

School of Computer Engineering
KIIT deemed to be University
Networks Laboratory Lesson Plan – Autumn'2023 (5th Semester)

Discipline: CSE/IT/CSCE/CSSE

Course name and Code: Networks Laboratory (IT 3095), (0-0-2, Cr: 1)

Instructor Name: Dr. Pinaki Sankar Chatterjee, [Email: pinakifcs@kiit.ac.in](mailto:pinakifcs@kiit.ac.in)

Instructor Chamber: Faculty Chamber-F211, Block-A, Campus-15

Technical Assistants Names:

Course Contents:

List of Experiments (Day wise):

Day-1

➤ *Aim of the experiment:*

1. Discuss what is networking and its significance in computer network. Discuss the components (i.e., h/w and s/w) required for data communication in a Computer Network. (Show the h/w components like Network Interface Card (NIC), Network Cable, RG-45 Connector, Hub, Switch, Router etc.)
2. Highlight the importance of socket programming as a s/w for data communication and the basic fundamentals required for doing socket programming using C.
3. Review of function, pointer, structure, structure with in a structure, pointer to structure, and command line argument concept using C programming Language.
4. What is little endian and big endian. Discuss the significance of endianness in computer network.

● **Assignments**

1. Write a C program to swap the content of 2 variables entered through the command line using function and pointer.
2. Write a C program to assign values to each member of the following structure. Pass the populated structure to a function Using call-by-value and another function using call-by-address and print the value of each member of the structure.
 struct student_info{

```

        int roll_no;
        char name[50];
        float CGPA;
        struct dob age;
    };

```

3. Write a C program to extract each byte from a given number and store them in separate character variables and print the content of those variables.

Input/Output:

```

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$ ./a.out 258
The input number = 258
digit in the 1st byte=2
digit in the 2nd byte=5
digit in the 3rd byte=0
digit in the 4th byte=0

```

4. Write a C Program to enter a number and store the number across the following structure and print the content of each member of the structure. Then aggregate each member of the structure to form the original number and print the same.

```

    struct pkt{
        char ch1;
        char ch2[2];
        char ch3;
    };

```

Input/Output:

```

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$ ./a.out 258
The input number = 258
digit in the 1st byte=2
digit in the 2nd byte=5
digit in the 3rd byte=0
digit in the 4th byte=0
1st member of the structure=2
2nd member of the structure=5,0
3rd member of the structure=0
The regenerated number = 258
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$

```

5. Write a C program to check whether the Host machine is in Little Endian or Big Endian. Enter a number, print the content of each byte location and Convert the Endianness of the same i.e. Little to Big Endian and vice-versa.

Input/Output:

```
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$ ./a.out 258
extracted byte from the LSB of the Number=2
extracted byte from the LSB of the Number=1
Memory representation of the Number
-----
Memory Address -> Value
-----
1668590496 -> 2
1668590497 -> 1
1668590498 -> 0
1668590499 -> 0
The LSB of the number is stored at the lowest memory address
Hence, the host machine is in little Endian
The Number is converted to Big Endian now
Memory representation of the Number
-----
Memory Address -> Value
-----
1668590444 -> 0
1668590445 -> 0
1668590446 -> 1
1668590447 -> 2
The number in Big Endian Format is 33619968
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$
```

Day-2

➤ Aim of the experiment:

1. Basics of Socket Programming.
2. Details of Connection less Socket programming APIs for TCP/IP stack using C.

• Assignment

1. Write a sender and receiver program in C by passing the IP address and the port number of each other through the command line arguments using connection less socket. Both of them will exchange messages with each other continuously. If any one of them will receive the “exit” message from the other end then both of them will close the connection. (Assume both the client and server are running with in the same host)

Input/Output:

```
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$ ./recv
Received Message::Hello
Received Message::Hi
Received Message::How r u?
Received Message::exit
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$ ./send
Enter the Message to Send:: Hello
Enter the Message to Send:: Hi
Enter the Message to Send:: How r u?
Enter the Message to Send:: exit
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_lab/Lab_By_Me/Lab1$
```

Day-3

➤ *Aim of the experiment:*

1. Demonstrate the packet Analyzer tool (Wireshark) to analyze the details of a packet which is captured during packet transmission in the network.

