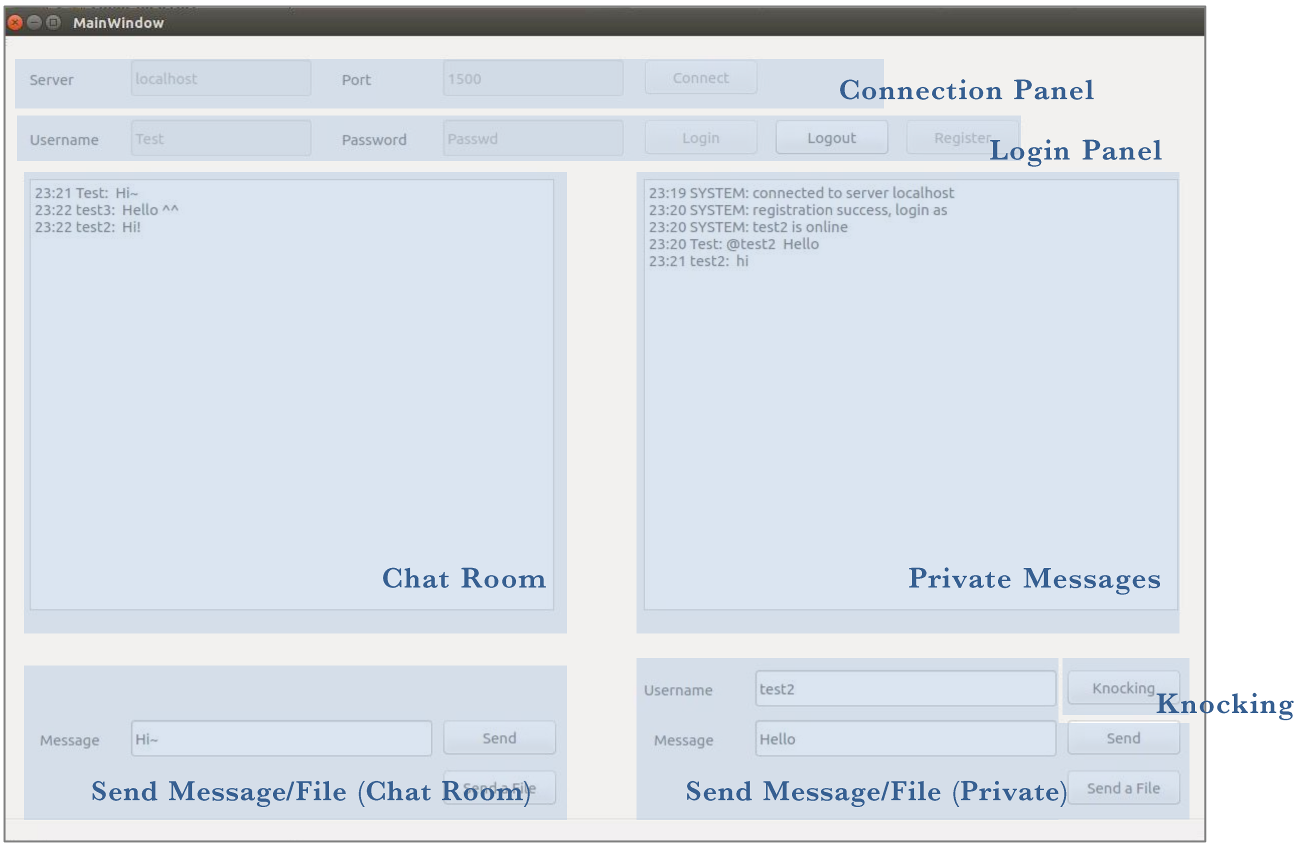
Computer Networks Final Project

Team 10

Role	Member
Protocol design, Code adjustment, Bonus part(offline message)	B02902009 黃昱維
Protocol design, Main architecture of server, client and GUI	B02902043 吳楹偉
Protocol design, Report	B02902049 葉為元

User Guide



Server: localhost
Port: 1500
Username: Test
Password: Passwd

```
23:21 Test: Hi~
23:22 test3: Hello ^^
23:22 test2: Hi!
```

```
23:19 SYSTEM: connected to server localhost
23:20 SYSTEM: registration success, login as
23:20 SYSTEM: test2 is online
23:20 Test: @test2 Hello
23:21 test2: hi
```

