Port Knocker

Mykytenko Ihor, s17288

The project fully implements the features described in the specification provided and was designed in accordance with it. Therefore, the normal flow of the program would look like this:

1. The server is launched with port numbers as its command-line arguments(minimum 5 is required), and a knocking sequence for authorization is formed out of those ports(in shuffled order), to which the server starts listening.

2. The client is launched with server IP address and port knocking sequence as it’s command-line arguments. Then it sends packets to the ports in this sequence and if it is correct, the client receives a TCP port number from the server, initiates a connection(which is logged by the server) and receives a greeting over TCP. If auth attempt is unsuccessful, client terminates after 2 seconds of waiting.

3. If the client sends the number of packets which is less than the length of the authorization sequence, it has a chance to later send the missing packets and authorize. In case there’s more packets sent, the ones which came after completing the sequence(valid or not) will stay in the map of auth attempts as a new authorization attempt.

The major drawback of the implementation is that assumes that the client knocks a sequence of the proper length. If there are more packets coming, the future authorization attempts for that client will be seriously complicated, although possible(It would be needed to send exactly the number of packets to fail authorization with invalid sequence of valid length, and then start anew).

Regarding the other required details, it covers the

requirements. Any number of clients can attempt to authorize at once, because the application assigns a separate thread to handle a dialogue after a successful authorization, and before that data on each port is processed with no regard to number of clients.

The code in general is commented well and logs the actions taken on various stages of execution. Therefore, I kindly ask to contact me in case of any uncertainty.