SOCKETS

5 mechanism(pipe/fifo/sm/mq/sem) =>
communication within same system

But through socket=>
Either in same or different machines

Socket=>2 processes in same or diff machines can communicate

Note: It is not system communication
but process communication

2 types
a)Connection oriented [TCP]
(client server)
(slow)
(more reliable)

b)Connectionless [UDP]
(no client server)
(takes address of dst and data content)
and simply sends it)
(fast)
(less reliable)
eg:live streaming has some data loss

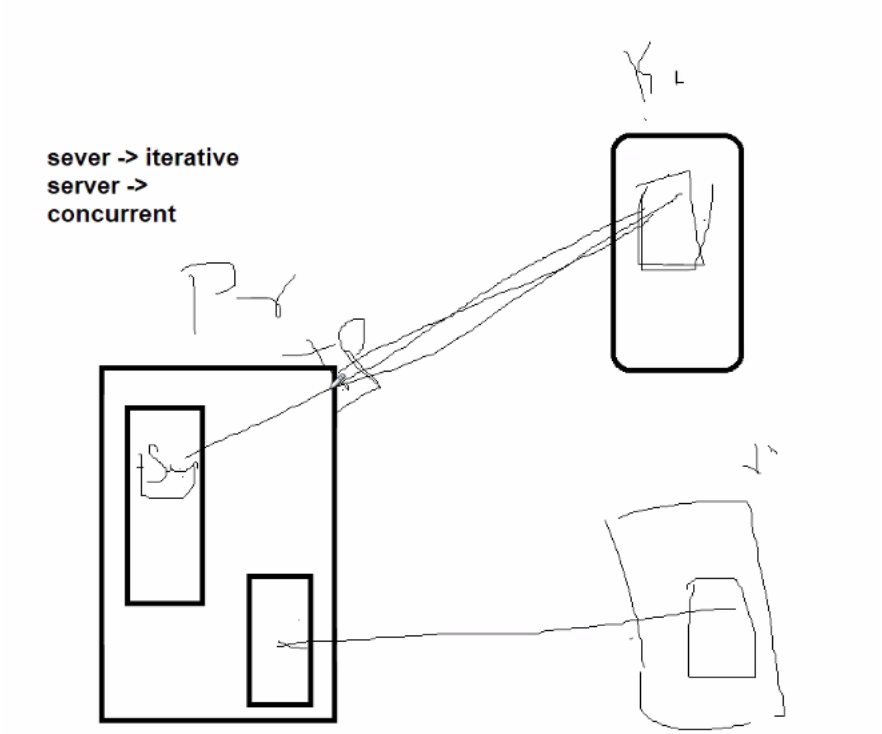
Note:Every system has 2 ip address
a)local ip address
b)ethernet/wlan ip address

cmd: ip a

Note: when both processes are in same system=>local/eth ip
different system=>ethernet

A server can handle only 1 client at a time=>ITERATIVE SERVER

A server can handle many client at a time =>CONCURRENT SERVER
1 client comes ...creates a child using fork()

