1 Assignment

This assignment is to familiarize you with traffic capture and analysis as well as ensure you can write a TCP server and client.

2 Part 1: Write a TCP client and server, 50 points

Write a simple TCP based client and server which can talk to each other. Your server must bind to a specific port and listen for connections.

Your server should be able to manage multiple clients simultaneously. Test and ensure your server can listen to at least 3 clients simultaneously.

Your server should print out a message when:

1. A new client connects.
2. After every 1,000 bytes received from a particular client (the message should identify the client uniquely and the total amount of bytes from the client). For example, if "client 1" sends a total of 2,500 bytes, your server should output:

```
Received 1,000 bytes from Client 1. Total: 1,000
Received 1,000 bytes from Client 1. Total: 2,000
```

Your server does not need to send any data to clients, but you may find this useful for debugging.

Write a simple TCP client which connects to your server and sends a random amount of data between 2,000 bytes and 1,000,000 bytes. Data the client sends to the server can be anything you choose (for example, just a bitarray of 0s).

3 Part 2: Use wireshark to look at throughput

Use Wireshark (https://www.wireshark.org/) to capture traffic between your client and server during one run.

You can run your client and server on the same machine (using localhost and the loopback device - details and tutorials specific to your OS are online), or run your server and client on separate machines. You'll need to figure out what device to capture on and have admin permissions on the machine you're capturing on.

In Wireshark, view and save a PDF version of a TCP throughput graph for your client/server connection. (Look under Statistics >TCP Stream Graphs > Throughput). Be sure to view/save the throughput for the appropriate direction to capture the data transmitted from the client to the server.

Answer the following questions (5 points each):

1. What device did you capture on? What network(s) were your client and server on? Eg. Did you run both the server and client on the same machine? Did you run the client on your laptop and your server on aludra.usc.edu?

2. What are the total number of application bytes your client transmitted to your server?
3. What is the maximum throughput in bits/s your client/server connection achieved?
4. Was there any packet loss or duplicate ACKs in your client/server connection?
5. Describe (if any) other traffic you captured other than the TCP connection between your client/server.

4 Turn In Instructions

You must submit a ZIP (.zip) or GZIP'ed TAR file (.tar.gz). These will be the *only* formats accepted. YOU MUST NAME YOUR ARCHIVE FILE USING YOUR USC USERNAME: E.G. {YOUR USC USERNAME}.tar.gz or {YOUR USC USERNAME}.zip.

Email your .zip or .tar.gz file to bartlett@isi.edu BEFORE 8pm on the deadline.

Included in your ZIP or TAR(.GZ) file should be the code for your client and server a README file (see below) along with the Wireshark throughput graph in PDF format (Part 2, worth 15 points). *Do not turn in any binaries - only text files (eg no compiled code/executables).*

Your README file is worth 10 points, so leave yourself time to complete the README. This README file should be plain text (no other format will be accepted) and should include the following headings/sections. Each of these items should be *clearly marked*:

1. AUTHOR: Your NAME and EMAIL.
2. BUILD: Instructions on how to compile your code under UNIX. Use a Makefile if possible. If you use Python (or some other scripting language), simply put "None" for this item in your README.
3. RUN: Instructions on how to run your code. For example:

   ```
   % python3.2 server.py
   % python3.2 client.py
   ```

4. BUGS: List things you know you were not able to complete or do correctly.
5. ANSWERS: The answers to questions 1–5 from part 2.