Socket Programming Assignment

Problem Description: Design and implement a socket programming system where a server manages client connections, maintains a dictionary mapping client names to public keys and facilitates secure communication and video streaming among clients. The system should allow clients to communicate with each other through the server securely, notify existing clients of new connections, and enable secure one-to-one communication by requesting public keys from the server for encryption. Additionally, the server should stream a video file requested by a client without actually saving the file at the client's end.

Public key Cryptography: In this problem statement, you can use the **RSA algorithm** to generate the public and private key pairs at each client, share the generated public key with the server and keep the private key secret to decrypt the cipher text. More details are available here.

Server Tasks: [20 marks]

1. *Client Connection Management*
   a. [(4+1+1) marks] Create a server socket to receive client connection requests and ask for their name (e.g., `Enter your name:`) and the client's generated public key (e.g., `Enter the public key:`).
   b. [1 mark] It maintains a *dictionary* and stores the client's name and its public key as the *value*.
   c. [1 mark] As it receives a new client connection request, the server broadcasts client's name and public key from the dictionary to all the connected clients.
   d. [(1+1+1) marks] If a client wishes to disconnect, it has to send a `QUIT` message to the server and its entry gets removed from the dictionary, and the connected clients are notified.
   e. Note that the server stores the client name and its corresponding public key in a dictionary. As a new client connects to the server, it's name and public key are stored in the dictionary and then, the server broadcasts this dictionary with all the connected clients. Hence each client stores the broadcasted information at its end. So, if client 1 wishes to secretly communicate with client 2, then it is not required to make a separate request for its public key to the server but instead get the key from the stored entry of the dictionary at its end. This way multiple requests for public keys to the server are avoided.
2. *Secure Communication Management*

   [(2+1+1+1) marks] A client uses the public key of another client with which it wishes to communicate, encrypts its message, generates a cipher text and broadcasts via the server to all the clients.

   [**Note:** The aim behind broadcasting the cipher text is to observe that the clients who do not have the corresponding private key cannot decrypt the cipher text and get the original message. Only the intended client of having the private key can decrypt the cipher text and get the message displayed at its end.]
3. *Video Streaming Management*
   a. [1 mark] The server maintains a directory containing multiple resolutions (e.g., 240p, 720p, and 1440p) for a video file.

b. [3 marks] Upon receiving a client's request, the server streams the video by sourcing frames proportionately in sequence from each of the available resolution video files. For example, the streamed video alternates between different resolutions, ensuring that the frames are sourced sequentially from each available resolution. Specifically, the first one-third of the frames originate from the 240p resolution video, followed by the next one-third from the 720p resolution video, and finally, the remaining one-third from the 1440p resolution video. This approach provides a balanced distribution of frame qualities throughout the viewing experience.

```
For example:
     Client_1: List Available Videos
     Server:          VIDEO_1_240p.mp4,          VIDEO_1_720.mp4,
VIDEO_1_1440p.mp4
     Client_1: Play VIDEO_1.mp3
     Server: Playing video VIDEO_1.mp3 in different resolutions
```

Client Tasks: [10 marks]

1. *Connection Establishment*
   a. [3 marks] Create a socket and connect to the server.
   b. [1 mark] Send a client name and its generated public key to the server.
2. *Secure Communication*
   a. [(1+1) marks] A client maintains a dictionary at its end. This will get updated whenever the server broadcasts the dictionary again which means either a new client is added to the network or a client disconnected from the network. If a client wishes to communicate secretly with another client then it fetches its public key from the saved dictionary (broadcasted by the server). Meaning that, if `client_1` wishes to share a sensitive message with `client_2`, then It encrypts the sensitive message using `client_2`'s public key and sends it to the server, which, in turn, broadcasts the cipher text (encrypted message) to all the clients.
   b. [(1+1) marks] Decrypt and display the received cipher text. For example, all clients other than `client_2` cannot decrypt the cipher text as they do not have the corresponding private key.
3. *Video Playback*
   a. [(1+1) marks] A client requests the server to list the available videos at its end and starts playing a video file by providing its file name from an available video file list that the server displays.

Note: Use Python language to solve this assignment.

Submission Instructions

Submit a single zip file (named after your roll number) containing

- Two Python files:
  - **`roll_no_server.py`** (contains the server program), and
  - **`roll_no_client.py`** (contains the client's program)

where **`roll_no`** is your IIT Dharwad roll number.

- **README** file to state any explanation of the code files, program structure, demo instructions, etc.
    - Also, record a demo video that covers all the above-listed tasks, and share its **link** in the README file.