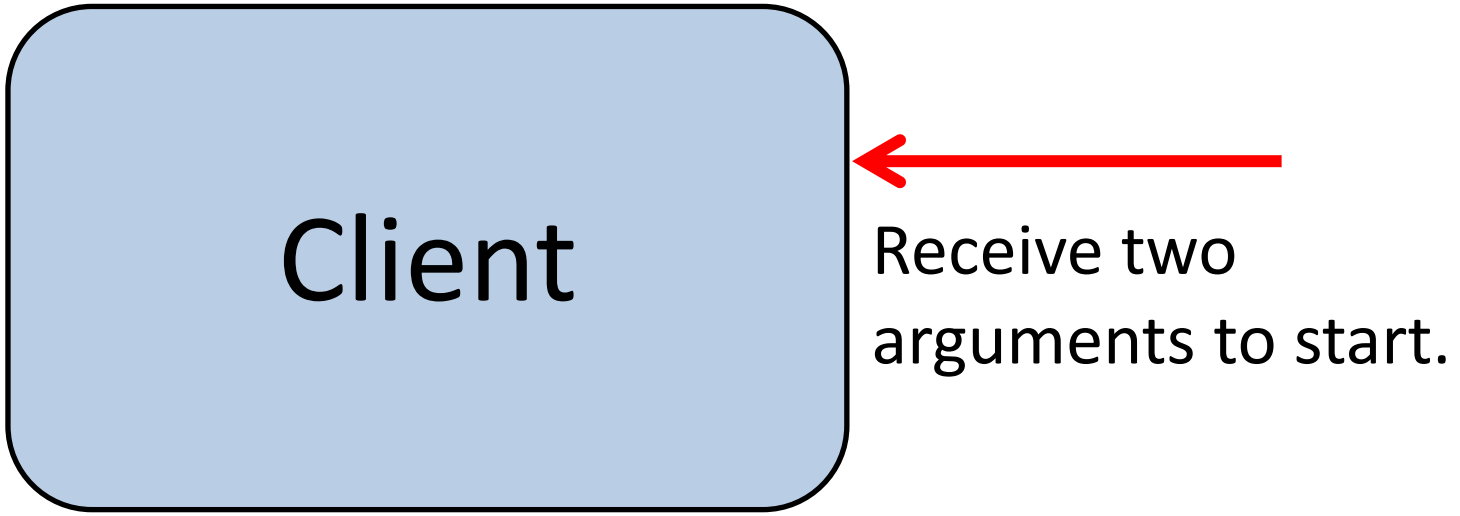
Homework1- Concurrent Chat Room

鄭力維

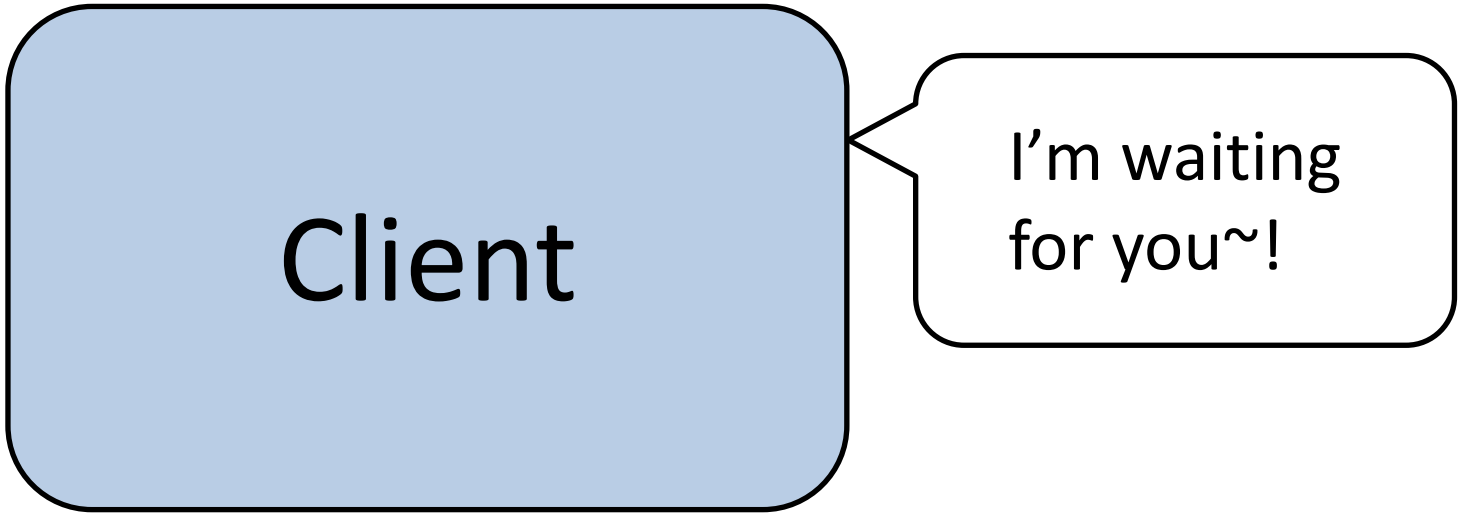
Client - startup

- You may give it a server IP and a port number as arguments in command line while starting the client.



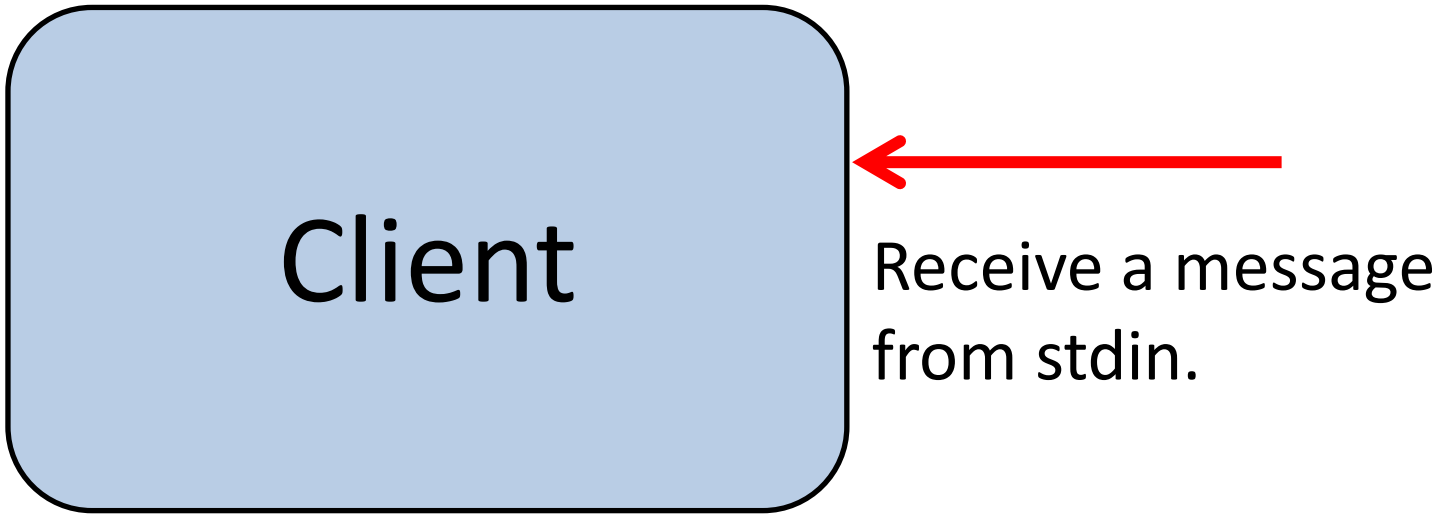
Client - startup

- If the arguments are correct, the client connects to the server.
- If the arguments are not correct or the client cannot connect to the server, wait for the user to re-connect again.



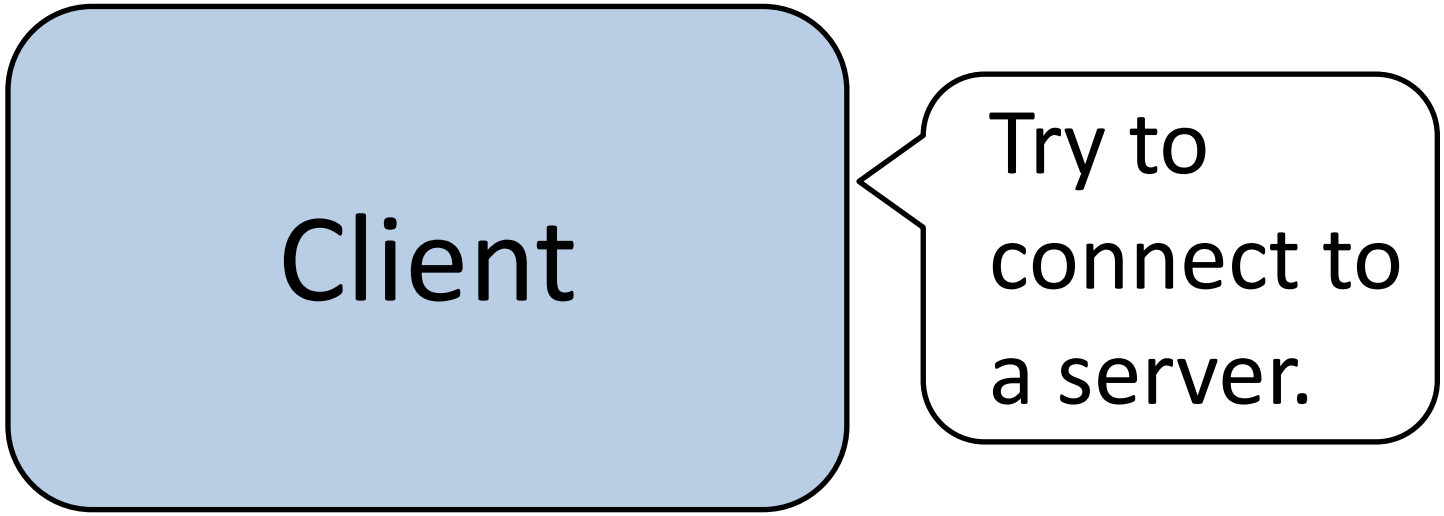
Client – messages from stdin

- While receiving a message from stdin, check if it is “/connect ”, “/quit ”, or other command.



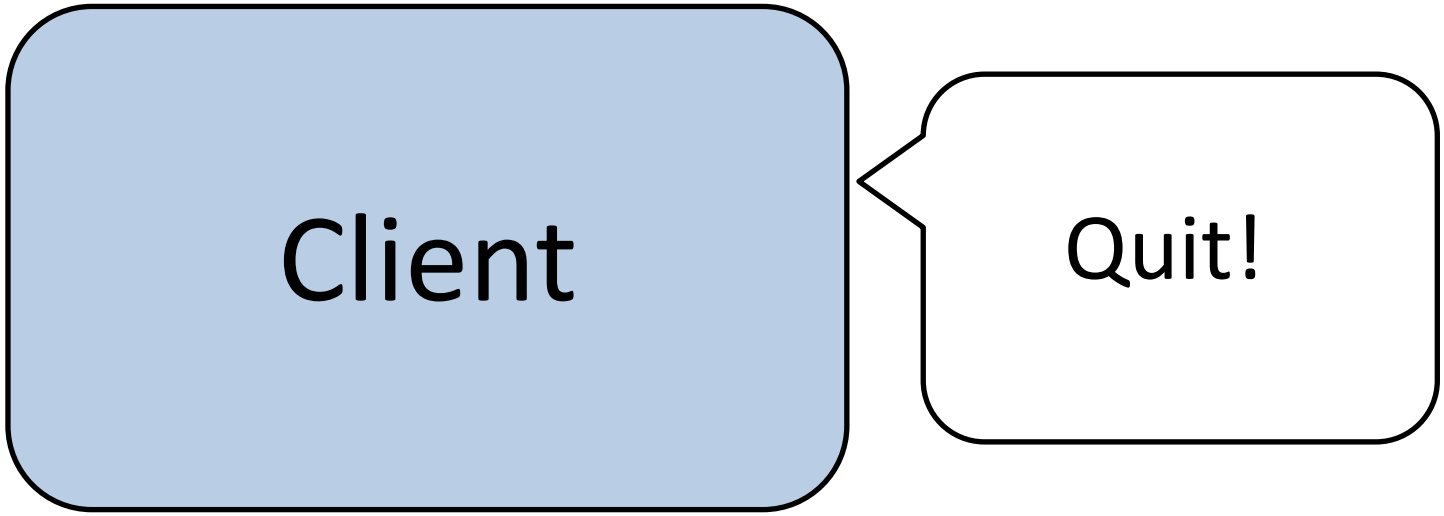
Client – messages from stdin

- If it is a “/connect ” command and the client has not connected to a server, handle it.



