THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Thapar Institute of Engineering and Technology, Patiala

Computer Science and Engineering
Network Programming Laboratory (UCS413)
Assignment:1

1. Familiarity with Lab environment and understanding Client-Server model, Unix basic commands, socket programming headerfiles, and elementary socket system calls.

   (a) Understanding and using of commands like ifconfig, netstat, ping, arp, telnet, ftp, finger, traceroute, whois etc.

   (b) Socket header files contain data definitions, structures, constants, macros, and options used by socket subroutines. An application program must include the appropriate header file to make use of structures or other information a particular socket subroutine requires.

   Commonly used socket header files are:

   - **/usr/include/netinet/in.h**: Defines Internet constants and structures.
   - **/usr/include/netdb.h** Contains data definitions for socket subroutines.
   - **/usr/include/sys/socket.h** Contains data definitions and socket structures.
   - **/usr/include/sys/types.h** Contains data type definitions.
   - **/usr/include/arpa.h** Contains definitions for internet operations.
   - **/usr/include/sys/errno.h** Defines the errno values that are returned by drivers and other kernel-level code.

   (c) Elementary socket system calls:

   - **socket() System Call:** Creates an end point for communication and returns a descriptor:
     int socket (int AddressFamily, int Type, int Protocol);
   - **Bind( ) System call:** Binds a name to a socket. The bind subroutine assigns a Name parameter to an unnamed socket. It assigns a local protocol address to a socket.
     int bind (int sockfd, struct sockaddr *myaddr, int addrlen);
   - **connect() System call:** The connect function is used by a TCP client to establish a connection with a TCP server.
     int connect(int sockfd, struct sockaddr *servaddr, int addrlen);
   - **listen() System call:** This system call is used by a connection-oriented server to indicate that it is willing to receive connections.
     int listen (int sockfd, int backlog);
   - **accept() System call:** The actual connection from some client process is waited for by having the server execute the accept system call.
     int accept (int sockfd, struct sockaddr *cliaddr, int *addrlen);
   - **send( ), sendto( ), recv( ) and recvfrom( ) system calls:** These system calls are similar to the standard read and write functions.
   - **close( ) system call:** The normal Unix close function is also used to close a socket and terminate a TCP connection.
     int close (int sockfd);