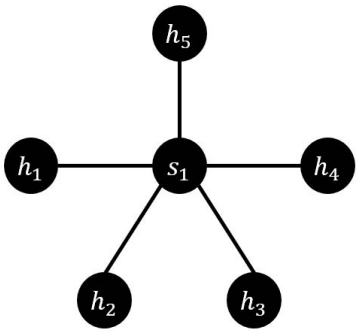
CS3611 Computer Networks (Spring 2023) Lab 3: Socket Programming

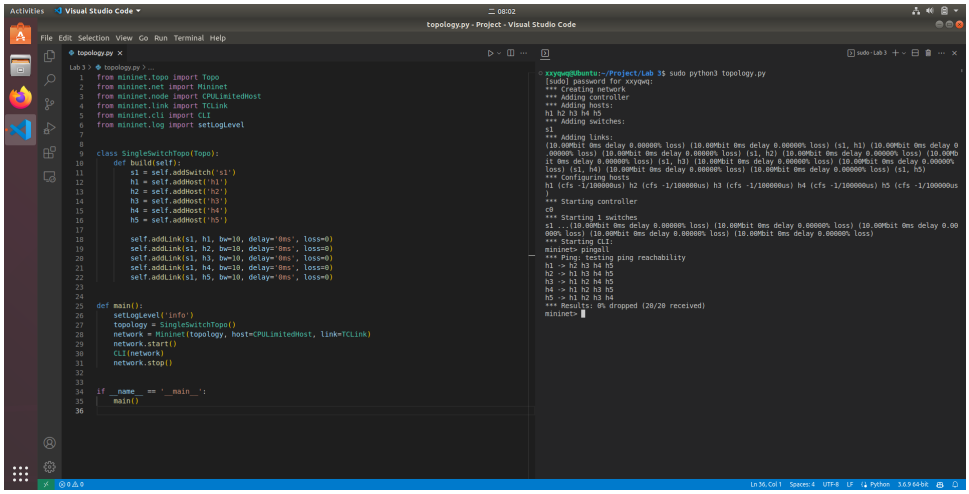
Xiangyuan Xue (521030910387)

1 Topology Construction

Build a topology with 5 hosts and 1 switch as shown in the following figure.



Define the topology in Mininet and start command line interface.



Use `pingall` command to test the connectivity of the topology. We can see that all the hosts are correctly connected and the topology is successfully constructed. Correspondence of hostname, IP address and port number is shown in the following table.

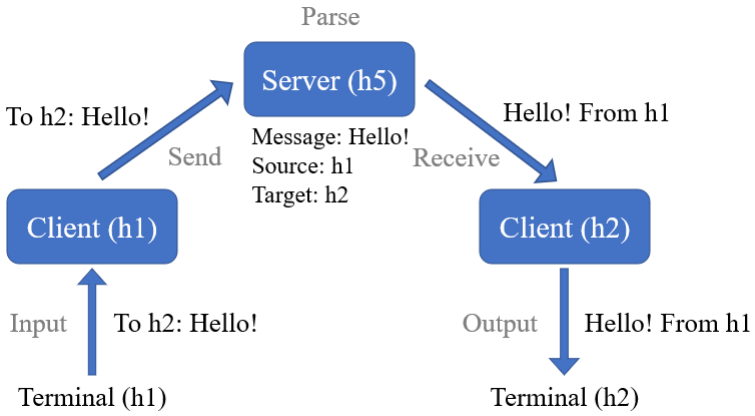
Host	Address	Port
h1	10.0.0.1	3366
h2	10.0.0.2	3366
h3	10.0.0.3	3366
h4	10.0.0.4	3366
h5	10.0.0.5	3366

2 Client-Server Model with TCP Protocol

To illustrate the implementation of the client-server model with TCP protocol, we describe the workflow as follows.

- Sender host accepts the message from the terminal and sends the message to server.
- Server receives and parses the message, and then sends the message to receiver host.
- Receiver host receives the message and prints it to the terminal.