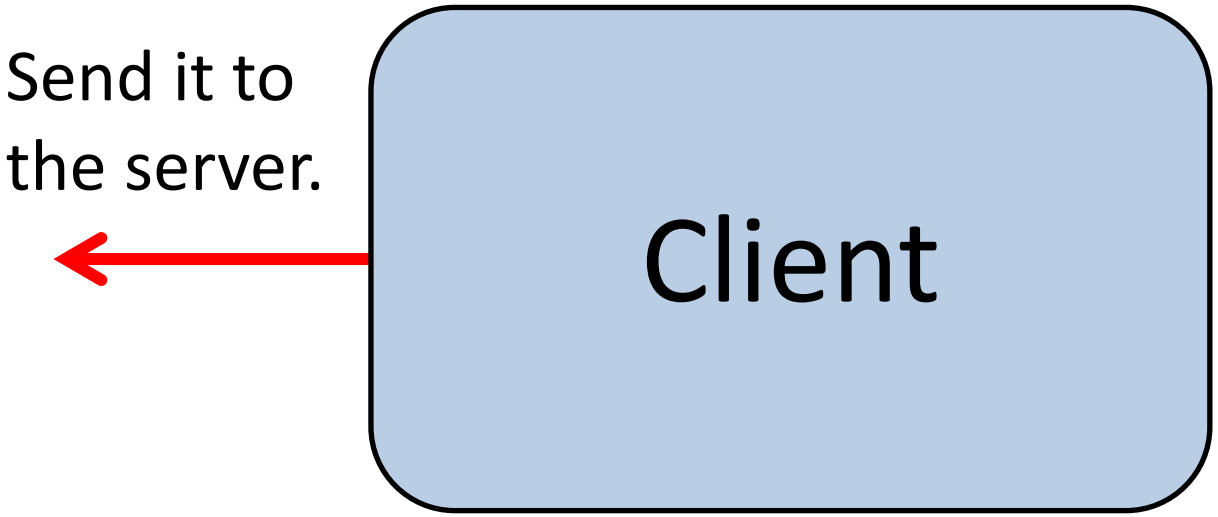
Client – messages from stdin

- If it is a “/quit ” command, end this process no matter it is connected to a server or not.



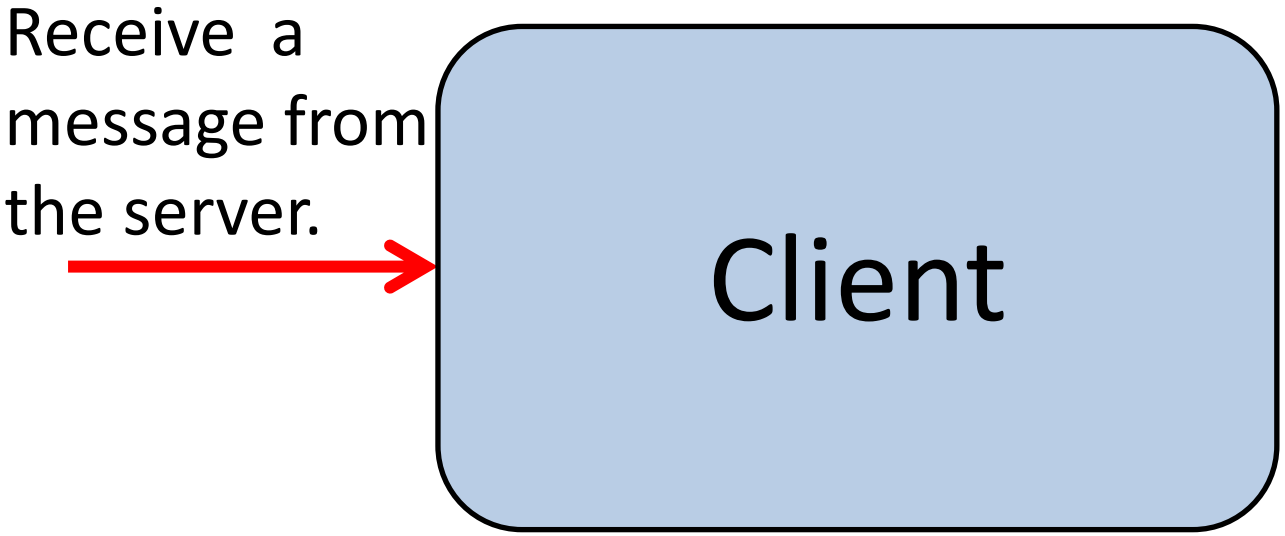
Client – messages from stdin

- Else send it to the server without modifying the content if it is connected to the server.



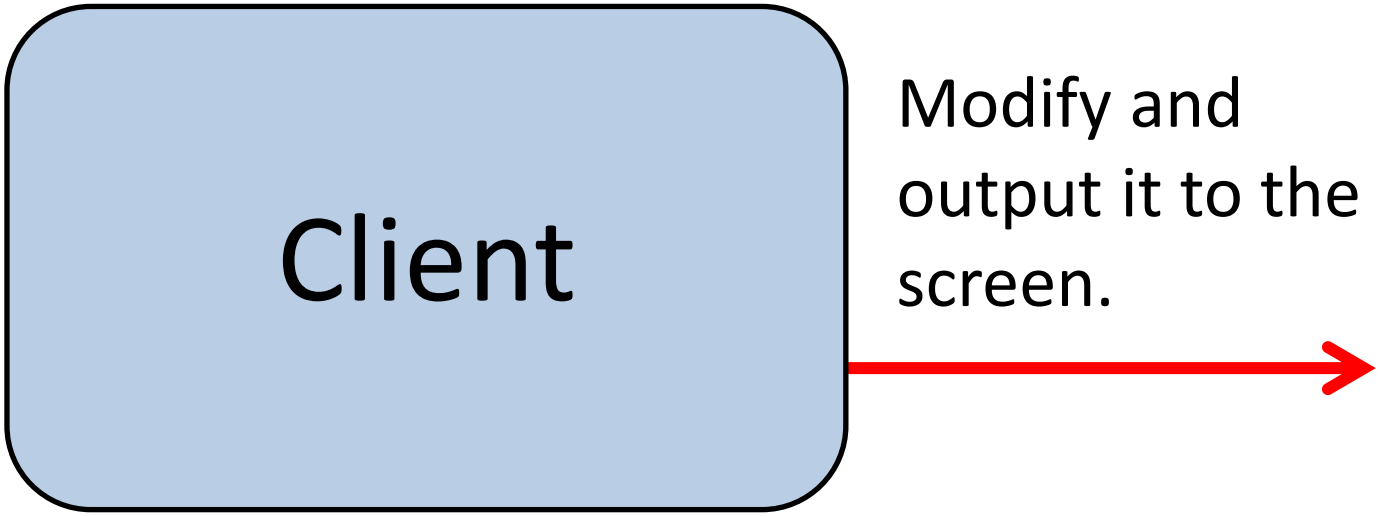
Client – messages from a server

- When the client receive a message from the server, check if it starts with “/serv ” or “/msg ”.



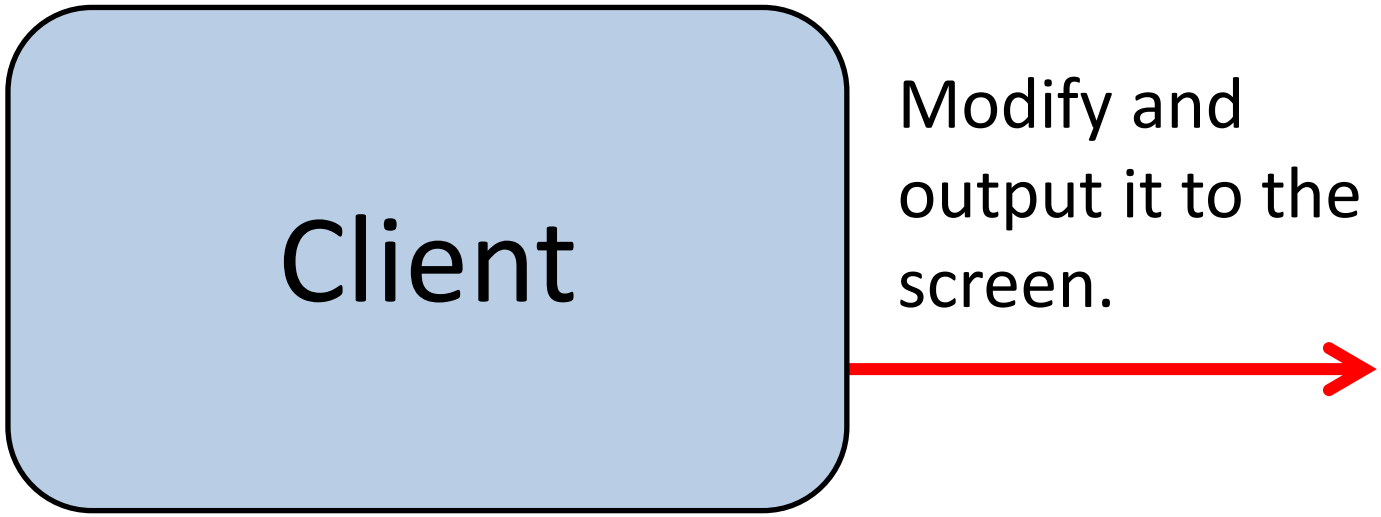
Client – messages from a server

- If it starts with “/serv ”, you should replace “/serv ” with “[Server] ” and output to the screen.



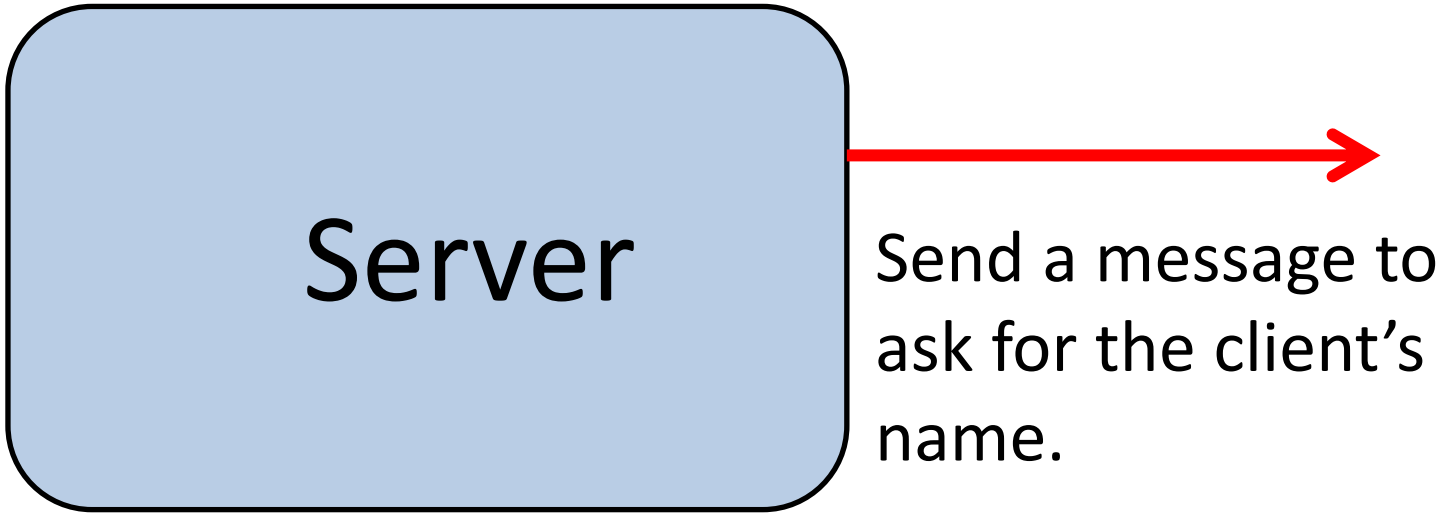
Client – messages from a server

- If it starts with “/msg ”, you should eliminate “/msg ” and output to the screen.



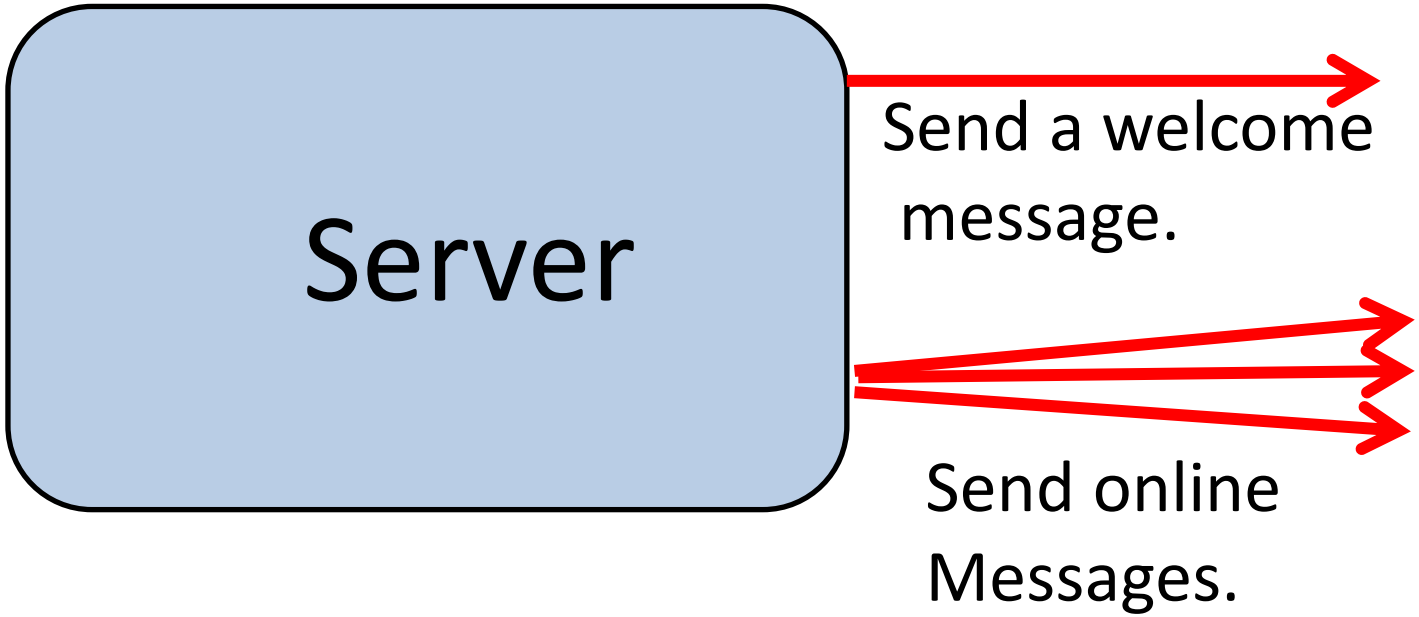
Server – setup user information

- While there is a client connecting to the server, ask the client for its username first.



