UNIVERSITY OF NORTH TEXAS
COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CSCE 5050 – Application of Cryptography

Appendix: Client/Server Chat System

Overview & Specifications

This appendix describes a client/server chat system that takes benefit from the power of threads. The server of this system is accommodating multiple clients as they send and receive data at different intervals, utilizing threads.

The diagram, shown in Figure 1, shows the basic idea for the threaded server operation.

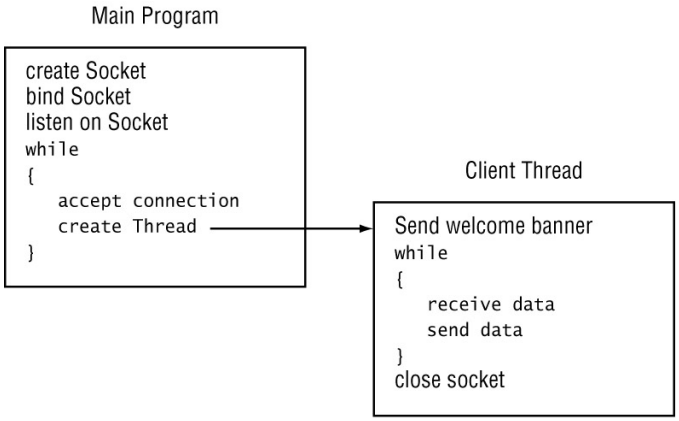

Figure 1: The threaded server's operation

The chat system includes two types of components: a chat room and the clients. The clients may appear at any time, but the chat room is a long-lived 'server' component. There is very simple user registration process; any chat client can send a message as soon as it connects to the server with a unique ID (i.e., registered). All messages are to be broadcasted to all clients connected to the chat room.

The chat server manages a chat session in chat rooms. At any given time there may be multiple chat clients in a chat session. The chat server is responsible for managing all the chat clients in the session and distributing chat messages. A chat client starts by requesting the communication parameters (server name and port numbers) for the chat server from a location server. The chat server will respond by welcome message and provides the chat client with the available list of chat rooms. The client then registers with the chat server (using a unique ID) to join one particular chat session. After joining the session, a chat client can allow the user to switch to one of the chat rooms and send and receive chat messages.

The communication between the client and server includes two types of messages: *control messages* and *chat messages*. The control messages will allow a chat client to perform tasks such as joining and leaving the chat session, creating a chat room, switching to a certain chat

room, etc. Chat messages are the public messages that delivered to all users in a given chat room. To send a public chat message, a chat client must be present in one of the chat rooms. The chat client sends all its public chat messages to the chat server, and it is the chat server's job to relay these messages to all the chat clients within that room.

Using byte-type arrays to read and write data makes stream classes inconvenient, especially when we are writing text in our chat communications. Stream readers and writers provide a convenient way to read and write text and binary data types to streams. This chat system utilizes the network stream, stream reader and stream writer to have the TCP socket accessible in the form of string data type stream. See Figure 2.

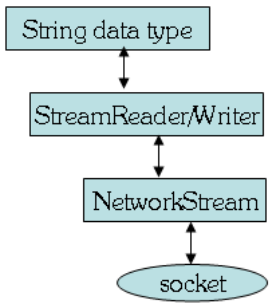

Figure 2: Using I/O Streams

Server Specifications

- Server is responsible for maintaining several Chat Rooms
- Server is capable to accept multiple Chat Clients at a time.
- Server accepts and responds to the following Chat commands (control messages). These commands can be in the form of button clicking or a value for TextBox.
    - **Register** – to connect to the server and join to the selected chat room.
    - **Create Room** – to create a new chat room and make it visible to all clients.
    - **Join** – to join to one particular chat room, this means the client will leave the current room and join the new one.
    - **Who** – Request for the names of all client in the room.
    - **Disconnect** – The client is no longer in the system
- Server prints a message (given in boxes below) on the server screen following every event.
- Chat Server keeps track of the chat client who join and leave each room, and display the following message to all chat client in this room

```
Client <client-name> has joined (or left) the room <room-name>.
```

- When a Chat Client starts up and registers with the Server. Server will display the following message to all chat clients on the system.

```
Client <client-name> has registered in the room <room-name>.
```

- A Client can create a new room. Server will respond to the client request and create the new room, adds this client to the new room, publishes the updated chat rooms list, and displays:

```
Client <client-name> has created and joined room <room-name>.
```

- When a client sends broadcast message to a room, the message is forwarded to all room attendance, and display the following only on the server screen:

```
Client <client-name> is speaking in room <room-name>.
```

Client Specifications

The first thing a Chat client does is to locate the remote Chat Service (IP address and port number) and register with the Server. Upon connection each new Client is presented with a list (ListBox) of available chat room. At the beginning only one chat room is available by default called "General". Additional rooms can be added later by the client, you can add some other rooms like, Science, Sport, Movies, Linux, Politics, etc. Client can execute one of the server commands mentioned above. After each command execution, SUCCESS/FAILURE status of each command should be printed out.

When Client creates a new room, it automatically becomes the first member of that room. If a room by this name already exists, an error message is returned.

Client can send messages to everyone in her room, just by typing the message and press enter (no command here).