- **Assignments**

1. Analyze the packets using Wireshark, that are captured by running both client and server (connection less) with in the same host.
2. Analyze the packets using Wireshark, that are captured by running the client in one host and the server in another host (connection less).

Day-4

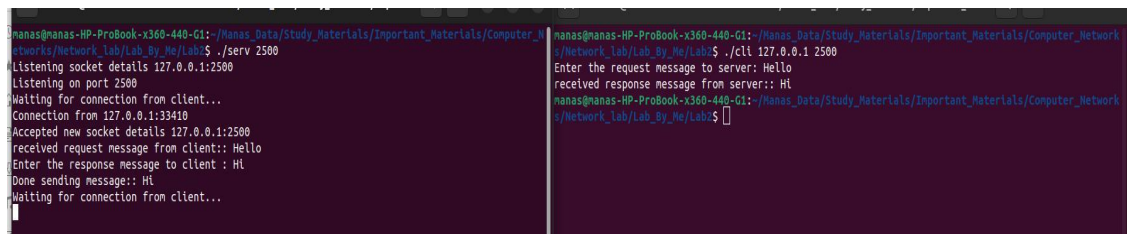
➤ *Aim of the experiment:*

1. Details of Connection Oriented Socket programming APIs for TCP/IP stack using C.

- **Assignment**

1. Write a connection-oriented client and server program in C using command line arguments. At the server side, pass the port number (to whom the server will bind to) in the command line. At the client side, pass the IP address and the port number of the server (to whom the client will connect to) as command line argument and carry out the following tasks.
 - i. Print all the relevant messages during the connection establishment at both the ends.
 - ii. After establishment of connection exchange messages. After message exchange is over then the client sends a “close” message to the server to tear down the connection.

Input/Output:



```
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/By_He/Lab2$ ./serv 2500
Listening socket details 127.0.0.1:2500
Listening on port 2500
Waiting for connection from client...
Connection from 127.0.0.1:33410
Accepted new socket details 127.0.0.1:2500
received request message from client:: Hello
Enter the response message to client : HI
Done sending message:: HI
Waiting for connection from client...

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/By_He/Lab2$ ./cli 127.0.0.1 2500
Enter the request message to server: Hello
received response message from server:: HI
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/By_He/Lab2$
```

Day-5

➤ *Aim of the experiment:*

1. Discuss how to design a Sequential Chart Server.

- **Assignment**

1. Write a connection-oriented client and server socket program using C where the server will behave as a chat server serving multiple chat clients but one at a time. When the chat server receives a “bye” message from a particular client then it terminates the respective connection with that client.

Input/Output:

```

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Simple Chart Server$ ./cli 127.0.0.1
Waiting for the message from the server....
Received message from server: Welcome to my server
Send message to server: Hello
Waiting for the message from the server....
Received message from server: Hi
Send message to server: How r u?
Waiting for the message from the server....
Received message from server: I am fine
Send message to server: bye
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Simple Chart Server$

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Simple Chart Server$ ./serv
You got a connection from client : 127.0.0.1
Send message to client :Welcome to my server
Received Message from the Client: Hello
Send message to client :Hi
Received Message from the Client: How r u?
Send message to client :I am fine
Received Message from the Client: bye
Connection Terminated

```

Day-6

- *Aim of the experiment:*

2. Discuss the overview of file transfer over a computer network.

- **Assignment**

1. Write a connection-oriented client and server program in C using command line arguments. Do the file transfer from the server as follows.
 - i. Server first sends the list of files present in the current directory at it's own end.
 - ii. After receiving the same, client send the name of a file it wants to download from the server.
 - iii. Finally, after receiving the same server uploads the file to the client.
 - iv. After sending the file, server closes the client connection at its own end.