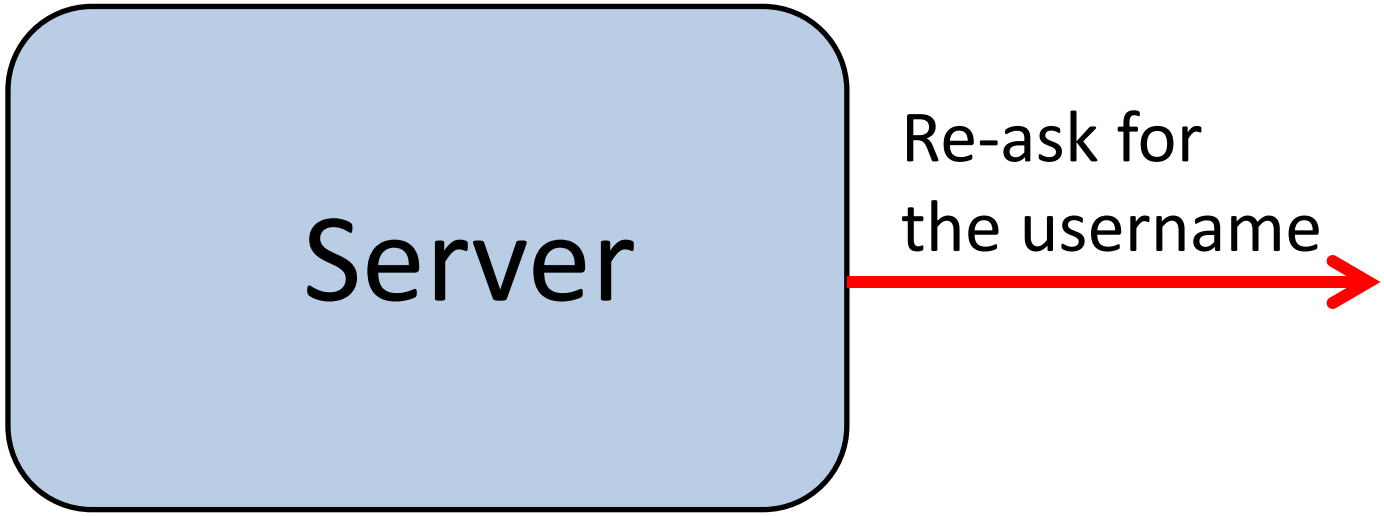
Server – setup user information

- If the receiving username is valid, the client is a valid user. Send a welcome message to it and send online messages to all other valid users.



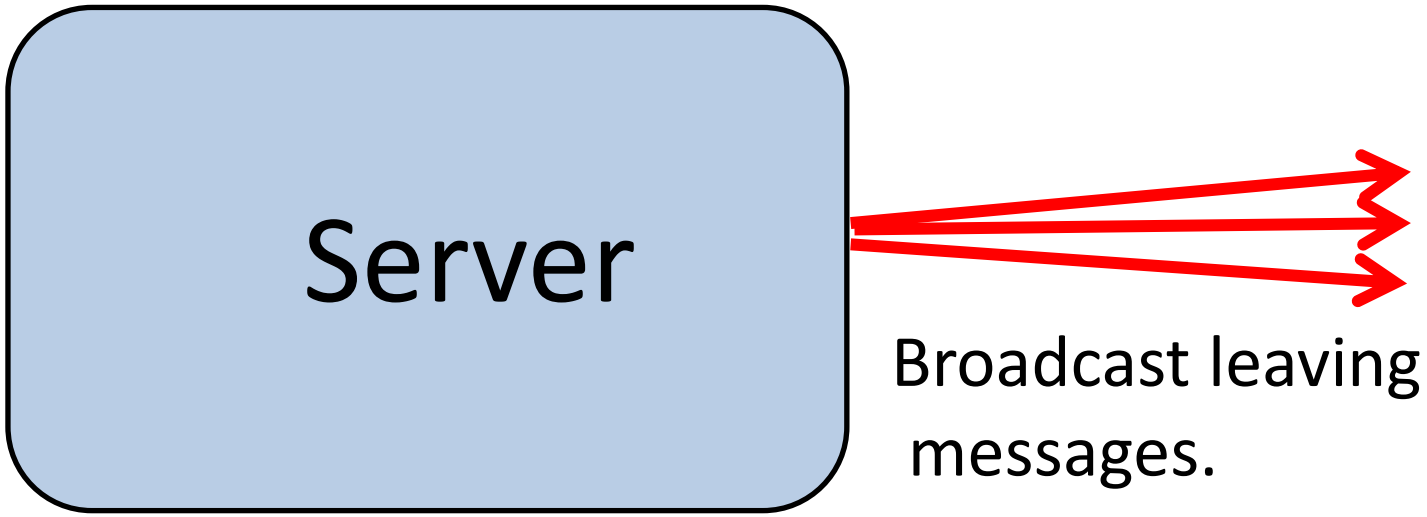
Server – setup user information

- Else the username is invalid, re-ask for the username again.



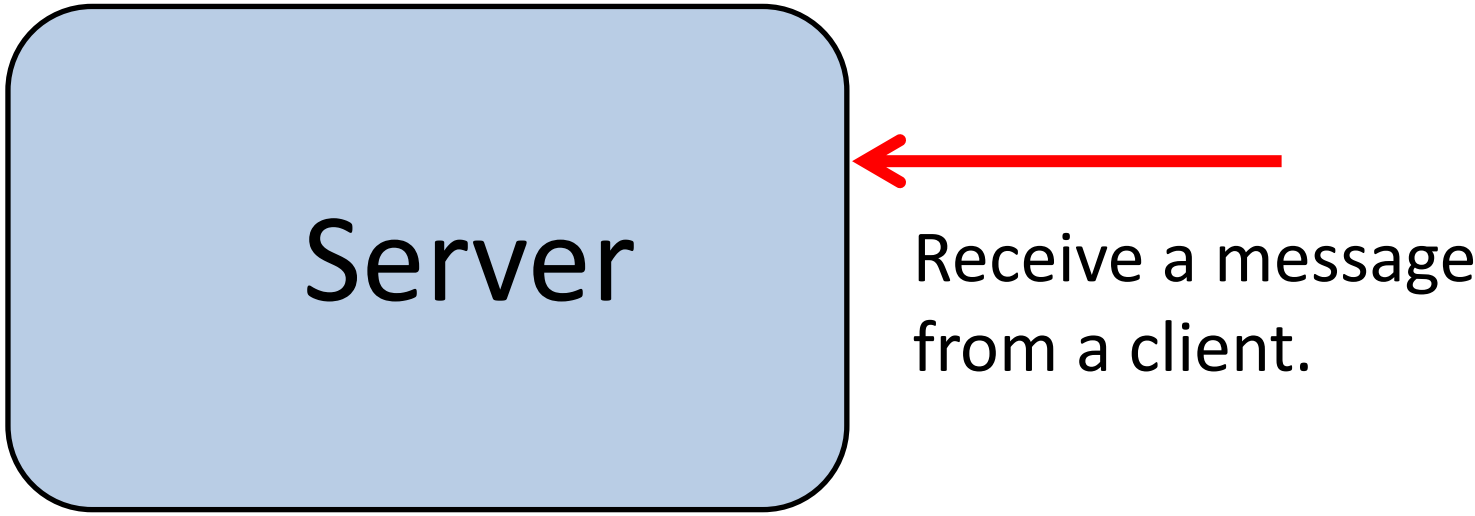
Server – leaving message

- While a valid user is leaving the chat room, send leaving messages to all other valid users.



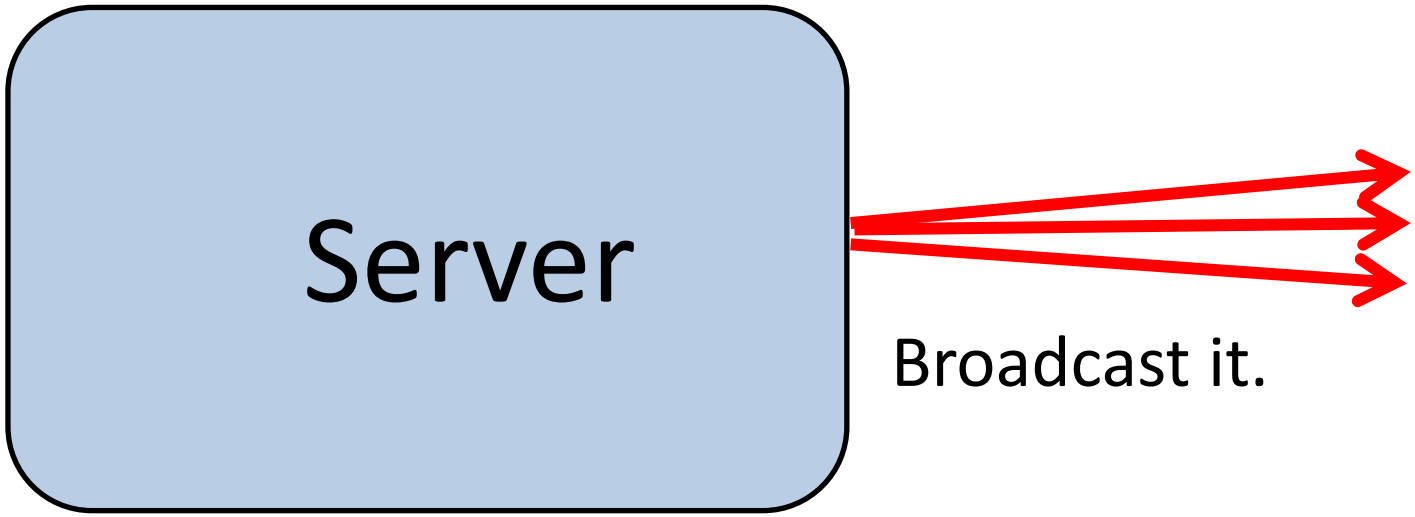
Server

- While receiving a message from a client, check if it is a valid command (except “/who ” and “/nick ”, other commands start with “/” is invalid).



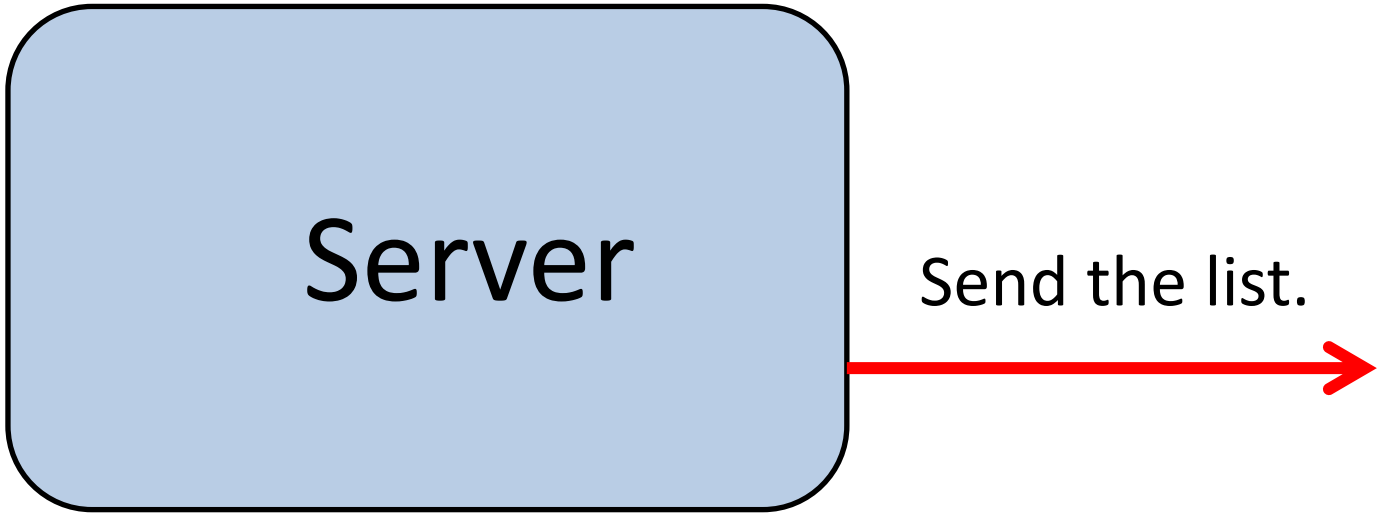
Server – chat message

- If the message does not start with “/”, it is a chat message, change it to proper format and broadcast it.



