Introduction to Network Programming

2014 Fall Semester Final Exam

Closed Network, Open Textbook and Closed USB disk

(1) 30% (a) 15% You should use the setsockopt() system call with a timeout to set timeout for a UDP socket. Write a UDP server that endlessly does the following functions:

- ✓ Take an integer number sent from a UDP client (for example, 8). If no integer number arrives, then waits.
- ✓ If an integer number is received, then take another integer number sent from the same UDP client (for example, 14).
- ✓ The second integer number must arrive within 3 seconds after the first integer number is received. Otherwise, the first integer number is discarded and the state of the UDP server is reset to the initial state. (That is, so far no integer number has been received.)
- ✓ If the second integer number arrives in 3 seconds, then the UDP server returns the sum of the two received integer numbers to the UDP client and the state of the UDP server is reset to the initial state.

(b) 15% You should use pthread() system calls to write a two-thread UDP client that endlessly does the following functions:

- ✓ Take input from the keyboard, one integer number at a time, until the user hits EOF. When the EOF is entered, exit the UDP client program.
- ✓ For each input integer number, send it to the UDP server.
- ✓ For the above two operations, these operations should be handled by one thread. Note that after entering an integer number from the keyboard, the user may wait an arbitrary amount of time (e.g., 1 second, 4 seconds, etc.) before entering the next integer number.
- ✓ While sending integer numbers to the UDP server, an integer number may come back from the server at any time. The client should use another thread to immediately detect the arrival of an integer number, receive it, and then print it on the screen.

(2) 30% You are requested to write a version of the ping program that "pings" all hosts on a specified subnet. Let's call this new program ``pingsubnet''. Its usage is "pingsubnet subnet_IP". For example, ``pingsubnet 140.113.17'' will sequentially ping 140.113.17.1, 140.113.17.2, 140.113.17.3, ..., to 140.113.17.254 with an interval

of 1 second. When pinging a specific IP address, the format of the output line should be "IP address (Host name) RTT=XXX ms". For example, when pinging 140.113.17.71, the output line should look like "140.113.17.71 (shieyuan3.cs.nctu.edu.tw) RTT=5.254 ms." This means that you need to use the gethostbyaddr() function on the specified IP address to find the name of the host that uses this IP address. If there is no response from a pinged IP address for 3 seconds, then the output line should be "IP address (Host name) no response". For example, "140.113.17.71 (shieyuan3.cs.nctu.edu.tw) no response." The following is an example output of pingsubnet:

#pingsubnet 140.113.17
140.113.17.1 (csns1-17.cs.nctu.edu.tw) RTT=0.853ms
140.113.17.2 (csns2-17.cs.nctu.edu.tw) no response
....
140.113.17.254 (e3rtn-17.cs.nctu.edu.tw) RTT=1.763 ms

(3) 20% The readline() function reads a complete line of data from a socket. An efficient version of its implementation uses an internal buffer to avoid repeatedly reading just one byte from the socket and then check whether the read byte is a '\n'. Instead, each time when it would like to read data from the socket, it reads as many bytes as its internal buffer can accommodate in just one read() system call. Then, internally when the main function of readline() wants to read a single byte of data and check whether it is a "\n", the single byte is directly returned from the internal buffer without calling the read() system call. The code of this efficient implementation is listed in Figure 3.18 on page 92 of the textbook.

Now you are requested to develop a **writeline(int sockfd, char c) function, where . sockfd is the socket of an established TCP connection and c is the provided char.** Using this function, a sending application program can send out a char c to the remote receiving application program over an established TCP connection. However, if the char c is not a "\n", it is stored internally inside a buffer without being actually sent to the receiving application program. Each time when the sending application program calls the writeline(int sockfd, char c) function, the provided char c is inserted into and stored inside the internal buffer. When the provided char c is a "\n", the writeline() will send the completed line to the remote application program in just one write() system call and clear the content of the internal buffer for later reuse. In short, the writeline() is the opposite of the readline(). Both of them use an internal buffer

to greatly reduce the need to call read() or write() system call frequently. After developing writeline() function, you should write an application program that calls this function to test whether this function works correctly.