Input/Output:

```

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2$ gcc ftpserver.c -o serv
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2$ ./serv 6000
Listening on port 6000
Waiting for connection from client...
Connection from 127.0.0.1:53664
Trying to open the file: client.c
Done sending file client.c
Waiting for connection from client...

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2$ ./cli 127.0.0.1 6000
Enter filename: client.c
bytes received=2000
bytes received=632
bytes received=0
Written file client.c-dl successfully
Enter filename:

```

Day-7

- *Aim of the experiment:*

3. What is I/O multiplexing and why it is required?
4. Discuss different types of I/O multiplexing.
5. Discuss how to design a concurrent chat server using fork().

- **Assignment**

1. Design a connection oriented concurrent chat server using fork() in C where the server will serve multiple chat clients simultaneously. When the chat server receives a “exit” message from a particular client then it terminate the respective connection with that chat client.

Input/Output:

```

manas@manas-HP-ProBook-x360-440-G1: ~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_1/hadling_multiple_clients_using_fork$ ./cli 12 7.0.0.1
Sending message to the server....
Hello
Waiting for message from the server....
Received Server Message: Hello
Sending message to the server....
Fine
Waiting for message from the server....
Received Server Message: Fine
Sending message to the server....
[

manas@manas-HP-ProBook-x360-440-G1: ~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_1/hadling_multiple_clients_using_fork$ ./serv
parent process
creating child process
Waiting for the message from the client::127.0.0.1->43852
Received Client Message: Hello
Sending back the message to the client::127.0.0.1->43852
Waiting for the message from the client::127.0.0.1->43852
Received Client Message: Fine
Sending back the message to the client::127.0.0.1->43852
Waiting for the message from the client::127.0.0.1->43852
parent process
creating child process
Waiting for the message from the client::127.0.0.1->58716
Received Client Message: Hi
Sending back the message to the client::127.0.0.1->58716
Waiting for the message from the client::127.0.0.1->58716
Received Client Message: exit
Closing the child process for client::127.0.0.1->58716
[

manas@manas-HP-ProBook-x360-440-G1: ~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_1/hadling_multiple_clients_using_fork$ ./cli 12 7.0.0.1
Sending message to the server....
Hi
Waiting for message from the server....
Received Server Message: Hi
Sending message to the server....
exit
manas@manas-HP-ProBook-x360-440-G1: ~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_1/hadling_multiple_clients_using_fork$

