In this project, you are going to implement the TCP-Friendly Rate Control (TFRC) protocol. TFRC protocol is designed to share bandwidth fairly with TCP. On the other hand, TFRC protocol is more suitable for multimedia network application than TCP since it does not exhibit large fluctuation in bandwidth.

TFRC protocol is an equation-based rate control protocol. In other words, its sending rate is expressed as a function of a number of network parameters. In particular, the sending rate $T$ of TFRC is computed using the equation below:

$$T = \frac{s}{R \sqrt{\frac{2p}{3}} + t_{RTO} (3 \sqrt{\frac{3p}{8}}) p (1 + 32p^2)}$$

where $R$ is the average round trip time, $t_{RTO}$ is the TCP time-out, $s$ is the segment size (typically 1500 bytes), and $p$ is the loss rate. In this project, you will use a simpler equation for computing the sending rate $T$ below:

$$T = \frac{1.2s}{R \sqrt{p}}$$

To simulate the real network environment within a single computer, we will use the architecture shown in the Figure below.
The sender sends data packets to the relay node. The relay node then forwards the packets to the receiver. The receiver sends the acknowledgment packets directly back to the sender. This is the same architecture as described in homework 3. Below are some details on the functions of the sender, relay node, and the receiver.

**Sender:**

The sender includes the following information in the header of a data packet:

1) The sending timestamp, indicating when the packet was sent
2) Sequence number for loss detection at the receiver

Depending on the average round trip time $R$ and the loss rate $p$, the sender will send data at the rate $T$ as computed in the equation above. Note that the unit of $T$ is bytes per second. Depending on the packet size, you have to set the interval between the packets appropriately to ensure that the sender is sending $T$ bytes per second on average.

The sender is responsible for computing the average round trip time $R$. The instantaneous round trip time $R_{\text{inst}}$ of a packet is computed by subtracting the timestamp in the ACK packet from the receiving time of the ACK packet. Note that the receiver is responsible for putting in the original transmission timestamp of a data packet in the header of the ACK packet for that data packet. The average round trip time $R$ is computed as $R = (1-a)R + aR_{\text{inst}}$ where $a$ is $0 < a < 1$. The average round trip time is updated upon receiving every ACK.

**Receiver:**

The receiver includes the following information in the header of the acknowledgement packet:

1) Packet loss rate
2) The original timestamp of the data packet for which it receives.
Computing packet loss rate at the receiver.
The receiver computes the packet loss rate using a sliding window of size $N$ (see the ppt slide on TFRC on how to compute this loss rate). The loss rate is computed upon receiving every data packet.

**Relay node:**
The function of the relay node is to simulate packet loss rate and delay. To simulate packet loss, the relay node implements the token bucket algorithm (already implemented in homework 3). An arrival packet at the relay node is dropped when the token bucket is empty. To simulate network delay, the relay node delays the transmission of a packet to the receiver by a random amount of time. This delay can be done as follows.

When a packet arrives at the relay node, it is immediately copied into some buffer $B$ (if it is not dropped due to the token bucket algorithm). The relay node immediately associates this packet with a transmission time equal to the receiving time at the relay node plus a random amount of time (simulating the network delay). Your program would then loop through this buffer, and send the packet only if the current time is greater than the transmission time. Since the random delay can potentially introduce out-of-order packets, we want to prevent this by ensuring the relay node only send out packets in order. This can be done as follows.

Packets in buffer $B$ are stored in the order of arrival at the relay node. We assume that packets arrive at the relay node in order, e.g., packet with sequence number 1 will arrive before packet with sequence number 2 at the relay node. Your program then only checks the oldest packet in term of arrival time. If the transmission time of this oldest packet is greater than the current time, then transmits this packet and moves on to the next oldest packet. Otherwise, don’t do anything. This mechanism effectively ensures that the relay node will send out packet in the order it receives. You can implement buffer $B$ as a queue.
The receiver sends the ACK back to the sender. The ACK contains the packet loss and the original timestamp to allow the sender to calculate the sending rate.

**Important note:**
In this project, you are not required to transmit the video stream.

After you have implemented all the functionalities, use the following parameters to run the following experiment.

1) Set a, the factor used to compute the average round trip time equals to 0.05.
2) Set the random amount of delay time is between [80 msec, 100 msec]
3) Set the size of the window for computing the packet loss rate to 1000 packets.
4) Set the packet size equals to 500 bytes.
5) Set the bucket depth to 500 packets
6) Set the token rate in the relay node at 150 packets per second and the maximum send out rate at the the relay node at 500 packets per second.
7) Set the initial sending rate at the sender at 200 packets per seconds.

Answer the followings:

1) Carefully plot the packet loss rate as a function of time from 0 to 5 minutes.
2) Carefully plot the sending rate as a function of time from 0 second to 5 minutes.
3) Repat (1) and (2) when the delay is set between [150msec, 200msec]

Note: you can record the sending rate and loss rate to a file, the use excel to make these plots.

Discuss the effects of increasing and decreasing the value of the following parameters with respect to the sending rate:

The factor used to compute the average round trip time a, the size of the window used to computed the packet loss rate, the packet size.

**What to turn in:**
1) Please use “project submission” as the subject in your submission email. I have a script that archives your emails automatically.

2) Turn in your source code of the sender, relay, and receiver

3) A detail description of how to run your modules in the “README.TXT”. Do not type the instructions for running the modules in your email. Put it in README.TXT.

4) A properly formatted document named “project.pdf” (pdf format). This file will consist of the plots and the discussions as described above. **Important: if your program does not function fully, describe the non-working symptoms. Compile everything in Linux.**

**Extra credits:**

1) Incorporating actual video into your system.

2) Computing the loss rate using the loss event method described in the paper
