ECS 152A: Computer Networks

Winter 2025

Project 1

(100 points)

Due Date: Friday February 21st, 5:00pm PT

Team: The project is to be done in a team of at most 2 students. You *cannot* discuss your code/data with other classmates (*except* your project partner)

All submissions (including your code) will be checked for ***plagiarism*** against other submissions as well as the public Internet. *This includes submissions from previous quarters*. Plagiarized submissions will be entitled to ***zero*** points and may result in the students involved failing the class.

Don't cheat. It's not worth it.

Project 1 consists of two parts:

1. Implementing Iperf Client
2. Implementing a proxy server

You may find it useful to look at chapter 1.4 in the textbook. It's relatively short and contains relevant, useful information on socket APIs and client/server models.

Both the bare bones description of the project, and the lack of starter code, are intentional. The initial pain in getting started and connecting the project with the course material can feel a bit jarring, but it will force you to define the implementation requirements, which is a central skill in computer networking. Even today, when the core protocols of the Internet have been around for decades, this skill is essential for application developers who need to use the network (and what application does not?).

Part 1: Implementing iPerf(*50 points*)

For this part, you will build a UDP server and client (both hosted on localhost) using the socket API in Python. You will send 25-200 megabytes of data from the client to the server. Your code should take the number of megabytes to send as an input, then generate a payload of that size. The server should measure the throughput (amount of data received / time taken to receive them) and send it back to the client. The client should then print the throughput value received from the server. You will report the computed throughput in kilobytes per second.

You will first implement this yourself and then get assistance from ChatGPT. In your submission, include your original implementation, a link to your chat session with ChatGPT, and your implementation after interacting with ChatGPT. Note that you may have to tweak the implementation suggested by ChatGPT. You will include your final implementation after making the required tweaks.

In addition to the throughput, your code should print the following for each run:

- The data sent from the client and its timestamp.
- The timestamp when the data is received on the server side.
- Both sent and received data.
- The IP addresses of the client and server.

Codefilename:
udp_server[name1]_[student_id1]_[name2]_[student_id2].py
udp_client[name1]_[student_id1]_[name2]_[student_id2].py

Report:proj1_1_[name1]_[student_id1]_[name2]_[student_id2].pdf

Remember to include the names of all programs you submit in your report.

Part2: Proxyserver (50 *points*)

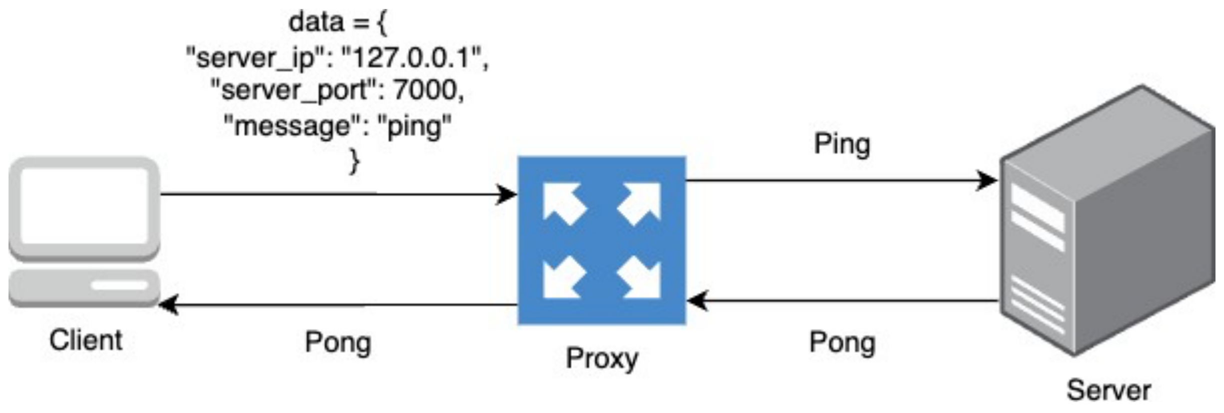
You have to implement ping-pong client servers using **TCP Sockets.** But instead of having the client talk to the server directly, you need to implement a proxy server that forwards requests from the client to the server. Your code should accept a 4-character string as an input to be forwarded (i.e., not just "ping" and "pong" but any 4-letter string).

The client should send data to the proxy server in the following JSON format:

```
data = {
"server_ip": "127.0.0.1",  #  The server's IP (destination) "server_port": 7000,  #  The
server's port (destination) "message": "ping"    #  The actual message
        }
```

The proxy server should extract the server's IP from the message and forward the message to the server. Once the server responds, the proxy forwards the message back to the client.

The proxy server also implements an IP blocklist consisting of several IP addresses. Whenever it finds that the server IP is in the blocklist, it does not forward the request and replies with an "Error" message instead.



You need to implement the client, proxy, and the server.

In addition to the required functionality above, your code should print the following for each run:

- Messages sent and received (e.g., "ping" or "pong").
- The JSON data being transmitted.

Code file name:
proxy_server[name1]_[student_id1]_[name2]_[student_id2].py
client[name1]_[student_id1]_[name2]_[student_id2].py

server[name1]_[student_id1]_[name2]_[student_id2].py

Report: proj1_2_[name1]_[student_id1]_[name2]_[student_id2].pdf

Testing Environment:

All submissions will be tested on Python 3+.

Late Submission Policy:

No late submissions are allowed. However, if you barely miss the deadline, you can get partial points up to 24 hours. The percentage of points you will lose is given by the equation below. This will give you partial points up to 24 hours after the due date and penalize you less if you narrowly miss the deadline.

$$\text{Total Marks you get} = (\text{Actual Marks you would get if NOT late}) \times \left[1 - \frac{\text{hours late}}{24}\right]$$

Late Submissions (later than 24 hours from the due date) will result in zero points *unless you have our prior permission or documented accommodation*.

——————————*Best of luck*——————————

Submission Page

Include this signed page with your submission

I certify that all submitted work is my own work. I have completed all of the assignments on my own without assistance from others except as indicated by appropriate citation. I have read and understand the university policy on plagiarism and academic dishonesty. I further understand that official sanctions will be imposed if there is any evidence of academic dishonesty in this work. I certify that the above statements are true.

Team Member 1:

Jinsi Guo	Jinsi Guo	
Full Name (Printed)	Signature	

Date

Team Member 2:

Victoria Lam	Victoria Lam	
Full Name (Printed)	Signature	Date