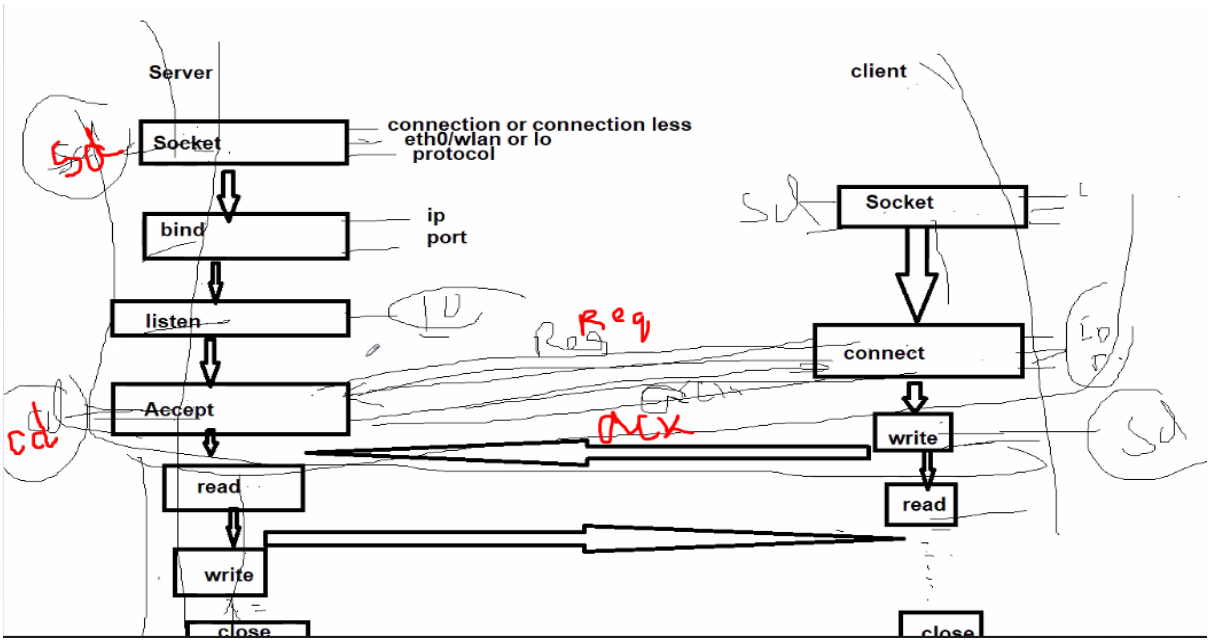


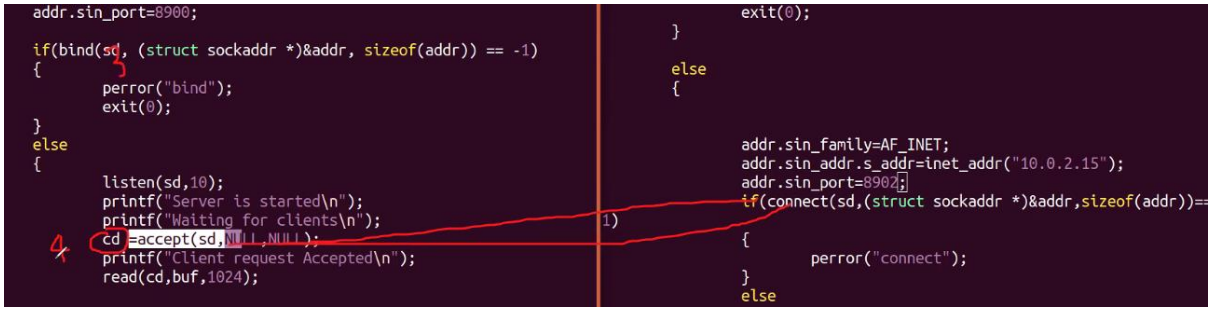
client should know the ip address and port no of server
server need not know

2 processes in server
They have same ip address but diff port no

CONNECTION ORIENTED COMMUNICATION

Socket steps





```
addr.sin_port=8900;

if(bind(sd, (struct sockaddr *)&addr, sizeof(addr)) == -1)
{
        perror("bind");
        exit(0);
}
else
{
        listen(sd,10);
        printf("Server is started\n");
        printf("Waiting for clients\n");
        cd =accept(sd,NULL,NULL);
        printf("Client request Accepted\n");
        read(cd,buf,1024);
```

```
        exit(0);
}

else
{

        addr.sin_family=AF_INET;
        addr.sin_addr.s_addr=inet_addr("10.0.2.15");
        addr.sin_port=8902;
        if(connect(sd,(struct sockaddr *)&addr,sizeof(addr))==
1)
        {
                perror("connect");
        }
        else
```

In order to communicate , we need a node=>socket
Socket is a communication end point

