Computer Networks Lab
Week-3
Aug 6, 2021

Venkata Naga Sai Ram Nomula
RA1911033010021
L2 - SWE
Aim:
To simple tcp/ip client server communication
Procedure:
STEP 1: CREATE A FOLDER (Regno)
STEP 2: CREATE a filename server.c
STEP 3: open or click server.c
STEP4: WRITE THE PROGRAM IN server.c
STEP5: CREATE a filename client.c
STEP6: open or click client.c
STEP7: Write the program for client.c
STEP8: OPEN A NEW TERMINAL
STEP9: Type cd foldername
STEP10: Type cc server.c
STEP11: Type ./a.out
STEP12: Open one more terminal
STEP13: Type cc client.c
STEP14: Type ./a.out 127.0.0.1
STEP15: Type any message, say hello in the client terminal
STEP16: Verify its received in the server

Code:
Server.c

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<netdb.h>
#include<arpa/inet.h>
#include<string.h>
int main(int argc,char*argv[])
{
int bd,sd,ad;
char buff[1024];
struct sockaddr_in cliaddr,servaddr;
socklen_t clilen;
clilen=sizeof(cliaddr);
bzero(&servaddr,sizeof(servaddr));
/*Socket address structure*/
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(2564);
/*TCP socket is created, an Internet socket address structure is
filled with wildcard address & server’s well known port*/
sd=socket(AF_INET,SOCK_STREAM,0);
/*Bind function assigns a local protocol address to the
socket*/
bd=bind(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
/*Listen function specifies the maximum number of connections that
kernel should queue for this socket*/
listen(sd,5);
printf("Server is running....\n");
/*The server to return the next completed connection from
the front of the
completed connection Queue calls it*/
ad=accept(sd,(struct sockaddr*)&cliaddr,&clilen);
while(1)
{
bzero(&buff,sizeof(buff));

/*Receiving the request message from the client*/

recv(ad,buff,sizeof(buff),0);
printf("Message received is %s\n",buff);
}
}
```

Client.c

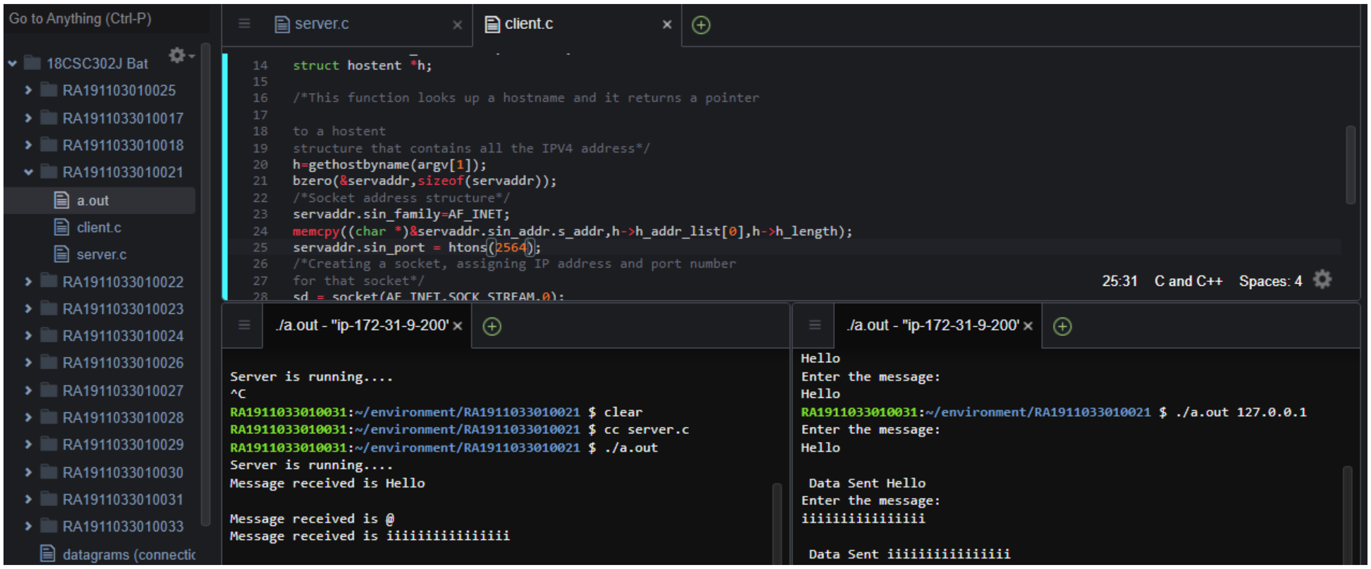
```
#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<unistd.h>
#include<netinet/in.h>
```

```
#include<netdb.h>
#include<arpa/inet.h>
int main(int argc,char * argv[])
{
int cd,sd,ad;
char buff[1024];
struct sockaddr_in cliaddr,servaddr;
struct hostent *h;

/*This function looks up a hostname and it returns a pointer

to a hostent
structure that contains all the IPV4 address*/
h=gethostbyname(argv[1]);
bzero(&servaddr,sizeof(servaddr));
/*Socket address structure*/
servaddr.sin_family=AF_INET;
memcpy((char *)&servaddr.sin_addr.s_addr,h->h_addr_list[0],h->h_length);
servaddr.sin_port = htons(2564);
/*Creating a socket, assigning IP address and port number
for that socket*/
sd = socket(AF_INET,SOCK_STREAM,0);
/*Connect establishes connection with the server using
server IP address*/
cd=connect(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
while(1)
{
printf("Enter the message: \n");
/*Reads the message from standard input*/
fgets(buff,100,stdin);
/*Send function is used on client side to send data
given by user on client
side to the server*/
send(sd,buff,sizeof(buff)+1,0);
printf("\n Data Sent ");
//recv(sd,buff,strlen(buff)+1,0);
printf("%s",buff);
}
}
```

Output:



```c
struct hostent *h;

/*This function looks up a hostname and it returns a pointer

to a hostent
structure that contains all the IPV4 address*/
h=gethostbyname(argv[1]);
bzero(&servaddr,sizeof(servaddr));
/*Socket address structure*/
servaddr.sin_family=AF_INET;
memcpy((char *)&servaddr.sin_addr.s_addr,h->h_addr_list[0],h->h_length);
servaddr.sin_port = htons(2564);
/*Creating a socket, assigning IP address and port number
for that socket*/
sd = socket(AF_INET,SOCK_STREAM,0);
```

```
Server is running....
^C
RA1911033010031:~/environment/RA1911033010021 $ clear
RA1911033010031:~/environment/RA1911033010021 $ cc server.c
RA1911033010031:~/environment/RA1911033010021 $ ./a.out
Server is running....
Message received is Hello

Message received is @
Message received is iiiiiiiiiiiiiiii
```

```
Hello
Enter the message:
Hello
RA1911033010031:~/environment/RA1911033010021 $ ./a.out 127.0.0.1
Enter the message:
Hello

 Data Sent Hello
Enter the message:
iiiiiiiiiiiiiiii

 Data Sent iiiiiiiiiiiiiiii
```