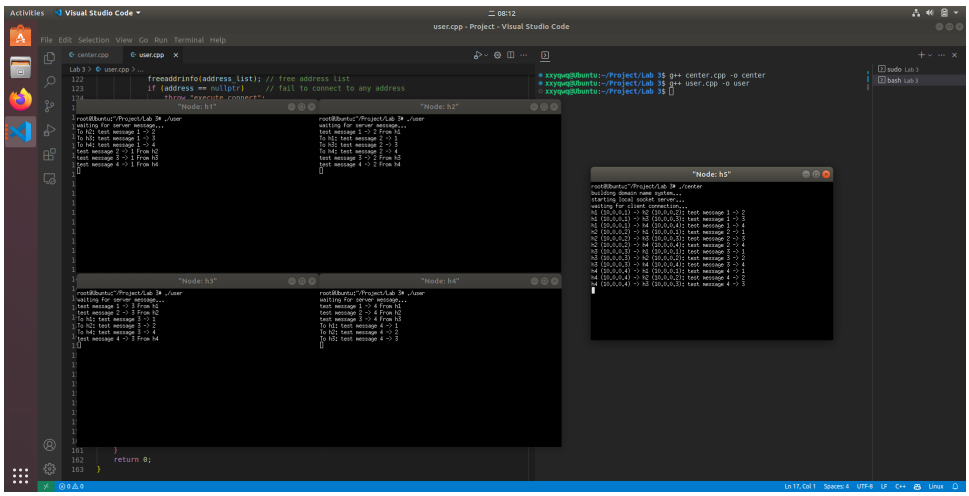
The following figure explicitly shows the workflow described above.



The server program consists of two procedures: `setup` and `execute`. The `setup` procedure builds domain name system and starts local socket server. The `execute` procedure is a minimal loop which keeps receiving message from the source host, parsing the message and sending the message to the target host.

The client program consists of two procedures: `setup` and `execute`, which is similar to the server program. The `setup` procedure starts local socket client. The `execute` procedure applies the select model, which is a non-blocking I/O model, to maintain multiple connections. The procedure keeps checking the terminal input and the socket connection and sends or receives message if required.

Compile `center.cpp` and `user.cpp`. Run `center` on h5 and `user` on h1, h2, h3, h4. We send $A_4^2 = 12$ messages to test the correctness of communication between hosts.



We can see that all the messages are correctly delivered from the source host to the target host. Logs are printed on the server terminal.

3 Client-Only Model with UDP Protocol

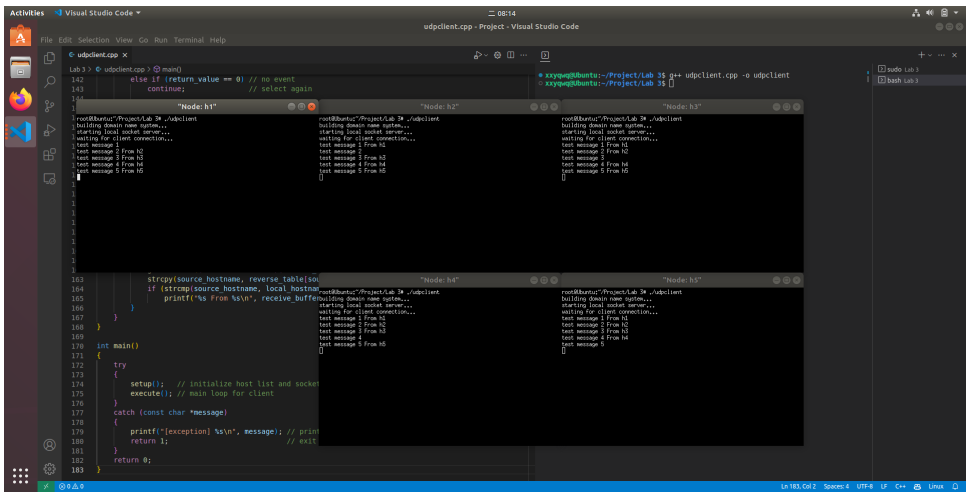
There is no server in the client-only model, so both the server function and the client function should be integrated into a single program. We describe the workflow as follows.

- Sender host accepts the message from the terminal and sends the message to broadcast address.
- Receiver host receives the message from broadcast address and prints it to the terminal.

The program consists of two procedures: `setup` and `execute`. The `setup` procedure builds domain name system and starts local socket server. The `execute` procedure applies the select model to maintain multiple connections. The procedure keeps checking the terminal input and the socket connection and sends or receives message if required.

Notice that the source host will receive the messages sent by itself. To filter out these messages, we fetch the local hostname and address in the `setup` procedure, which is implemented with `ifaddrs.h` by traversing the network interface and ruling out the loopback interface. When a message is received, we compare the source hostname with the local hostname. The message will be printed only if the source hostname is different from the local hostname.

Compile `udpclient.cpp`. Run `udpclient` on h1, h2, h3, h4 and h5. We send a message on each host to test the correctness of message broadcasting.



We can see that all the messages are correctly displayed on the terminal of each host and the messages sent by the hosts themselves are effectively filtered out, which indicates that the message broadcasting is correctly implemented.

References

[1] Bryant R E, David Richard O H, David Richard O H. Computer systems: a programmer's perspective[M]. Upper Saddle River: Prentice Hall, 2003.

[2] Beej's Guide to Network Programming. https://beej.us/guide/bgnet/.