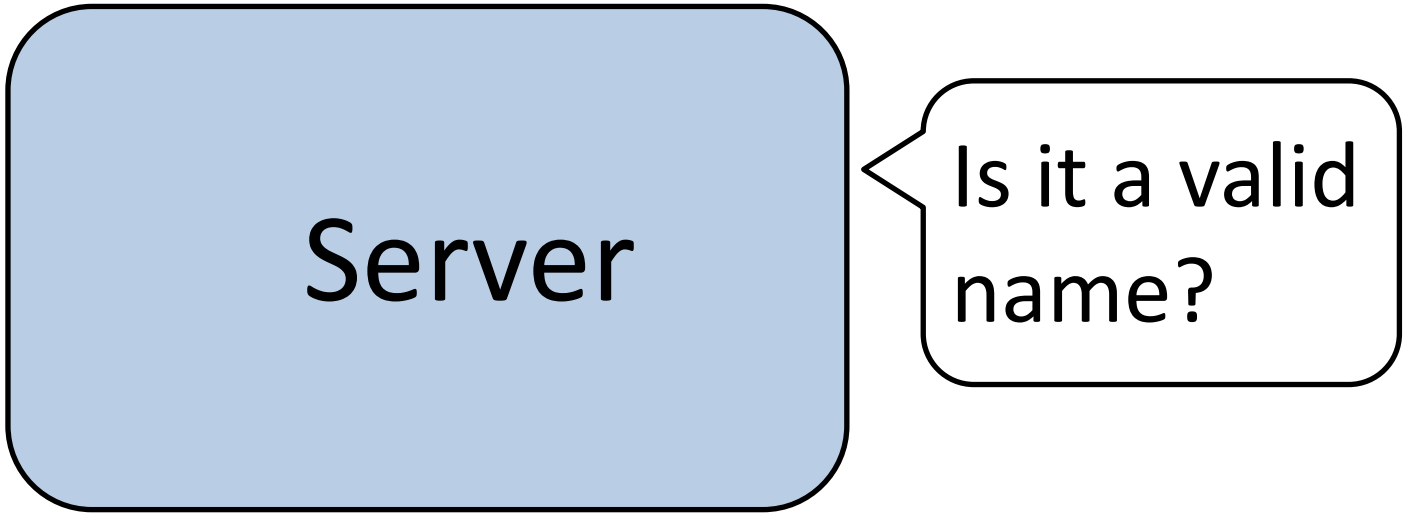
Server – who message

- If it is the “/who ” command, which means that the user asks for the information about all the valid users, server sends the list to the client.



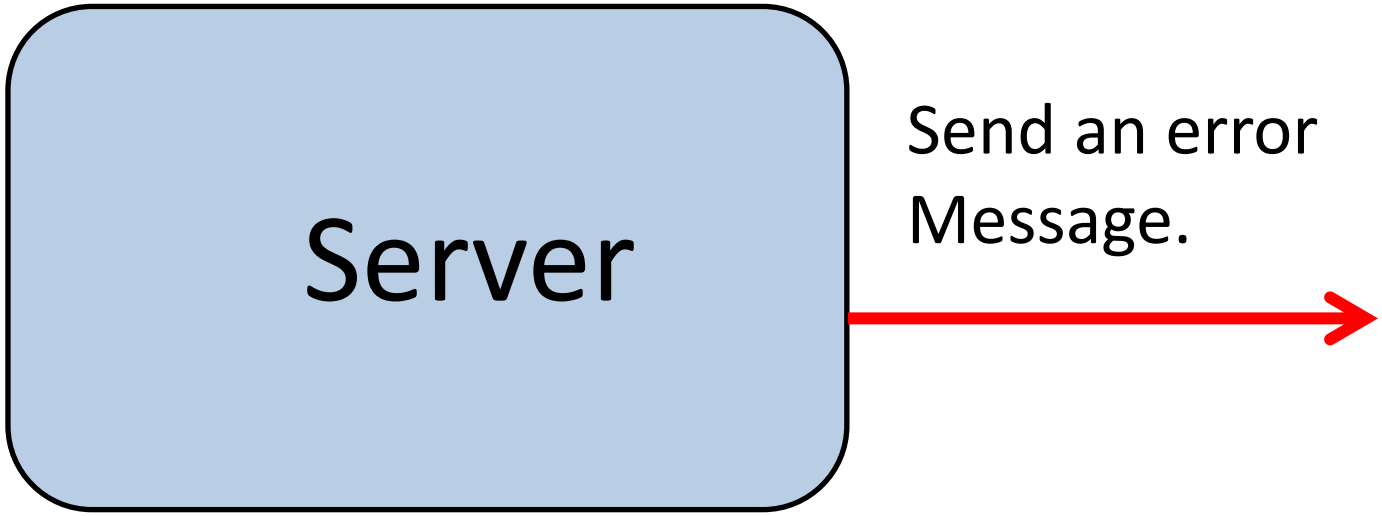
Server – nick message

- If it is the “/nick ” command, meaning the client wants to change it’s username, check if there is a username following the “/nick ” and the username is valid.



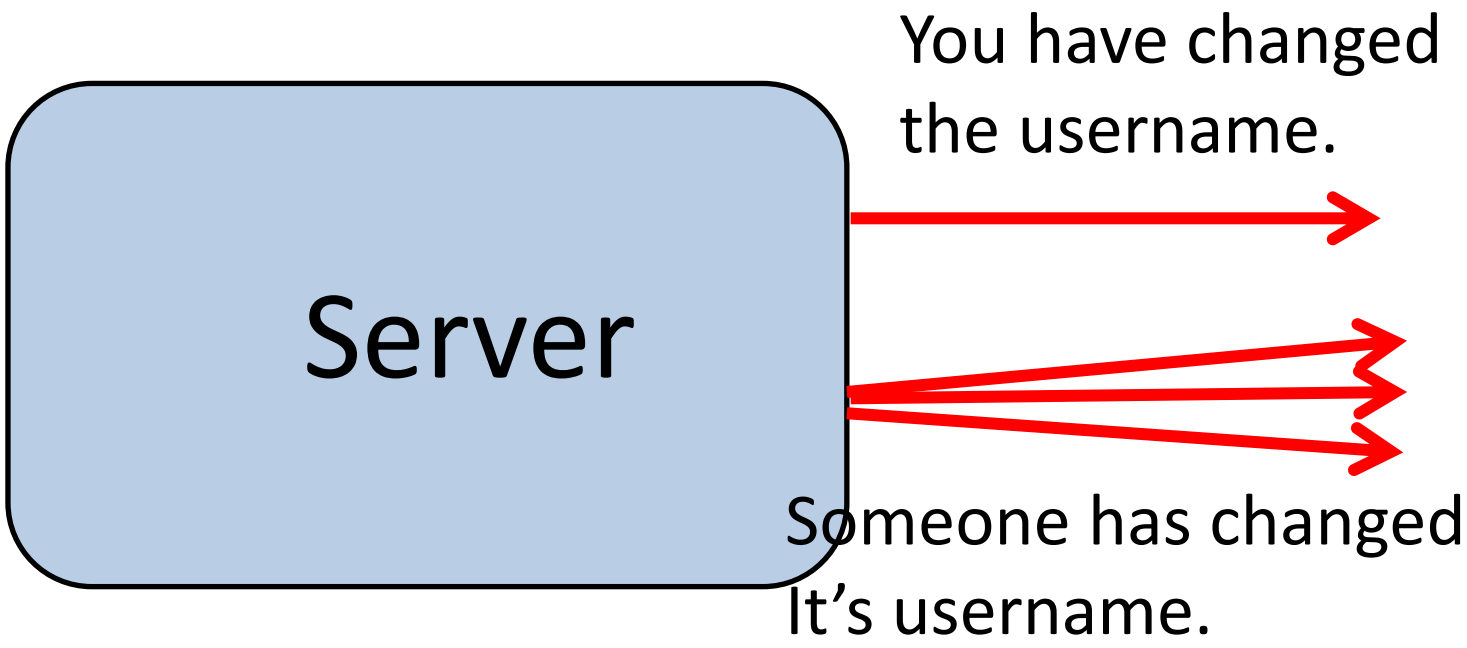
Server – nick message

- If there is no username or the username is invalid, server should send error message to the client.



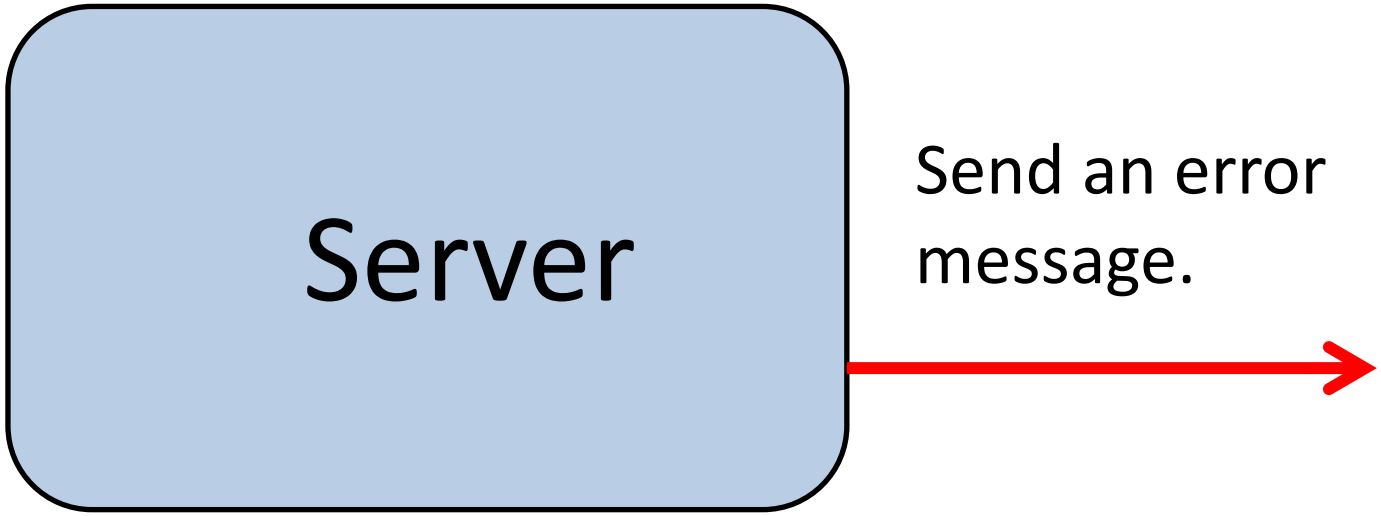
Server – nick message

- Otherwise rename the user, then send username-changed messages to it and other valid users.



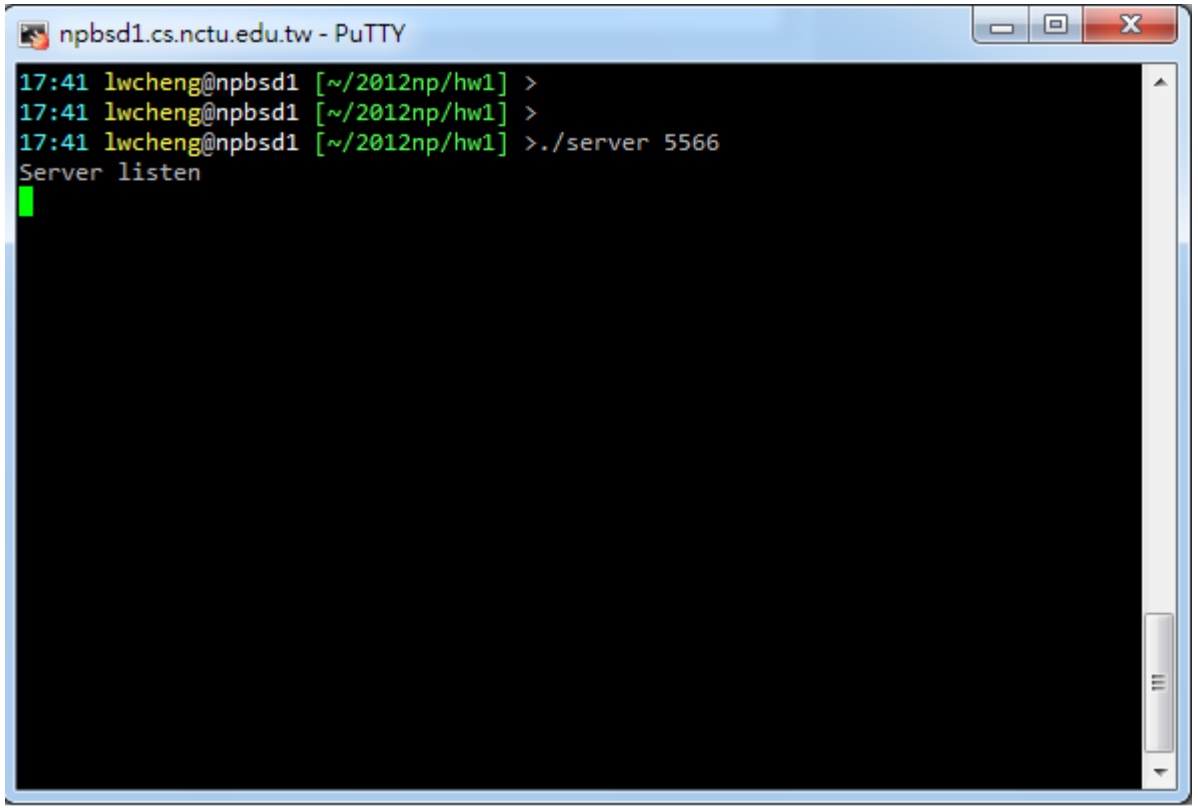
Server – invalid command

- If it is an invalid command, server sends an error message to the user.



