Programming Assignment 1

CST 311, Introduction to Computer Networks

PLEASE READ THE ASSIGNMENT DESCRIPTION CAREFULLY BEFORE YOU START.

The assignment must be submitted electronically to Canvas.

In this assignment, you will learn the basics of socket programming for UDP and TCP in Python. You will learn how to send and receive datagram packets using UDP sockets and TCP sockets. Throughout this part, you will gain familiarity with the client-server architecture. The assignment is a warm-up to help you with the setup, and you will develop and write the client side of a client-server application in the next assignment.

Instructions

This assignment requires you to execute the Python client and server code for a UDP socket programming example and also to read and understand a TCP example. The goals of this part are 1) to learn about how UDP and TCP work and how to program with UDP and TCP and 2) to make sure your computer is set up and working, ready for the next assignment where you will be writing your own code.

The link below is a tutorial that shows three different iterations of building a socket server and client with Python:

Python Socket Programming Tutorial

The link provides templates for TCP socket programs for both the client and server and instructions on how to run them. You should read through, at the least, the section on how to run the scripts. Anything beyond that is beneficial for your understanding, but not necessary for this assignment. **Try to understand what each line of the TCP versions of the scripts is doing.**

WHAT TO DO:

Step 1 - Download this zip file, extract and retrieve the `udp_client.py` and the `udp_server.py` files.

Step 2 - Modify `udp_client.py` to request input from the user for a string, and send this user-provided string to the server instead of the predefined string found in the template. **The program should keep asking for the user input until the user provides** `"done"`.

Step 3 - Modify `udp_server.py` to resend the received string in ALL CAPS.

You will be using your PC to run both the client and server code in the Mininet virtual machine in a virtualized network.

Step 4 - Follow the instructions from Lab 1 to set up a default network in the Mininet VM with two hosts `h1` and `h2`.

Step 5 - We will run the server on host `h1` and the client on host `h2`. You first need to change the HOST variable in `udp_server.py` to the IP address of the host `h1`, then also change the HOST variable in `udp_client.py` to the IP address of the host `h1` since the client on host `h2` will communicate with the server on host `h1`. To get the IP address of host `h1`, run the `ifconfig` command as we did in Lab 1. To run a command on a host in Mininet, use the following command format:

```
<host_name> <command>
```

, where `<host_name>` is the name of the host that we want to run the command on, and `<command>` is the command that we want to run. For example, to get the IP address info on host h2, we can perform the following command on Mininet:

```
mininet>h2 ifconfig
```

You also need to specify the port number for the communication. Port numbers from `1024` to `49151` are the User Ports and are the ones to use for your own protocols.

Then copy `udp_server.py` to `h1` and `udp_client.py` to `h2`.

The following note linked below may help you with this step:

How to Copy a File from Your PC to the Mininet VM

Step 6 - Run the copied programs on their corresponding hosts. You first need to run the server program on `h1` first, then run the client on `h2`. Please refer to the sample run in the “What to Hand in” section below.
Use the following test cases below to produce the expected outputs:

Input:	**Output**:
`csumb`	`CSUMB`
`cst`	`CST`
`311`	`311`
`is pRettY FuN !!!`	`IS PRETTY FUN !!!`

What to Hand in

You will need to hand in the following:

1. The modified `udp_client.py` and the `udp_server.py` files. Please make sure that you submit the final versions of the files running on the Mininet hosts.
2. Submit screenshots of the client/server programs running on two different hosts in Mininet **for all four (4) test cases given above**, for example:

```
mininet-vm% sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> h1 python3 udp_server.py &
mininet> h2 python3 udp_client.py
Input sentence: cst
Received: CST
Input sentence: hellO
Received: HELLO
...
```

Note that the `&` character makes the command run in the background. Please make sure you run with `python3` command instead of `python` for the correct version of Python. When running with the test cases mentioned in Step 6 of the previous section, please perform all test cases in one connection.

Please include the screenshot(s) in a PDF file and submit this file. Do not compress the required files to a Zip file.

Your submission should include the following files:

- `udp_client.py`
- `udp_server.py`
- The PDF with the screenshot(s).

Grading Rubric

Task	**Point**
Setting up the server correctly.	15
Setting up the client correctly.	15
Correctly implementing the server program.	15
Correctly implementing the client program.	15
Generating correct outputs for the test cases.	40
Total	**100**