National University of Singapore

School of Computing

CS2105 **Assignment 2** (10 Marks) Sem 2 AY22/23

Submission Deadline

31st March (Friday) 11:59 pm. 2 marks penalty will be imposed on late submission (Late submission refers to submission or re-submission after the deadline). The submission folder will be closed on **7th Apr (Friday) 11:59 pm.**

Objectives

In this assignment, you will implement a client and server for file transfer. After completing this assignment, you should

- be able to construct application layer packets
- have a good understanding of communication protocol design.

Group Work

All the work in this assignment should be done individually. However, if you find the assignment too difficult,

- you are allowed to form a group with another student
- maximum two students per group.
- Group submission is subject to **2 marks penalty** for each student.

Under no circumstances should you solve it in a group and then submit it as an individual solution. This is considered plagiarism. There will be no acceptance for excuses such as forgetting to declare as group submission. Please refer to the "Special Instructions for Group Submission" and "Plagiarism Warning" on page 2 for more details.

Grading

- We accept submission only of Python 3 (3.8.10 in particular) programs.
- Programming languages other than Python 3 are not allowed.
- We will test and grade your programs on the `stu` server. Please make sure that your programs run properly on `stu` and not only on your own system.
- By default, we use the `python3` program installed in folder `/usr/bin/python3` on `stu` for grading. Unless stated otherwise for individual tasks, you are allowed to use libraries installed in public folders of `stu` (e.g. `/usr/lib`) only.

- Your programs will be graded automatically using scripts.
  - Please make sure that your programs behave exactly the same as described in this document, because the grading scripts are unable to award partial marks.
- A set of testing scripts is released to you in the assignment package.
  - These test scripts cover common cases that your programs are expected to handle.
  - During actual grading, we will use extra cases to test your programs. Hence, passing all the released test cases does not guarantee that you will get full marks.
  - In addition, we will detect fraudulent cases such as hard-coding answers within programs and deduct marks accordingly.

We will deduct 1 mark for failure to follow instructions.

Program Submission

- For individual submission, please name your two source file as `Server-<Matric number>.py` and `Client-<Matric number>.py`; and submit it in a single zip file `<Matric number>.zip`, to the `Assignment_2` on canvas.
- Here, `<Matric number>` is your matriculation number which starts with letter A. An example file name would be `Server-A0165432X.py`.
- Note that file names are **case-sensitive** on `stu`.

You are not allowed to post your solutions to any publicly accessible site on the Internet.

Special Instructions for Group Submission

- For group submission, please include matriculation numbers of both students in the file name, i.e. `Server-<Matric number 1>-<Matric number 2>.py`.
- An example file name would be `Client-A0165432X-A0123456Y.py`.
- For each group, there should be one designated member who submits the file, to avoid problems caused by multiple branches within a group.
- **Do not change the designated submitter!** If the group needs to upload a new version, it should be done by the same designated submitter as well.

Plagiarism Warning

You are free to discuss this assignment with your friends. However, you should refrain from sharing your program, program fragments, or detailed algorithms with others. If you want to solve this assignment in a group, please do so and **declare it as group work**.

We employ zero-tolerance policy against plagiarism. If a suspicious case is found, the student would be asked to explain his/her code to the evaluator in face. A confirmed breach may result in zero marks for the assignment and further disciplinary action from the school.

Question & Answer

If you have any doubts on this assignment, please post your questions on `piazza` before consulting the teaching team. However, the teaching team will NOT debug programs for students and we provide support for language-specific questions as a best-effort service. The intention of Q&A is to help clarify misconceptions or give you necessary directions.

FAQ

We will collate your questions here: link

The Unreliable Channel

We have implemented 3 simulators on the machine `172.28.176.63`, to mimic unreliable channels. Your server and client have to connect to the simulators and transfer files.

- The simulator waits for both the client and server to connect the simulator via `TCP` connection before enabling the channel.
- The architecture can be found in fig:1
- Task
  - connect to the simulator with a handshake. The handshake is similar to Assignment 1, with an extra character `C` to signify client and `S` to signify server. The handshake details can be found in fig:2
  - The simulator will send you the current waiting list number; you need to wait till you receive `0_`. We allow only 30 students to connect to a simulator at any given time.
  - Then you transfer file from the server to the client. The input_file_name will be provided as command line parameter to the server
  - Save the received file to the file name provided to client as command line parameter
- You earn 1 mark per correct file transfer in the case of `Reliable Channel`.
- You earn 2 mark per correct file transfer in the case of other channels.
- Forgot to mention; the simulator really hates making friends. It will timeout and terminate in 100 seconds.

