Application Programming Assignment (Evaluation)

CSCI 396: Computer Networks

Spring 2024

This assignment is designed to give you programming experience with application layer code. Feel free to collaborate with others and use resources, but all code and writeups must be your own.

Due Date: Friday, February 23rd at 12:00 pm.

For this assignment, you will be provided starter code on Moodle. The starter code contains a server and a client for both the TCP and UDP transport layer protocols. The code is there to help you; you are welcome to use it or start from scratch as you prefer, and your grade will not be impacted by your decision.

1. (30 points) **Ping.** In this problem, you will design your own (much simpler) version of the traceroute program. You should use UDP as your transport layer protocol so that you can explicitly see when packets are dropped. Your program will use the client server model. The client will send a packet of data to a remote host. The server on that host will return the data back to the client (this is known as echoing).
   (a) (5 points) **Preparing your testing setup.** You will need to have one machine act as the server, and another machine act as the client. You are encouraged to ask a classmate to act as one of the machines for testing. Ensure that you are able to get a basic echo server (like the provided code) working. As part of this testing process, you should use Wireshark to catch your messages. Provide the Wireshark capture of both client and server messages as part of your submission.
   (b) (10 points) **Simulating errors.** In order to fully test our ping program, we will need to have packet loss. However, Reed's network is small and (relatively) well maintained, so there probably won't be that much. You will need to simulate errors. We will use a simple error model where each packet is "dropped" uniformly at random. In this case, "dropping" the packet will simply mean that the server does not respond. Modify your server code to take an input parameter representing the percentage of packets to drop on server creation and drop packets in the manner specified. Submit your resulting code.
   (c) (15 points) **Ping Client.** Write the client side program for our ping application. Your client should perform the following functions:
      1. Send ten pings to the server.
      2. For each ping that is echoed, print the response message from the server.
      3. For each ping that is echoed, calculate and print the round trip time (RTT) for that packet.
      4. For each ping that is not echoed, print "*" for the RTT of that packet.
      5. Compute and display the average delay and packet loss rate (i.e., received / sent).

      During development, you can test on your own machine (without needing a second). To do this, start by running the server code first, and then run your client using your localhost IP (essentially make your computer ping itself). Once your code is complete, test it using the setup you created in the first part of this question. You should submit your code and the Wireshark capture of your tests using the setup in the first part.
2. (35 points) **HTTP.** In this problem, you will develop a web server capable of handling HTTP requests. We will follow HTTP 1.1, meaning that you should use TCP as your transport layer protocol and the client server model.

(a) (15 points) **Server.** You will need to write your web server. It should have the following functionality:

1. Accept an HTTP GET request.
2. Parse the accepted HTTP request.
3. Retrieve the requested object from the local file system.
4. If the object exists, craft an HTTP response containing the object.
5. If the object does not exist, craft a response using the appropriate status code.

Your server only needs to handle one request at a time. It does not need to handle other HTTP request methods. You should submit your code.

(b) (5 points) **Web Server Testing.** You can now test your server. You will need to have one machine act as the server, and another machine act as the client. You are encouraged to ask a classmate to act as one of the machines for testing. On the machine acting as the server, create an HTML file in the same directory your server file is in. Run your server program. Find the IP address of the host running the server. On the client machine, open a browser and provide the corresponding URL. You should test both objects that exist on the server and objects that do not. Provide your HTML file and images of your browser for both tests.

Testing Example. Assume your setup contains the following conditions:

- Your HTML file is named "test-object.html".
- Your IP address is "123.42.90.210".
- Your server is listening on port 6789.

You should input the URL "http://123.42.90.210:6789/ilovenetworks.html" (without quotes) into the address bar of your browser. If you do not include a port number (preceded by a :) after the IP address, the browser will assume the default for HTTP, port 80.

(c) (15 points) **Web Client.** You will also write a simple web client to go with your server. Your client should have the following functionality:

1. Take as input an IP address, port, and object path.
2. Connect to the server at that IP address using a TCP connection.
3. Send an HTTP request using the GET method to the server for the specified object.
4. Display the server's response.

You should submit your code and a Wireshark capture of the packets sent in both directions.

3. (10 points) **Code Style.** Your code should follow good style. There are no specific guidelines (beyond commenting appopriately), but your code should be easy to follow and should use clear and efficient methods for solving problems.