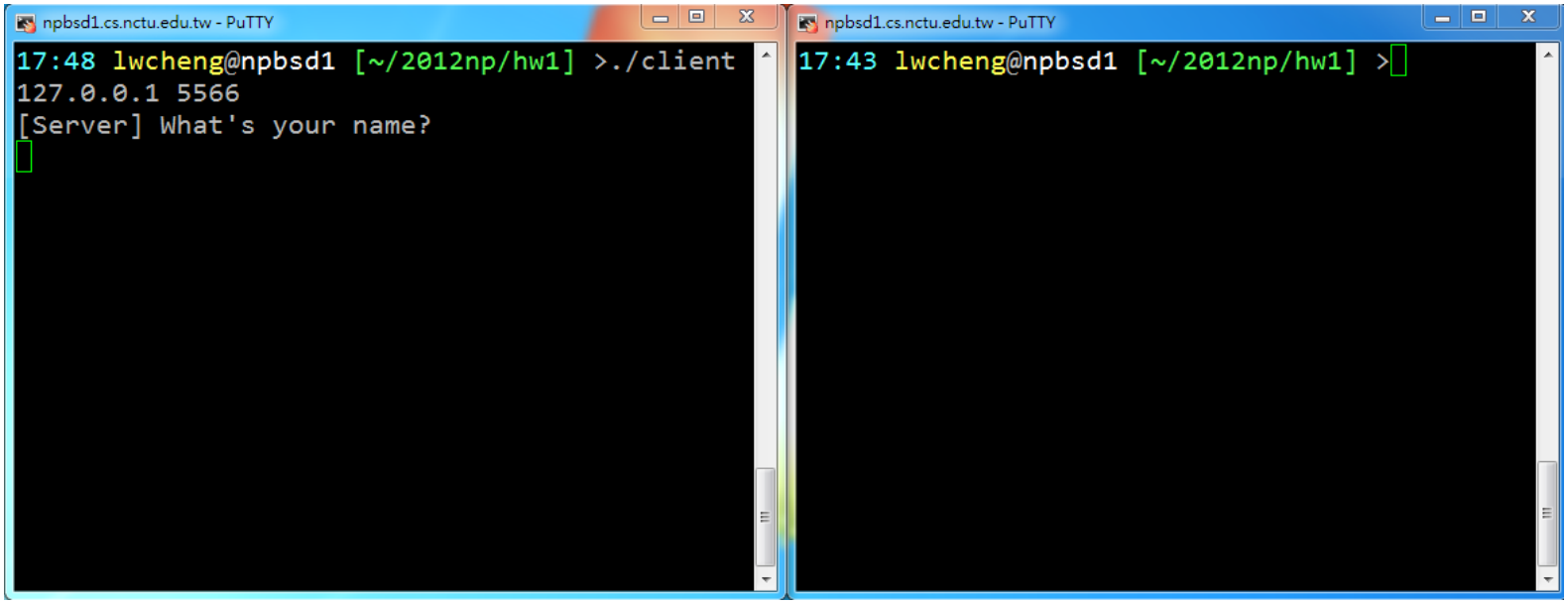
Demo flow in 2012/11/2

- Run the server with a port number.



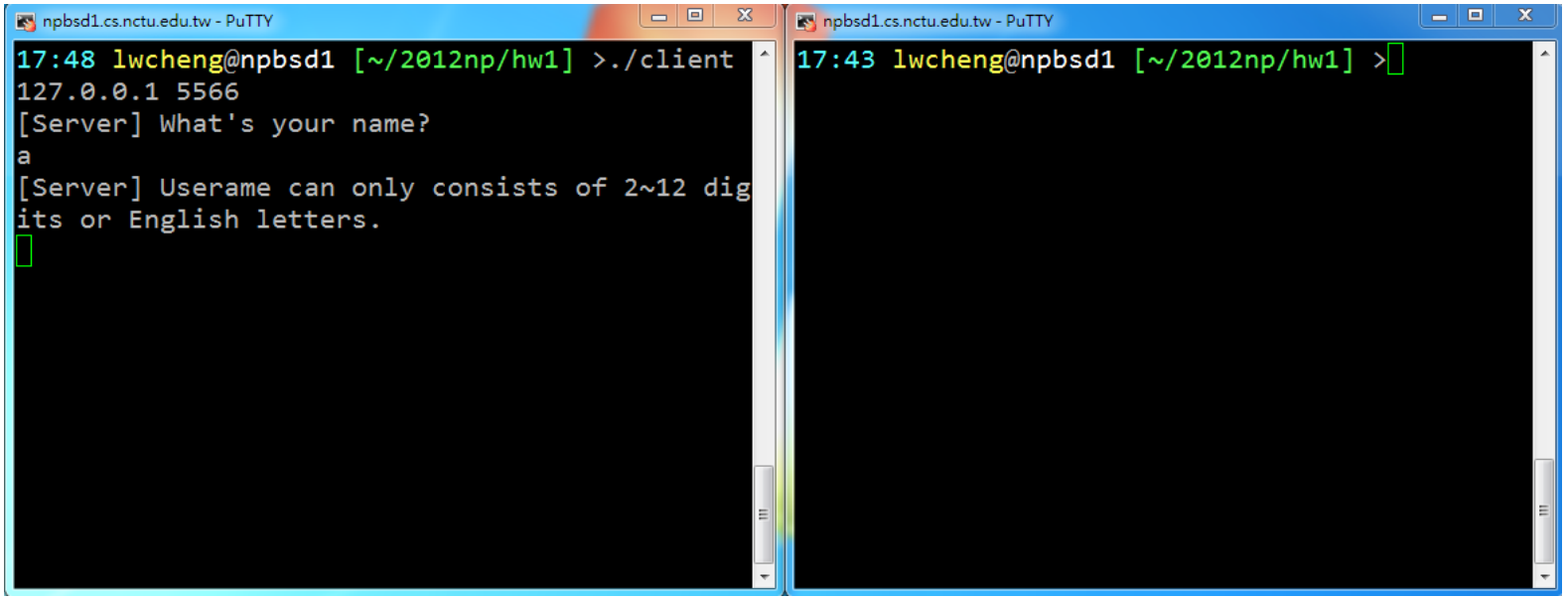
Demo flow in 2012/11/2

- Execute one client with two arguments (in the left terminal).
- After executing it, the server asks for the username.



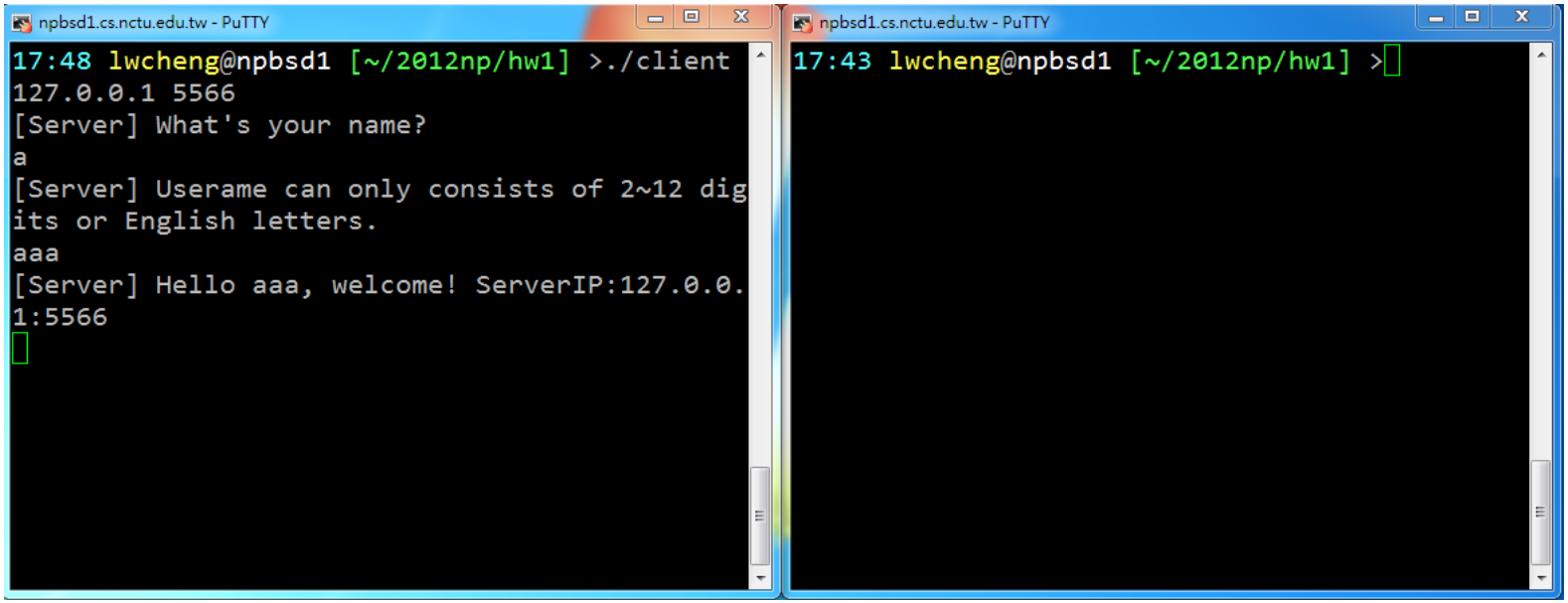
Demo flow in 2012/11/2

- The left client sends “a” as its username.
- Since the name is invalid, the server sends an error message to the client.



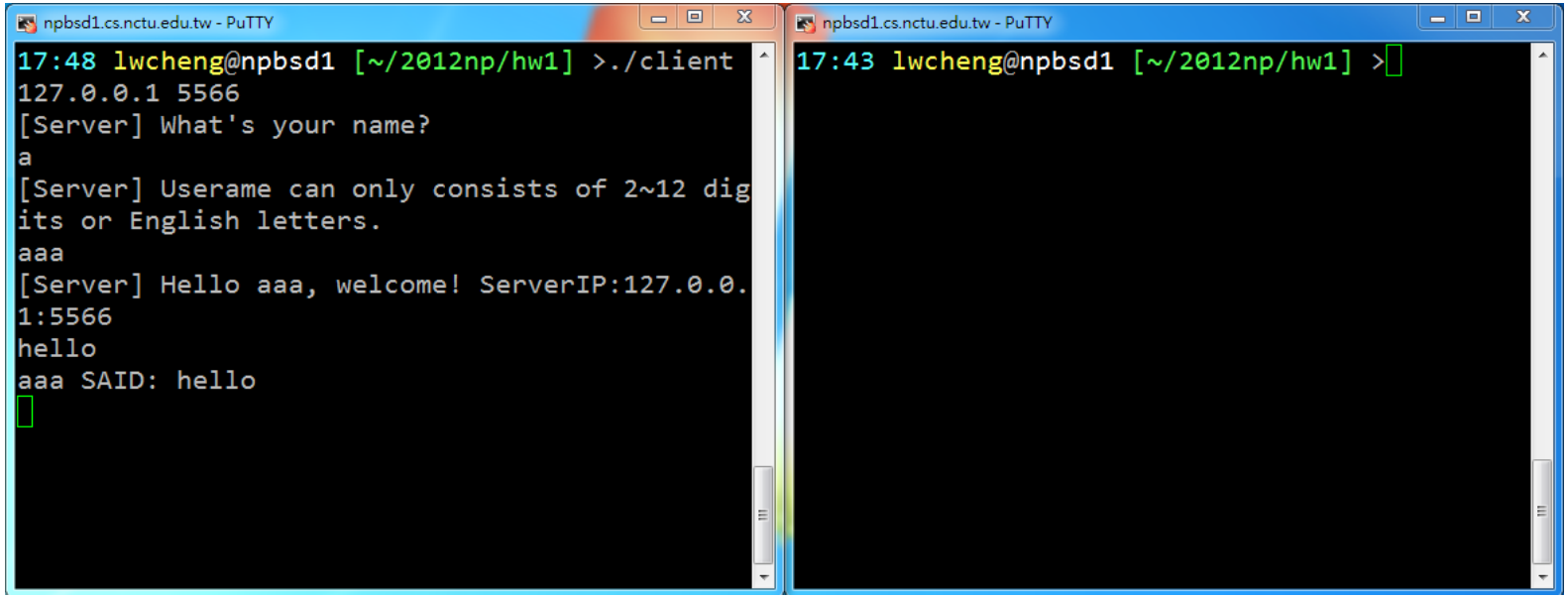
Demo flow in 2012/11/2

- The left client sends “aaa” as its username.
- The server sends welcome message to the left client because “aaa” is a valid username.
- Now the left client is known as aaa.