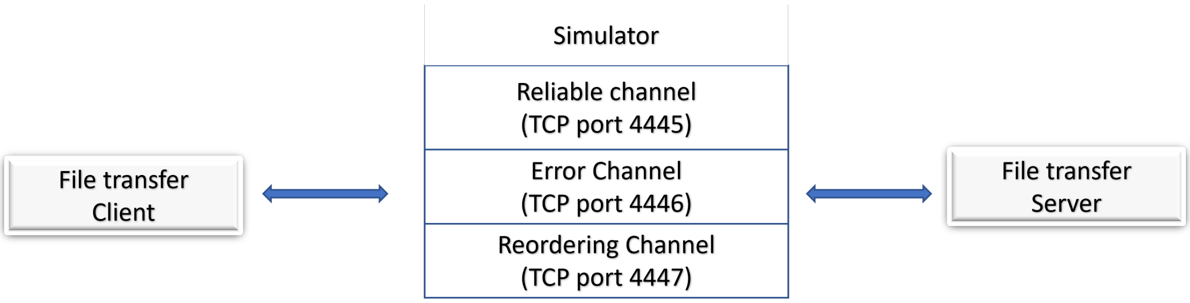
Simulator
Reliable channel (TCP port 4445)
Error Channel (TCP port 4446)
Reordering Channel (TCP port 4447)
File transfer Client
File transfer Server

Figure 1: Architecture.

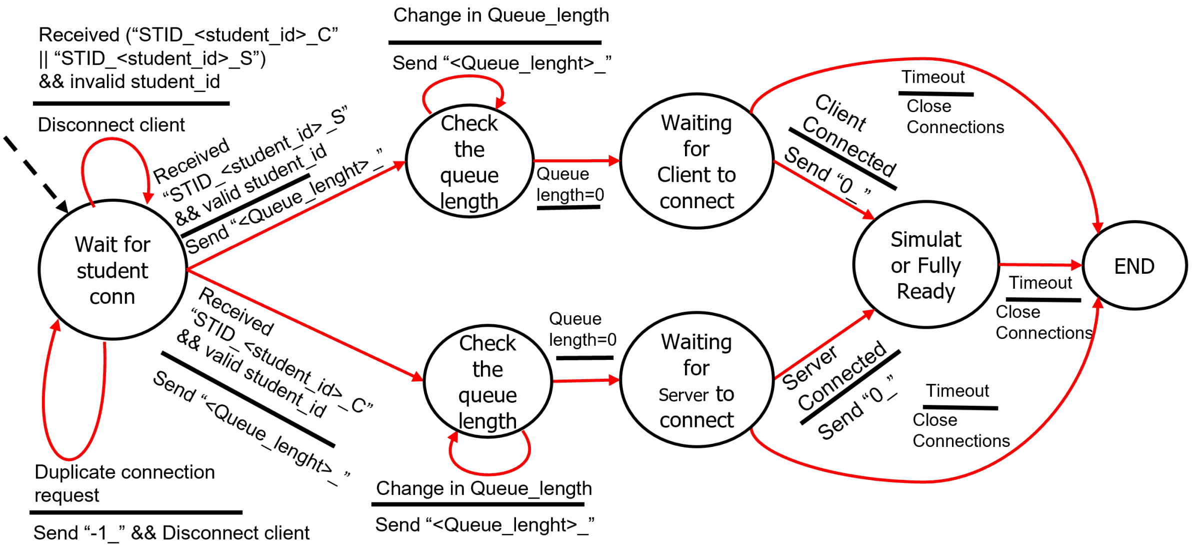

Figure 2: Handshake FSM.

Let us understand the Simulator further:

- All the simulators have the following constraints
  - The packet size from the server is fixed to 1024 B.
  - The packet size from the client is fixed to 64 B.
  - Partial packets will not be delivered
  - No packets will be dropped
- The 3 Simulators are:
  - *Reliable Channel (Mode 0)*: This channel introduces no error or reordering of packets. Fig 3
  - *Error Channel (Mode 1)*: This channel introduces errors in packets. Fig 4
  - *Reordering Channel (Mode 2)*: This channel reorders packets. Fig 5

Testing Your Program

To test your program, please use your SoC UNIX ID and password to log on to `stu` as instructed in Assignment_0.