Socket determines is it conn oriented or conn less
Socket determines is it local or eth ip address
Socket determines the protocol (tcp/udp)

Note: port no should be > 8080

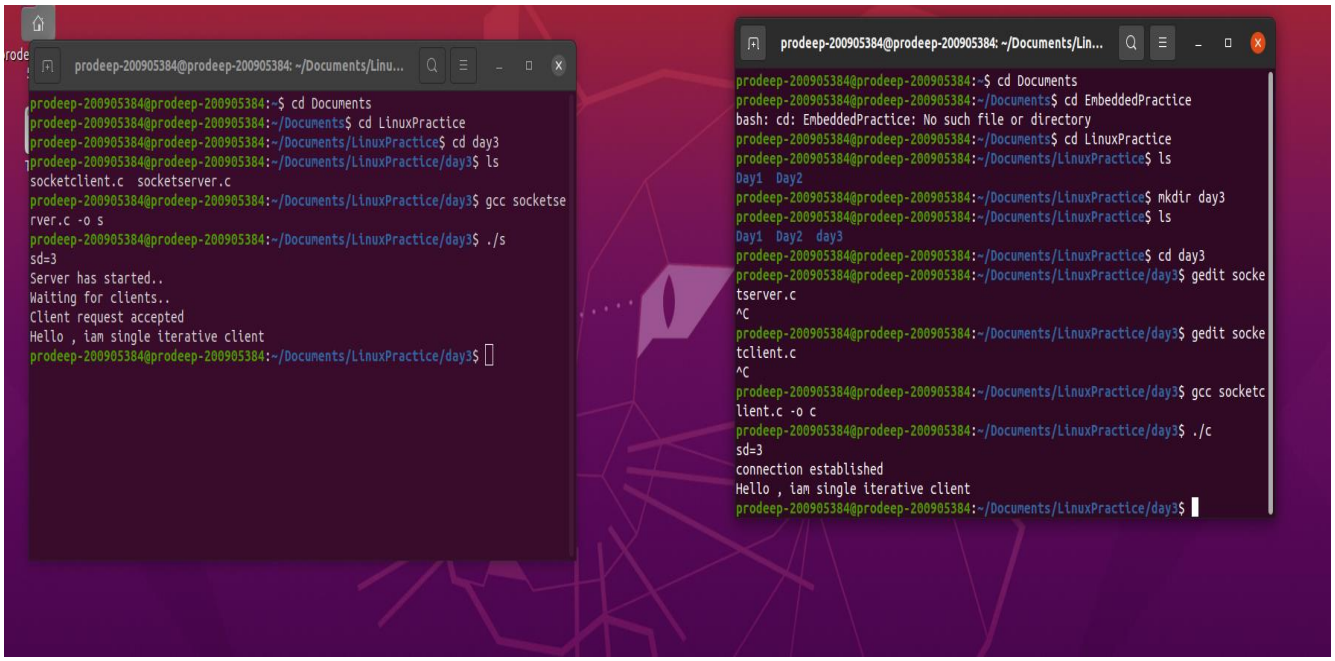
Only when server is in listen mode =>it is operational and can accept request from client

