Computer Networks Assignment-1

Written by Akilesh B, CS13B1042

September 1, 2015

Goal:

The objective of this assignment is to design an end-to-end application layer protocol which is similar to “ping” program except that it is running at the application layer.

Protocol design:

This system is designed similar to UDP to send and receive packets of data (Datagrams) without trying to establish a connection. I just send packets of data to an IP address. I implemented this protocol (both the client and server parts) in Java.

Message exchanged:

My protocol allows client to send a String or an integer etc. This is inturn converted into a byte array and sent to the server.

Messages rules and syntax:

I'm sending the sequence number which is an integer inturn converted into a byte array after converting it into a string. Server receives this and sends the exact same thing back to the client. This facilitates me to check if the packets received back at the client are out of order.

- Loss rate: Server has a tunable parameter *LOSS_RATE*. This parameter is specified on line number *23* of the UDPServer.java . Every time, after receiving the packet from the server, I generate a random number between 1 and 100. If the random number is less than LOSS_RATE specified, then the packet is dropped by the server and not sent back to the client else, it is sent back. This design approximately ensures *LOSS_RATE* % of packets are dropped on an average. This randomized implementation is quite practical because we don't know when the server will be down and packet will not reach back to the client.

- Average delay: Server has a tunable parameter *AVERAGE_DELAY*. This is determined by an random gaussian function. The mean and variance of this random gaussian distribution can be specified on line number *41* and *42* respectively of the UDPServer.java . Server waits for this much time before sending back the packet to the client. This is implemented using timer class which allows us to perform an action after a certain delay.

- The client sends 10 ping requests to the server separated exactly by one second. This is implemented by using Timer and Timer task class in java which allows us to schedule a task to occur repeatedly. The client waits for exactly one second for a reply, beyond which it assumes the packet is lost in the network. This was achieved by using ExecutorService in java. The service is shutdown after one second.

- My own timer is implemented in both server and client, instead of using *setSoTimeout* because this protocol is entirely at the application layer. By using *setSoTimeout*, I'm making changes in the transport layer. This also ensures that no packet is dropped at the server during the time of delay.
- IP address is localhost (specified in line 32 of UDPClient.java) .
- To calculate round trip time (*rtt*), system time is measured just before sending the packet (*t1*) and just after receiving the packet (*t2*) at the client. The difference (*t2* - *t1*) gives round trip time. Maximum, Minimum, Average, Standard deviation are then calculated accordingly.
- Errors are also appropriately handled.

Result:

Hence, an end-to-end application layer protocol similar to "ping" is designed and implemented.