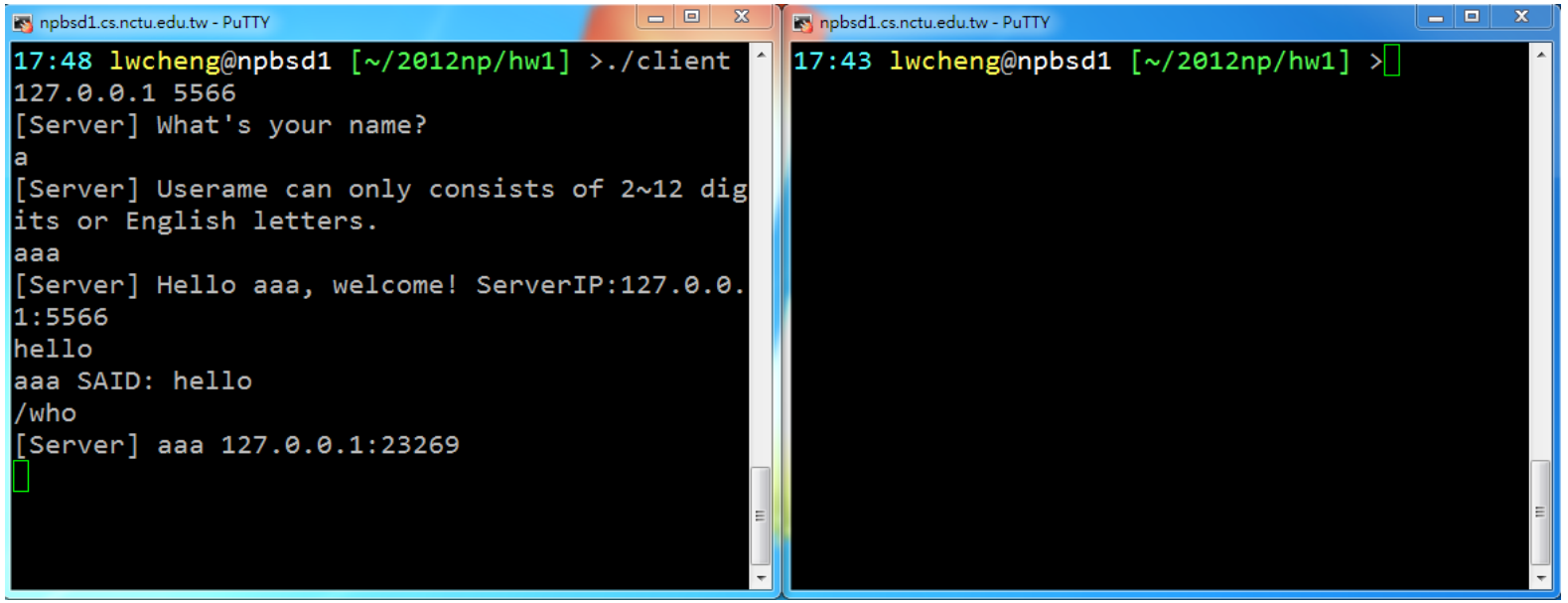
Demo flow in 2012/11/2

- aaa sends “hello” to the server.
- The server knows this is a chat message, and so broadcasts the chat message with proper format.



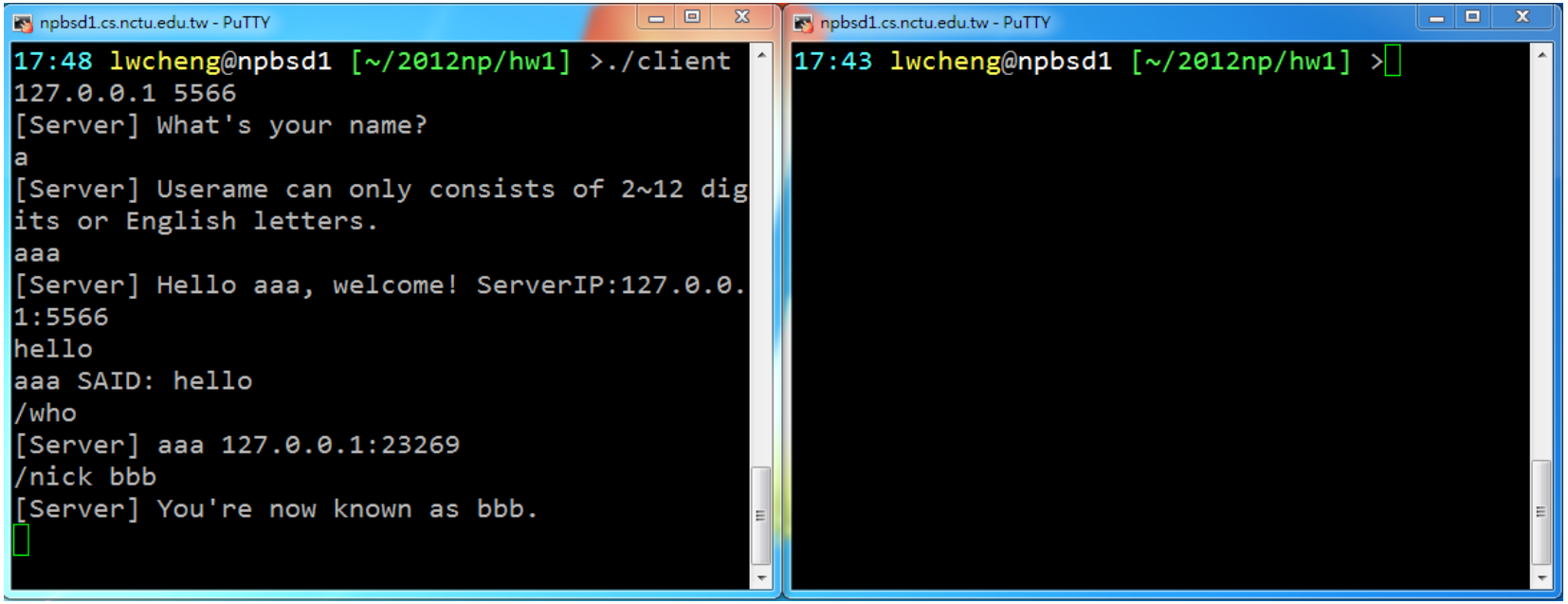
Demo flow in 2012/11/2

- aaa sends “/who” to the server.
- The server recognizes it as a who message, and then sends user list to aaa (only aaa online now).



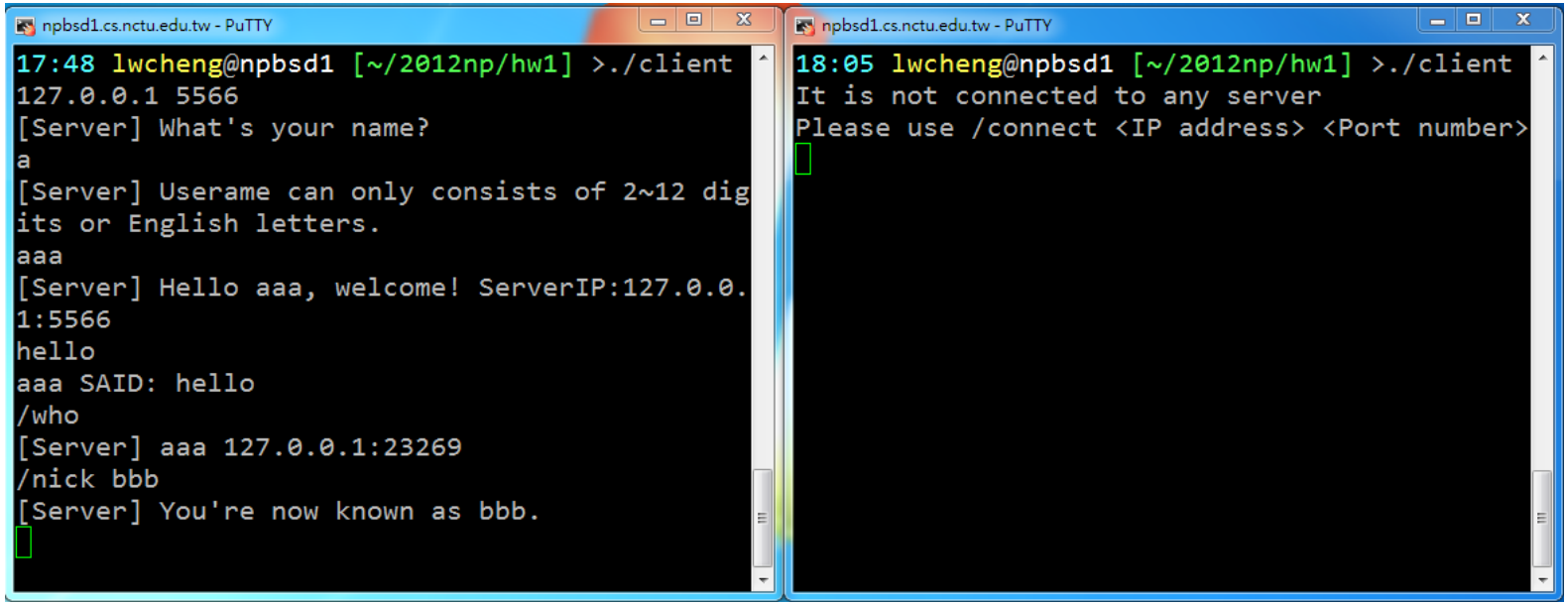
Demo flow in 2012/11/2

- aaa sends “/nick bbb” to the server.
- Since it is a nick message and bbb is a valid username, the server renames aaa as bbb.
- Now the left client is known as bbb.



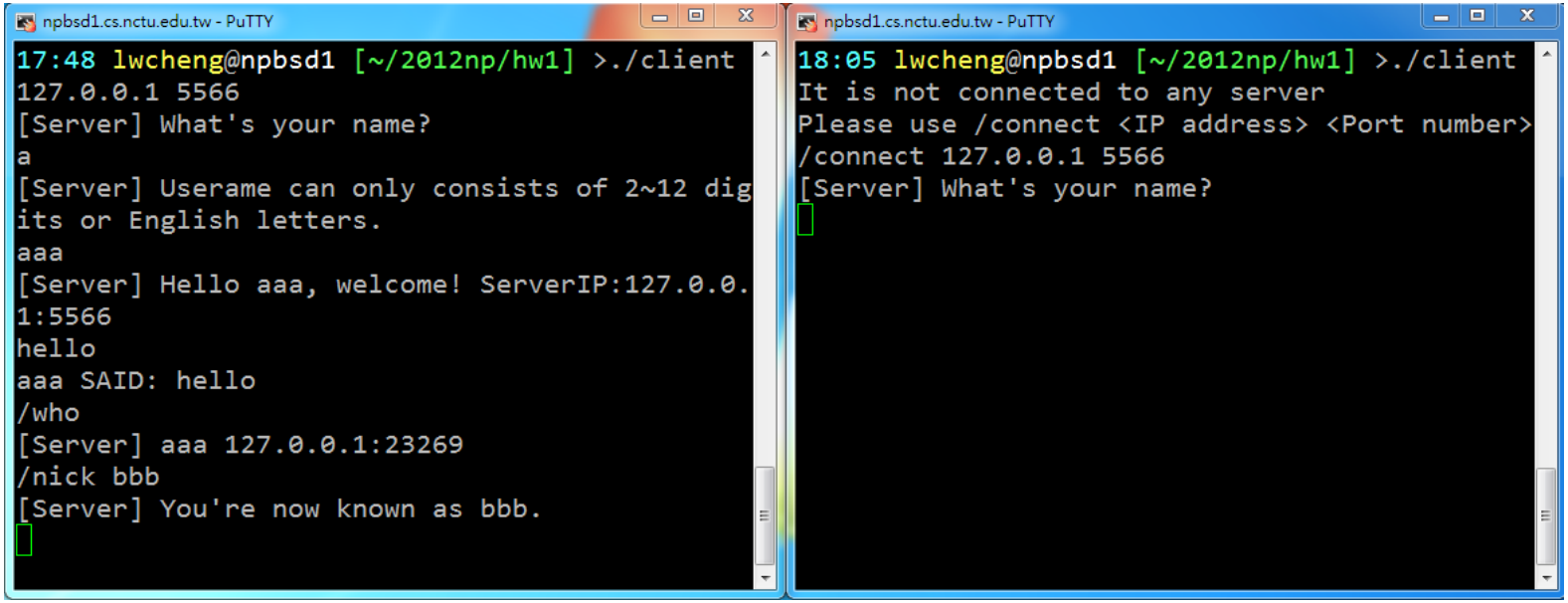
Demo flow in 2012/11/2

- Execute another client with no argument (in the right terminal).
- Since there is no IP and port number, it does not connect to a server but wait for user's input.



