CSC257/457: Computer Networks
Project 1: Rock Paper Scissors

September 8, 2021

Objective

Your objective for this project is to implement a networked game of *Rock, Paper, Scissors* (RPS). The rules of the game, when played in person, are simple: two people each internally choose one of rock, paper, or scissors, there is a countdown, then each player simultaneously reveals their choice. Rock beats scissors. Paper beats rock. Scissors beat paper.

For this assignment, you are asked to write a single program which can serve as both the game client and the game server. You are free to use either transport layer protocol as you see fit. (I.e., you may use UDP – see the corresponding wireshark lab – or TCP). The purpose of this assignment is to give you practice with both socket programming AND developing network application protocols.

Program usage depends on commandline arguments:

1. `./rps player server port`
2. `./rps player client addr:port`

In both cases, the first commandline argument represents a player's name. The first invocation should create a RPS server on the specified port, and the second should contact a server at the specified port and address combination. You are encouraged to use local IP addresses for debugging your code, but you should test your ability to communicate across a live network at least once before submitting (E.g., cycle1 vs cycle3 on the instructional network).

Note that your program should serve as both client and server for your own implementation of RPS, but you are not expected to develop programs which can interoperate with other class-members (though that would be a happy accident).

Protocol Specifics

1. The server process must be executed first.

2. Before any network activity, the server process must greet the invoking player and state that they are waiting on an opponent.

3. Before any network activity, the client process must greet the invoking player.

4. The client initiates communication to the server and communicates the client player name – exact details are up to you.

5. The server must acknowledge the client and communicate the server player name – exact details up to you.

6. Once both client and server have completed this initial "handshake", they must communicate to their local users the name of their opponent, then prompt their local user for a move.

7. Once both client and server users have selected moves, the processes must inform each other of the selected moves.

8. Once client and server *both* confirm their users have selected moves, they then exchange moves.

9. After confirming that moves have been successfully exchanged, both processes begin a dramatic countdown, and then print both players moves and the name of the winner at the end. The dramatic countdown can simply be printing "3, 2, 1" with one-second pauses between characters. This should be hard-coded and does not have to involve the network.

10. Both processes terminate – cleanly bringing down any open sockets.

Collaboration

You are allowed to collaborate with a partner on this project. You will both receive the same grade for the project. Your writeup must include a section titled "Collaboration" which indicates who worked together, and how the workload was distributed. One student should submit the source code and writeup, the other student should submit a text file indicating that they collaborated with a partner, and name the partner.

Writeup

You must produce a *short* writeup explaining your implementation of these general protocols. You must indicate if you collaborated with anyone regarding the protocol you developed. You should state whether you chose to use UDP or TCP as your transport layer protocol, and clearly indicate how you performed the initial handshake and move exchanges.

Submission Instructions

Upload a ZIP archive with everything (source code and writeup) to BlackBoard before the deadline.

You must also include a plain-text `README` file in your src directory which describes how to build and run your program, and includes the names and UR NetID's for each contributing team member. Zip your `src` directory, `README`, writeup into a single file and upload to blackboard.

This project is due by 11:59pm EDT on September 24th 2021.