Contributors

Justin Xiao	xulongx2
Jerry Guo	zemingg2
Tiancheng Shi	ts15

Files in the directory:

chatserver.py

```
The server receive user names and encrypts users' passwords. It contains
functions that create the TCP socket, build connections to the clients, and
implementing operations like Broadcast Messaging(BM), Private Messaging(PM), View
Chat History(CH), and Close Socket for specific client(EX).
Functions include:
```

- sendint(data) : parse int into byte
- receiveint(data): parse byte into int
- get_key(val, my_dict): get the key of client, if cannot find the client, return "Unknown client".
- sendtoClient(sock, type, message): send message to client.
- chatroom (sockets, clients, address, client_keys): implementing the features:
    - Encrypt clients' passwords.
    - BM:
        - Server sends the acknowledgment back to the client to prompt for the message to be sent.
        - Server receives the message and sends that broadcast message to all other client connections.
        - Server sends the confirmation that the message was sent.
    - PM:
        - Server sends the list of current online users.
        - Server receives the information and checks to make sure the target user exists/online.
        - The server encrypts the private message.
        - Server sends the confirmation that the message was sent or that the user did not exist.
    - CH:
        - All chat history is stored in a file on the server.
        - sends the chat history to the client.
        - Client and server return to "prompt user for operation" and "wait for operation from client" state, respectively.
    - EX:

        - Server receives the operation and closes the socket descriptor for the client.
        - Server updates its tracking record on the socket descriptors of active clients and usernames of online users.

chatclient.py

```
It contains functions that build the connection with the server, and implementing
operations like Broadcast Messaging(BM), Private Messaging(PM), and Close Socket
for specific client(EX).
```

- sendint(data) : parse int into byte
- receiveint(data): parse byte into int
- accept_message(): handle incoming message from the server.
    - BM:
        - Client sends operation (BM) to broadcast a message to all active clients.
        - Client sends the broadcast message to the server.
        - Client receives the confirmation.
        - Client and server return to "prompt user for operation" and "wait for operation from client" state, respectively.
    - PM:
        - Client sends operation (PM) to leave a message to a specific client.
        - Client receives the list of online users from the server.
        - Client prompts the user for the username (of the target user) to send a message to, and the message to be sent.
        - Client sends the username and the message to the server.
        - Client receives the confirmation from the server.
        - Client and server return to "prompt user for operation" and "wait for operation from client" state, respectively.
    - CH:
        - sends operation (CH) to view the chat history.
        - receives and then displays the chat history on the screen.
        - Client and server return to "prompt user for operation" and "wait for operation from client" state, respectively.
    - EX:
        - Client sends operation (EX) to close its connection with the server and end the program.
        - Client should close the socket and exit.

To run/test the code:

```
1. Send all of these three files from local to remote using the command "scp
(local address) (remote address)"
    After that, run“ls”to see if these files have been uploaded successfully.
    eg.  scp ./Desktop/IS496/chatserver.py
zemingg2@student00.ischool.illinois.edu

3. Connect to two different student machines in four terminal windows with the
command "ssh (remote address)"
    Determine which terminal is the server and which terminals are clients.
    eg.  ssh zemingg2@student00.ischool.illinois.edu
```

```
         ssh zemingg2@student01.ischool.illinois.edu
         ssh zemingg2@student02.ischool.illinois.edu
         ssh zemingg2@student03.ischool.illinois.edu

4. Run the server script in the server terminal by typing "python3 chatserver.py"
+port
    eg.  python3 chatserver.py 410002

5. Then in the clients' command line windows, type "python3 chatclient.py"+
server's address + server's port + current client's name
    eg.  python3 chatclient.py student00.ischool.illinois.edu 410002 Jack

6. Set the password for each client.

7. Enter the operation each client want to do. Typing in:
    "BM": ask the server to broadcast a message to all other availible clients.
    "PM": ask the server to send the message privately to a specific target
client.
    "EX": close its connection with the server and end the program.
    The server closes socket and goes back to "wait for connection" state
```