Listen =>decides max no of clients

Accept=>blocking system call & waits for client connections

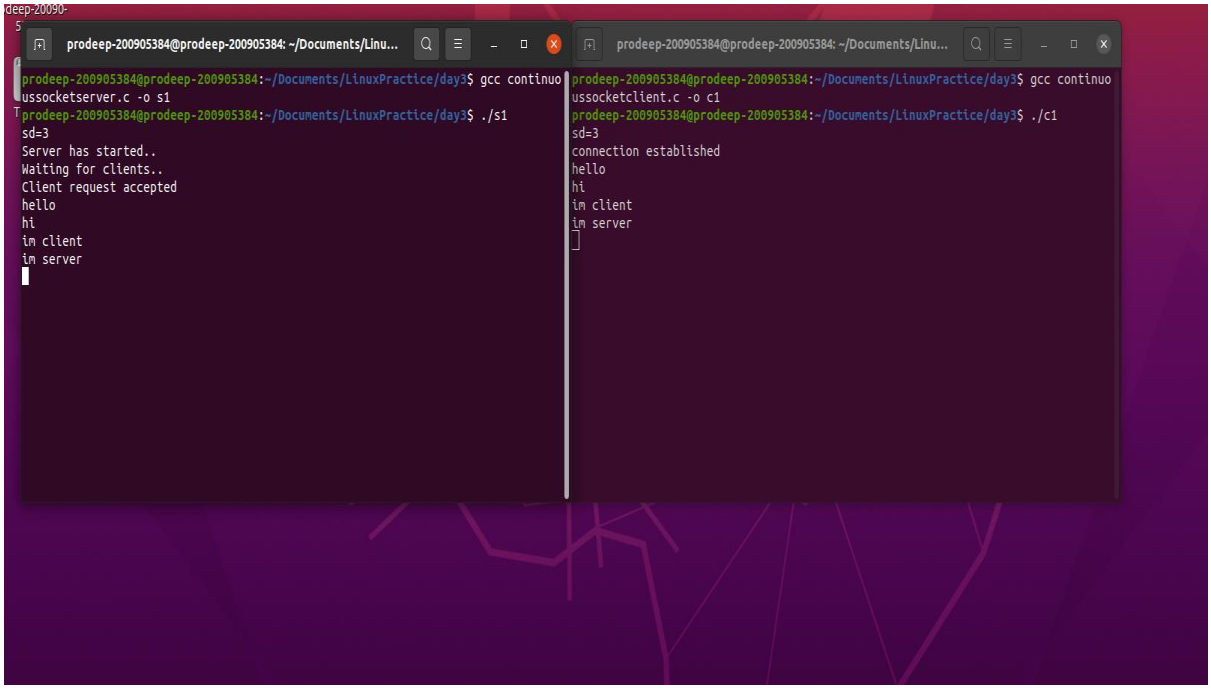
Server has diff client descriptor for every client