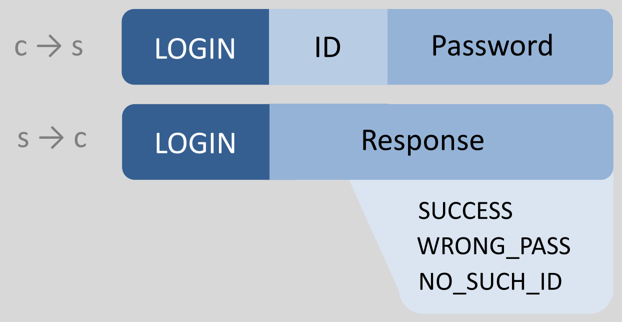
Message: Hi~

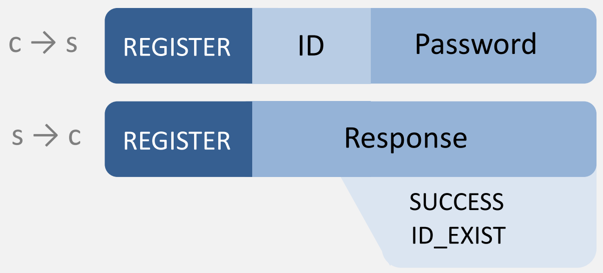
Username: test2
Message: Hello

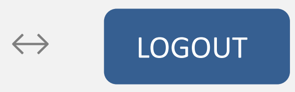
Protocol

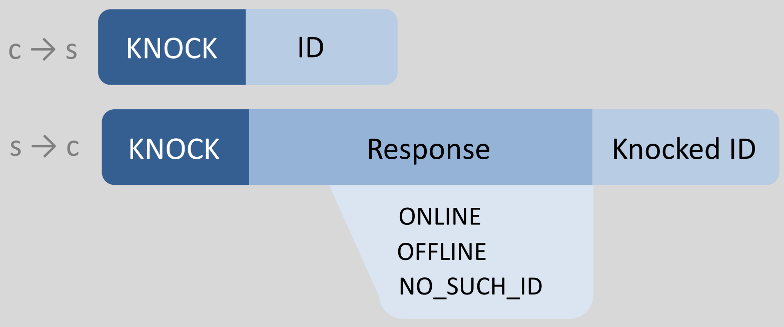
In every message, we start with an one-byte number to represent its type.(ex: LOGIN=1, LOGOUT=2...), following up with some corresponding information. The message used for client → server and server → client may have different format

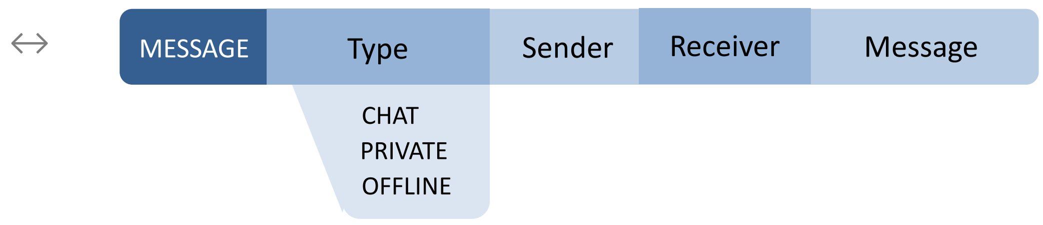
All messages are appended to 256 bytes before being sent.

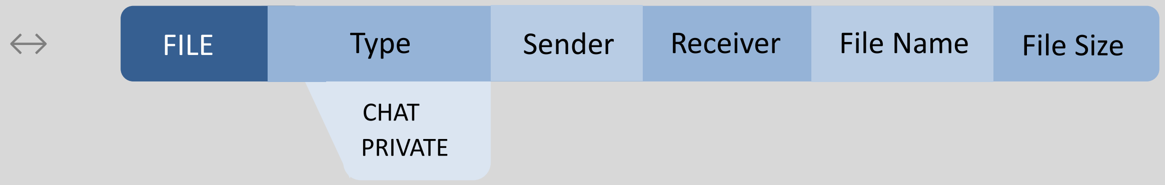












Process Flow

Main flow

Client connects to server → Server creates a new thread to handle the client

→ (all the other operations)

→ Client exits and closes the socket → Server closes the socket and the thread exits

For login/logout/register/knocking, it runs like:

Client sends a request message

→ Server checks the user list, tries to serve the request and reply if success

For message sending

Client sends a message

→ Server sends the message to all receivers and back to sender as well

→ Sender receives the reply (its own request) and knows the transmission is done

For file sending:

Client sends a request message

→ Server sends the request message to all receivers and back to sender as well

→ Sender receives the reply (its own request) and starts to send file (by 256 bytes packages)

→ Server transmits file packages to all receivers

Bonus(offline message):

Server receive message request and found receiver is offline

→ mark as an offline message

→ send back to sender and save one copy in memory

a client login and server found there's offline messages for it

→ Server sends offline messages to the user and delete them from memory

Challenges

We decided to use Qt as our GUI tool since all of us have experience writing C++ code. Then we faced a problem. The two threads on client, one for sending message and one for receiving message, cannot both access Qt GUI as we need. Thus we let main thread connect with GUI, and use signal to communicate with the other thread.

Then another problem popped out. Using signal would cause error message on Qt Creator, the IDE for Qt, due to its inner error. We looked on the internet and solved it by adding an #include “main.moc” statement at the end of the code.