```

Day-8

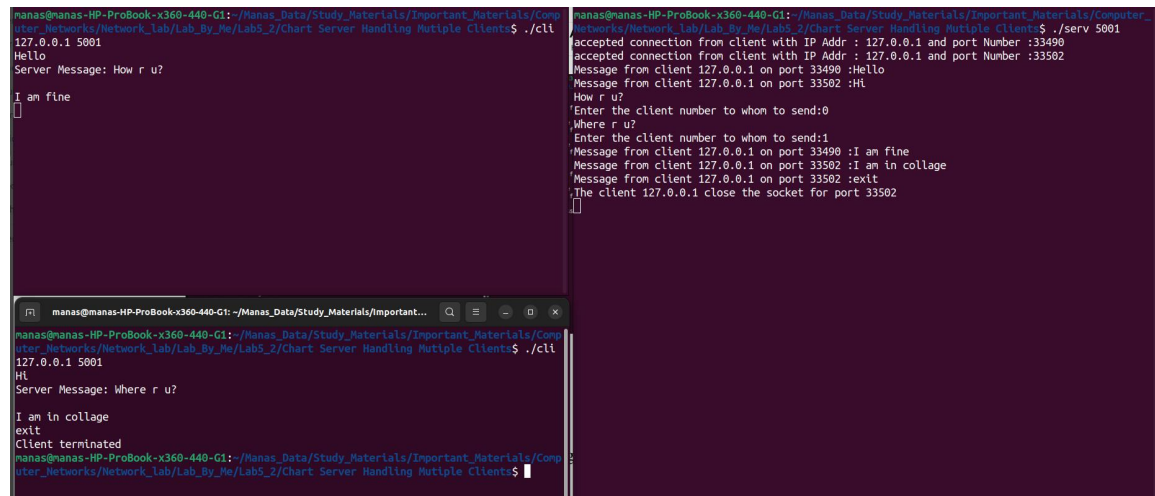
- *Aim of the experiment:*

1. Details of I/O multiplexing using select() API.
2. Discuss how to design a concurrent chat server using select().

- **Assignment**

1. Design a connection oriented concurrent chat server using select() in C where the server will serve multiple chat clients simultaneously. When the chat server receives a “exit” message from a particular client then it terminate the respective connection with that chat client.

Input/Output:



```
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Chart Server Handling Multiple Clients$ ./cli
127.0.0.1 5001
Hello
Server Message: How r u?
I an fine

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Chart Server Handling Multiple Clients$ ./serv 5001
accepted connection from client with IP Addr : 127.0.0.1 and port Number :33498
accepted connection from client with IP Addr : 127.0.0.1 and port Number :33502
Message from client 127.0.0.1 on port 33498 :Hello
Message from client 127.0.0.1 on port 33502 :Hi
How r u?
Enter the client number to whom to send:0
Where r u?
Enter the client number to whom to send:1
Message from client 127.0.0.1 on port 33498 :I an fine
Message from client 127.0.0.1 on port 33502 :I an in collage
Message from client 127.0.0.1 on port 33502 :exit
The client 127.0.0.1 close the socket for port 33502

manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Chart Server Handling Multiple Clients$ ./cli
127.0.0.1 5001
Hi
Server Message: Where r u?
I an in collage
exit
Client terminated
manas@manas-HP-ProBook-x360-440-G1:~/Manas_Data/Study_Materials/Important_Materials/Computer_Networks/Network_Lab/Lab_By_Me/Lab5_2/Chart Server Handling Multiple Clients$
```

Day-9

➤ Aim of the experiment:

1. Introduction to network simulator tool (NS2/NS3/Packet Tracer) and its application in research work.
2. Demonstration of how routing works using NS2/NS3/Packet Tracer.

• Assignments

1. Simulate routing of packets in a LAN with in the same subnet.
2. Simulate routing of packets in a LAN with different subnets.

Day-10

➤ Aim of the experiment:

1. Comparison and analysis of some routing protocols using NS2/NS3/Packet Tracer.

• Assignment

1. Compare and analyze DSR and AODV protocol using NS2/NS3/Packet Tracer.

Grading Policies:

- *Continuous Evaluation components:* Continuous evaluation for 60 marks Consists following components:
- **Lab participation (10 Marks):** Students' participation in the lab based on their attendance and engagement.

- **Lab records (10 Marks):** Neatly written lab records based on the assignments to be evaluated.
- **Continuous evaluation (based on Lab skills, 20 Marks):** Students' lab skills will be assessed through hands-on activities and involvements in doing assignments during the lab hour.
- **Use a variety of evaluation methods to get a comprehensive assessment of student learning. (20 Marks)**
 - End semester evaluation: *Comprehensive assessment of student learning and performance. (40 Marks)*

Practice Problem Sets: – Problems on socket configuration and communication on different application protocols will be given.

Reference Materials: –

1. Beej's Guide to Network Programming Using Internet Sockets(https://beej.us/guide/bgnet/pdf/bgnet_a4_c_2.pdf)
2. Unix Network Programming, Volume 1: The Sockets Networking API (Addison-Wesley Professional Computing Series) by W. Stevens, Bill Fenner, and Andrew Rudoff,
3. Wireshark Tutorial for Beginners - YouTube(<https://www.youtube.com/watch?v=TkCSr30UojM>)
4. Cisco Packet Tracer(<https://www.netacad.com/courses/packet-tracer>)