50.012 Networks - Project (Strawman Submission)

Team Name: TeleTubbs
Team Members: Tracy Yee Enying (1002379), Valerene Goh Ze Yi (1002457), Sng Xue Le Candace (1002276), Rahmathulla Shameena Nilofar (1002532)

Title of Project: Chat Server Application

Main Idea of Project: To create a simple chat server application for single/multiple client(s) that monitors and displays the traffic (number of packets sent) in real time and indicates the level of congestion (categorized into low, normal and high) to the user(s). Feedback on congestion control is also provided to the user in the form of a real-time plot of the congestion window size with time. The user(s) can interact with the application by entering a number in the input field to set the rate of sending messages to the server and observe the changes accordingly. Refer to Figure 1 below for the rough layout of the application.

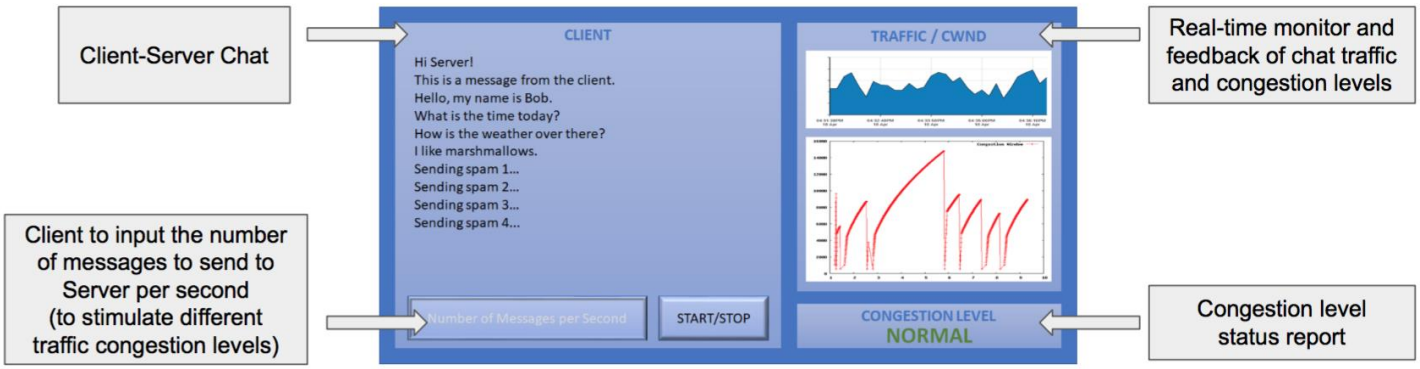

Fig 1. Application Dashboard

Features:

1) Send and receive messages
2) Monitor traffic and detect congestion
3) Feedback on congestion control
4) Multiple clients (additional feature)

Note: These four features will be implemented in order from (1) to (4). We aim to accomplish at least features (1) and (2), only working on features (3) and (4) if time permits. Hence, if we were to reduce functionality, we would remove features from the bottom-up, i.e. removing (4), followed by (3), if necessary.

How we intend to implement the idea:

1) Setup a simple server that accepts a single client connection through TCP. This is done using Python and sockets.
2) Setup a web application (simple GUI) with the client-server chat to allow for one-way messaging. The chat will only display the messages sent by the user. The server will not reply the client and the client is free to continue sending messages to the server. This is done using Python and Flask.
3) Monitor chat traffic and plot a (real-time) graph for the number of packets sent and the level of congestion. Display graph on web application. This is done using the Pusher library in Python.
4) Automate the user's inputs using a Python script (user can set the rate of messages sent on the web application) to observe the changes in the traffic, congestion level and congestion window size when the rate of messages sent changes.
5) (Implement feedback functionality by monitoring and plotting the congestion window size in real-time. This is also done using the Pusher library in Python.)
6) (Allow multiple client connections to the server and plot the graphs for each user separately.)

***Note:** Similarly, steps 1 to 6 for the implementation will be carried out in order, hence if we were to reduce functionality, we would remove these steps in the following order: (6), (5), real-time plotting for (3).

Network topics incorporated into the project: Web API (main topic), Application Layer, Transport Layer