Computer Networks **Instructor: Dr. Deepak Tosh**
Spring 2023 **Due Date:** 03/26/2022

Programming Assignment # 2a: Reliable File Transfer

In this assignment, your goal is to write **two Python-based socket** programs: 1) for file transfer server, and 2) for a client that interacts with the server in getting the file. These do not need to implement a full featured file transfer protocol, like ***sftp***, instead, basic file transfer functionality is all that is needed. This is a **multi-part assignment** and initially we will start with a file transfer service over TCP sockets. *Note that the server should be able to transfer any kind of files (text, pdf, or multimedia file). You are required to create the client-server programs in such a way that they can transfer files with extension txt, jpeg, pdf, mov, mp3, and mp4.*

Task-1 (Implement TCP-based File Transfer Server-Client Application)
Specifically, you need to build two python programs (rft1Server.py and rft1Client.py), which must do the following:

1. When the server is started, it should ask for a port number. Your server should create a TCP socket then listen for connections on that port.
2. When the client is started, it should ask for an IP address and a port number of the server. The client should connect using a TCP socket to the server listening at the give IP address and port.
3. The client should provide a **RETR** command to retrieve a particular file. Upon user input, the client should send the file name to the server. For example, "RETR *myfile.pdf*" will ask the server to send the file *myfile.pdf.*
4. The server should receive the message from the client, check if the requested file exists, and if so, send its contents to the client.
   - You need to make sure that the TCP packets' payload does not contain more than 1000 bytes.
   - Thus, you would have to send several packets back-to-back to the client depending on the size of the file.

   Hint – Use small text files first, then go for larger and different types of files.
5. When the client receives the contents of the file, it will append them to a new file, and when file is completely sent (can be detected with end-of-file indicator), it should close the connection.
6. The server should still be running and able to accept another client connection.

Note: No special python modules related to file transfer is allowed to be used.

Environment to Execute the Programs
Choice – 1:

- You can install two UBUNTU 18+ VMs on your VirtualBox (with host-nly network mode) and then use client program in one VM and server program in another VM to test your programs.

Choice – 2:

- You can just use one UBUNTU 18+ VM in VirtualBox and in the home folder, you can create two subfolders named "Server" and "Client", which are the home folders for your server and client programs respectively.

How to run your codes?

- In one terminal, you would need to navigate to Server subfolder and execute your server program using "python3 rftServer.py". Similarly, in another terminal, you have to navigate to the Client subfolder to run the client program using "python3 rftClient.py".

Here is a sample client-server interaction format that you should be following while building your program. Left one is client's and right one is server's terminal. RED means the client/user is providing the input.

After client terminates, you must test the correctness of the sent & received files using a DIFF command – "**diff -s recvdFile sentFile"**

```
cliNode> python3 client.py
Provide Server IP: 10.0.0.3
Provide Port#: 8080
You are now connected! Enter your
commands now.
RFTCli> RETR myfile.pdf
Received myfile.pdf
RFTCli> CLOSE
```

```
srvNode> python3 server.py
Listen at port# 8080
Listening for connection at 8080…

Connection accepted from 10.0.0.2.
Asking for file myfile.pdf
Sending the file ...
Transfer Complete!
Connection closed, See you later!

Listening for connection at 8080…
```

As outcomes of this assignment, you would need to provide the following as:

1. Python programs that show outputs as shown above with proper code documentation. Lack of code documentation will cost 5% of the grade.
2. Report containing your strategy to solve the problem, evidence, and multiple execution samples (i.e., sending different kinds of files), instructions for running the submitted programs, and references used. Comprehensive report with no grammatical error is expected and it carries 10% of this assignment's total grade.

References:

[1] Socket programming in python: https://realpython.com/python-sockets/