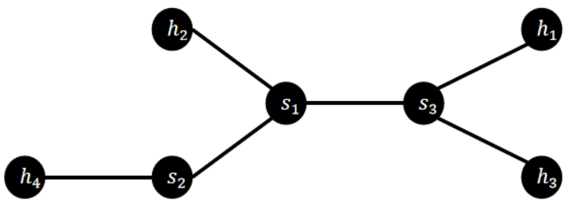
Lab 3: Socket Programming

This lab aims to help you get familiar with socket programming, which is a commonly used tool for messaging and file transmission in internet. In this lab, we will build a file transmission system which contains a sever and a client using TCP and UDP protocols respectively. You need to program in C++ language. You can refer to https://beej.us/guide/bgnet/ for more useful information.

1. File Transmission System implementation using TCP and UDP protocols (50 points)
   Requirements:
   a) Use Mininet to build the following topology, which contains 4 hosts. Set the link bandwidth for (s1,s2), (s1,s3) as 10Mbps and the loss rate as 0% . (You may reuse the codes in lab2.) Pick h_4 as the server and h_1 as the client.



   b) The workflows of this file transmission system are as follows: The client connects with the server and send the requested file name to the server. Then the server can transmit the requested file to the client if the file exists. (In Mininet, the command to open the terminal of host h1 is 'xterm h1'.)
   c) You are supposed to build the file transmission system using socket programming with TCP and UDP protocols respectively. You need to write four cpp files, namely, tcp_server.cpp, tcp_client.cpp, udp_server.cpp and udp_client.cpp. The cpp files should be complied by following commands.

```
g++ tcp_server.cpp –o tcp_server        g++ tcp_client.cpp –o tcp_client

g++ udp_server.cpp –o udp_server        g++ udp_client.cpp –o udp_client
```

   And you are supposed to run the executable files on sever terminal as

```
./tcp_sever    ./udp_sever
```

   And run the executable files on client terminal as

```
./tcp_client sever_IP_address file_name    ./udp_client sever_IP_address file_name
```

   Please obey the above instructions carefully because we will test your code with exactly the same procedures.

d) You are supposed to test the transmission condition with respect to different file size when transmitting files using TCP and UDP protocols, and put your screenshots and conclusions into your report. We provide 3 test files in Lab3_files.zip, you can download it from https://jbox.sjtu.edu.cn/l/010nOs

   i. Use TCP and UDP protocols to transmit files (test_1.png, test_2.png, test_3.png) from the server to the client. Record the transmission time of the above tasks for TCP and UDP protocols.

   ii. Set the packet loss rate of the link (s1,s2) as 5% and 30% respectively. Then retransmit the above three files using TCP and UDP protocols. Write your findings in the report.

2. UDP protocol does not provide reliable transmissions. Can we modify the UDP protocol to make it support reliable file transmissions? (50 points)
Hint: To improve UDP, you may borrow the idea of TCP protocol. That is, you may assign each UDP packet a serial number. The receiver collects the packets by the order of serial number. If the receiver successfully receives the packet, it will send back an ACK packet, carrying the serial number of that packet. After receiving the ACK packet, the sender can transmit the next packet with serial number adding one.
Requirements:
Please define a struct with a header including the serial number of UDP packet. Modify the transmission logic as described in hints to realize the reliable transmission of UDP protocol. You need to write two cpp files, namely, reliable_udp_server.cpp and reliable_udp_client.cpp. And the two files should be complied by following commands.

```
g++ reliable_udp_server.cpp –o reliable_udp_server
```

```
g++ reliable_udp_client.cpp –o reliable_udp_client
```

And you are supposed to run the executable files on sever terminal as

```
./reliable_udp_sever
```

And run the executable files on client terminal as

```
./reliable_udp_client sever_IP_address file_name
```

Verify your improvement of UDP enables reliable transmission by comparing with the original UDP protocols when transmitting test_3.png on links with high loss rate. Please obey the above instructions carefully because we will test your code with exactly the same procedures.

Your final submission .zip file should include a pdf report, tcp_server.cpp, tcp_client.cpp, udp_server.cpp, udp_client.cpp, reliable_udp_server.cpp, reliable_udp_client.cpp. Please put the screenshots of hosts' terminals in your report.