**Network**

Compile the Server Program:

gcc server.c -o server

Run the Server:

./server

Explore Socket Options:

* Modify the server code to experiment with different socket options.
* Test with Telnet or Netcat:

telnet localhost 8080

* Or using Netcat:

nc localhost 8080



Packet Inspection with [Wireshark](https://www.wireshark.org/docs/wsdg_html_chunked/):

* Use Wireshark to inspect network traffic. Capture packets on the relevant network interface and filter by IP address or port number.
* Install Wireshark:

Capture Packets:

* open Wireshark and start capturing packets on the network interface associated with the server.

Filter Packets:

* To focus on the packets related to their server, students can apply filters in Wireshark. For example, they can filter packets by IP address or port number.
* To filter packets based on the IP address of the server (replace 192.168.1.1 with the actual server's IP address):

ip.addr == 192.168.1.1

To filter packets based on the port number the server is using (replace 8080 with the actual port number):

tcp.port == 8080

Analyze Packets:

analyze individual packets to understand the details of the communication. They can look at protocol information, source and destination addresses, payload data, and more.

Follow TCP Stream:

Wireshark has a feature called "Follow TCP Stream" that allows users to see the entire conversation between a client and a server. This is useful for understanding the flow of data.

Right-click on a packet and select "Follow" -> "TCP Stream."

Save Capture:

Encourage students to save their Wireshark capture for further analysis or for inclusion in their documentation. They can save it in a format like PCAP (Packet Capture).

Apply Display Filters:

Display filters are entered in the filter bar at the top of the Wireshark window.

Performance Testing:

* Use tools like ab (Apache Benchmark) for basic performance testing:

ab -n 1000 -c 10 http://localhost:8080/