Program for iterative server

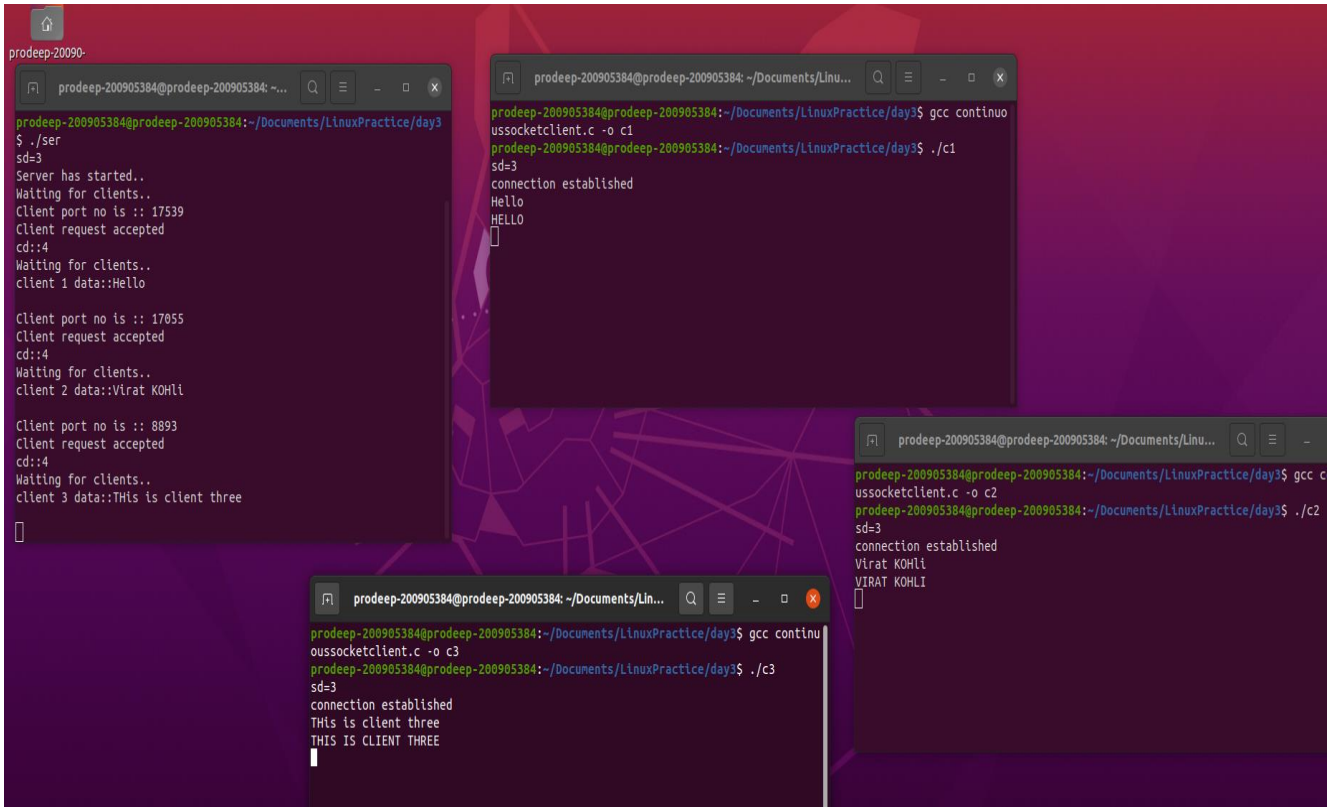


```
prodeep-200905384@prodeep-200905384: ~/Documents/Linu...
prodeep-200905384@prodeep-200905384:~$ cd Documents
prodeep-200905384@prodeep-200905384:~/Documents$ cd LinuxPractice
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice$ cd day3
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ ls
socketclient.c  socketserver.c
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ gcc socketse
rver.c -o s
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ ./s
sd=3
Server has started..
Waiting for clients..
Client request accepted
Hello , iam single iterative client
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$
```

```
prodeep-200905384@prodeep-200905384: ~/Documents/Lin...
prodeep-200905384@prodeep-200905384:~$ cd Documents
prodeep-200905384@prodeep-200905384:~/Documents$ cd EmbeddedPractice
bash: cd: EmbeddedPractice: No such file or directory
prodeep-200905384@prodeep-200905384:~/Documents$ cd LinuxPractice
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice$ ls
Day1  Day2
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice$ mkdir day3
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice$ ls
Day1  Day2  day3
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice$ cd day3
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ gedit socke
tserver.c
^C
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ gedit socke
tclient.c
^C
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ gcc socketc
lient.c -o c
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ ./c
sd=3
connection established
Hello , iam single iterative client
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$
```



Program for concurrent server




```
//server parent process is just accepting connections
//server child is converting to uppercase
```

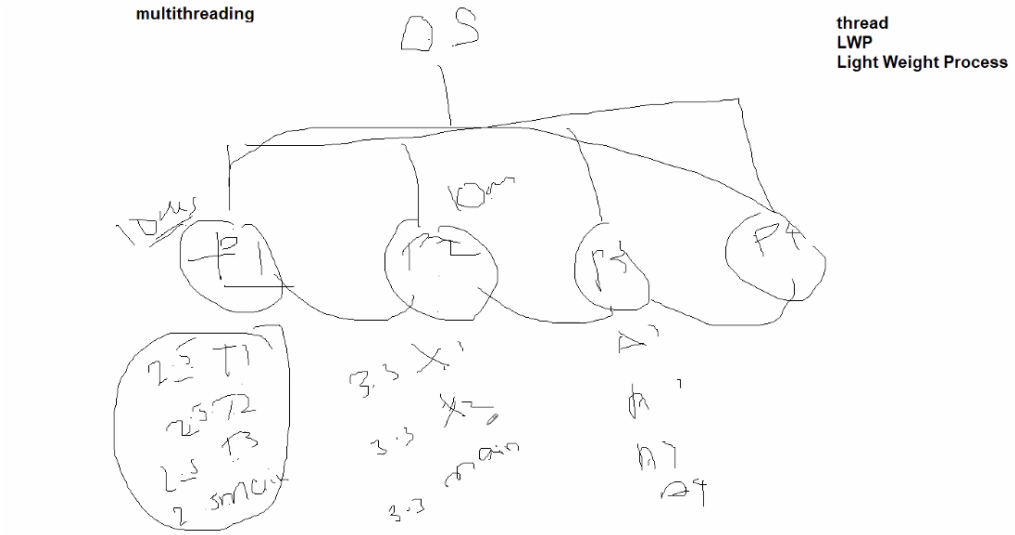
//cd changes every time new client added ...in accept()
//same client prg c1,c2,c3

