ELE 420/520 Cyber-Physical System Security

Project 1 **Due: Feb 13th, 11:59 pm EST**

Socket Programming

1 Project Description

In this project, you will use a network (already implemented in Mininet), whose topology is shown in Figure 1. In this network, we will include three host machines, i.e., "Server," "Forward," and "Client." The information flows in the following steps:

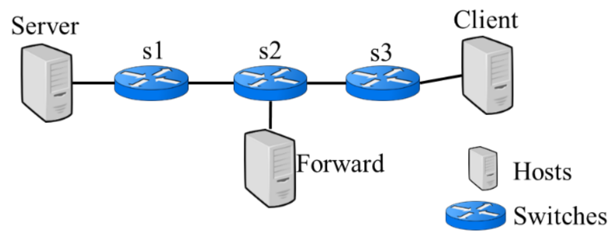
[1] The *Client* machine can only send a request including the content "hello from client" in the application layer payload to the *Forward* machine (please refer to our socket programming tutorials on how to use *bytes* object in Python).

[2] The *Forward* machine will forward the request from the *Client* machine to the *Server* machine. However, it will append the original message in the request, e.g., "hello from client", with " from forward." In other words, the request sent from the *Forward* machine to the *Server* machine includes the content "hello from client from forward" in the application layer payload.

[3] When the *Server* machine receives the request from the *Forward* machine, it replies directly by adding the content in the request with bytes " from server". Note that the *Server* machine only responds to the *Forward* machine.

[4] When the *Forward* machine receives the response from the *Server* machine, it forwards the responses to the *Client* machine by adding the content in the response with bytes "from forward".

[5] Ultimately, you will expect that the *Client* machine receives a response carrying the application layer payload with contents like "hello from client from forward from server from forward". The communication terminates.



Host	***IP Address***	***Port Number***
Server	10.0.0.1	65432
Forward	10.0.0.2	65432
Client	10.0.0.3	NA

Figure 1. Network topology used in Project 1.

2 Implementation

If you have not already done that, you can obtain the starter code (including tutorials) related to this course at https://github.com/hugolin615/ELE420520Spring2026/tree/main/project1. You should be able to download these codes in any directory of your Raspberry PI machine; I recommend directly in your home directory. Before doing the assignment, please refer to our Mininet tutorial or Mininet walkthrough on its official website to make sure that you have basic understanding of running mininet. Also some sample codes of socket programming is provided at https://github.com/hugolin615/ELE420520Spring2026/tree/main/socket_program.

You should implement in the following files under the directory *project1*; the locations where you should input your codes are marked with #TODO.

- ***client.py***: refer to *simple_client.py* in our tutorial to implement the Client machine
- ***server.py***: refer to *simple_server.py* in our tutorial to implement the Server machine
- ***forward.py***: this is a challenging part, but not that challenging. The Forward machine should include a "server" socket to receive requests from the Client machine and a "client" socket to make a connection to the Server machine so that the Forward machine can forward messages between the server and the client.
- ***build_net.py***: no implementation is needed in this script. This is the script to build a communication network in Mininet based on the topology shown in Figure 1.

To simplify the task, you should run your codes in the following order.

[1] Entering Mininet CLI console by running your *build_net.py* (running commands "sudo python ./build_net.py" in the shell console). This will open four Xterm consoles, one corresponding to the *Server*, two corresponding to the *Forward*, and one corresponding to the *Client* machine.

[2] Execute the *server.py* in the "Server" console.

[3] Execute the *forward.py* in the "Forward" console. If your implementation is correct, print out messages will show that the *Forward* machine is connected to the *Server* machine.

[4] Execute the *client.py* in the "Client" console. If your implementation is correct, print out messages will show that the *Client* machine is connected to the *Forward* machine. Follow up the hint from the print out message to continue.

[5] To finish, type "exit" in the Mininet CLI console.

(**Other tips**). Please test all your scripts by using *Python3*. In my tutorial, I link the shortcut command "*python*" to my *python3* executable.

3 Turn-in and Grading.

Please submit a single PDF file, named as "*Last Name_First Name_Project 1.pdf*", including the following contents:

- **Python scripts (*client.py*, *forward.py*, *server.py*) including your implementation (50%).**
- **A snapshot of running the Python scripts in Mininet (25%).** After running the scripts, take a snapshot to show the consoles of all three machines. Please place the three Xterm in the desktop properly so that I can see the outputs clearly. Please name this snapshot as *running_mininet.jpg (or running_mininet.png)*
- **Also submit a sperate Network trace file recorded by Wireshark (25%).** In Mininet, in the Forward Xterm that is not running any script, you start Wireshark in that console. Using Wireshark to monitor the network traffic that will go through the "Forward" machine, while you run your Python scripts again (following the steps [1] through [5]). Save what you recorded and that is what we usually referred as a network trace file (in the format of .pcapng or .pcap). Please name this trace file as *network_trace.pcap* or *network_trace.pcapng*.