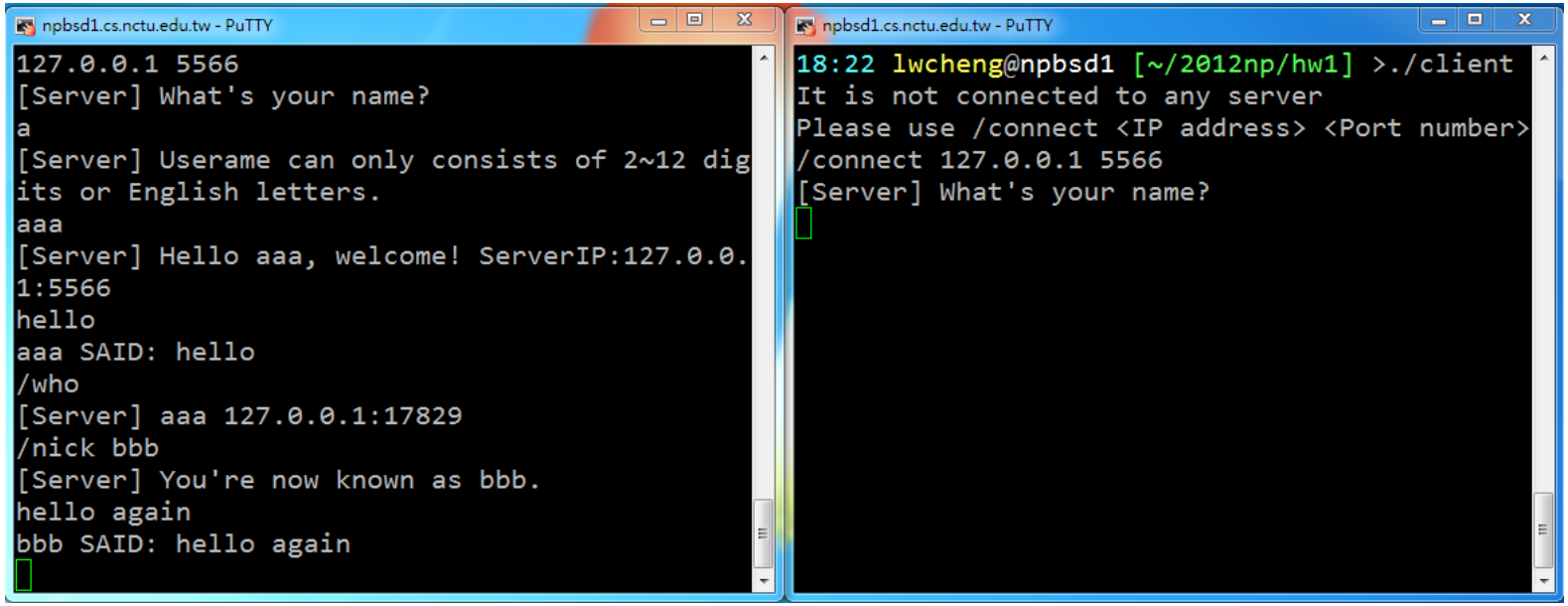
Demo flow in 2012/11/2

- The right client uses “/connect 127.0.0.1 5566” to connect to the server.
- The server accepts the connection and asks for a username.



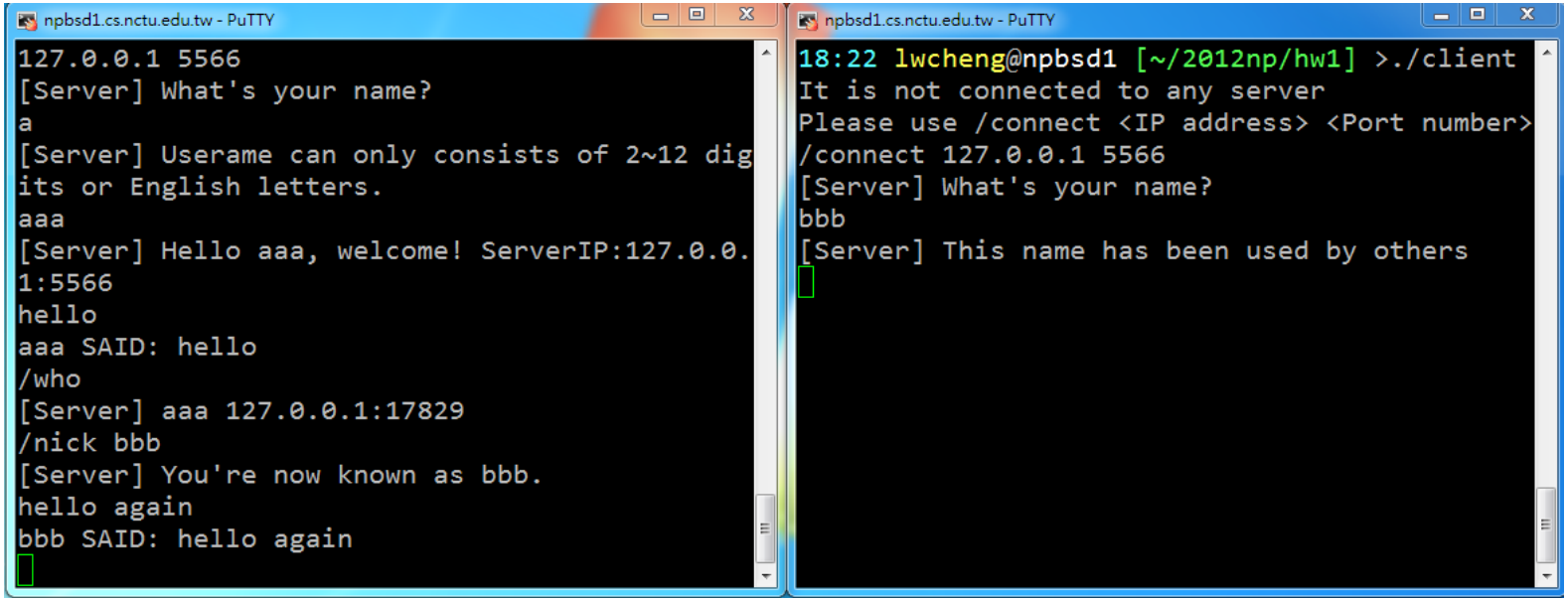
Demo flow in 2012/11/2

- bbb sends “hello again” to the server and the server broadcasts it.
- The right client would not receive the message, because is an invalid user now.



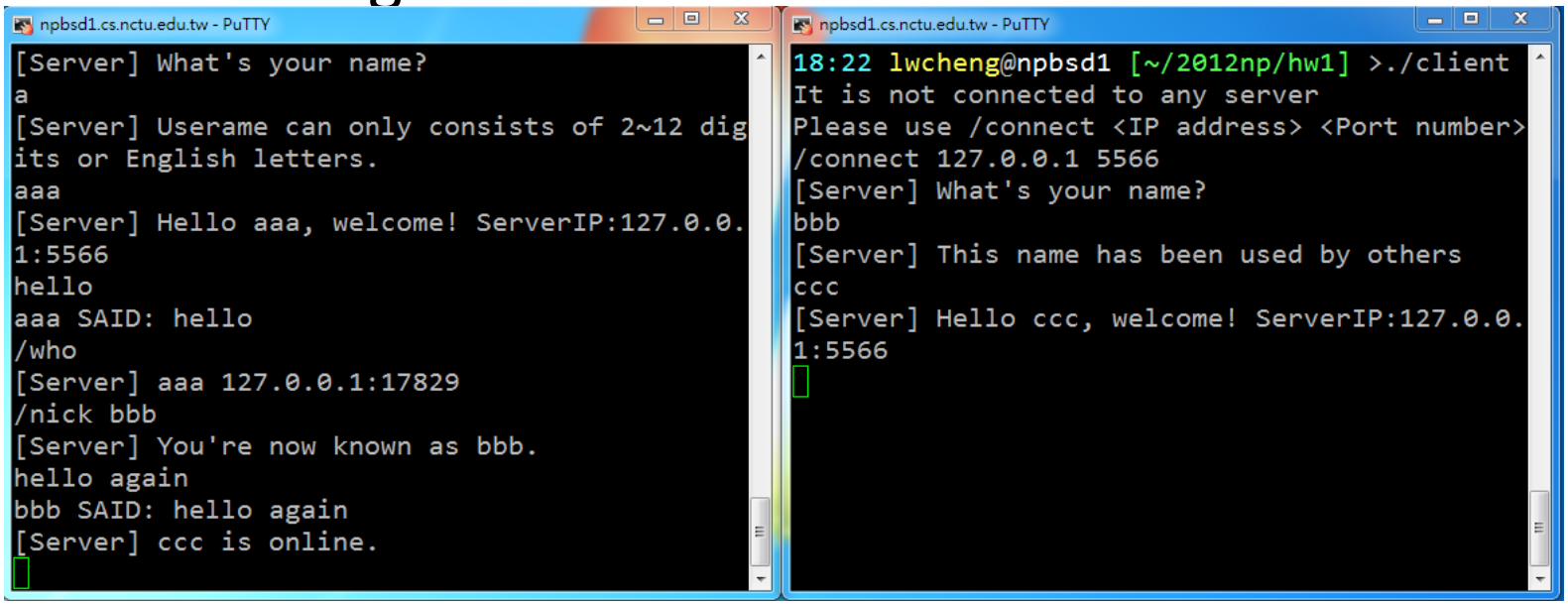
Demo flow in 2012/11/2

- The right client sends “bbb” to the server.
- The server refuses the username (bbb) because there is already one user using it (the left client).



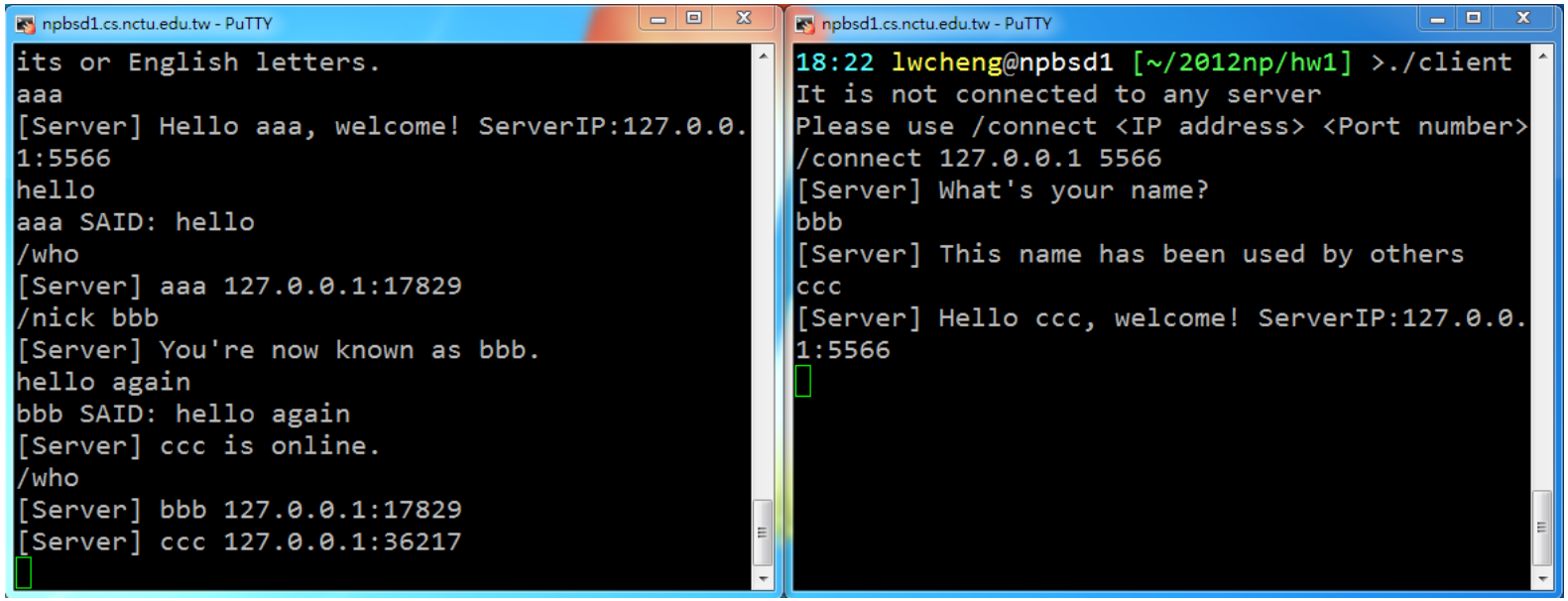
Demo flow in 2012/11/2

- The right client sends “ccc” to the server.
- The server sends welcome message to the right client because “ccc” is a valid username, and also sends online message to all other valid users (bbb).
- Now the right client is known as bbb.