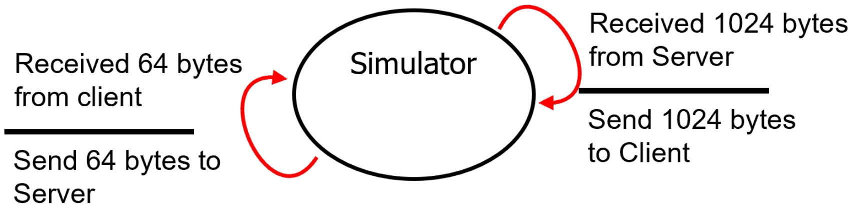
Simulator
Received 64 bytes from client
Send 64 bytes to Server
Received 1024 bytes from Server
Send 1024 bytes to Client

Figure 3: Reliable channel FSM.

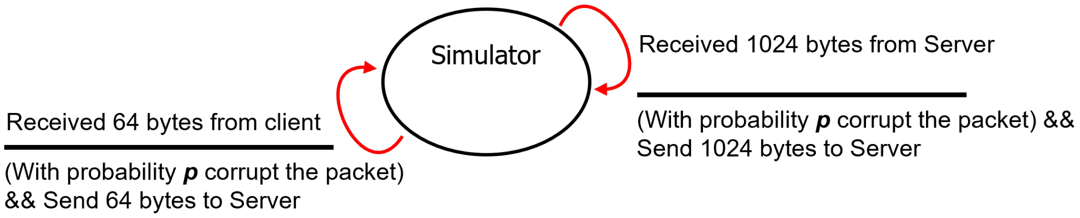
Simulator
Received 64 bytes from client
(With probability *p* corrupt the packet) && Send 64 bytes to Server
Received 1024 bytes from Server
(With probability *p* corrupt the packet) && Send 1024 bytes to Server

Figure 4: Error channel FSM.

- Since you have 2 programs, you will be running them on 2 separate terminals
- Your program should receive five **command-line argument** as the following command shows:

  ```
  python3 Sever-A0165432X.py  <ip_address> <port_num> <input_file_name> <mode> <st
  key>
  ```

  ```
  python3 Client-A0165432X.py <ip_address> <port_num> <output_file_name> <mode> <s
  key>
  ```

  - `<ip_address>`: The IP address of the machine on which the simulator is running.
  - `<port_num>`: The port number of the `TCP` socket corresponding to the simulator.
  - `<input_file_name>`: The file to be transferred
  - `<output_file_name>`: The destination file name
  - `<mode>`: Is the integer input corresponding to the simulator we are connecting to
  - `<student_key>`: The 6 digit key is the same as the one used in Assignment_1.
- During the evaluation, we will change the input files, the IP address and the port number; hence your code should accept all the above mentioned command line parameters.
- Note that your program should not read from `stdin`.
- Your program can print anything to `stdout` or `stderr`, and our test script will silently ignore them.
- We also release a set of grading scripts to you under the **test** folder.
- To use the grading script, please upload your program along with the `test` folder given in the package, to `stu`. Ensure that your program and the `test` folder are

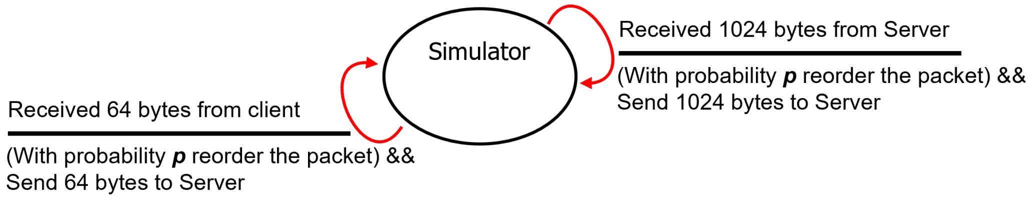
Simulator
Received 64 bytes from client
(With probability ***p*** reorder the packet) &&
Send 64 bytes to Server
Received 1024 bytes from Server
(With probability ***p*** reorder the packet) &&
Send 1024 bytes to Server

Figure 5: Reorder channel FSM.

in the same directory. You can run the following command for help:

```
bash test/FileTransfer.sh -h
```

- To run the test script for the server, kindly use the `-s` option

All of you will be connected to a single simulator, hence start the assignment early to avoid "congestion" during the last few days.

Template

We have released the template of the code.

- The entire code structure is implemented.
- Handshake code is implemented.
- Few helper functions like Packet reception code are implemented.
- Kindly implement only the part of the code with `#TODO`

Enjoy!!!