MULTITHREADING

Processes: Heavy and bulky
System is loaded with a lot of resources
Lot of overhead needed =>pcb ,tables

We can achieve multitasking by thread which is better than process

Threads :Lightweight processes
Many Threads exist inside 1 process
Atleast 1 thread is running =>Main thread
Even if we dont create a thread , 1 thraed already exists =>
Main Thread



Os doesnt know about threads
OS only knows about process

eg:P1 has 3 threads and 1 main thread
OS has alloted 10ms to P1

th1/th2/th3/main th=>each get 2.5 ms

MULTICORES
eg:quadcore=>4 cores

In the above example, in p1
1st core=>th1=>10ms and so on

Note:Multicore utilized better in multithreading

All threads share the same memory,pcb,code/data/stack/heap segment

---

fns execute serially
f1();
f2();
f3();
//if f1 goes in infinite loop ..f2 & f3 wont execute

threads execute concurrently

Thread->pthreads.h file

to compile:
gcc thread_demo.c -o test -lpthread
./test

Note:main thread has to wait for other threads to complete
Threads alone cannot survive without main thread

We can have same func , executed by many threads but different arg

---
limitations of thread

1)If one thread has issue out of 5 threads in a process,
=>entire process will be terminated

2)race condition

to achieve synchronization in threads =>mutex locks
makes threads to wait & run sequentially
application:print multiplication table

```
void myfun1()
{
        int count=0;
        while(1)
        {
                printf("in myfun1\n");
                //printf("in myfun1 process id = %d thread id = %lu\n",getpid(),pthread_self());
                printf("count = %d\n",count);
                count++;
                if(count == 60)
                        pthread_cancel(tid2);
                if(count == 100)
                        pthread_cancel(tid3);
                if(count == 140)
                        pthread_cancel(pthread_self());
                usleep(500000);
        }
}
```

```
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ gcc threadsync2.c -o ts2 -lpthread
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$ ./ts2
Thread 15 is trying to take lock
Thread 15 takes lock
Multiplication Table of 15
Thread 6 is trying to take lock
Thread 12 is trying to take lock
15 x 1=15
15 x 2=30
15 x 3=45
15 x 4=60
15 x 5=75
15 x 6=90
15 x 7=105
15 x 8=120
15 x 9=135
15 x 10=150
*******************************Thread 15 releases the  lock
Thread 6 takes lock
Multiplication Table of 6
6 x 1=6
6 x 2=12
6 x 3=18
6 x 4=24
6 x 5=30
6 x 6=36
6 x 7=42
6 x 8=48
6 x 9=54
6 x 10=60
*******************************Thread 6 releases the  lock
Thread 12 takes lock
Multiplication Table of 12
12 x 1=12
12 x 2=24
12 x 3=36
12 x 4=48
12 x 5=60
12 x 6=72
12 x 7=84
12 x 8=96
12 x 9=108
12 x 10=120
*******************************Thread 12 releases the  lock
prodeep-200905384@prodeep-200905384:~/Documents/LinuxPractice/day3$
```
