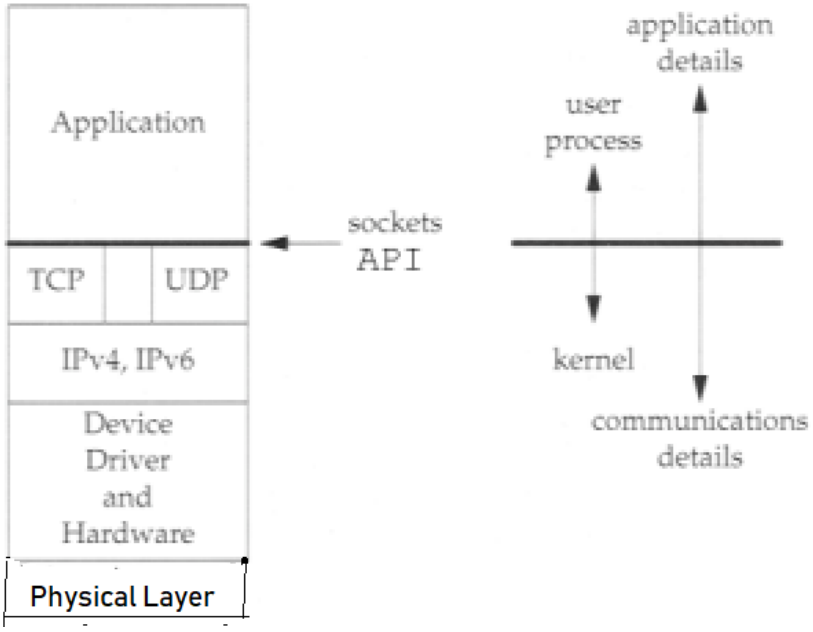
# Lab 5:
# Introduction to Socket Programming

Introduction:

A socket is an endpoint used for in Internet protocol communication; an IP address followed by a colon and a number: 172.217.2.110:80. The latter is the socket connection for **www.google.com** web server. This lab not meant to be a complete investigation of socket programming but is designed to 'get your feet wet' and begin to recognize ways in which networking tools can be used for troubleshooting.

The Client / Server Model is the basic architecture of socket programming. The server program executes first, creating a socket and listens for data to be received; the client executes second; it makes a connection to the listening port and sends the first network packet to the server. After initial contact, either the client or the server is capable of sending and receiving data through a full duplex connection.

The TCP protocol handles lost packets, duplicate packets, packet sequencing, flow control and congestion control. The diagram below gives a graphic view of socket programming using the TCP/IP protocol stack.




Ports

Sockets are the UNIQUE endpoints of an Internet communication which allow data to be exchanged. There are 65,535 ports on a PC. Suppose you are running an ftp client, a telnet client, and a web browser concurrently. When data arrives at the data-link layer, to which application should the packet

be delivered? Well part of the packet contains a value holding a port number, and it is this number which determines to which application the packet should be delivered. Some of the common protocol ports are listed below:

| Protocol | Port |
|---|---|
| File Transfer Protocol (FTP) | 20/21 |
| Secure Shell (SSH) | 22 |
| Telnet | 23 |
| Simple Mail Transport Protocol (SMTP) | 25 |
| Domain Name System (DNS) | 53 |
| Dynamic Host Configuration Protocol (DHCP) | 67/68 |
| Internet Message Access Protocol (IMAP) | 143 |

Ports 0 – 1023, are called “well-known ports” and are reserved by the IETF. Servers or clients that you create will not be able to bind to these ports unless you have root privilege.

Ports 1024 – 65535 are available for programs you write but beware other network applications maybe running and using these port numbers as well, so do not make assumptions about the availability of specific port numbers. **Keep your port number above 59000 to avoid conflict with other applications**.

1. Find a YouTube video that describes socket programming. I recommend the following video, where a simple socket connection is set up between server and client.

**[Python Tutorial -- Networking](https://youtu.be/XiVVYfgDolU)**

This video is written in Python, but you are free to use whatever language you prefer. Make sure that whatever tutorial you choose, that there is a server and a client, and that there is a simple 'Hello World' message being passed between the two.

You are free to _copy_ the code from whatever video you follow, without modification. *This isn't a programming lab*. But I do recommend that you type in manually whatever code you are given and do make sure to *attribute* whatever tutorial or video you choose to follow in your program's comments.

**2.** Once you have your server compiled and running. Type netstat -a to show the open port. Notice the port is “listening for client connections”. Take a screen shot. Name the file **learnname_listening.jpeg [0.4 Marks]**

**3.** Compile and run your client. Type netstat -a and show the connected port. Notice the port is now “established”. Take a screen shot and name your file **learnname_established.jpeg [0.4 Marks]**

4. Your client will connect to the port on localhost that you have opened. Instead of "Hello World", you should send **your username** to the server**. [0.4 Marks]**
5. Capture that packet using WireShark. Use a filter to capture only traffic on localhost. **[IPv4 loopback address is: 127.0.0.1 and IPv6 loopback address iss ::1] [0.675 Marks]**
6. 
7. The filter command is:

    - For Ipv4 enabled systems use command: ip.addr==127.0.0.1
    - For IPv6 enabled systems use command: ipv6.addr == ::1

(Hint: **install either utility Npcap or rawcap)**.

8. Save the capture file and submit it as well. Name the file **learnname_packet.jpeg**

Grading:

- **learnname_listening.jpeg**
- **learnname_established.jpeg**
- **learnname_message.jpeg**
- **learnname_packet.jpeg**
- **Executable code saved as a text file.**

Submit the files using the link available for Lab 5 in Graded Work. Remember replacing **learnname** with **your name** for submission.

This project will be due in a week. Good luck!