(4) 20% Use the thread-specific data technique to convert the above writeline() function to a thread-safe function. Let's name this thread-safe function writeline_r() to denote that it is reentrant and thread-safe. Write a multi-threaded server program in which each thread sends its data over its own established TCP connection to a different TCP client. Let these threads call this function to send their data to their clients simultaneously to test whether this function is really thread-safe.

Introduction to Network Programming：2014 Fall Semester Final Exam Specification

(1) Format:

./server <SERVER PORT>

./client <SERVER IP> <SERVER PORT>

Note:

For example, if the client first input “8”, then input another number “14”. And the waiting time “t” must be random from 1s to 5s.

The client should print the following information:

send 8

wait 1s

send 14

wait 3s

recv 22

(2) Format:

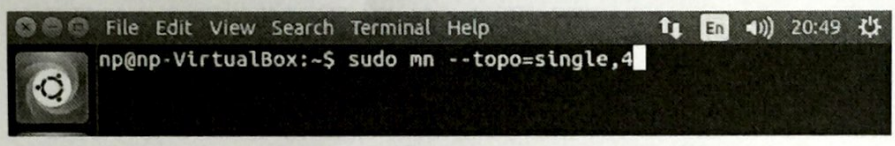
#pingsubnet<space>140.113.17

140.113.17.1<space>(csns1-17.cs.nctu.edu.tw) <space>RTT=0.853ms

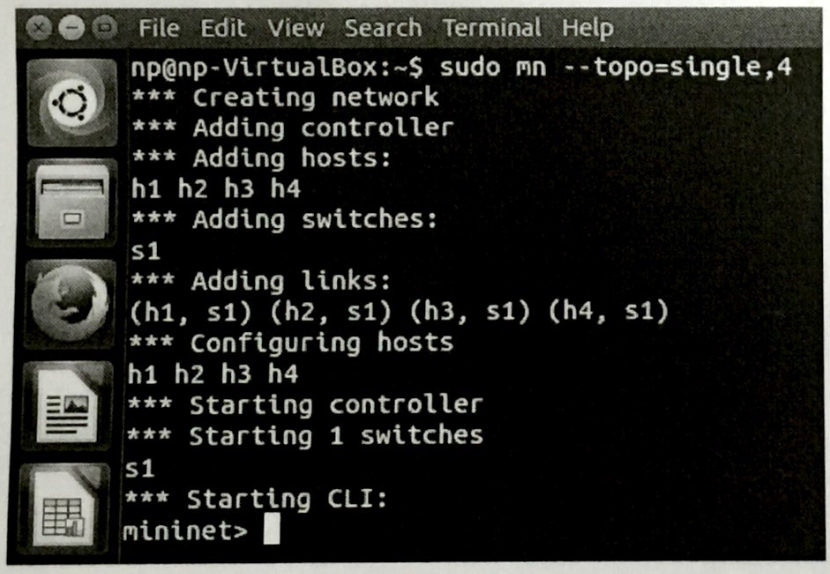
140.113.17.2<space>(csns2-17.cs.nctu.edu.tw) <space>no<space>response

You can use Mininet to emulate a small network. Shows below:

