CS13B1040
Akshita Mittel

COMPUTER NETWORKS: ASSIGNMENT 1

THE “PING” PROTOCOL

PROTOCOL DESIGN

The application layer protocol was made using python strings. This is because, it is the most fundamental data type that can be used as an abstraction. The protocol has the following header and data in proper order:

HEADER:
Client host = protocol[:9]
Client port = protocol[9:13]
Packet ID = protocol[13:14] → Here the ID was taken to be a one digit number, since we were asked to only send 10 packets. The format of this particular entry can be modified easily to provide an universal ID.
DATA:
Time = protocol[14:27] → This is the actual time at which the packet was first sent. It is represented as a floating point number with uniform length. Which can later be unpacked.
Message = protocol[27:] → The rest of the protocol consists of the actual message that the user wants to pass.

myServer.py

This is the program which contains the code for the Sever. A UDP mechanism is used for both the client and the Server, since the rigid TCP mechanism allows no scope for packet loss.

First the Host and Port are defined. After-which they are bound to the UDPSocket that was created in line 10.

For the current purpose, I did not keep a Timeout for the server as we can manually stop it at our own convenience, hence the actual message transmission occur in a continuous while loop.

The Server takes two arguments, the first is the data loss rate and the second is the average delay in milliseconds.
The average delay is incorporated using a time.sleep(). This time method stalls the current thread for a brief period of time (in this case, the time specified by the arguments).
Since the time.sleep only halts a specific thread and not the entire process, a new thread is created, ensuring that the server does not stall when the sleep is called.

The server retrieves the message from the client using the function recvfrom, and specifying the host and the port.
A random number is generated which determines whether the packet is to be returned or not. If the packet is to be returned, it is sent back to the client using sendto. The address where the message has to be sent was received with the client message in recvfrom.

myClient.py

All the data that has to be calculated is initialized and set as a global variable to ensure that the values remain intact across the functions.

startPings:

The function startPings, first assigns the Host and the port and then in an iterative way sends 10 pings using the function Ping.

The function Ping returns the delay that was present in a round trip to the server and back. If the delay shows the message “Request timed out.” , then it will be taken as a missed packet and the corresponding message will be printed.

The maximum, minimum and total delay are calculated with each iteration of the ping.

Towards the end of the function, all the previous collected data is used to calculate, the maximum RTT, minimum RTT, average RTT, stdDev of RTT, missed packets, and packet loss rate.

ping:

This function first establishes a UDP socket and sets the timeout to be 1 second. It then creates a certain protocol using the protocol function. The documentation for the PROTOCOL is defined above.

It then notes the time, just before sending the packet to the Server. After sending the message to the server it tries to get the “ping” back. The connection times out, it closes the client and returns the corresponding statement, else, if the try block works, it closes the client socket and returns the actual RTT.