowstart, fastrtx) false

   ii) Use RFC793edu TCP agent with additive increase and multiplicative decrease
   - set additiveinc true, others false, set the slow start threshold to 1

   iii) Use RFC793edu TCP agent with exponential increase and multiplicative decrease
   - set exponinc true, others false, set the slow start threshold to 1

   iv) Use RFC793edu TCP agent with additive increase and slow start
   - set slowstart true, others false

   v) Use RFC793edu TCP agent with additive increase, slow start and fast retransmission
   - set slowstart true, set fastrtx true, others false

   NOTE! Use simulation time of 15 seconds. You can also use longer times, as long as the figures you draw are clear.

a) For each option

- plot the congestion window
- explain what happens in the congestion window evolution.
- Specifically identify in each picture which phase is
  - additive increase
  - exponential increase/slow start
  - time out
  - fast retransmit

b) • Compare the number of packet arrivals and packet drops at the bottleneck queue between the different options. (Don't just list the values, but also discuss the reasons and consequences)
• What are the throughputs,

$$\text{Throughput} = \frac{\#\text{packet\_arrivals} - \#\text{packet\_drops}}{\text{simulation\_time}},$$

for the different options (i-v)? Which is the best? Why?

2. **Simulate the different TCP options**[Worth 5 points]

Exercise 1 dealt with the main building blocks of TCP: Congestion avoidance, slow start, fast retransmissions. These parts together give the TCP version named Tahoe. Other versions we study are TCP Reno (fast Recovery and possibility for delayed ACKs) and TCP Vegas (a rate based congestion avoidance algorithm)

Change the TCP connection to

i) TCP Tahoe

ii) TCP Reno

iii) TCP Vegas

As in the previous exercise, plot the congestion window for each option and explain what happens in the congestion window evolution.

3. **Change the TCP maximum window parameter to see which choice keeps the pipe full.** [Worth 5 points]

Use

- Option (iii) from exercise 1
- TCP Reno
- TCP Vegas

a) For each of the three, study and answer:

- Which value of max window gives the best results?
- Explain why too small or too big values are not good.

b) How important is the choice of maximum window size for each of three options studied?

4. **Simulate how TCP connections interact with other TCP connections. Try all the combinations of Tahoe, Reno and Vegas.** [Worth 5 points]

   NOTE: As the links are identical Tahoe-Reno is the same situation as Reno-Tahoe. So you need to study altogether six combinations:

   - Tahoe-Tahoe
   - Reno-Reno
   - Vegas-Vegas
   - Tahoe-Reno
   - Tahoe-Vegas
   - Reno-Vegas

   NOTE: Use long enough simulation times. At least several hundred seconds.

   a) What are the throughputs and number of drops for each alternative? Specifically answer which combinations, and **why**, have
   - the best total throughput?
   - the worst total throughput?
   - the most packet drops?
   - the least packet drops?

   b) When different TCP connections interact, the result might be that some connections get an unfairly large share of the bandwidth for some reason.
   - What combinations, and **why**, obtain the most fair throughput between the links (that is, share bandwidth as equal as possible), and what combinations obtain unfair throughput.