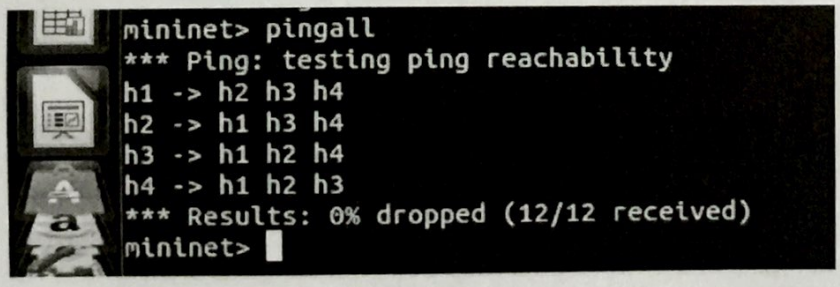
I. Start Mininet



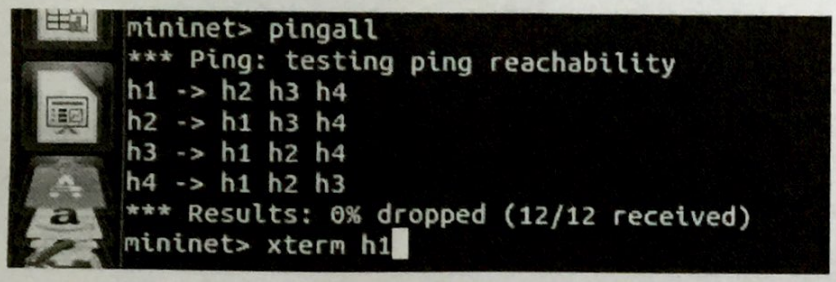
II. If Mininet start successfully, you will see like this.



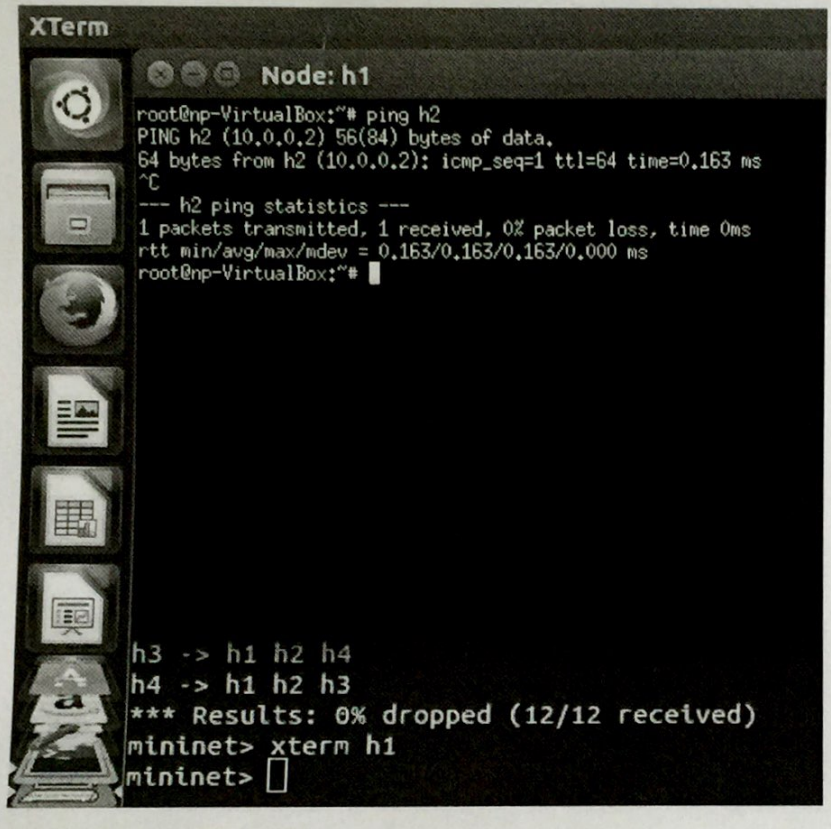
III. You can use “pingall” to check every connections between hosts.



IV. You can use "xterm h1" to start the terminal of h1.



V. Finally, you can excute your program in the terminal.



(3) Format:

Only upload the writeline.h

I. Put your void writeline(int sockfd, char c) function in the "writeline.h".

II. The output string of your writeline(int sockfd, char c) should contain "\n".

III. The line string length < 1024.

(4) Format:

writeline_r.h

I. Put your void writeline_r() function in the "writeline_r.h".

II. The line string length < 1024.

The arguments in writeline() function can be

writeline_r(int sockfd, char c) or

writeline_r(int sockfd, char c , ...).