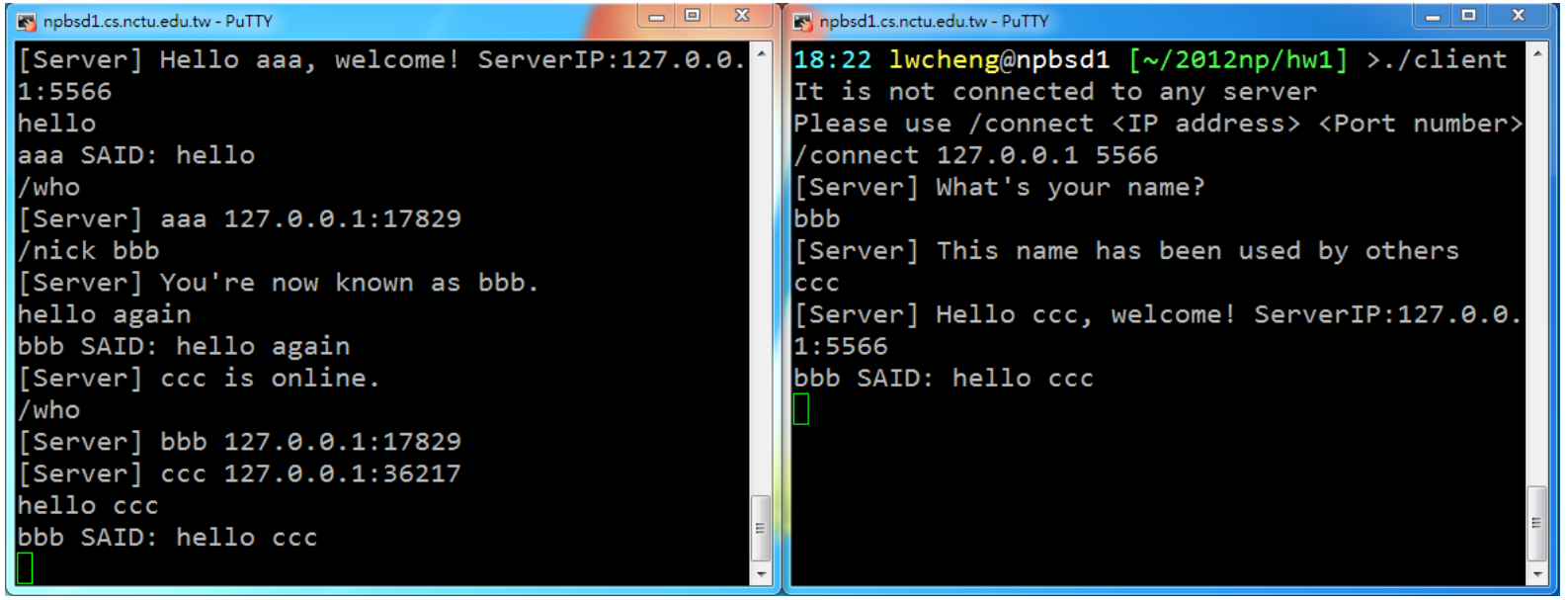
Demo flow in 2012/11/2

- bbb sends “/who” to the server.
- The list sent by the server includes two users (bbb & ccc) because there exists two valid users.



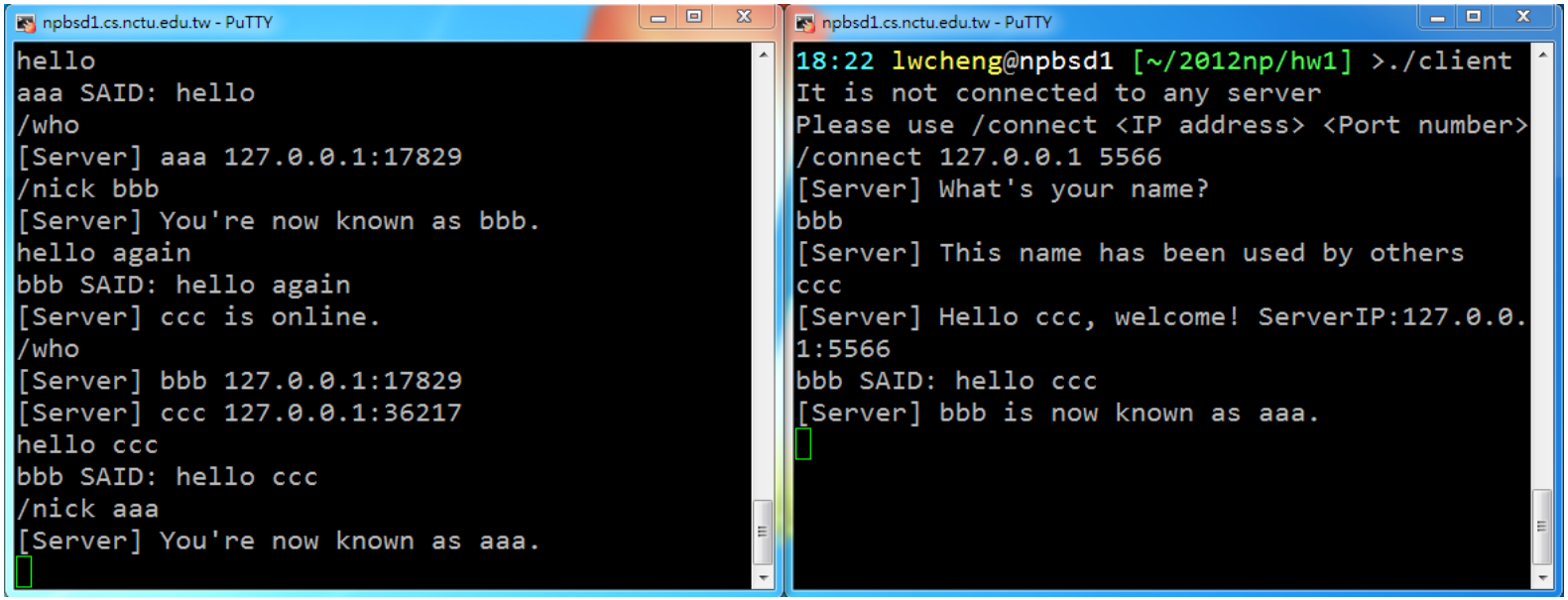
Demo flow in 2012/11/2

- bbb sends ”hello ccc” to the server.
- The server broadcasts the chat message to all valid users (bbb & ccc can see that).



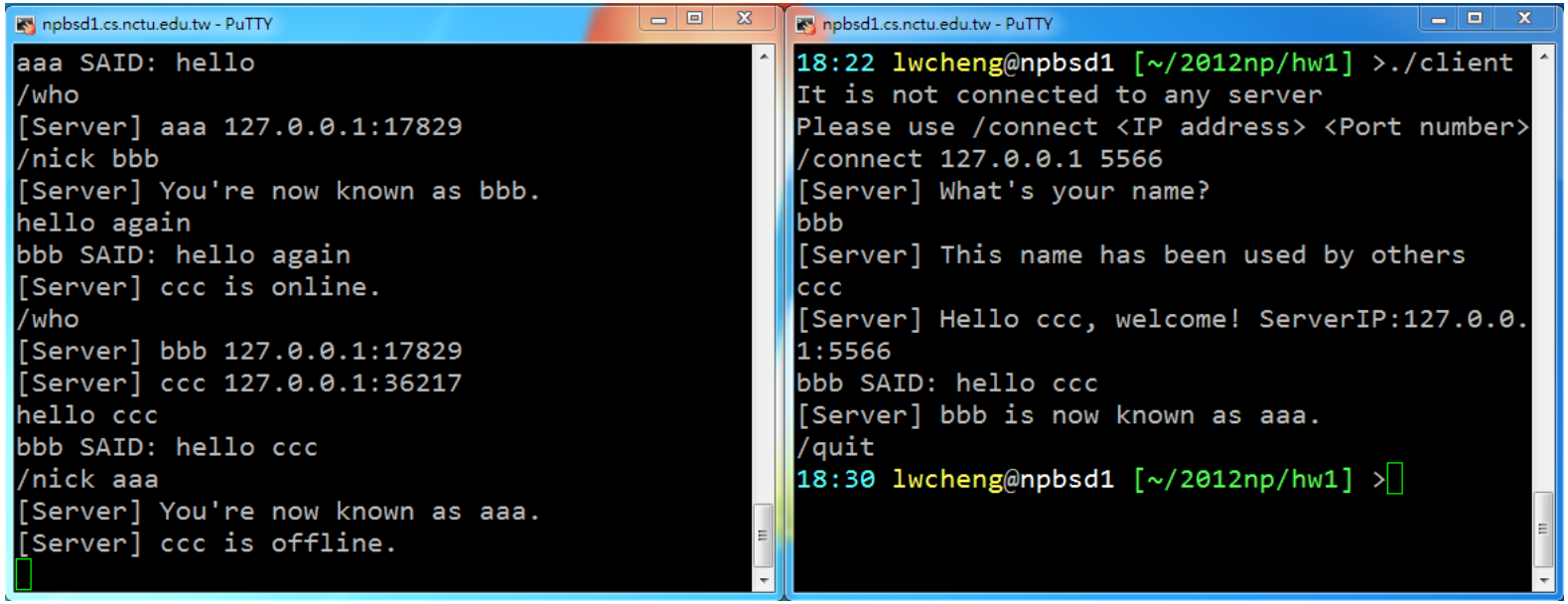
Demo flow in 2012/11/2

- bbb sends ”/nick aaa” to the server and then is renamed as bbb.
- bbb is notified that it is known as aaa.
- Other valid users are notified bbb is known as aaa.



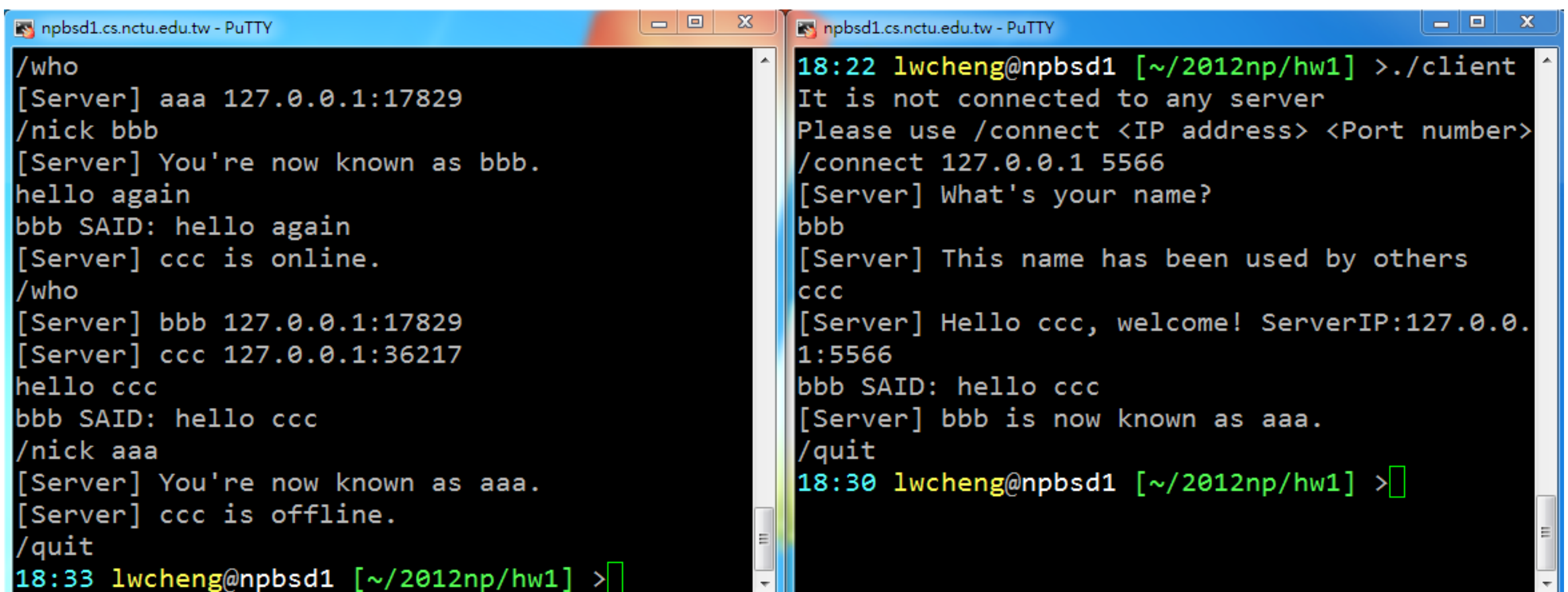
Demo flow in 2012/11/2

- ccc uses command “/quit” to end the process.
- While server realizes ccc quits, it sends leaving message to all valid users.



Demo flow in 2012/11/2

- bbb uses “/quit” to leave.



Requirement

- All messages transmitted between a server and a client should end with a newline (“\n”).
- If a message sent by a server is a chat message, it should start with “/msg ”, else start with “/serv ”.
- A user with a username is a valid user, or it is an invalid user.
- A server should not send online/leaving messages and chat messages to invalid users.
- Server/Client should NOT crash or be hanged.

Hint

- You can use select() for constructing the concurrent Server/Client program.

IMPORTANT

- Due date is 2012/11/21 23:59
- Tar your source code and upload it.
- Hand-in format: <Student ID>_<version>.tar
- Score will decrease 25% each day after due until the code being uploaded.
  Score = Original Score * (75%) ^ (Days of delay)
- DO run your Server/Client on nplinux0~4 or npbsd0~4.
- If you have any question, you can post it on the course forum (http://nsl22.cs.nctu.edu.tw/).